Mirror Force

A Visual Tool to Aid Scrum Teams

STUDENTS

Marco Antonio Almeida (105347)
Stijn Frankfoorder (108164)

SUPERVISOR

Jacob Nørbjerg

PAGES | CHARACTERS

117 pages | 232.518 characters including spaces

CBS
COPENHAGEN
BUSINESS SCHOOL
HANDELSHØJSKOLEN

MASTER THESIS | E-BUSINESS | MAY 2018

Abstract

Agile practitioners focus on working code rather than spending too much time with tools. Nonetheless, gaining and maintaining awareness is essential for the success of Agile software projects. A way to do so is through dashboards, a useful project management tool. This research studies the relationship between Agile team members with their tools and provides a solution for those teams. Overall, this research answers the following question: *How are Scrum teams affected by their tools and how can dashboards aid those teams?*

To collect the necessary data, we have conducted semi-structured interviews with twelve individuals from six different companies so that we could outline characteristics of the tools being used. The findings from the interviews, mixed with the theory of Information Visualization and Scrum inspired the design of our prototype, named *Mirror Force*. Later, the prototype was evaluated, through a usability test followed by a group interview.

From the individual interviews, we have identified 11 categories of requirement and constraints in the tools being used. The solution was designed in the form of an integrated ceremony-based dashboard system. From the usability test, we have found that some design heuristics were successfully applied in the prototype, however, the dashboard that represented the work of the Product Owner did not fully succeed due to the addition of elements that were still unknown by the participants.

While the research process was conducted with rigor, several challenges were faced in order to manage the relationship with the participating companies. Although it did not deeply impact our data gathering process, we continuously had to make adjustments to our data gathering plan. The evaluation of our prototype, however, showed us the potential of the solution where an assortment of directions could be taken in order to create a product useful for Scrum teams.

Table of Contents

List of Tables

List of Figures

1. Introduction

1.1. Background

Our eyes are our windows to the world. It is through them that 90% of our knowledge is yielded, placing it as the human's most dominant sense (Pocock, 1981). Indeed, their supremacy over our other senses is so strong that they reduce the impact of our other faculties. Only with closed eyes are we fully able to savor the taste of a simple apple, the smell of a properly aged wine, the sound of T-Bone Walker's guitar, and the touch of the loved one. The importance that visual perception has in cognitive systems is, therefore, paramount.

Nevertheless, vision alone rarely is the sole responsible for intellectual work. To accomplish cognition, the support from other cognitive tools is most needed, such as pencils and paper, calculators, computers, and information systems (Ware, 2004). The particular relationship between the visual system and the computer is an interesting subject. The first, a powerful pattern-finding mechanism and the latter, a powerful data storage and information processor. Through a display, a computer can provide visualizations of large quantities of data that can be quickly interpreted by the human eye if presented well (Ware, 2004). Information Visualization is the name given to the study of visual representations of data to boost human cognition.

Inside organizational settings, Information Visualization takes place in a variety of forms that aim to improve awareness, collaboration, coordination, and decision-making (Gutwin, Penner & Schneider, 2004; Tory & Möller, 2004). One form of visualization, which has been increasing in popularity is accomplished through dashboards (Paredes, Anslow, & Maurer, 2014; Yigitbasioglu & Velcu, 2012). A dashboard is defined as a "visual display of the most important information needed to achieve one or more objectives; consolidated and arranged on a single screen so the information can be monitored at a glance" (Few, 2006, p. 34). Although the origin of dashboards in an organizational setting can be traced back to the early 1980s, it was only a decade later that its potential was realized. With the advent of data warehousing and the introduction of the Balanced Scorecard, managers needed a method that could easily monitor the pre-established metrics and compare them with the ever-increasing amount of data generated by the business (Few, 2006).

Nowadays, the use of dashboards has expanded to different settings. One such situation where dashboards are useful is in the management of software projects, fostering awareness through communication and coordination (Treude & Storey, 2010). In this paper, our focus leans towards a specific methodology of software project management, also known as Agile.

Agile is an approach to software development that challenges the traditional way of developing software. It is a term that is used to cover many different methodologies towards software development that share the same common goal: being able to quickly respond to changes (Coram & Bohner, 2005). Scrum and Extreme Programming (XP) are among the most popular Agile methodologies.

The focus of Agile practitioners is on working code rather than spending too much time with tools. Nonetheless, gaining and maintaining awareness of various aspects such as project status and bottlenecks is essential for the success of Agile software projects (Treude & Storey, 2010). Thus, there is a need for lightweight Agile tools that can provide at a glance reports, such as dashboards, to aid these teams to quickly convey the information they need, increasing their overall awareness (Paredes et al., 2014).

1.2. Motivation

The inappropriateness of tools has been mentioned as one of the factors that lead Agile teams to failure (Chow & Cao, 2008). In a lightweight process where individuals and interactions are perceived as more important than processes and tools, one would assume that the tools would have more of a supporting role. However, it is part of our own experience that tools sometimes take over the processes and team members start to be limited to the extent of features provided by their tools.

There are some studies that investigate the relationship between Agile software development teams and their tools (Azizyan, Magarian, & Kajko-mattson, 2011; Dubakov & Stevens, 2008). However, they pursue a statistical focus, leaving little room for understanding how the features of the tools affect the daily work of teams. Moreover, since their research, many new Agile project management tools have reached the market, meaning that these tools could affect Agile development teams in yet unknown ways.

Furthermore, other studies propose the creation of new visualization tools that would facilitate parts of the existing processes in an Agile team (Biehl, Czerwinski, Smith, & Robertson, 2007; Jakobsen et al., 2009; Mateescu, Kropp, Burkhard, Zahn, & Vischi, 2015). These solutions, on the other hand, focus specifically on the development team and not so much on the other supporting roles, such as the project and product manager.

Our motivation with this research was to reduce the gap between these two areas of research. By following a qualitative approach to understand the usage and needs of Agile tools, we designed and tested a solution based on Information Visualization that can support all the different roles of an Agile team.

1.3. Research Question

This research pursued two objectives: to understand what is the relationship of the members of Agile teams with their tools and to provide a solution that those same teams would benefit from for being easily deciphered. To reach these objectives, this thesis had two approaches to theory building. First exploratory, where we looked into real-world organizations, their Agile practices, and their tools, and last explanatory, summoning not only the findings from the first part of the research but also the existing theory in order to create an artifact that would optimize their process.

It is noteworthy that the main contribution of this investigation is the evaluation of the prototype, however, without a proper comprehension of the requirements and constraints of the tools Agile teams use, the later part of the research would hardly be able to convey any benefits for the investigated teams.

Consequently, this thesis' research question is designed in such a way that combines both our objectives:

> *How are Scrum teams affected by their tools and how can dashboards aid those teams?*

The first clause highlights the exploratory nature of the research, investing time in understanding which tools already exists and what can be improved. The second clause, on the other hand, has explanatory features, combining theory with practice to form a tool that aims to ease the pain points mentioned beforehand.

1.4. Process and Delimitation

The research question has been scoped to a specific methodology of Agile called Scrum, due to the fact that this was the framework followed by all the investigated teams. Moreover, the solution has also been scoped to a dashboard, which is a specific implementation of Information Visualization, an already proven method to foster communication and coordination (Eckerson, 2010). These considerations establish the boundaries of our research making it more achievable, considering the available resources, and yet being challenging enough.

The research process consists of two stages. First, a knowledge gathering phase, with the focus on understanding the constraints and requirements that Scrum teams have when using their tools. The second phase was characterized by an evaluation of a prototype, built by connecting the findings from the first stage with the theory of Information Visualization. This prototype, called *Mirror Force*, portrays the core contribution of this thesis.

Since the proposed artifact comes in the shape of an information system, it made sense to look into the current literature on research methodologies for designing information systems. Design Science was identified as a common way to describe and defend the choices made when designing software, including dashboard based tools (Peffers, Tuunanen, & Rothenberger, 2007; Santiago Rivera, & Shanks, 2015). Although we opted to align our research methodology with the traditional social sciences paradigms, we acknowledged the relevance of Design Science by outlining its relation to our investigation.

One multiple-case study was conducted with six Scrum teams from companies in different industries. Through a series of interviews, we have looked into their process, their needs, and the difference between the Scrum roles when using these tools. We have initially assumed that, even though these roles are connected by the same project management tools, they have different needs to fulfill their daily tasks, therefore, they need specific perspectives from those tools.

From the analysis of the interviews, we were able to identify the most important requirements and constraints they had in their current set of tools, which would be plotted in a solution in the form of a dashboard. Moreover, we found some support for our initial assumption that different Scrum roles had different needs from their tools.

Though we did not explicitly expect to see human interaction as a relevant outcome of this part of our study, it evolved during this research as an understanding of how the tools are used. Whenever the tool fails to quickly convey information, colleagues tend to communicate to solve issues. Tools can tell what has happened during an iteration, but they seldom can tell why it happened, individuals can tell why.

The output from the interviews was considered the most valuable input for devising the prototype. Furthermore, we have also done a systematic review of the project management tools that those teams use in order to gather inspiration for a new tool. This served as a diagnosis step, which allowed us to identify a gap where our solution could be placed.

Theories on Information Visualization, Design Heuristics, Displays, and Agile also played a part in explaining why the solution took the format it did. Our proposed solution, *Mirror Force*, acts as an integrated ceremony-based dashboard. It facilitates meetings, providing an agenda that is connected with the many different tools being used by the teams.

With a prototype of *Mirror Force*, we revisited the same companies to demonstrate and evaluate the proposed solution. This feedback session served as the final checkpoint of the research, allowing us the chance to corroborate the findings from the interview and test the prototype against those predefined requirements.

The evaluation was done through a usability test followed by a group interview. The analysis of this step served as a guideline for where the attention should be turned to on future iterations in the development of a new version of the prototype, and to show the relevance of the Design Heuristics.

The main contribution of our research is the development of a prototype that positions itself as a tool that facilitates human interaction. Thereby, insights are given to where the attention should be turned to for further development of dashboards that can facilitate the Scrum process. Moreover, we provide a list of requirements and constraints of Agile project management tools. This list can be transferred to other studies that look into the relationship between Scrum teams and their tools or it can be expanded to the entire Agile context.

A practical contribution is made by the ratification of relevant design principles. These principles are related to artifacts that should be included in dashboards, which can aid software designers

in the development of their solution. Such artifacts include, but are not limited to, the ability to configure the dashboard and the inclusion of appropriate information to increase usability.

1.5. Thesis Structure

This thesis is divided into six chapters, each one focusing on one specific aspect of the research. It follows a traditional logico-deductive project report structure, starting with the introduction, then going from theory to method, analysis, discussion, and finally, reaching a conclusion. Moreover, in order to project the answers of each of the research question clauses, the data collection, analysis, and discussion have been grouped into two chapters. Each chapter represents the path taken for solving a specific clause of the research question.

The *introduction* provides the background, overall motivation, and research question for this investigation and its report. It sets the scope of the work, briefly describing its dimensions and limitations, defines the research questions which were aimed to be solved by the end of the research process, and finally details how the report is structured, easing the process of the readers on understanding the scope of the project, while still leaving some suspense to the reader.

In this research, Information Visualization represents the core of the *theoretical foundation* and Agile project management the setting. The second chapter is devoted to guiding readers on how the existing literature on those topics supported our research process. Whenever the theory becomes pertinent to the investigation, a link is made at once, in order to assure that those arguments are relevant to the overall construction of this work.

The research *methodology* is discussed in chapter 3. There, every step taken in order to design this investigation is accounted for, from considerations on how our assumptions may affect our process and how they relate to our philosophy, to how we developed a strategy which would altogether create a dependable, credible, transferable, and fair research. Although the description of the research methodology followed a more traditional social science approach to the methodology, we also accounted for the Design Science Research Methodology, as it provided an empirical framework towards answering our research question.

Chapters 4 and 5, “How Are Scrum Teams Affected by Their Tools?” and “How Can Dashboards Aid Scrum Teams?”, serve as checkpoints in our research. The results and discussion in

chapter 4 feed new theory to chapter 5. Both chapters have a similar structure. They start by describing what the data collection plan was, following the guidelines from a methodology that was already in place. We document every step taken in order to gather and analyze data, comparing the original plan with what really occurred. Later, the findings are presented. Lastly, we close these chapters by providing our own reflections on the results, being critical on what went right, what went wrong, and what was learned.

In *conclusion*, we briefly review the work done and draw upon the ultimate goal of this thesis: answering the proposed research question. These reflections allow us to share our own understanding of the case companies investigated, the designed solution, and which directions the research could go in future iterations.

2. Theoretical Foundation

The theoretical foundation presented in this chapter is developed to form a groundwork that can help to answer the research question: *“How are Scrum teams affected by their tools and how can dashboards aid those teams?”.* The theory of Information Visualization sets the groundwork. Here, it will be explained how the combination of visual elements and computational power can help in understanding complex information. The Scrum framework forms the context and relevant elements, such as the roles and the ceremonies will be explained.

2.1. Information Visualization

Visualization has been embedded into human communication for a long time. It has helped humans to understand its own history by caveman drawings and has supported in exploring the world through mapping (Brown, 1979; Tversky, 2001). These examples already demonstrate the value of visualization. However, through the expansion of computational power and development of computer techniques, new, dynamic, and interactive visualizations became common (Hegarty, 2011). These visualizations can have a positive impact on human cognition if performed correctly. This amplification of cognition through a computer-supported visual data representation is the foundation of the theory of Information Visualization (Card, Mackinlay, & Schneiderman, 1999). It includes complex and large quantities of information to be represented, transformed and interacted with. By these means, it turns complex data streams into an understandable representation, which allows its users to extract information from patterns, structures, and trends (Yi, Kang, Stasko, & Jacko, 2008).

2.1.1. The Visualization Process

Information Visualization is built upon the perceptual system (Robertson, Card, & Mackinlay, 1993; Tversky, 2001; Ware, 2004). The human perception system consists of several subsystems, including the visual, olfactory, and auditory system. Through our visual system, we obtain more information than through all other systems together (Ware, 2004), making it the most powerful perceptual subsystem. Ware (2004) explains the visualization process through four stages. First, there is the collection and storage of the data, which is the most time-consuming stage. Collection and storage can include all sort of data, including quantitative data, such as monthly sales, or categorical data, such as the names of sold products. This stage

is followed by the transforming-stage, which turns data into something we understand, this can be a comparison, a graph, or any other understandable representation. Next, a hardware is needed that can display the graphical algorithm. Once the data is represented on hardware, the human cognitive system perceives visual information such as color and shape. This is when the information enters the processing memory system, which, when simplified, has three segments: the sensory, short-term, and long-term memory (Atkinson & Shiffrin, 1968). To remember, information should be efficiently processed from the sensory to the long-term memory (Huang, Eades, & Hong, 2006). Once the user stored the representation of the external display, an internal construct is made by giving meaning to the display representations, which allows using the information (Hegarty, 2011).

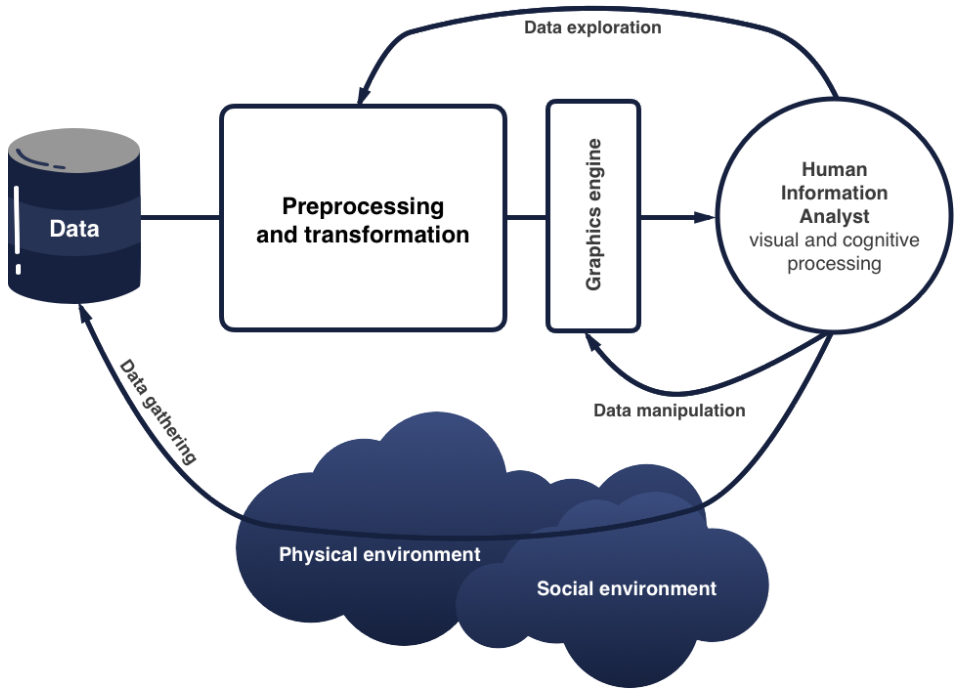


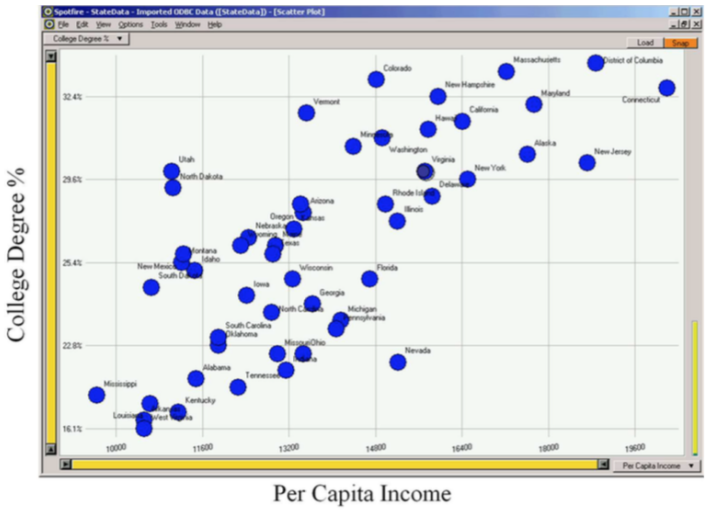
Figure 1. A schematic diagram of the visualization process. Adapted from Information Visualization: Perception for Design (p. 4) by Ware, 2004.

2.1.2. The Value of a Picture

“A picture is worth a thousand words” is a well-known saying. Fekete, Van Wijk, Stasko and North (2008) explain the reasoning behind this sentiment, by giving an example. Figure 2 (a), which represents a thousand words, shows the names of 50 states in the U.S. in the left column, the correlating percentage of citizens with a college degree in the middle, and the per capita

income on the right. When asking which state that has the highest incomes, Figure 2 (a) can provide the answer by the means of a quick scan. However, when asking if there is a correlation between the college degree and the per capita income, it becomes difficult to answer using the spreadsheet. Many data points need to be remembered and compared, which makes answering this question a complex process. Answering this questions using Figure 2 (b), a visual representation of the same data in the form of a scatterplot is easier. This visual representation demonstrates there is a correlation between the two variables. Thereby, the act of visual plotting serves as a way to improve communication of data.

State	College Degree %	Per Capita Income
Alabama	20.6%	11486
Alaska	30.3%	17610
Arizona	27.1%	13461
Arkansas	17.0%	10520
California	31.3%	16409
Colorado	33.9%	14821
Connecticut	33.8%	20189
Delewere	27.9%	15854
District of Columbia	36.4%	18881
Florida	24.9%	14698
Georgia	24.3%	13631
Hawaii	31.2%	15770
Idaho	25.2%	11457
Illinois	26.8%	15201
Indiana	20.9%	13149
Iowa	24.5%	12422
Kansas	26.5%	13300
Kentucky	17.7%	11153
Louisiana	19.4%	10635
Maine	25.7%	12957
Maryland	31.7%	17730
Massachusetts	34.5%	17224
Michigan	24.1%	14154
Minnesota	30.4%	14389
Mississippi	19.9%	9648
Missouri	22.3%	12989
Montana	25.4%	11213
Nebraska	26.0%	12452
Nevada	21.5%	15214
New Hampshire	32.4%	15959
New Jersey	30.1%	18714
New Mexico	25.5%	11246
New York	29.6%	16501
North Carolina	24.2%	12885
North Dakota	28.1%	11051
Ohio	22.3%	13461
Oklahoma	22.8%	11893
Oregon	27.5%	13418
Pennsylvania	23.2%	14068
Rhode Island	27.5%	14981
South Carolina	23.0%	11897
South Dakota	24.6%	10661
Tennessee	20.1%	12255
Texas	25.5%	12904
Utah	30.0%	11029
Vermont	31.5%	13527
Virginia	30.0%	15713
Washington	30.9%	14923
West Virginia	16.1%	10520
Wisconsin	24.9%	13276
Wyoming	25.7%	12311



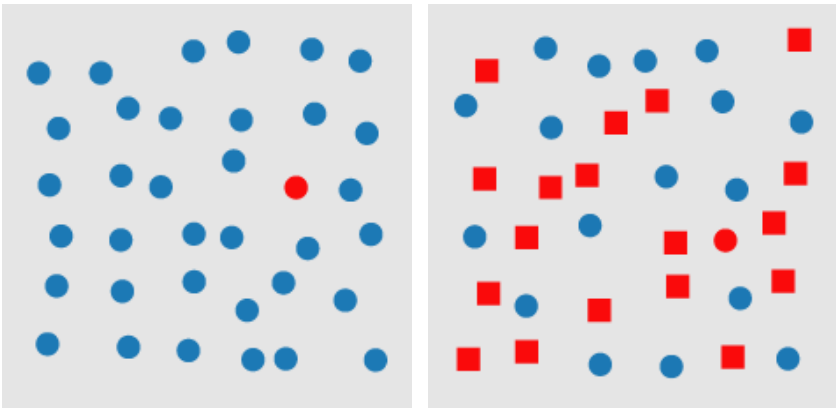
(a) (b)

Figure 2. Comparison between a thousand words and a picture. Reprinted from The Value of Information Visualization (p. 6) by Fekete et al., 2008.

Through this example, Fekete et al. (2008) show the benefits of visualizing information. Further understanding of the value of Information Visualization is given in the following two subsections, from a perceptual and cognitive perspective.

Perceptual Perspective on the Value of Information Visualization

“Visual displays provide the highest bandwidth channel from the computer to the human. We acquire more information through vision than through all of the other senses combined” (Ware, 2004, p. 2). This implies that vision is the most efficient manner to transfer knowledge to the brain and plays a crucial role in our perceptual system (Fekete et al., 2008). Ware (2004) describes two psychological theories that can explain the perception of visual elements. On a low level, the preattentive processing theory states that some visual elements can be detected quickly and accurately by our low-visual system (Perera, Goodman, & Blashki, 2007). Preattentive processing can be done based on different visual features such as color and line length. A classic example is displayed in Figure 3 (a), containing many blue dots and a single red dot. Here, the recognition of the red dot is easily made due to the color difference. However, our preattentive processing has strict boundaries, which is explained by a related example. When the same field of dots contains a large quantity of both red and blue dots as in Figure 3 (b), the task could not be solved with preattentive processing but would need a longer process (Fekete et al., 2008).



(a) (b)

Figure 3. An example of preattentive processing. Reprinted from Attention and Visual Memory in Visualization and Computer Graphics (p. 1171) by Healey and Enns, 2008.

Where our preattentive processing is based on recognition, Gestalt principles are based on the understanding of an image. The Gestalt principles originate from the Gestalt movement, initiated by Wertheimer, Koffka, and Kohler in 1912 and influenced German psychology (Wertheimer, 1923). The principles, described in Table 1, focus on how people perceive visually grouped items and small items within the boundaries of bigger items. Both types tend to show potential

associations (Kohler, 1967). The objective of this movement is to describe principles on how our brain works when interpreting an image and thereby, impact our perception (Koffka, 1935 in Fekete, et al., 2008).

Gestalt principle	Meaning
Similarity	Similar elements tend to be grouped together
Proximity	Things that are close together are perceptually grouped together
Continuity	Visual elements that are smoothly connected or continuous tend to be grouped
Symmetry	Two symmetrically arranged visual elements are more likely to be perceived as a whole
Closure	A closed contour tends to be seen as an object
Relative Size	Smaller components of a pattern tend to be perceived as objects whereas large ones as a background.

Table 1. The Gestalt principles.

Cognitive Perspective on the Value of Information Visualization

Ware (2004) states that Information Visualization if presented well, can support human cognition, thereby, providing a number of benefits to its users. The first way, in which users can profit from Information Visualization is, as explained by Card et al. (1999), done by increasing memory and general processing resources. In this case, the external display itself can store a huge amount of information, which can be adopted by the visual system as a resource (Hegarty, 2011). Additionally, data in Information Visualization can be clustered, which prompts two advantages. First, it allows quick detection of trends and patterns (Robertson, Card, & Mackinlay, 1991). Figure 2 (b) is an example in which a trend between two variables can be discovered instantaneous. Second, the effort needed to search for data can be reduced (Larkin & Simon, 1987), as large volumes of data can be shown at a glance and in a small space. When required, the user can zoom in to understand the information on a detailed level.

2.1.3. Classification of Visual Display

The benefits described in the previous subsection can be realized through different forms of visualizing. Hegarty (2011) makes a distinction between three different categories of visual displays: iconic, relational and complex displays. Iconic displays are representations of objects that themselves are visual-spatial entities. A roadmap on your navigation system is an example of such a display since it provides a visual representation of the road as an object.

Relational displays show a relation between quantitative data sets or categorical and quantitative data. In these displays, visual and spatial properties represent entities and properties that are not necessarily visible or distributed over space. Variables, such as color, shape, and location are the representing dimensions of the display (Few, 2004). An example of a relational display is shown in Figure 2 (b), which shows the relation between two quantitative variables.

Complex displays became reality by developments in information technologies. These have led to new opportunities and challenges of how to visualize large and complex data sets, with researchers in Information Visualization focusing more on visualization of abstract information spaces. This category is present when a display provides information that can be zoomed and filtered in an interactive manner. For example, a digital map that allows students to find the room for their next class.

2.1.4. Examples in Software Development

Information Visualization and the related display categories are widely used and their value can be best explained by real examples. Examples of Information Visualization in software development include code changes, project tracking, and dashboards (Paredes et al., 2011; Ware, 2004). The knowledge in this field emanates from academic research and professional practice. In this subsection, we provide examples from both fields.

Information Visualization in Research

The research conducted by Biehl et al. (2007) in the area of Information Visualization focuses on the creation of an interactive dashboard, named *FASTDash*. The objective of this tool is to enhance ongoing activity awareness for small development teams. Information Visualization is

applied to increase team activity awareness by highlighting real-time information of team members' current activities in the common code base (Figure 4). With *FASTDash*, a developer is able to see who has which file open, or which files are currently being worked on (Biehl et al., 2007).

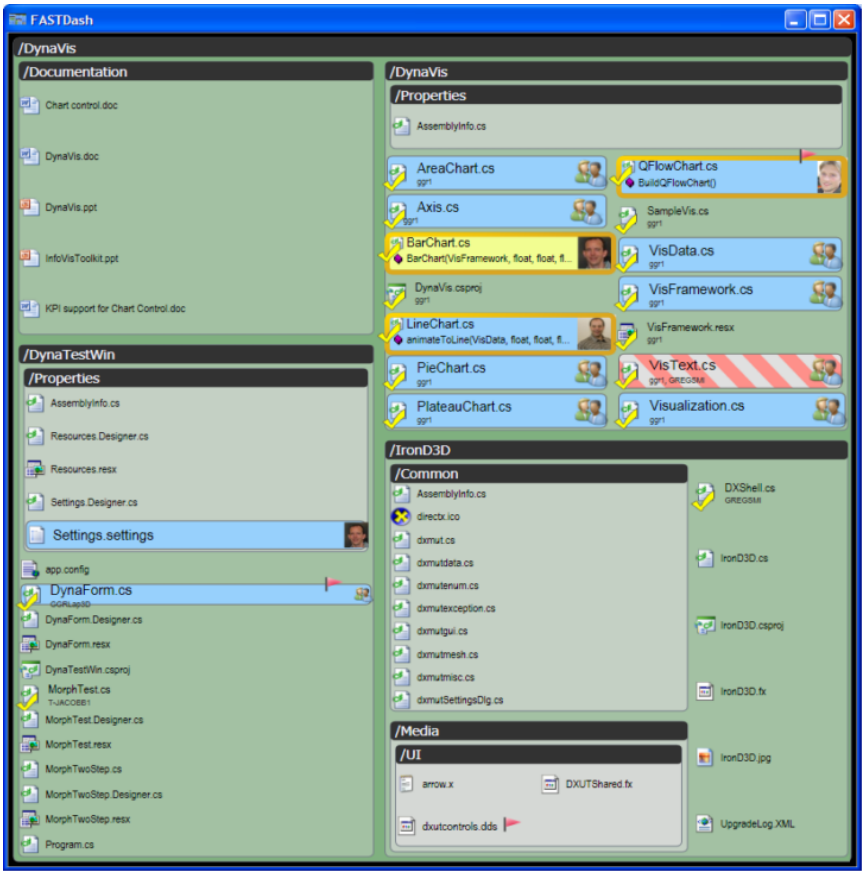


Figure 4. *FASTDash* view showing three programmers working in a shared code base. Reprinted from FASTDash: A Visual Dashboard for Fostering Awareness in Software Teams (p. 1313) by Biehl et al, 2007.

A comparable research in the field of Information Visualization comes from Jakobsen et al. (2009). They argue that *FASTDash* is limited as its focus is merely on increasing ongoing activity awareness. In their own research, they build on top of *FASTDash* and create *WIPDash*, which combines code-based level awareness with the overall status of the project. These objectives are represented in a large screen visualization for co-located teams (Treude & Storey, 2010). As can be seen in Figure 5, Information Visualization is utilized in a large screen containing two visual elements. On the left side, a spatial depiction of project fields is shown,

including detailed stories. The right side represents several modes that can show detailed information about work items (Jakobsen et al., 2009).

A more recent study on Information Visualization in Agile software development using dashboards is called aWall, a large multi-touch wall system designed to support and foster collaboration and communication in development teams (Mateescu et al., 2015).

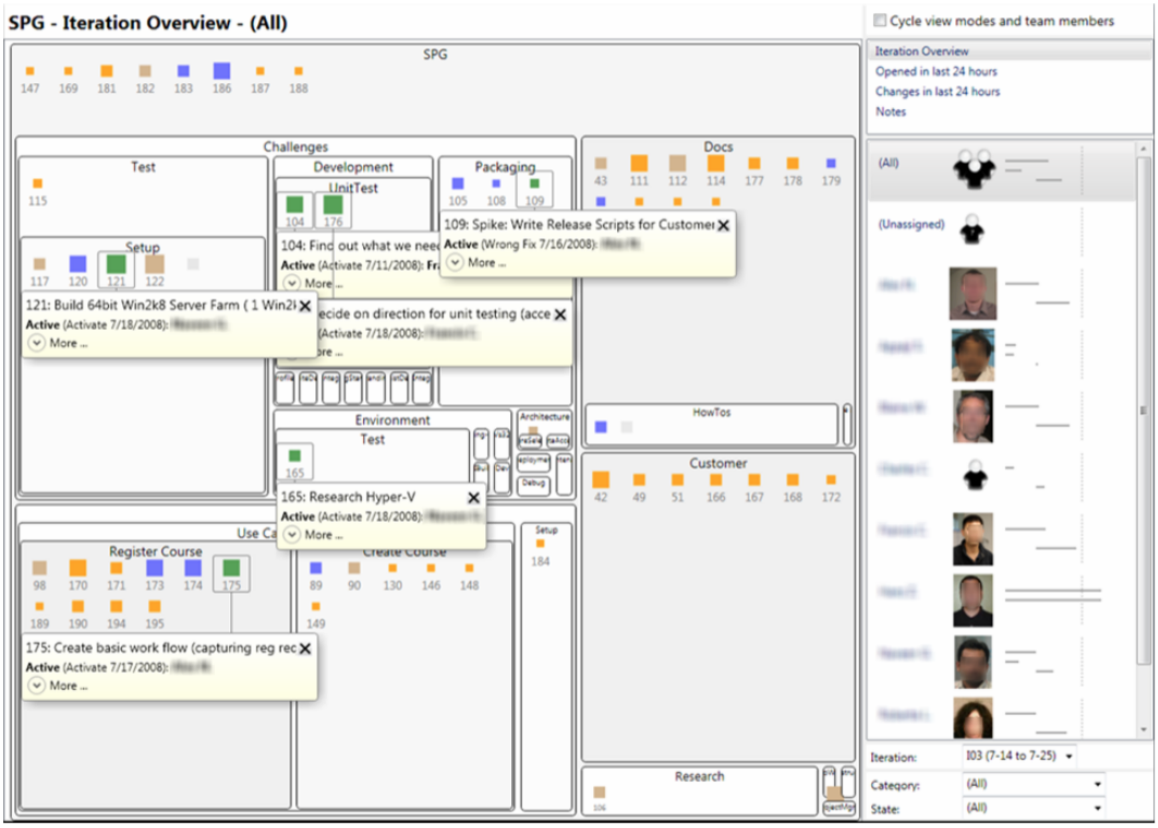


Figure 5. *WIPDash* showing the iteration view for the current iteration of a project. Reprinted from WIPDash: Work Item and People Dashboard for Software Development Teams (p. 796), by Jakobsen et al., 2009.

Information Visualization in Practice

From professional practice, many examples can be given of tools that use Information Visualization. One example of Information Visualization in the professional field is a project management tool. A quick Google search on *"IT project management tools"* gives over 10 million results. *ScrumWise* is one of them, focussing on increased communication and better teamwork for Agile software development teams. Opposed to the tools described in the academic literature, professional tools encompass many different features as it is aimed to

accommodate complex processes. Despite not being researched, these tools and their features, serve a practical relevance. Hence, their pertinence is not based upon research, but by their business value.

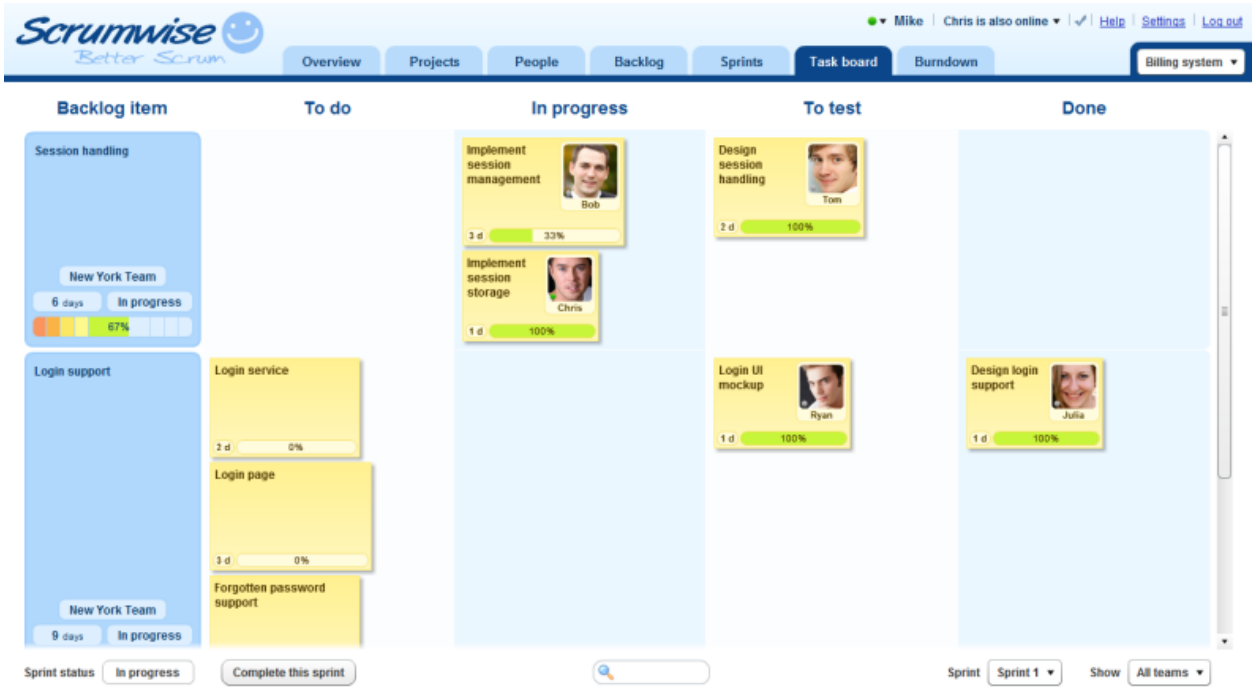


Figure 6. Taskboard from *ScrumWise*.

2.1.5. Dashboards

These three examples take advantage of Information Visualization, each in their own way. The focus of this study is to utilize Information Visualization by creating a dashboard. The definition of a dashboard provided by Few (2006), is based on a shared set of characteristics. According to the author, a dashboard is a visual display of the most important information needed to achieve the ambition, which is concentrated in one screen allowing information to be screened in a single view. Yigitbasioglu and Velcu (2012) build on this, as they add that a dashboard should allow the user to identify, explore, and communicate focus points that need corrective action. Its primary task is to visualize big data sets in a concise manner to identify patterns, and rarities (Lempinen, 2013). Therefore, a dashboard is expected to improve decision making at a glance (Yigitbasioglu & Velcu, 2012).

The principle of dashboards comes from the car industry. Here, a dashboard is used to inform the driver about essential information such as speed, fuel level, and revolutions per minute (Yigitbasioglu & Velcu 2012). In an organizational context, dashboards visualize key indicators at a glance that allow monitoring and navigation of in-depth information (Lempinen, 2013; Treude & Storey, 2010). The objective of a dashboard is to provide a visual representation of a large amount of information in a concise manner, to recognize patterns and trends. The effectiveness of a dashboard, meaning how well a dashboard serves to identify, explore, and communicate focus points depends on how it is represented.

The representation of a dashboard can be analyzed from two perspectives, the functional and the visual perspective (Yigitbasioglu & Velcu, 2012). Functional features represent *what* a dashboard is supposed to do and should always be in line with the objective of the dashboard to ensure correct decisions. Visual design, on the other hand, is concerned with *how* information can be presented efficiently (Yigitbasioglu & Velcu, 2012). Both will be explained in the following subsections.

Functional Dashboard Design

As mentioned, a functional fit should ensure that the elements in a dashboard are in line with its objectives (Yigitbasioglu and Velcu, 2012). Therefore, a suitable functional dashboard design depends on its context. The context of our research is Scrum development teams. For this reason, the framework presented by Storey, Čubranić, and German (2005) can be useful as a starting point to include the right functional elements. This framework is used for describing, analyzing, and comparing visualization of human activities in software development. Hence, the framework, as found in Table 2, can act amongst others, as a formative evaluation mechanism to guide the design of our proposed solution.

Dimensions	Characteristics	Features
Intent	Role	Team, Developer, Maintainer
	Time	Present, Recent Past, Historical
	Cognitive support	Authorship, Rationale, Time, Artifacts
Information	Change management	Releases, Branching
	Program code	Syntactic Units (e.g., files), Semantic analysis

	Defect tracking	Defects, Changes
	Documentation	Design, Architecture
	Informal communication	Email, Instant messages
	Derived	Single source, multiple sources
Presentation	Form	Text, hypertext, graphical
	Kinds of views	Statistical views, graph views, special views
	Techniques	Visual variables (hue, transparency, position), animation
Interaction	Batch/Live	Offline, online
	Customization	Level of customization
	Queries	Query language, filter widgets
	View navigation	Overview plus detail, zoomable views
Effectiveness	Status	Availability, scalability, interoperability
	Cost	Economic cost, installation, learning, usage
	Evaluation	Adopted, case study, user study

Table 2. A framework for understanding visualization tools for human activities in software development. Adapted from On the use of visualization to support awareness of human activities in software development (p. 196) by Storey et al., 2005.

Even if a functional fit exists, a pitiful visual design can confuse the user. The following subsection will reflect upon the principles, also called heuristics, that should be considered when visualizing information.

Visual Dashboard Design

When it comes down to visual design, a single form of visual representation that is best in all situations does not exist (Hegarty, 2011). There are, however, rules on how to create a good design for a specific situation. These rules are called design heuristics. Design heuristics are used to create and evaluate the design of an information system, based on a set of principles. In this way, heuristics form a foundation for shared knowledge on design (Zuk, Schlesier, Neumann, & Carpendale, 2006). Moreover, they help to communicate new ideas by creating a shared language between its users (Gamma, Helma, Johnson, & Vlissides, 1994). Using

heuristics is a relatively fast and cheap process. For this reason, they are used as guidelines to evaluate existing and create new concepts (Zuk et al., 2006). The design principles described in here serve as guidance for the creation of our prototype, which is described in chapter 5.

It is recognized that the evaluation of the design principles is challenging but essential (Forsell & Johansson, 2010). The existing literature on heuristics in Information Visualization focuses on its representation. Forsell and Johansson (2010) describe heuristics that are useful for assessing complex interactive visual displays of Information Visualization. Based on an extensive literature review, Yigitbasioglu and Velcu (2012) summarize the most important findings that are relevant to the design of a dashboard. Hegarty (2011) describes the main principles for display design, inspired by the literature on cognitive science. To ensure that all the important heuristics are applied in the creation of our dashboard, the subsequent part will present an extensive list, which is inspired by the three aforementioned articles. The principles listed by Hegarty (2011) are leading, however, they are supplemented with heuristics related to dashboard design. Overall, the elements included in this list are based on both general and dashboard design principles.

Group	Principle	Description	Source
Task specificity	No Single Best Display Principle	There is no best display independent of the objective	Hegarty, 2011
	Proximity Compatibility Principle	Integrated activity need high proximity displays, focused activities need low proximity displays	Wickens & Carswell, 1995
Display Expressiveness	Relevance Principle	Only appropriate data should be displayed	Forsell & Johansson, 2010
	The Principle of Capacity Limitations	There is a maximum of information that can be shown	Iselin, 1988
	The Principle of Recognition	The user should rather recognize than remember displayed information	Forsell & Johansson, 2010
	Data-ink Ratio	Have as much ink available to data for emphasis on the most relevant information	Tufte, 2001
Display	The Principle of	Use the Gestalt Principles for	Tversky et al,

Perception	Perceptual Organization	correctly perceiving data	2002
	Apprehension Principle	Use visuals that correctly represent the activity performed	Kosslyn, 2006
	The Principle of Discriminability	The visual differences between variables should be substantial to be able to distinguish them	Kosslyn, 2006
Display Semantics	The Principle of Compatibility	Natural mappings should be used between a display and its meaning, to ensure understanding	Kosslyn, 2006
	The Principle of Dimension Representation	Match the dimensions of represented variables in terms of scale	Bertin, 1983
Display Pragmatic	The Principle of Salience	Design the display with a focus on the most important information	Dent, 1999
	The Principle of Informative Changes	Be consistent with multiple displays	Kosslyn, 2006
Display Usability	The Principle of Appropriate Knowledge	The principle of sufficient information to help the user	Kosslyn, 2006
	The Principle of Visual Momentum	Provide visual aids to help users make connections between different displays	Wickens & Holland, 2000
	The Principle of Minimal Action	Minimize the number of actions necessary to accomplish a goal or a task	Forsell & Johansson, 2010
	The Principle of Preference	Being able to customize or chose the preferred display increases usability	Wilson & Zigus, 1999
	At a Glance Principle	A dashboard should fit into a single screen and should allow drop-down for additional information	Few, 2006

Table 3. Design heuristics: principles and their sources.

Task-Specific Principles

Hegarty (2011) states that there is no such thing as a ‘‘best’’ or ‘‘worst’’ display independently of the task to be performed. Visual displays are used for different purposes, such as information accumulation and exploring (Ware, 2004). A display that is adequate for one task, may be inadequate for another. Tabular information, for example, is more suitable for symbolic tasks

such as extracting specific values and make an overall judgment (Vessey, 1991). Graphs, on the other hand, allow better correlation estimates and decrease the time spent on tasks (Schulz & Booth, 1995).

The statement that there is not a single way of displaying that is best for all situations is supported by Wickens and Carswell (1995). They argue that integrated activities, which need multiple sources to be integrated with, are best facilitated by high proximity displays. On the other hand, focused activities, which only need one display to perform the task at hand, are best facilitated by low proximity displays.

Hegarty (2011) questions whether it is optimal to have an all information in a single display or provide modification options and present information across multiple displays. The answer, in line with the core principle that no single display is best for all situations, is that it depends on the meaning of the display. For example, when details of a complex system are needed to be represented, it might be valuable to use multiple displays that permit flexibility of visualization (Smith, Bennett, & Stone, 2006). On the other hand, when displaying high-level trends, a single display would be sufficient (Gillan, Wickens, Hollands, & Carswell, 1998). Regardless of this choice, when several displays visualize related information, the design should always show the connection between them to avoid confusion (Woods, 1984). This can be achieved, among others, by keeping elements consistent across displays.

Display Expressiveness Principles

Expressiveness relates to the content that is shown on the display. First, only relevant data should be presented. Showing too much information can be counterproductive, as the working memory can only handle a maximum of information, meaning that information overload can lead to confusion, conceal important information, and cause inconsistent decisions (Forsell & Johansson, 2010; Yigitbasioglu & Velcu, 2012). The core of this concept is explained by the U relationship between information load and decision accuracy, which states that there is a maximum of information to be processed by a person (Iselin, 1988).

Another implication of the limited working memory is that visual displays should facilitate the user to recognize the information rather than memorizing the provided data (Kosslyn, 2006; Forsell & Johansson, 2010). In this way, heavy burden on cognitive processing can be

prevented and used for other means. If a visual display is designed to recognize information, it can serve its purpose as a way to store information as described by Card et al. (1999).

Expressiveness is also reflected in the data-ink ratio mentioned by Tufte (2001), who states that as much ink as possible on a screen should be used for data to maximize the focus on the most relevant information (Yigitbasioglu and Velcu, 2012). All non-data ink, such as background pictures is excessive, therefore, should be avoided. This principle does not apply to all types of data to ink ratio, though. An exception mentioned by Gillan and Richman (1994) is increasing the thickness of the x- and y-axis.

Display Perception Principles

To establish that displays are perceived in an intended way, the features should be suitable for the tasks being performed (Vicente, 2002). The importance is confirmed by Cleveland and McGill (1984), who emphasize the importance of visual dimensions which can be accurately assessed. An example is provided by Tversky, Morrison, and Bertrancourt (2002) as they argue that animation is not more suitable than graphs to highlight main aspects.

Next to the choice of animation, Gestalt principles, such as similarity in color use or consistency in pattern use, affect the way a display is perceived (Goldstein, 2008; Forsell & Johanson, 2010). This is important as there is a natural human tendency to group elements presented on a display (Hegarty, 2011). Too many lines or inconsistent patterns can confuse the user. In relation to colors (Kosslyn, 2006) argues that the visual difference between different variables should be substantial to be perceived as different.

Display Semantics Principles

A visual display is easier to understand if the visualization is in line with its objective. Perception of the display is directly depending on the mapping of data elements. Forsell and Johanson (2010) suggest that the mapping of data elements should be reinforced by realistic techniques and symbols. Based on the literature, this done in two ways. First, by choosing a natural mapping between graphic forms and its meaning. Second, by relating dimensions of the representation with the scales and measurements it is supposed to represent (Bertin, 1983; Mackinlay, 1986; Zhang, 1996).

Display Pragmatics Principles

Pragmatics in design deal with the context in which visual displays acts. A core principle in this knowledge field is salience, making the most important information pertinent (Dent, 1999). In their study on weather displays, Hegarty, Canham, and Fabrikant (2010) find support for the effects of salience, also called bottom-up, displays.

Similarly, Kosslyn (2006) argues that users expect changes between displays to carry information, meaning that large changes across properties of a display that do not carry information should be avoided.

Display Usability Principles

Usability is the degree to which a user can effectively and efficiently complete a task using a tool. To adopt a software application, one should understand the specifics (Mandel, 1997). One way to assure understanding of the tool is by offering help, either beforehand or when needed. This offering can be done by displaying sufficient knowledge to understand the information that is displayed. Examples are showing legends in charts, or providing a step-by-step explanation on how to understand a widget (Kosslyn, 2006). Another way to offer aid is by the ability to undo actions, and by showing grid lines to the user (Yigitbasioglu & Velcu, 2012; Forsell & Johansson, 2010; Amer & Ravindran, 2010). Finally, users can be given a referential connection between different displays. An example is highlighting related anchors between different displays, by which confusion can be prevented (Forsell & Johansson, 2010; Wickens & Hollands, 2000).

Minimal actions concern the workload with respect to the number of actions necessary to accomplish a goal or a task (Forsell & Johansson, 2010). This is especially true for dashboards since they are expected to aid decision making at glance. Flexibility can also increase usability. Flexibility is reflected in the number of possible ways of achieving a given goal. It refers to the means available to customization in order to take into account working strategies, habits and task requirements (Forsell & Johansson, 2010).

Finally, a dashboard should fit on a single screen but allow data to be drilled down (Few, 2006). This idea is supported by Shneiderman's (2003) Visual Information Seeking Mantra, which states that visual guidelines should support an overview first, then zoom and filter, and finally details on demand. Providing the opportunity for additional information is essential, and

increases the usability of a dashboard as patterns can be identified at first, while additional information can be shown if the user needs this.

2.2. Awareness in Software Development

Awareness is a relative concept. It is the internal human state of consciousness including feelings and perceptions about one's surroundings (Scheier, 1976). The roots of the concept can be found in psychology and neurology. Over time it became an important concept in the field of computer-supported cooperative work (CSCW), as it was realized that maintaining awareness in a workspace is difficult, though essential for effective coordination of software projects (Gutwin & Greenberg, 2002).

Storey et al. (2005, p. 193) describe awareness as "knowing who else is working on the project, what they are doing, which artifacts they are or were manipulating, and how their work may impact other work". In software development projects, awareness consists of being aware of technical and social elements of development, together with present and future articulation work. A more concise definition is provided by Dourish and Bellotti (1992), as they describe it as an understanding of the activities of others, which provides a context for one's own activity.

This context is used to ensure that individual contributions serve the groups' purpose as a whole, and to assess individual actions in light of these group objectives. In turn, this knowledge can be used for coordination of tasks and thereby collaboration among team members (Dourish & Bellotti, 1992). The relevance of awareness for collaboration is supported Storey et al. (2005). They explain the importance of awareness in software development as follows: software development is a human-intensive and human-driven activity, and the realization of a software project of a reasonable size, includes multiple team members (Froehlich & Dourish, 2004). A key issue in collaboration between those team members is knowing what is going on (Endsley, 1995), which is awareness.

There is a great variety of awareness described in the information system literature. Examples include group, situational, and emotional awareness (Endsley, 1995; Jakobsen, et al., 2009; Guzman & Bruegge, 2013). Sometimes it can be difficult to understand the differences between these several forms of awareness. Treude and Storey (2010) try to simplify the concept of

awareness in software development, by classifying the concept into two main categories: high-level and low-level awareness (Treude & Storey, 2010).

Treude and Story (2010) argue that high-level awareness in software development is more related to project management issues. An example of high-level awareness is group awareness. Gutwin et al. (2004) describe group awareness as who is working on what, which module are they currently working on, what are they doing, and what are their following plans.

Low-level awareness, on the other hand, is more related to detailed tasks such as changes in code and incoming new tasks (Treude & Storey, 2010). An example of low-level awareness is called change awareness, which is described as the ability to observe asynchronous changes in a collaborative record over time (Tam & Greenberg, 2006). In software development, a code repository can be such a record. Seeing the change increases awareness, as it gives the user an impression of the activity in the source code (Dix, 1994).

2.2.1. Gaining Awareness

The main takeaway from the previous subsection is that awareness is essential for successful communication and coordination (Treude & Storey, 2010). Likewise, good communication and coordination are essential for successful software development (Chow & Cao, 2008; Krauter, 1995). However, software development becomes increasingly complex, giving each project a unique history (Froehlich & Dourish, 2004). Therefore, staying aware of important artifacts, such as project status and bottlenecks, becomes more difficult (Treude & Storey, 2010).

One way to gain awareness, mentioned in the literature, is using an Agile software development methodology. Agile software development teams sit down together frequently, to inspect the project and understand the constant stream of complex data, such as the change of requirements. Examples of such gatherings are ceremonies that allow for planning and revising of the software to be delivered. These events throughout the process allow for reflection, making problems visible and taking learnings from the past (Lehtonen, Eloranta, Leppanen, & Isohanni, 2013). Thereby, using an Agile methodology increases awareness, as those teams are working in short iterative development cycles and team-members meet each other on a regular basis (Schwaber & Beelde, 2002).

In the following subsection, we will give a comprehensive explanation of the Agile philosophy, hereafter, we will focus on a specific Agile methodology named Scrum which forms the context of our research.

2.2.2. Scrum

In software development, the term Agile is used as an umbrella-term to describe incremental development methodologies (Beck et al., 2001), such as Scrum (Sutherland & Schwaber, 2007). The Agile Manifesto was established in 2001 by a group of 17 software practitioners with the objective to provide an alternative for the heavy documented software development methodologies (Beck et al., 2001). Proponents of Agile methods assume there is a high degree of unpredictability during a project (Nerur, Mahapatra, & Mangalaraj, 2005). Abrahamsson, Salo, Ronkainen, & Warsta (2002) state that this unpredictability is partly caused by the business community since they demand fast and nimble software development for continuous changing requirement.

To serve this uncertainty the best way possible, the Agile Manifesto outlined twelve principles, available in Appendix A, for an iterative and people-centered approach for software development (Beck et al., 2001). These principles form the basis for many software development methods such as Extreme Programming and Scrum (Coram & Bohner, 2005). Even though these methods differ, they share a set of key values, as each method has a focus on individuals and interactions, working software, customer collaboration and responding to change (Cockburn, 2002; Dingsøyr, Nerur, Balijepally, & Moe, 2012).

Many of the elements written down in the manifesto were not new. Brian Randell from IBM, for example, recommended iterative development already in 1968 (Larman & Basili, 2003) and Scrum, an Agile framework, was published 15 years before the Manifesto (Takeuchi & Nonaka, 1986). However, the novelty of the method can be explained in two ways: by the strong focus on effectiveness and maneuverability, and by setting people as key actors for the success (Highsmith & Cockburn, 2001).

Scrum is an Agile methodology and was introduced by Takeuchi and Nonaka (1986). It was named after the scrumdown formation used in rugby, in which forwards position in a tight formation and specific positions and aim to get an ‘out-of-play-ball’ back into the game through teamwork (Schwaber, 1987). The overall idea of Scrum is to develop information systems that

are subject to the influence of external factors, which can change throughout the process of development. Therefore, the objective is to deliver a system that can be used on the day of delivery (Schwaber, 1987). In the Scrum Guide, written by Schwaber and Sutherland (2017), it is emphasized that the process framework is built upon three pillars: transparency, inspection, and adaptation. A definition of the three pillars will be provided and followed by an explanation of the most significant elements of the Scrum process framework: the process, the team, artifacts, and events.

Transparency means that essential elements of the process must be visible to those accountable for the result.

Inspection relates to the fact that Scrum users must frequently examine if they progress towards the goal. There should be a balance between of inspection frequency, it should not get in the way of achieving its goal.

Adaption is needed when it is determined that one or multiple aspects go outside of the acceptable limits. This also means that if needed adjustments to the process can be made.

Scrum Process

As described by Abrahamsson et al. (2002), the Scrum process has three main phases: the pre-game, development and post-game phase (see Figure 7). The pre-game phase consists of two sub-phases: *Planning Phase and Architectural Design Phase*. During the Planning Phase, it is decided how the system to be built is going to look like, who will be part of the team, which resources are available and if there are any other needs, such as a training. In the Architectural Design Phase, a high-level design is made in which it is decided how items from the backlog are going to be implemented.

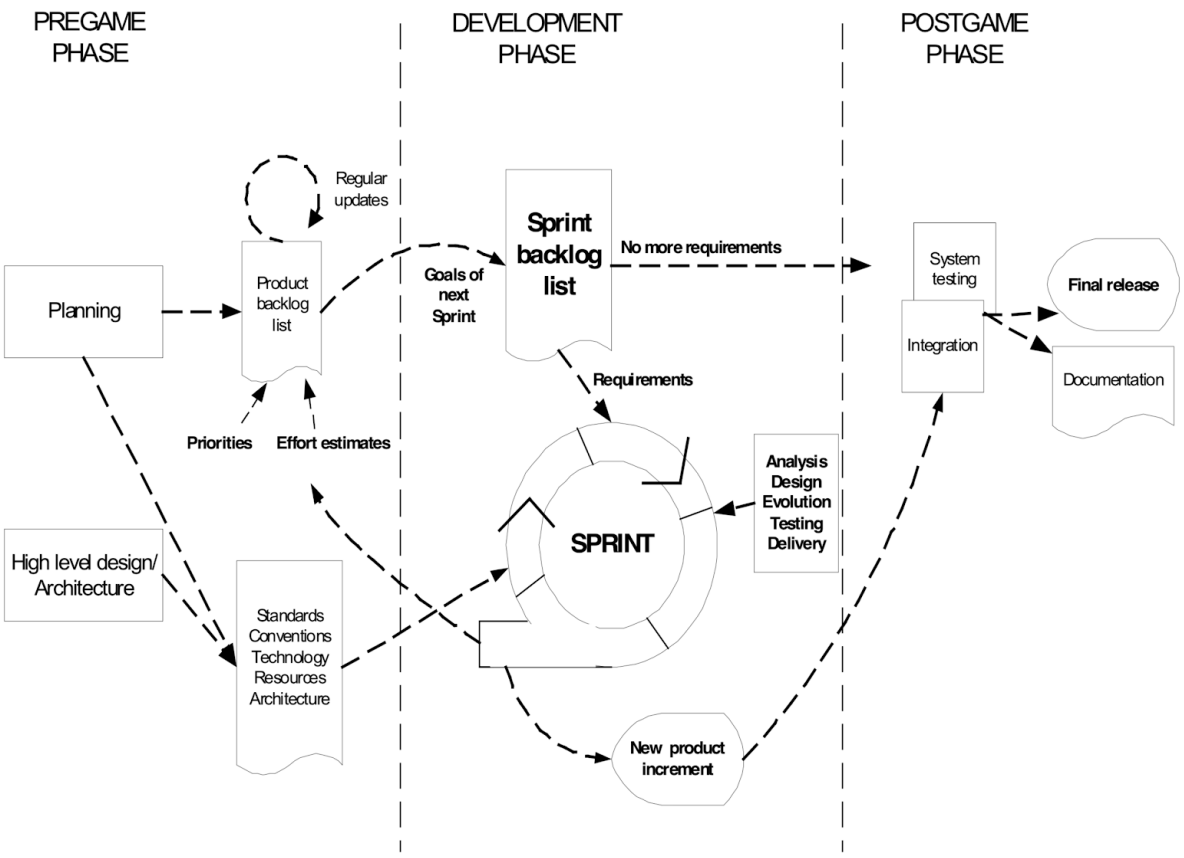


Figure 7. Scrum process. Reprinted from Agile software development methods: Review and analysis (p. 28) by Abrahamsson et al., 2002.

The start of the development phase is built upon on the Sprint Backlog. This phase is characterized by a high degree of uncertainty, in which the Agile ideology is strongly recognized. The ambiguity is caused by technical and environmental factors such as time, dependabilities, and external requirement, which all influence the development process (Abrahamsson et al, 2002).

The last phase, the post-game phase, is when it is agreed that the requirements for the systems are completed, and thereby the project is closed. Once entered into this phase, there is no requirement left to build and no new requirements can come in. In this phase, the product should be made ready for release, which includes activities such as integration, testing of the system, user training and documentation (Schwaber, 1987)

Scrum Team

There are three main roles within the Scrum team: Product Owner, Development Team, and a Scrum Master (Sutherland & Schwaber, 2007). The teams are small and self-organized and together they are responsible for delivering the product (Rising & Janoff, 2000). Due to this self-organizing nature, a Scrum team has a substantial amount of responsibility. Therefore, planning, decision making, and assigning new tasks to team members is often done by the team itself (Schwaber & Beedle, 2001).

The **Product Owner** has a conception of how the final product should look like and is responsible for maximizing the value of the product. To get this conception, the Product Owner assembles new ideas from the stakeholders and decides the priority (Abrahamsson et al., 2002). Depending on the nature of the organization and the type of product, the Product Owner can be either someone from within the company or a customer.

The **Development Team** include the ones who actually build the product. Typically, the Development Team consists of designers, developers, testers, and researchers. It is desired to have all required expertise in the team to increase speed and short communication lines. Although each team member has its expertise, it is desired that members of the Development Team can take multiple roles. Next, by building the actual product, team members have an important role in delivering input to the Product Owner.

The **Scrum Master** is the gatekeeper of the team, which means he or she should not be there to state what the team should do, but merely serve the team. This includes protecting Development Team from work overload and taking difficulties out of the way. The Scrum Master can be a member of the Development Team. The Scrum Master cannot be the Product Owner, as they have conflicting interests.

Scrum Artifacts

Product Backlog. In Scrum, the Product Backlog is an organized list of all that is needed for the product. The requirement can come from stakeholders such as internal departments, direct clients or team members. The Product Backlog can be updated at all times with new items or requirements. The Product Owner has the responsibility to maintain the Product Backlog, which means ensuring content, sorting its availability.

Sprint Backlog. From the Product Backlog, items are selected to be part of the Sprint, which results in a Sprint Backlog. The Sprint Backlog is the way to reach the Sprint Goal and by these means an estimate about the work to be done. All the requirements in the Sprint Backlog have a precise estimate in terms of effort. These requirements are ordered chronologically, meaning that the top features should be done first. The Sprint Backlog is often used as a communication tool during the Daily Standup, to clarify what people are working on (Schwaber & Sutherland, 2017). A story is a tactic for items that are included in the Backlog.

Scrum Events

Sprint Planning. The Sprint Planning is the first event of a Sprint with the objective to clarify the work to be done in the next Sprint. It is the responsibility of the Scrum master to make sure everyone is present, to communicate the objective of the planning and to make sure the planning is done within the time-limit (Schwaber & Sutherland, 2017). Two questions are central in the Sprint Planning:

1. What is going to be delivered in the next Sprint?
2. How are we going to ensure that the work, needed for this Sprint, will be delivered?

Daily Scrum. As the name states, the Daily Scrum is a daily meeting for the entire team, which lasts for about 15 minutes. There are two central themes in the Daily Scrum: what work is going to be done the next day, and what work is done the previous day. The Development Team is responsible for answering these questions, and by these means, it communicates if the objectives of the Sprint can be achieved (Sutherland, 2007). The objectives of the Daily Scrum is to have direct communication, avoid unnecessary meetings, increase the overall knowledge of the team and promote fast decision making (Schwaber & Sutherland, 2017).

Sprint review. A Sprint Review is held at the end of a Sprint and has a strong focus on the product to be built. During this meeting, it is analyzed which steps are made towards the goals. During this informal session, it is also discussed how the Backlog should be adjusted to maximize the value of the product and the goal for the next Sprint is defined (Schwaber & Sutherland, 2017; Sutherland, 2007).

Sprint Retrospective. While the focus of a Sprint Review is on the product, the focus of the Sprint Retrospective is on the team. During the retrospective, the general theme is how the

team can improve for the coming Sprint. The ceremony is between the Sprint Review and the following Sprint Planning and has the objective to understand the Sprint in terms of people, relationships, tools and the overall process. Next, it is analyzed from a human perspective which elements went well, which have room for improvement, and how these improvements can be implemented in the next Sprint (Schwaber & Sutherland, 2017).

Scrum Tactics

Burndown Chart. The name of the burndown chart is derived from the fact that remaining work is burning down over time. The monitoring tool is used in an attempt to guide the team to the realization of the Sprint Goal and to predict the progress of the Sprint, by showing the aggregated work on a day-to-day basis (Sutherland & Schwaber, 2017). In general, the tool can form a basis for a conversation, for example, it indicates when work is added (Berczuk, 2007).

Figure 8 shows an example of a burndown chart. The y-axis indicates how much work still needs to be done; the x-axis shows the number of days left; the red line shows the ideal line, (which is the total points that should be done per day to reach the Sprint goal); the blue line shows the work that remains to be done.

Scrum Board. The Scrum Board is a way to visualize the existing user stories of a Sprint. The objective of a Scrum Board during the Daily Scrum is to aid the discussion of the tasks at hand. Next, it is a passive information display throughout the day to show the status of the work (Esbensen, Tell, Cholewa, Pedersen, & Bardram, 2015). The board usually consists of a state of the story (which can often be seen in the top of the board). A basic set-up of the states of a Scrum Board is *To Do, Doing,* and *Done,* depending on the preferences, however, it can be extended with additional states as *Review* and *Testing*. It also provides information related to the story, such as who is responsible for it, the number of points, and its description (Sutherland & Schwaber, 2007).

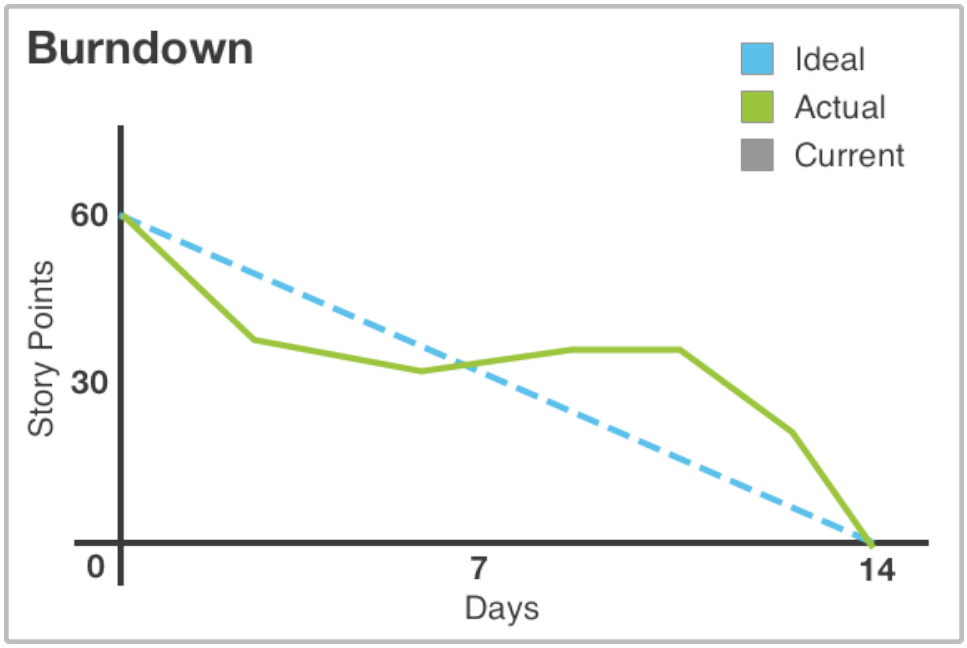


Figure 8. Burndown chart.

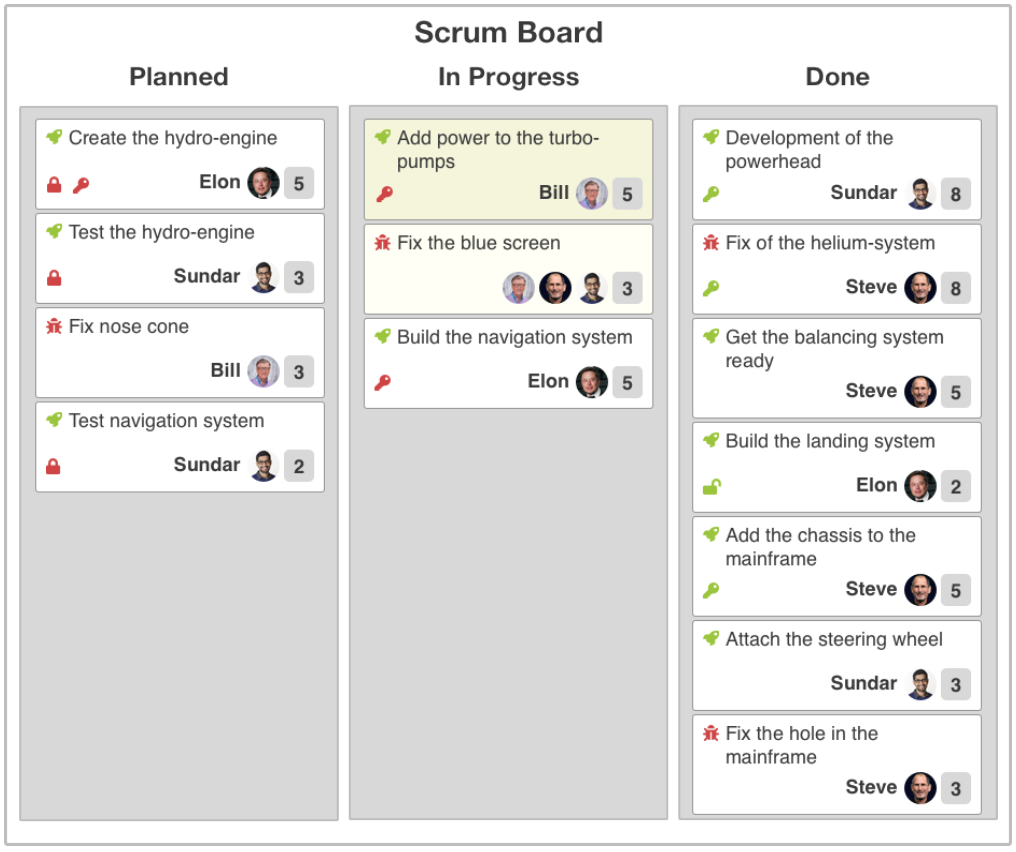


Figure 9. Scrum board.

2.2.3. Tools

In section 2.2.1. we argued that using Agile methodologies, such as Scrum, increase awareness, as those teams are working in short iterative development cycles and team-members meet each other on a regular basis (Schwaber & Beelde, 2002). However, Biehl et al. (2007) found that merely having a Scrum team does not provide sufficient support for ongoing awareness of team-members actions. This has to do with the nature of software development, which is best described as rather complex, having many interdependencies among team members. To increase awareness, Scrum software development teams make use of tools that provide information to support their development process (Treude & Storey, 2010).

These tools can take a variety of forms, such as dashboards or a code repository, and they often make use of Information Visualization. One early example of the use of Information Visualization is SeeSoft, a code based visualization tool with the objective to increase awareness, by showing who is working on which lines of code (Eick, Steffen & Sumner, 1992). Later, web 2.0 introduced new communication and collaboration possibilities such as wikis, tags, and feeds. Subsequently, researchers and software developers looked for opportunities to transfer these elements to aid software development (Treude & Storey, 2010). This resulted in new types of tools and Information Visualization techniques, such as Codebook, a social medium for software development (Begel & DeLine, 2009).

Paredes et al. (2014) conduct a systematic mapping study on the existing literature on Information Visualization used by Agile software development teams. Their findings suggest that “visualization techniques are valuable in Agile software development because they lead to understanding, collaboration, and self-organization. Visualization techniques help Agile teams to increase knowledge sharing when designing, developing, communicating, and tracking progress” (Paredes et al, 2014, p. 163). Thereby, tools and virtual environments are used in software development to increase both types of awareness (Biehl et al., 2007). A dashboard is a specific type of tool and in software development, it can be used to display a project overview, bottlenecks, and tracking. Thereby it provides a form of high-level awareness (Treude & Storey, 2010).

So far, the literature on awareness has provided us with valuable insights for our research. First, it can be concluded that awareness is essential for the success of software development teams. Moreover, through tools such as dashboards, Information Visualization can aid awareness in the software development process.

One of the objectives of this research is to create a dashboard that can aid Scrum teams. Two articles that can help to understand which elements are important to include in a dashboard for Scrum teams are described below. Azizyan et al. (2011) conducted a survey among software practitioners to discover the most and the least satisfactory factors with the use of tools by Scrum teams. They found that the most satisfying aspect for Scrum teams to use a tool was ease of use. Ease of use is also an important concept in the Technology Acceptance Model (TAM), which explains the acceptance of computer-based programs by its users (Davis, 1986). There, the ease of use functions as an intermediate variable, between external variables and the acceptance of a computer-based program. As our objective is to create a tool will be usable, we will use the concept of ease of use as a measurement during the test of the proposed solution.

Azizyan et al. (2011) provide two other important factors for a project management tool: customizability and prize. In contrast, they also identified that lack of integration is experienced as the most negative aspect of the use of a collaborative tool. Hence, an important learning is that our proposed solution should allow for integration. Mashup development is a way to avoid the lack of integration. A mashup is an application generated by combining content, presentation, and functionality (Yu, Benatallah, Casati, & Daniel, 2008). An example of a mashup is housingmaps.com, which combines Craigslist and Google Maps. The principle can be also applied to enterprises to combine public knowledge or business knowledge to their own needs (Elmeleegy, Ivan, Akkiraju, & Goodwin, 2008). An example is the integration of a code repository into a communication channel or vice versa.

Chow and Cao (2008) explored what the critical success factors of Agile software development projects are by conducting a quantitative study. Data were collected from 109 Agile teams, in a diverse group in terms of team size and industries. Their research included a large number of factors, mentioned throughout the literature, that could potentially affect the Agile software development projects. The research had the outcome that only 6 factors could, to some extent, explain the success of Agile development projects. Surprisingly, they failed to find confirmation

for some granted essentials for Agile development success, such as executive support or Agile appropriate projects. They did, however, find support for the following factors:

- A correct delivery strategy
- A proper practice of Agile software engineering techniques
- A high-caliber team
- A good Agile project management process
- An Agile-friendly team environment
- A strong customer involvement

Each of these 6 success factors had multiple attributes, which helped to explain the success. These success attributes included but were not limited to, good progress tracking mechanism, communication through daily face-to-face meetings, and a strong customer relationship (Chow & Cao, 2008). Both the factors and the attributes found in their research will be considered during the development of the dashboard.

2.3 Summary

In this chapter, we presented the foundation of our theoretical framework. First, we explained the essence, process, and value of Information Visualization. Subsequently, we shifted to dashboards, the form of Information Visualization used throughout this research, by introducing the categories of visuals, the existing literature, and the design principles related to the creation of dashboards.

In the second part of the theoretical foundation, we shifted to the concept of awareness in software development, and how both Scrum and Information Visualization can aid the development of this concept in software projects. Hereby, the Scrum literature was also introduced to set the context of this research.

3. Methodology

Saunders, Lewis, and Thornhill (2016) compare the process of uncovering data analysis and data collection techniques as similar to the action of peeling an onion. This allusion serves as the foundation that describes how this research came to be. Basically, before reaching the conclusion of which data analysis and data collection techniques we would use to answer a certain research question, we should first look into the aspects that lay the philosophical foundation of the research, ultimately reaching a reliable, valid, and coherent investigation. This chapter is dedicated to the methodological considerations done to shape this research.

A different approach to traditional thesis outlines, however, has been taken when describing the data collection and analysis. Due to the fact that this was a two-phased research, where the findings and reflections from the first step highly influenced the approach taken on the second phase, we have decided to move the data collection and analysis for each phase to chapter 4 and 5. We hope this will ease the reading process, as the story of this research is told chronologically.

The last section of this chapter relates this thesis with Design Science research methodology, which greatly extended our research design, especially due to the fact that we created an Information System. Nevertheless, the traditional approach to explain the methodology was taken, mostly because we wanted to build a strong argument towards the philosophy and approaches taken to the theory development. Design Science, therefore, served as a guiding light on which activities should be executed and when.

Figure 10 summarizes the path taken in order to reach the data collection and the data analysis. The choices taken in each step are highlighted, providing an outline of how our decisions are mapped compared to the pool of existing options for building a research.

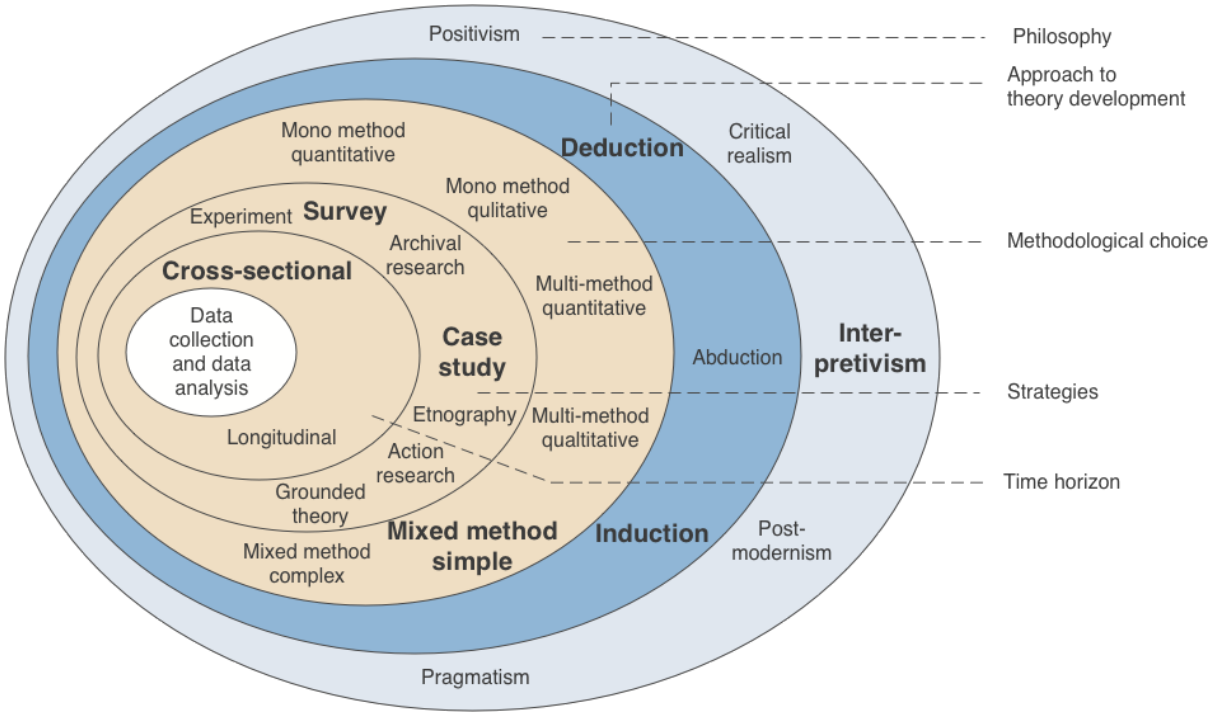


Figure 10. Thesis research onion. Adapted from Research Methods from Business Students (p. 124) by Saunders et al., 2016.

3.1. Research Philosophy

First, let us dive into the assumptions that lay the foundation for this research. On an ontological standpoint, we believe that software development Scrum teams construct their own reality, as each of them belong to a different setting, comprised of different organizational cultures with different individuals and goals, which means that the investigation is based “upon perceptions and experiences that may be different for each person, and change over time and context” (Eriksson & Kovalainen, 2008, p. 14).

To complement our perceptions on how reality is constructed in these environments, an appropriate approach to define the epistemology of this investigation, essentially defining how knowledge can be produced and argued for. The narratives told by the participants will be regarded as truth for the contexts they are witnessing. Moreover, our own experience with Agile also affects on how we perceive the truth in these contexts. This research aims to “create new, richer understandings and interpretations of social worlds contexts” (Saunders et al., 2016),

assuming that each role in Scrum such as Product Owners, Scrum Masters, and the Development Team experience the work done on a project in a different way.

This research, therefore, builds on top of an **interpretivist** approach, also known as constructivism, believing that reality is socially constructed (Saunders et al., 2016) and relying "as much as possible on the participants' views of the situation being studied" (Cresswell, 2014, p. 37).

3.2. Approach to Theory Development

With the philosophy in place, the next step going deeper into the 'research onion' is to delimitate how the theory will be embedded in the investigation. The objective of this research is twofold: first, understand how Scrum teams are supported by the tools that they use and how these tools visually help them to become aware of their everyday work; and secondly, test a prototype that could improve the awareness of these teams.

The first part of the investigation, together with the already existing theory on Agile software development and Information Visualization, feeds the necessary theory to the second part of the research, which can either validate or reject the created prototype. Therefore, this research possesses two approaches to the development of the theory.

To match our research philosophy, we chose to begin the research by focusing on understanding the individuals' meanings and rendering the complexity of the situation (Cresswell, 2014). This is considered an **inductive** inference, as it generates theory from the collected data that was used to explore or explain a given phenomenon (Saunders et al., 2016).

Following up the inductive approach we look into building mechanisms that protect the research from bias (Cresswell, 2014) and to bring "a testable proposition about the relationship between two or more concepts or variables" (Saunders et al., 2016, p. 146). This **deductive** step connects the theory of Agile software development and Information Visualization with the findings from the investigated Scrum teams, evaluating the learnings, ultimately providing a theory on the use of dashboards within Scrum teams.

3.3. Research Design

The research design consists of three steps that lead to the data collection and data analysis: a methodological choice, one or many research strategies, and the time span that the research will take. It basically is the general plan on how the research will be conducted in order to answer the research question and to also establish coherence between the chosen philosophy and approach to the theory development with the data collection and data analysis (Saunders et al., 2016).

3.3.1. Methodological choice

Being a constructivist research, this study relies on understanding the experience that the participants have with Agile and the tools that support their work. This data mostly come as words represented through meanings and stories. The richness of the information is essential in order to extract information that would lead to a new theory, therefore, a qualitative investigation is dominant on the inductive part of the research. The deductive part, on the other hand, is conducted with the purpose of testing the generated theory in order to validate the previous learnings, hence, a dominating quantitative approach is more suitable.

As a whole, the methodological choice in this study could be classified as a **partially integrated mixed methods** research, since it only uses qualitative and quantitative methods at a specific stage (Saunders et al., 2016). The reasoning behind mixing these two methods is to provide a higher level of confidence in the research. Using only one specific method could affect the findings as each of them come with their own weaknesses and strengths. Therefore, by combining both approaches, there is an opportunity of reducing this effect, leading to a greater confidence in the conclusions and providing a better understanding of the studied subject (Creswell, 2014; Saunders et al., 2016).

Overall, the purpose of this research was to mix exploration and explanation. First, by looking into how Scrum teams interact with their existing tools, understanding how these tools support their work and what are their needs and limitations. Then, by designing a solution that creates casual connections with the collected data and the theory that offers an explanation on how the participants can improve their process.

3.3.2. Research Strategies

This subsection describes the plan of action taken in order to answer the previously defined research question. It is classified as the step that links between the philosophy and methodological choices with the data collection techniques (Denzin and Lincoln, 2011). Similar to the choice of method, the strategy is split in two, in order to accommodate an appropriate plan for each of the methodological choices taken.

When choosing a strategy to go along with the qualitative part, two approaches seemed inviting. Doing an action research would place us in an organizational context, providing us with the opportunity to "develop solutions to real organizational problems through a participative and collaborative approach" (Saunders et al., 2016, p. 189).

An action research would provide a richer understanding of how the tools affect a team's cognitive capabilities and a more well-suited solution could be developed for that specific team. However, following this approach could have a negative effect on answering the proposed research question. Due to the longitudinal nature of action research, this strategy is more suitable for medium- or long-term projects as it usually runs over several iterations (Saunders et al., 2016).

On the other hand, a case study also seemed fitting, as our interest floats on a given phenomenon (Yin, 1994; Blichfeldt & Andersen, 2006); how Scrum teams are affected by their tools. Case studies also offer the opportunity of generating insights that can lead to empirical decisions and the development of theory (Yin, 1994; Saunders et al., 2016). However, Yin (1994) states that a rationale for doing a single case study happens ones it is a unique or extreme case that is going to be the subject of the study.

In this research, we were not dealing with a critical case, as recommended for single-case studies, nor we had sufficient resources to actively iterate over organizational issues, as recommended for action research. Therefore, we have chosen a multiple-case study design, as its evidence "is often considered more compelling, and the overall study is therefore regarded as being more robust" (Yin, 1994, p. 45). By investigating different teams that participate in different organizational contexts, an opportunity to follow a replication logic was placed, allowing us to identify similar issues that were faced in the different settings being studied. As recommended by Yin (1994), to complement the **multiple-case study**, an organization was selected as a *pilot*

case, in order to test if the data collection technique was properly designed and could be used on the other studied organizations without further changes.

During the last part of the research, where the quantitative approach is more prominent, a method that could evaluate and identify how effective the proposed solution was would seem more fit. In addition to that, being able to render numerical results from this phase was very important as they served as guidance for arguing the areas of focus in future research. On that account, a **survey** came across as being a good fit to reach this objective. Data gathered from surveys “can be used to suggest possible reasons for particular relationships between variables and to produce models of these relationships” (Saunders et al., 2016). Establishing the relationship between the results and the theory was an important part of this research as it answered our research question and marked the conclusion of this investigation.

3.3.3. Time horizon

Since this research is constrained to be due on a certain point in time, a **cross-sectional** study is more fitting and this is also a reflection upon all the methodological choices mentioned previously. The duration of the research is of five months and the context provided by the investigate are, therefore, a ‘snapshot’ of the period this research was conducted.

3.3.4. Quality

Finally, we conclude the discussion on our research design taking a look at the quality of the research. When talking about quality of a research, two criteria are commonly used: reliability and validity (Saunders et al., 2016). However, there are some controversy on the use of such criteria on research that follows use qualitative methods, especially the ones coming from an interpretivist approach (Golafshani, 2003). Stenbacka (2001, p. 552) simplifies the discussion on validity and reliability by stating that the former is “valid if the informant is part of the problem area and if he/she is given the opportunity to speak freely according to his/her own knowledge structures”, and the concept of the latter “is even misleading in qualitative research. If a qualitative study is discussed with reliability as a criterion, the consequence is rather that the study is no good.”

Since this research follow a mixed method, where the qualitative part establishes the foundation of the research feeding data into the quantitative phase, it seemed more fitting to choose more

appropriate criteria that would better describe the quality of this investigation and accommodate our philosophy. We are, therefore, going to use *dependability*, *credibility*, *transferability*, and *authenticity* as a way to describe the quality of this research.

Dependability in an interpretivist approach, focus on documenting all the steps taken in order to be understood and evaluated by others (Saunders et al., 2016); accordingly, all the actions done are recorded and made available, both as part of our data collection techniques section or in the appendices when serving only as supplementary information.

As for credibility, this criterion is established by creating a sense of trust and rapport with the participants (Saunders et al., 2016). It also ensures that the meaning given by the participants are taken into consideration (Golafshani, 2003). This way, our research will rely on triangulation mechanisms to test and validate during the quantitative phase that the findings from the qualitative stage were properly gathered.

Similar to dependability, transferability “provide the lenses for evaluating the findings of a qualitative research” (Golafshani, 2003, p. 600). Through comprehensive documentation on how the research was designed and analyzed, it should be possible to allow the reader to judge the transferability of this study to another similar setting that could be researched.

The authenticity criteria are created specifically for interpretivist research, where reality is socially constructed and is broken down in four more categories, such as fairness, ontological authenticity, educative authenticity, and catalytic authenticity (Schwandt, Lincoln & Guba, 2007). However, since Schwandt et al. (2007) express that the last three categories are still limited in terms of providing a good description, we will then describe only how we aim to achieve fairness in this research. Fairness comes in place when the inquiry confronts a situation of value-pluralism due to the fact that these groups or individuals create their own reality. It is the responsibility of the researchers to expose and explain all the conflicting realities to the reader. Our aim is to provide all this information when applicable, either on the analysis or the discussion.

3.4. Design Science

There is a close and conscious connection with the research design described previously and the Design Science research methodology. Design Science in Information Systems can be

described as a “research paradigm in which a designer answer questions relevant to human problems via the creation of innovative artifacts, thereby contributing new knowledge to the body of scientific evidence. The designed artifacts are both useful and fundamental in understanding that problem” (Hevner & Chatterjee, 2010, p. 5). The Design Science research methodology proposed by Peffers et al. (2007) is described as a framework comprised of six sequential steps that can be iterated as deemed necessary. The activities are described below, and how they are connected to our research:

1. *Problem identification and motivation*

Due to the interpretivist nature of this research, the problem identification comes only after the analysis of our qualitative research step. It would only be possible to identify the existing issues by understanding how Scrum teams interact with their process and tools. Surely, we do start with the assumption that there are some challenges being faced with the current set of tools that they use and how the different roles in the teams feel about them. Sections 4.2 and 4.3 analyze and reflect upon these problems which motivate our study.

2. *Definition of objectives for a solution*

Similar to the first step, the objective of our solution would become clear as we gathered data and reflected upon what were the issues faced by the participants and what lacked in the current toolkit that they were using. The requirements of the solution emerged from the categories of challenges that appeared from the case study. These requirements are described in section 5.1.

3. *Design and development*

At this stage, we connected the findings from the analysis and the theory in order to produce a testable prototype. The objective was to embed the contribution of our research in our prototype. The description of the solution can also be found in section 5.1.

4. *Demonstration*

The participant companies were revisited and the artifact was shown. The prototype was presented and a usability test was conducted. The format of the demonstration is described in section 5.2.

5. *Evaluation*

After gathering the results of the demonstration, we verified how well the solution performed compared to the predefined objectives. The performance is evaluated in terms of ease of use and the results discussed. The analysis and discussion of these findings guided us on which paths this research could take in the future. Sections 5.3 and 5.4 contain the results and reflections on the evaluation step respectively.

6. *Communication*

As the final step, communication serves as a way to disseminate the knowledge provided by the research. In fact, this very paper stands for that. It comprehensively describes all the activities taken and documents the findings and reflections upon the subject matter of this study.

Even though the steps taken in this research were as the ones described in Design Science research methodology, we opted to use it as a practical extension of our work (Peffers et al., 2007). Design Science follows a more pragmatic paradigm to solve problems, which is not the purpose of this research. Following the interpretivist approach, we want to understand and reflect upon the people and their environment. Even though the main contribution of this research is indeed the created prototype, the process of understanding the reality of these team members were of utter importance. Therefore, we have chosen to describe the methodology in a more social sciences traditional manner.

4. How Are Scrum Teams Affected by Their Tools?

This chapter focuses on the process towards answering the first clause of the research question. It describes what were the data collection techniques and how the data was analyzed. Moreover, it presents the results from the data collection and projects our reflections upon the findings to the next stage of this study.

4.1. Data Collection and Data Analysis Plan

Before visiting the teams to understand their Agile practices and how they used their tools, it was important to devise a plan to pick the right sample and choose the right tactics for doing a proper data collection and data analysis.

As a precondition, we would only reach out to companies that use Agile software development methodologies. Moreover, being able to investigate organizations with diverse team sizes and from different industries would be a plus, as the results could broaden the horizon of the research, allowing us to create an intersection of the these teams' practices, independent of the business they are in.

Our ambition was to investigate a total of five teams, and from each of these teams, interview the individuals that would represent the roles of a team member, project manager, and a product manager, for a total of 15 interviews. The choice of the number of interviews was a conscious decision, as Saunders et al. (2016) recommend undertaking 5 to 30 interviews; sometimes conducting more than 12 homogenous interviews can lead to data saturation, i.e., interviews stop bringing any additional information. Furthermore, one of these companies would serve as a pilot, enabling us to test our interview guide and improving it, before visiting the other organizations.

To collect the necessary data in this phase, semi-structured interviews were chosen as a reliable technique. Questions would be encompassed under certain themes coming from the theory, such as teams collaboration and process awareness, and would follow a free flow, allowing questions to be asked in an unspecified order. It was also a good fit to this stage since the goal

was to explore how teams are using these tools and at the same time explain why (Saunders et al., 2016).

All the interviews, with the exception of the pilot case company, would be recorded and transcribed. The purpose of the pilot was to evaluate the quality of the interview guide, validating the questions' design. The interview guide was constructed under three different themes: demographics, Agile practices, and benefits of Information Visualization.

In the first moment, we would get familiar with the company and the setting that the team belongs to. This step serves as an icebreaker, leading the way towards a more natural conversation. Later, we ask about their Agile practices, such as which methodology are they using and how loyal are them to the core principles, and we also ask about their relationship with the customers. The last block of the interview is dedicated to covering the different benefits of Information Visualization. This part was considered the focus of the interview, as its purpose is to identify how the team handles their tasks with their current set of tools.

A total of 16 main questions were made with the possibility of pursuing those specific themes for further comprehension if deemed necessary. The interviews were estimated to be around 45 minutes long. During the interviews each one of us took a specific role, either acting as the interviewer, following the interview guide and making sure that all the main questions were answered, or as the notetaker, supporting the interviewer when more clarification on the questions was needed.

An additional question was added to the interview guide after the pilot interview. It served as a summary of the interview to understand what people do to become aware of their environment after leaving it for a while, in this case, during one or two Sprints. Below, you can find the complete interview guide with its 17 questions and what type of Information Visualization it is connected to:

Demographics

1. Could you describe the what the company does in a few words?
2. What is the organization size, and what is your team size and composition (developers, designers, testers)?
3. What is your role within the team? (PO, PM, SD)

Agile process

4. Which Agile methodology are you using? Why?
    a. How are you using it? What elements of the method are you using?
    b. Do you talk to the customer during the process? If so, how do you interact with the customer during the development process?
    c. For how long have this team been using the current method?
    d. How loyal are you to the methodology? Do you use all artifacts?

Information Visualization - Supporting tools

5. What type of tools are you using that support your software development process?
    a. Why are you using this particular tool(s) and not other ones?
    b. Could you maybe explain or potentially show us how you use them?
    c. When do you use them and for what reason?
6. Why are you using this tool?
7. What do you think in general about the tool/process that you're using?
    a. Is everybody using it the same way? Why?
    b. Do you think there are any issues with it? Could you give an example of what is wrong with it?
    c. Is there something you would like to see improved?
8. Does the tool you use help to establish a communication channel with your customer?

Information Visualization - Collaboration

9. How does your project management tool facilitate you to produce a certain output with a colleague?

Information Visualization - Low-level Awareness

10. How do you track changes in the project's artifacts? And how does this influence the progress of the project? Can you give us an example?
11. How do you handle tasks that need input from more than one team member, due to the specialization of skill sets, for example, design, development, and testing?
12. How often do you use your project management tool to ask for clarification in the requirements?

Information Visualization - High-level Awareness

13. How your current process provides you with the possibility to be aware of what another team member is doing? Can you give us an example?

14. How do you know where in the process (project) are you currently in? (e.g.: 50%, 100%) Does the tool provide any support with this information? Is it relevant to know it?
15. When you are looking for information about the project or the project status, what are you looking for? What type of information would be valuable to you?
16. How do you rely on the data (e.g. issues on the ticket system, chat messages, sprint velocity) that you currently have based on your toolkit? Does it help you make better decisions? How?

Interview Summary

17. What are the actions that you take after coming from a 2-week or a 1-month holiday to get up to speed on the project that you're currently assigned?

Transcription of the interviews took place, quickly after we finished all the interviews. The purpose of transcribing was to ensure a higher quality analysis, however, the process of manually transcribing a one-hour recording can take up from six to ten hours (Saunders et al., 2016). Considering that we intended to interview 15 individuals on 45-minute long interviews, this would mean from 67.5 to 112.5 hours of interview transcription.

To reduce the amount of effort spent, we used a web service called Trint (TRINT, 2018), an artificial intelligence speech to text converter embedded with a text editor. Although the tool's support was relevant, we still had to take an active stance towards reviewing what was done and correct the existing inaccuracies. With Trint, it took approximately 3 hours to complete the transcription of each 45-minute interview.

The audio transcription was done individually, each one is responsible for the half of the files and the transcripts were shared in order to do the analysis. The analysis of the transcripts followed the framework proposed by Burnard (1991) on analyzing interviews transcripts, with a few adaptations of the original framework.

Burnard (1991) defines 14 step towards building a detailed and systematic artifact of themes and issues addressed in the interview and connect these themes together into categories. First, the moment of analysis of the interviews started at the same time the interviews were conducted. Through memos, considerations that attract the notetaker's attention were recorded during the interview. These notes were used for later discussion after the interview.

A shared folder was created to hold all the transcripts. The transcripts and the notes from those interviews are then copied over to a spreadsheet to facilitate the analytical process. We both read all the texts to become familiar with the data, proofread the content, and have an overall perspective of the possible themes. The latter was done through the use of author memorandums, a specific column for what one of us would identify as important with the purpose of noting down the relationships between the different interviews (Brinkmann & Kvale, 2015).

After the first read through, a more detailed and individual analysis of the interviews was done. Each one of us created our own code books, going through the statements and establishing our own headings and categories. The headings stand for a way of describing the content of a certain statement (Burnard, 1991). We then started narrowing down the number of categories by grouping them. Overall, this meant that the interviews were analyzed twice, the first individual and then in a group, in order to increase quality and reduce bias.

Later, we group together to discuss our findings and merge the categories that we have identified. This new set of categories are then described and are given a core statement from the interviews to aid in the recognition of the categories with similar meanings. With these core statements in place, we made the last comparison to make sure that the final set of categories would cover all important statements from the interview. Table 4 displays a snippet from the finalized codebook.

The categories are the elements that are accounted for to represent the number of mentions in the interviews that would be used in the findings section to identify the most relevant categories in this phase of the research.

To conclude the analysis we did a systematic comparison of the project management tools used by the participants and cross-referenced them on the most mentioned categories from the interviews. We have created a project with tasks that resembles the work necessary to be done in a thesis project and converted the tasks into stories, then tried to input this information in the different tools and tried to run fake sprints to see the behavior of these tools. Thereby, we were able to experience the strengths and weaknesses of each individual tool, in relation to the findings from the interviews. This last stage would serve as an inspiration to not only answer

how the Scrum teams are affected by their tools, but would also serve as input to design the prototype.

Statement	Heading	Category	Notes
Basically, I would say I mostly do rely on kind of ambient awareness.	I mostly do rely on kind of ambient awareness.	Sprint tracking	
So I follow everybody can and should look at a ready pull request that comes into the platform. So through that, I know of the changes that are happening now, that happened in terms of what people are working on in this instance. I would say that I don't know and I don't care and if I need them then I will try to ping them on Slack, even though they are next to me because like people might have headphones on they might be in the zone, working on some specific problem. So I think out of respect I would prefer that.	I don't know and I don't care, and if I need them then I will try to ping them on Slack, even though they are next to me because like people might have headphones on they might be in the zone	Goal oriented	
But that being said half the time I would just poke them on the shoulder bother them saying I do need help. Yeah. So I try to be respectful of their attention.	I would just poke them on the shoulder	Offline communication	Link to the Agile manifesto
At the same time sometimes it's also human contact, right? That's also good in and of itself.	It's also human contact, right? That's also good in and of itself.	Offline communication	

Table 4. Excerpt from the codebook of the interview with Martin (personal interview, March 13, 2018).

4.2. Findings

4.2.1. Demographics

Six teams from different companies were interviewed in total. One more firm than the original plan, since the five intended companies could not provide us with the desired number of interviews. The teams ranged from a variety of industries, such as banks, media, Internet software & services, and IT services. The teams investigated had their offices in either Denmark or the Netherlands. Table 4 summarizes the companies, their industry and their which office

participated in the interview. Next, we provide a more thorough description of the company and the teams.

Name	Industry	Location
Adapt	IT Services	Copenhagen, Denmark
Blue Billywig	Internet software & services	Hilversum, the Netherlands
Draad	IT Services	Nijkerk, the Netherlands
Exitable	IT Services	Weert, the Netherlands
Karnov Group	Media	Copenhagen, Denmark
Santander	Bank	Copenhagen, Denmark

Table 5. Summary of the interviewed teams.

Adapt is a digital agency, founded in 1998. Within this field, they operate on a strategic and operational level. The latter includes the design, creation, and delivery of digital products. Adapt's mission is to aid their customers with their digital strategies. They operate from six worldwide offices, including Barcelona, Boston, and Vilnius. Their main office, however, is located in Copenhagen. 70 of a total of 130 employees are located in the Danish office. The teams at Adapt are divided into client teams, and they are fully responsible for a specific customer. This does not only include the development process, but also budgeting, planning, communication, and so on. These teams usually range from four to sixteen members. The team that we interviewed had Interflora, a flower delivery network, as their customer. Even though Adapt uses Scrum as their Agile methodology, the team members share other roles besides the Scrum roles. The user interface designer also acts as a Product Owner, for example, bringing in the requirements from the customers.

Blue Billywig is a Dutch company, operating in the online video platform market. They have two products: a video platform and a video player. With this, they host, publish, and analyze the video content. These services are provided by an in-house team, located at Hilversum, the Netherlands. A total of 35 people work there and the product development team has around 8 people, and it consists of developers, a Product Owner, and a designer. The teams do not have the role of Scrum Masters, the argument behind this decision is due to the seniority of the team, which considered that the role of a coach or facilitator was not necessary.

Draad is a small design agency from Nijkerk, the Netherlands. They position themselves as a no-nonsense design agency: they build websites and prejudice a slow process. They develop websites with a team of 9 people located in the Netherlands, and one in Romania. Their focus on Scrum was limited to the use of a Scrum Board, the Daily Scrum, and Sprint Planning. Draad was chosen as the company that would be our pilot case since they had stated beforehand that they could not fully participate in the research process. Furthermore, from a preliminary conversation, they had mentioned that they really vouch for a lean process, therefore, they do not use many tools.

Exitable is a design agency from Weert, the Netherlands. They focus on the development of websites and have a strong relationship with Ketch App, an app development company. In total 18 people work here, who all take interchangeable roles. From Exitable, we had only interviewed one individual, who acts as a back-end developer and strategist, mostly taking the role of a Scrum Master in the team.

Karnov Group is the largest legal, tax, and auditing information provider in Scandinavia. Currently, the organization employs 230 individuals throughout their Danish and Swedish offices. Their development department is composed of 35 people, where 6 belonged to the team we interviewed. That team was responsible for Karnov Group's online platform where users can search and annotate the documents available in the service. Here the role division was more strict and we were able to have an interview with all the roles described in the Scrum framework.

Santander Consumer Bank is an international operating bank, and part of the Spanish Banco Santander Group, focusing on loans and credit cards. 250 people work in the Danish office, with 30 working in development. The main focus of the team we interviewed was to handle the compliance issues coming with the new General Data Protection Regulation (GDPR), which would be enforced May 25th, 2018. The Scrum teams in Santander mainly consist of 3 developers, a Scrum Master, and a Product Owner. The Scrum Masters and Product Owners are shared across teams, though.

4.2.2. Agile Process

Twelve semi-structured interviews were conducted: 1 for the pilot case company and 11 that were actually accounted for in the analysis (Appendix B). This diverges from the original plan since we aimed for 3 pilot and 12 regular interviews, but after conducting them, they were

considered a reasonable number of interviews, due to the repetition of information. From the Dutch companies, it was only possible to interview people from one specific role, either a Scrum Master or a Product Owner. The interview with the Scrum Masters from Santander was, in fact, a semi-structured group interview, due to the fact that both were new hires from the bank and the lack of available time to conduct the interviews.

The pilot interview was done with Draad's project leader. Although they were doing a process similar to Scrum, with planning and refinement sessions, the process was done using nearly no tools. They used the board to see what everyone was doing and a spreadsheet to hold the features that were needed. There was very little reliance on the toolkit they had, and most of the work was done through face to face communication. The outcome of this interview was that we needed a way to grasp the whole goal of the interview and see if the tools aided in the process, so a new question was added to the interview guide which was: *What are the actions that you take after coming from a 2 weeks or a 1 month holiday to get up to speed with the project that you're currently assigned?*

Scrum was the methodology appointed by all the teams as their process towards project management. All of them stated that they were using all of the Scrum events and had the specific roles, although sometimes, those roles responsibilities were shared or partially fulfilled. However, as it is common with the Scrum framework, the teams had their own extensions to the method, so they can fit the process to the organization culture.

Many of the teams had added their own set of events into the process. These ceremonies served as a way to either improve or adjust the sprint as they go, which is in line with the Agile and Scrum methodology of "inspect and adapt" (Schwaber & Sutherland, 2017). At Adapt they have a meeting called Story Time, where the team sits together and start to refine and estimate the backlog items, also known as stories (Niklas, personal interview, March 28, 2018). While Blue Billy added User Story Review, as a moment for the Development Team to present the finished stories (Linda, personal interview, March 8, 2018).

Some of the interviewees had difficulty in positioning themselves with their roles in Scrum, even after stating that they were using the methodology. As an example, Sandra (personal interview, March 28, 2018) mentioned that her position leaned more towards a project manager than Scrum Master, although she did all the tasks related to facilitating the work of the team.

We have, however, classified their roles based on their tasks, following the Scrum methodology, therefore the interviewees would be placed either as a team member, Scrum Master, or Product Owner (Schwaber & Sutherland, 2017). Therefore, the interviewees had the following representation: 3 team members, 4 Product Owners, and 5 Scrum Masters. The division of roles within the team depended on the size of the company and the maturity of the Scrum method in place. In Santander, for example, they had two Scrum Masters responsible for multiple teams, which catered the opportunity to be distant from the team and share knowledge between teams (Michael & Stine, personal interview, March 22, 2018).

The major difference in terms of the role was the one from the Product Owners. These participants had very different backgrounds, and even though they shared some activities in common, like backlog grooming and customer relationship, their other tasks focused on different areas. Their backgrounds ranged from data analysts, web designers, business informaticians, and computer scientists. This difference is reflected in a lower degree of categories that intersect the needs for this role.

It is notable, however, that all the teams stated that they assiduously conducted the Scrum events, such as the Daily Sprint, Sprint Planning, and Sprint Review. These ceremonies acted as a fixed time-boxed event where face-to-face communication came in place to discuss their daily issues, the bottlenecks identified, and the achievements of the last Sprint.

During the interviews, the teams mentioned the different tools that they use and where they help in the process. A total of 31 software were identified and they were plotted on a table for further analysis. This was a casual classification with the sole purpose of serving as inspiration when designing our prototype, the function description is based on the information available on the related companies' website. The table is available in Appendix C.

Commonly mentioned tools were used for communication, such as Slack, referred by all companies, have the main objective to foster information exchange. Hosting for version control services, such as GitHub and continuous delivery platform like Jenkins were also frequently mentioned. Our interest, however, was mainly in their project management tools. It is the tool that is used by all the roles in the team, and it is also where the participants mentioned the most painful issues.

Four project management tools were referenced during the interviews: Microsoft's Visual Studio Team Services, Atlassian's Jira, Pivotal Tracker, and Asana. These tools were analyzed and compared with the categories of requirements mentioned by the participants. The summary of the systematic review of the project management software is available later in this chapter. First, we present the findings on what were requirements the interviewees had.

4.2.3. Requirements and Constraints in Agile Tools

During the first stage of our analysis, we have identified 45 different types of requirements and constraints from the interviews. This number, however, was reduced to 19 after constantly reviewing it to an extent that doing more merging would remove the meaning we wanted to convey. In the context of our study, we define a category as a *requirement* if the category is wanted and the tooling can provide sufficient support for this. On the contrary, a category is labeled as a *constraint* if the category is wanted but the tool cannot provide sufficient support. Figure 11 provides an overview of all the categories and their number of mentions during the interviews. The horizontal axis stands for the given names of the categories and the vertical axis represents the frequency of their mentions.

From the interviews, it became clear that some categories were more relevant than others during the conversation. Since the number of categories is still high, and they would be used as input to the design of the prototype, we established a threshold of at least 10 mentions for the categories that would be further analyzed and be used as inspiration for the design of the prototype, meaning that the categories being inspected range from *stakeholder engagement* to *group awareness*.

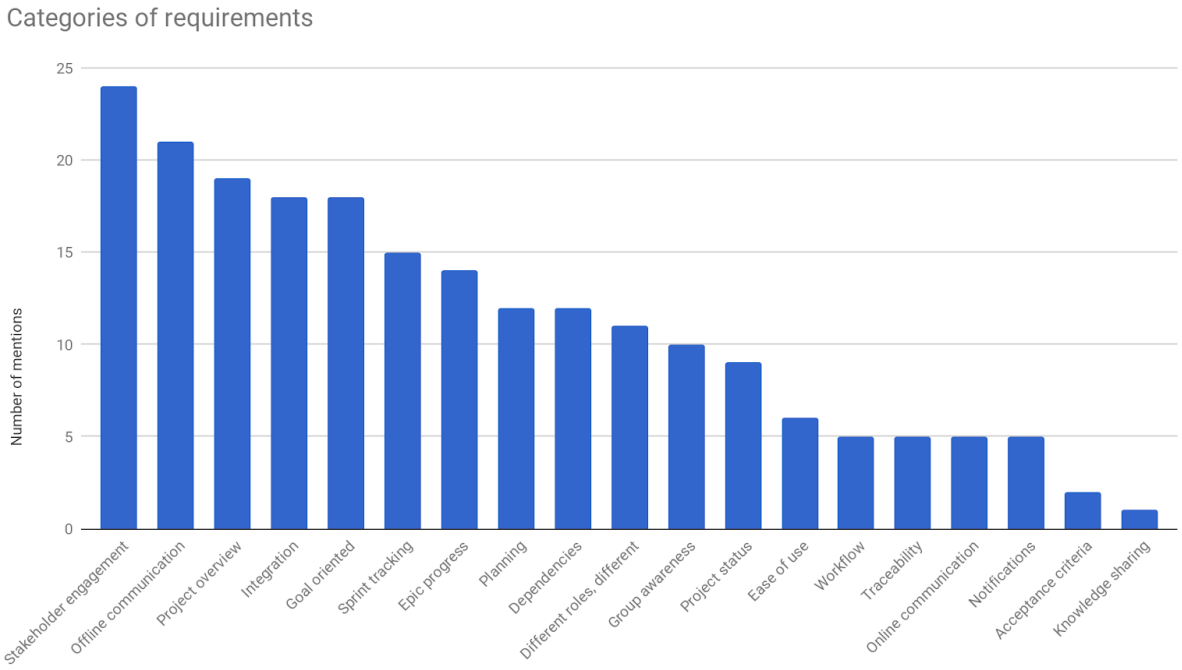


Figure 11. All mentioned categories of requirements and constraints.

Although there is still the possibility of merging some of these categories, like offline communication and online communication, which could be simply defined as communication, we still wanted to make known the differences between them.

To complement the chart on the number of mentions Figure 12 represents a stacked bar chart that displays the distribution of mentions. The mentions have also being weighted since there wasn't the same number of participants of each role. Next, in order of priority, we describe what each of the most relevant categories meant and we link them to a core statement from the interviews that represents what they are trying to convey.

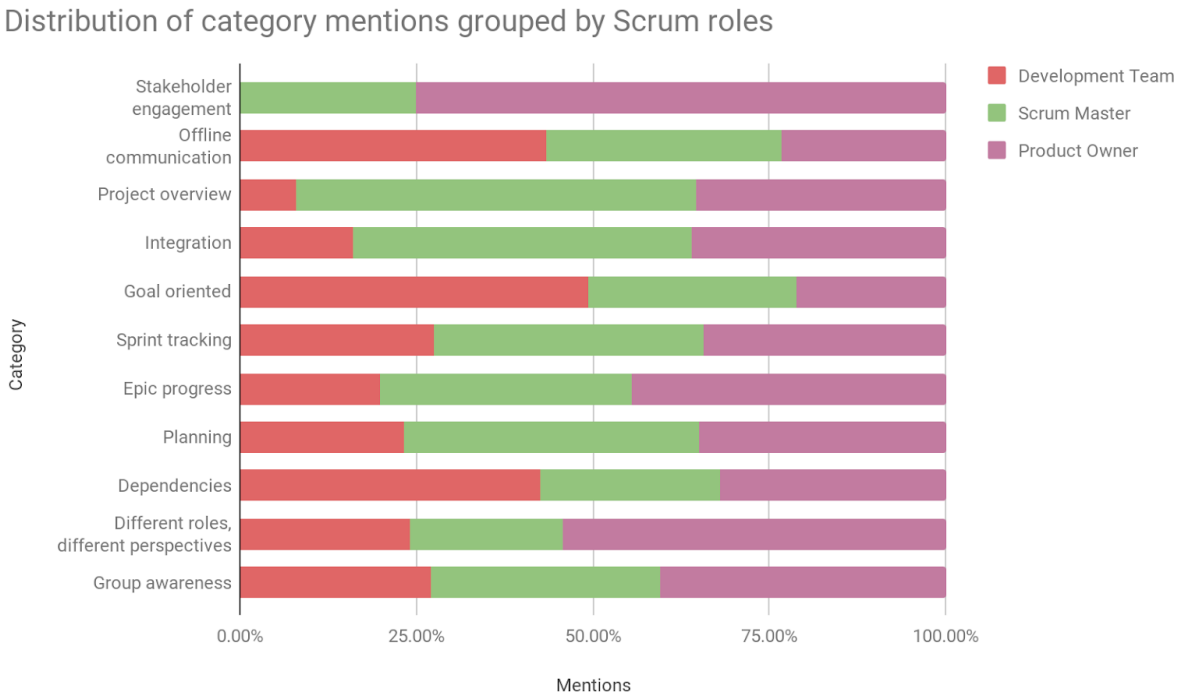


Figure 12. Mention distribution of the studied categories grouped by Scrum roles.

Stakeholder engagement was the most representative category, as it was mentioned 24 times throughout the interviews. It describes the relationship between team members and the interested parties of the solution being developed. Stakeholders include clients, users, other development teams, and other departments within the organization. This category is best described as a requirement for Scrum teams, as it is difficult to keep stakeholders engaged in the development process. Some mentioned that there is a lack of 'agility' on the stakeholder's side and occasionally, they are not participating in the discussions done within their toolkit. Stakeholder engagement is described as an important means to align the development within the Scrum team and their stakeholders. We have found that two of the interviewees were hired specifically to handle this issue. Stakeholder engagement was only mentioned by Product Owners and Scrum Masters. From a Scrum perspective, it is important to engage stakeholders, to optimize the product that is being developed.

> "But then there is other product related things I need to keep up on. What's our go to market strategy? Is marketing ready? Is our content department ready with the content that needs to go in here? Is our sales department trained, ready to go? Is our education

department up to speed on what's going to come? Is our customer support department ready?*”* (Ivan, personal interview, March 13, 2018)

Offline communication. This category was mentioned second most often and represents situations where the interviewee explicitly mentioned that they would walk to reach out to their colleagues for a face-to-face conversation, and ask for clarifications. We have identified that offline communication was mostly used when the tools were lacking in terms of providing sufficient explanation. Online communication tools, like Slack, were also mentioned, but only when it concerned short, incomprehensive, or distant communication.

> “I would just poke them on the shoulder, bother them saying I do need help… At the same time sometimes it's also human contact, right? That's also good in and of itself.” (Martin, personal interview, March 13, 2018)

Project overview represents the statements that are concerned with the ability of the project management tools to provide at a glance reporting. The interviewees often mentioned this when they wanted to have an overall perspective of the project: *How many bugs do we have? When is the next release? How are we doing in terms of velocity?* It was a common topic during conversations with Scrum Masters, as they usually felt the need to grasp statistics of the project to report to their leaders.

> “I think Pivotal has a bit of a limitation on how to display stuff, and for me, it is important to see the overall state fast without spending too much time and going too deep into stuff.” (David, personal interview, March 13, 2018)

Integration relates to the connection between one or more tool, as a way of extending their power to complete tasks. An example is the use of their communication tool (Slack) to automatically notify on failing builds coming from the continuous integration service (Jenkins). Some challenges were mentioned during the interviews, such as the poor development of custom integrations, which made the software harder to use or not even reliable. It was also noted that some integrations do not fully satisfy the team’s needs. However, the existence of integrations was highly appreciated, and tools with a set of good integrations, such as GitHub and Slack, were continuously praised about that. Simply due to the fact that would reduce the time that they need to switch between tools to do minor checks.

> “They are very positive about this one [Visual Studio Team Services] because it's apparently very good to combine with some of the other tools that we have been using.” (Michael, personal interview, March 22, 2018)

Goal oriented. Having the goal in the back of their minds would be a good and simple way to stay focused. Whenever interviewees mentioned that they needed to focus on their tasks at hand, also relating their activities to the Sprint goals, this category was used. Members of the Development Team are the ones commonly mentioning this requirement as they have to focus on their work. Similarly, the Scrum Masters need to keep track of the goal, so it is also part of their workflow. Four out of the five teams mentioned that they used Sprint goals, however, the toolkit they had did not offer any way of tracking these goals.

> “One of the things Scrum wants to do is make sure that the development team can focus on developing stuff, so anything that is not developing self is a distraction.” (Linda, personal interview, March 8, 2018)

Sprint tracking relates to the analysis of a project’s progress during single iterations, therefore, acting as a way of raising short-term awareness. This is done through conventional tools from Scrum, like the burndown chart, and the Scrum board. The objective of Sprint tracking is to discover the overall likelihood of achieving a certain goal and not, as it was emphasized, used as a way to account someone for the work being done.

> “Both as a Scrum Master and project manager I would know what we're trying to build and then I would see it for myself if we are on track or not.” (Sandra, personal interview, March 28, 2018)

Epic Progress is very similar to Sprint tracking, however, its purpose is set on tracking a higher level, ie. the tracking of epics. Epic is a tactic commonly used in Scrum to handle items that can be grouped to achieve a specific goal. It is used to show the evolution of the project. Epic progress is categorized as a requirement since the issue is usually raised when Agile teams feel they need to know where they are in the project, how much has been completed, and how much has yet to be done. Epic progress has more mentions between Product Owners than in *Sprint tracking*.

> “I actually could make a very nice horizontal bar of your feature or epic and show you how far we are of the total amount of story points or stories.” (Morten, personal interview, March 22, 2018)

Planning. This category represents the mentions that matched the interviewees' needs to look ahead to prepare the work for the future Sprints. Within the Scrum process, this was mostly done in the Sprint Planning ceremony.

> “Even though Jira can be difficult... I was thinking that the toughest work is always to get the project broken down adding story points. You do different tasks so you can actually do some kind of forecast.” (Sandra, personal interview, March 28, 2018)

Dependencies are the relations that some tasks intrinsically have. From the interviews, it was stated that dependencies could relate to tasks as well as projects. There are different types of dependencies between the tools, for instance, tasks that block another, or tasks that are within the same group of features (sometimes this is defined as an epic), duplicated tasks that have the same goal, but arrived from different stakeholders. Consequently, dependencies serve as communication tools, a useful mechanism to identify and connect challenges between tasks, projects, and teams.

> “Sometimes we create relations between the user stories. This need to be finished before that one. Then we finish this one in Sprint two and we will start this one in Sprint five, maybe.” (Niklas, personal interview, March 28, 2018)

Different roles, different perspectives. In some interviews, the participants admitted that the team members had different tasks that they needed to accomplish with their tools. Although in some events these were given as assumptions, other interviewees had actual experience within the other Scrum roles. Statements that show empathy with the colleagues that have different jobs had this category assigned.

> “Now, I'm thinking about our team lead. He has some specific scenarios that he wants to support with the tool and he doesn't think Pivotal support those, but I don't really have those requirements, so I don't know if it's in the position of team lead that you would have those requirements or if it is like a personal preference.” (Martin, personal interview, March 13, 2018)

Group awareness. The category group awareness was created to reflect the need for having an idea of who is doing what and which workload they have during the Sprint, contextualizing their own work. Both online and offline tools were used to achieve awareness.

> "Because it is only up to the Daily Scrum where you can get an idea of how is it going for the colleagues and are we on track or are we not on track." (Niklas, personal interview, March 28, 2018)

Some of the findings from this step can be related to the work of Azizyan et al. (2011). Through a survey on understanding usage and needs of tools for Agile teams, they have found five categories were important when selecting a tool. In order of relevance were: ease of use, customizability, price, availability of reports, and integration with other systems. Ease of use was also mentioned during our interviews, however, with only six mentions, it wasn't a category prominent enough.

We have also identified that integration was an important part of their process, being the third most mentioned category in the interviews. However, when we asked them the reason why they were using the tools for project management they were using, the most common response was due to the fact that the tool was already in place before they started in the organization and they were either satisfied or got used to it.

Concepts like pricing were not mentioned during the interviews. Customization was a characteristic that was absorbed by some of the created categories. In Azizyan et al. (2011), the reason of asking this question relates to the fact that they were investigating teams that could be using tools that were not custom made for Agile project management tools, like Microsoft Excel, for instance. So it was related to the ability of the tool to adapt to their process. There were a few categories that were related to customization, such as the *different roles, different perspectives*, and *workflow*. In the first, the users would have distinct needs from the tool.

The reporting system was also broken down into different categories, as we looked for more details in that area. So instead of simply classifying the statement "I want to know… how long does it take to resolve bugs." (Ivan, personal interview, March 13, 2018) as a report availability, we opted to group it in a specific category called *project overview*. There are other groups that refer to specific types of reports like the burndown chart, which relates to *Sprint tracking*.

From the different roles that we interviewed, it is also possible to see where their focus is placed. For team members, their attention was directed to the current task at hand, mainly talking to their colleagues (offline communication), completing their tasks in due time (goal-oriented), and being aware of the relationship between tasks (dependencies). The project management tool served as a feed for the necessary tasks, where the support of the team as a whole was more important.

Scrum Masters, on the other hand, were more interested in how the project was going. They were also interested in the goals, but the perspective was more on tracking and reporting results. They wanted to have an overview of how the Sprints were going and the ability to plan ahead for the next Sprints. The Scrum Masters were also responsible for improving the process, therefore, integration with other tools to optimize the workflow was also continuously mentioned during the interviews with participants of this role.

Product Owners provided a more strategic output. They have strongly mentioned the relationship with the customers, and the lack of tool support to handle it. Most of them mentioned that the handling of stakeholders was done through emails. They were also the most interested in knowing who was doing what in the project since they were not as close to the team as the team themselves.

4.2.4. Systematic Comparison of Project Management Tools

After having a clear understanding of how the investigated Scrum teams were using their tools, and grasping what they were looking for, we have done a systematic comparison of the existing project management tools mentioned during the interviews, as they were commonly mentioned as the place to go to have an understanding of the project by all the roles. The systematic comparison was done not only to establish how these tools currently provide a solution to the identified categories but to also inspire us when designing a new artifact. Next, we evaluate how these tools handle the requirements mentioned by the interviewees. The full systematic comparison can be found in Appendix D.

In a fake project that mimics the work that needs to be done when writing a thesis, we created different tasks, estimated them, assigned them to one of us, and plotted them on a release plan on paper first. Then we feed this data into the different tools, testing the tool and comparing how

it manages certain tasks against the categories mentioned in the interviews. First, an introduction of the teams' project management tools is made necessary.

With the mission of helping "humanity thrive by enabling all teams to work together effortlessly" (Asana, 2018), Asana is a generic project management tools that can handle different types of project. It is a task-based tool that is simple by default. A user adds the tasks to a list which is shared by the team. This was the tool used by Exitable.

Pivotal Tracker is a project management tool that provides an "intuitive Agile workflow" (Pivotal Software, 2018). The tool provides a process to guide the development. It comes with a velocity calculation that automatically adjusts the Sprint's velocity as it goes. Karnov Group uses this tool.

Atlassian's Jira is a popular tool for project management and "it is built for every member of your software team to plan, track, and release great software" (Atlassian, 2018). Jira is a customizable tool for project management that adheres to different teams' workflows. In their homepage, they highlight the features that were also mentioned by the interviewees as requirements, such as tracking, planning, reporting, and integration. This tool was used by Adapt.

Microsoft's Visual Studio Team Services is a suite of tools for software development teams. It does not only provide an Agile project management interface, but it also contains continuous delivery, code repository, packaging, and testing tools within the same service (Microsoft, 2018). This tool was used both by the Santander and Blue Billywig teams.

Below you will find the results of the analysis of these tools in comparison with the findings from the interviews.

Stakeholder Engagement

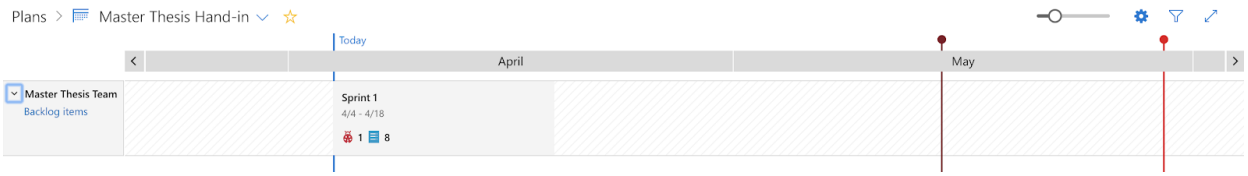


Figure 13. Plans add-on in VSTS.

Microsoft provides a free add-on called “Plans” in VSTS, which offers a perspective that goes beyond the current Sprint, displaying what items are planned to be done and in which Sprint. Moreover, it offers the feature of managing milestones that the team should be mindful of. It is a useful tool to communicate to stakeholders what is planned to be built in the following Sprints and also to communicate to the team when certain features are expected to be done. None of the other tools provide a roadmap or stakeholder management tasks without the use of an integration created by third parties.

Offline Communication

Since these tools are available online it is hard to convey how they could be used for offline conversations, however, they do boost communication, according to the interviews, as they provide the information that needs to be clarified during the meetings, such as the stories description and acceptance criteria, burndown charts, and reports.

Integration

All the project management tools investigated offer some sort of directory for integrating with the most popular web services available for free, such as Google Drive, GitHub, and Slack. These integrations facilitate the process of notifying team members on changes in any of these tools, facilitating the process of being aware of modifications in the project. It was not possible to extract the number of extensions the extension directory of these different tools had.

Goal oriented

Figure 14. Jira Sprint goals.

Sprint goals are one of the requirements of the Sprint planning session described in the Scrum guide (Schwaber, & Sutherland, 2017). With that being said, however, the only tool that provided

such a feature was Jira, but the Sprint goal wasn't very evident on the user interface. The arrow on Figure 14 displays where the Sprint goal is placed in Jira Scrum board. VSTS does not have an explicit widget available for the Sprint goal but allows general notes which can be used to note down the goals. However, as with Jira, the Sprint goal in VSTS is not prominent.

Sprint Tracking

Among the most common measures to track Sprints are the use of Scrum boards, burndown charts, and burnup charts. The different tools provide some sort of ability to track the work being done. Asana, Jira, and VSTS offer a Scrum board, and also offer reports to track the remaining amount of work, although they are available in different screens. Pivotal Tracker has a Sprint backlog that shows the tasks that are planned to be done.

In all tools, however, the charts are found in other screens, sometimes more than two clicks away, which makes the process of getting an overview of the Sprint's current state time-consuming.

Epic Progress

The progress of the project is usually related to the epic management. Apart from Asana, all the other tools are able to handle epic management by default, usually displaying how many items are missing, and providing a simulation that, based on the team's velocity, when all the items would be completed. Pivotal Tracker has the Epic management that is easier to use. It provides reporting and with very few clicks it is possible to find all the items that belong to a certain epic. Moreover, it provides the opportunity to offer epics in the main screen, here, through a bar chart its progress can be easily understood.

Project Overview

VSTS offers a customizable dashboard with a large variety of widgets that can be placed, virtually anywhere in the screen, enabling users to create their own views with the information that they usually need to refer to. Jira also offers this feature, but users are only able to have one dashboard at a time. These dashboards are usually provided as an empty screen, leaving to the user to choose the with what suits them best.

Pivotal Tracker and Asana lack this feature, with users having to go on different screens to get an overview of the project.

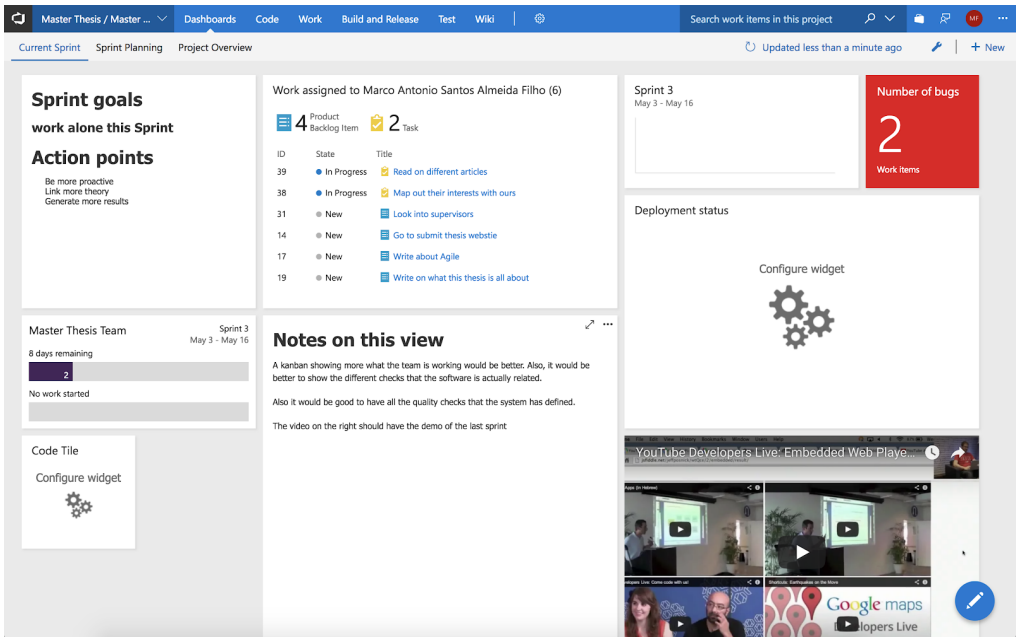


Figure 15. A VSTS custom dashboard.

Planning

Sprint planning was a common issue faced by the organizations. Tools like Jira and VSTS do not offer real-time updates when doing the planning so all participants in the process must also refresh their screens to update the changes. Pivotal Tracker provides the possibility to remove the noise of having too many boards on the screen and focus only on the product backlog and sprint backlog itself. Moreover, Pivotal Tracker provides automatic calculation on which items could be added on the following sprint based on the team's velocity.

Dependencies

Asana is the only tool that does not provide a connection between the different items in the backlog. The interviewees were usually confused on the visual representation used by the tools to reveal that a certain item blocked another. They usually only mark a relationship between them and rely on the task's description to figure out which tasks are the related ones.

Different Roles, Different Perspectives

All of the tools display the same content, independent of the role of the user. There is, however, the opportunity to configure the homepage for VSTS users, setting shortcuts to the most commonly used views. Similarly, Pivotal Tracker and Jira provide users with the feature of managing a personal dashboard to combine perspectives across projects.

Group Awareness

All tools offer visualization on who is doing what in the project. During the interviews, teams praised the Scrum board as a good way to be aware of what their colleagues are doing. This feature is represented in all tools but Asana.

Overall, we have found that the teams, although criticizing some elements on their project management tools, were satisfied with the features that were provided by the tool. We found a confirmation of this in our systematic comparison, as the tools covered many of the requirements mentioned in the interviews. The interviewees, especially the team members, gave immense attention to the fact that the approach to be Agile is through face-to-face communication. It was also found that the tools do not provide a way to handle the stakeholders, although it was commonly mentioned that maintaining a good relationship with them was of utter importance.

4.3. Discussion

We have reached the end of the first milestone of the project. It is time to reflect upon our findings and discuss what we have learned that helped to shape the next step: the design of the prototype. In this section, we discuss the answer to the first clause of the research question, the implications of the results for theory building, and what were the strengths and weaknesses of the research process until this moment.

These interview sessions aimed to respond how are Scrum teams affected by their tools. The word *affected* here stands for the ability to make a difference to the team, be it positively or negatively. First, it was beneficial for our research that all teams were using Scrum. Our initial plan was to investigate Agile teams in general, but all the interviewed teams were Scrum practitioners.

Although the focus was on project management tools, we were open to hearing about the other tools to complement their understanding of their projects. As a result, more than 30 tools were mentioned throughout the interviews, and many different requirements were raised during the conversations.

The most interesting finding was indeed the frequent reference to communication as a way to solve the daily issues, not because of communication itself, as this was expected, but the use of specific language that referred to the act of walking or touching to reach out to their colleagues. This was surprising especially due to the fact of the existence and popularity of chat tools like Slack and HipChat. This confirmed that the teams followed one of the values of the Agile Manifesto: “Individuals and interactions over processes and tools” (Beck et al., 2001). This cultural aspect of the teams can be attributed to the fact that they are mostly co-located.

Offline communication was a central argument whenever the tool seemed to lack the ability for clarification. Michael (personal interview, March 22, 2018) summarized this feeling by stating that the tools can tell the what – did we complete all the planned tasks? – however, it cannot tell the why – was someone sick and that’s why we were not able to successfully end the sprint?

The importance of offline communication in the interviews also changed our approach towards how we communicate on this thesis, bringing to the readers a more relaxed conversation. Although we were already leaning towards either an interpretivist or critical realist approach when considering the philosophy of this research, our way of writing followed a traditionally positivist method: instead of writing ‘us’ we would use ‘the researchers’.

Nevertheless, the most mentioned issue was the lack of stakeholder engagement in the process and the lack of tool support for handling the stakeholders. From the interviews, we understood that the Product Owners came from different backgrounds and the tasks that are not specific to the role of Product Owner are prioritized differently. Moreover, the stakeholders of the teams were different. The teams at Karnov Group and Santander had to handle a substantial alignment with the other departments before releasing a product, whereas at Adapt and Exitable the focus was to establish a transparent relationship between their customers, bringing them closer to the process. At Adapt, they even add their users to Jira, however, some users are reluctant to adhere to the system.

Certainly, there are theories that focus only on the management of stakeholders in projects, however, this is not the focus of this research and the solution built used only what was mentioned in the interviews with the knowledge on Information Visualization.

Stakeholder engagement and offline communication were not mentioned in the related research (Azizyan et al., 2011; Dubakov & Stevens, 2008). Both studies take a perspective on tool usage and needs, but this is done following a quantitative approach, without going deeper in understanding what are the fallback strategies that individuals take when the tools are insufficient. Azizyan et al. (2011) also had its survey with twice the number of low-level managers than developers. We have here stricken a balance between the roles, so they are all considered relevant when designing the prototype.

In the introduction, we stated that even though different roles are connected by the same project management tools, they have different needs to fulfill their daily tasks. Meaning that we started this research with the assumption that different roles need specific perspectives from those tools. While the findings, deducted from the interviews, are not as evident as we expected, they do give the indication that different perspectives are needed to some extent. Stakeholder engagement, for example, is a perspective that is mainly interesting for Product Owners, while goal orientation is mostly mentioned by members of the Development Team. Based on these findings, we decided to design the proposed solution with multiple dashboards, where each display does not serve a single Scrum role but has a focus on one of them.

With that being said, the 20 most mentioned requirements and tools summarized how the interviewed Scrum teams were affected by their tools, and the top 11 served as an inspiration on designing a data visualization tool that would tackle those issues on a different angle.

The research process suffered some challenges at this stage. Establishing a contact with the companies were challenging, especially due to the format of the data collection plan of doing interviews with 3 individuals of the same team. This meant that a total of 3 hours would be used and many of the approached companies were not interested in spending these resources. This would serve as a triangulation mechanism as interviews from different roles in the same company would give us the opportunity of doing a 360-degree analysis on their processes and their tools.

Eventually, we had to accept the fact that we would not be able to reach our goal with the interviews if we did not permit individual interviews. We have also broadened the range of the companies reaching out to other countries than Denmark. Although we do not think that this change influenced the results for answering the this step's research question, we do believe that knowing less of the companies' processes had an effect on the decisions on doing the prototype. We were not able to fully accommodate the processes of those teams in our design.

Another interesting reflection on the conversation is that some participants were a bit resistant to state that they were using Scrum, even though when they talk about the process because they did not believe they were completely following the process. However, when we asked about the elements of the methodology that they used, they mentioned all the roles, artifacts, and events of Scrum. When inquired what were they missing, they were usually mentioning tactics for Scrum, like the Scrum board, and the act of breaking down stories into tasks and assigning hours to them, for example.

During the interviews, we observed irritations by most of the participants towards the tool usage. These irritations often related to actions not possible, or requirements not available. However, after we conducted the casual systematic comparison of the project management tools, it appeared that most of these actions and requirement were available. Hence, it seems to be the case that participants occasionally lack appropriate information to utilize their tools.

The Scrum Guide defines Scrum as a "lightweight, simple to understand, hard to master process framework" (Schwaber, & Sutherland, 2017, p. 3). The hard to master part refers to the fact that Scrum is a continuous improvement framework, where inspection and adaptation of the process are also executed over the iterations. This means that the process will never be perfect. Therefore, we could see that all the teams were using Scrum, although they were at different levels of maturity. Issues were actually commonly related to the roles. In some settings, the differences between the job of a Product Owner and the Scrum Master were unclear, whereas in other situations it seemed that they were completely interchanged, the Scrum Master was the one reporting the budget and prioritizing the issues.

After analyzing the tools being used and the outcome of the interviews, we have also come to the conclusion that the toolkit that the teams have is extensive and it already suffices most of their needs. Our tool, therefore, had to follow a different approach to reach out to those teams. An interviewee warned about the cost of moving between tools, might be too expensive,

especially when a large amount of data has already been produced in one tool. Therefore we relied on the fourth most mentioned category, integration, as the keyword that would define out tool.

Tools like GitHub and Slack were constantly praised during the interviews. Their comprehensive marketplace combined with an easy installation and configuration make a powerful tool that can lead to a creative mash-up of tools that facilitate the whole development process.

Let us consider the following scenario where there is no communication between the tools. First, a developer would have to push his changes to the code repository so it could be shared with the team. Later, he would have to do the same process on another server to run his tests, to make sure that the changes made did not cause any bugs. If all the tests passed, he would be able to push his changes to the production server. After that, he would go to the chat application and notify the team that his changes were successfully deployed on the production server. This is not only time-consuming but it is also an extremely error-prone process.

Now, for an integrated continuously delivery environment. The developer pushes his changes to the code repository. The code repository then triggers an event where the code is sent to a build server that will test the changes. If all the tests pass the build server sends a message to the code repository stating that the tests passed successfully. Now the repository can push the code changes to the deployment server, which in return notifies the whole team that code changes were deployed in the chat application. Compared to the previously mentioned scenario, there was only one action done by the developer, compared to four.

Project management tools can benefit from this tools, providing information about the different used tools in one place and connecting them whenever possible. After learning the importance of integration in software development, we have decided that integration would be part of the core functionality of the tool.

The findings demonstrate goal orientation to be an important requirement as it is mentioned 18 times. However, in our systematic comparison of the project management systems, it was found that none of the tools have a prominent place reserved for such a feature. This is noticeable as the inclusion of this element would be relatively easy to implement, and does not take up much space. For this reason, it was decided to include this functionality in the design of our proposed solution.

Next, we reveal the answer to the second clause of our research question. We describe the process of designing the tool and connect what we have learned so far with the theory of Information Visualization. After running the evaluation step of the prototype we discuss the findings and the research as a whole.

5. How Can Dashboards Aid Scrum Teams?

The second part of our research targets on the feasibility of dashboards to aid Scrum teams in their development process. A chronological structural approach is used to provide an answer to our research question. In line with Design Science, it was decided to create and test a prototype, before creating a fully functioning tool. First, an explanation of the design of the prototype will be given. The prototype is central in the subsequent part of gathering our information.

5.1. Designing the Prototype

The design of our prototype is inspired by three elements discussed in this research so far, (1) the literature, which reveals comprehensive knowledge on dashboard design principles and demonstrates the essential elements of the Scrum methodology, (2) the interviews, showing the relationship between Scrum teams and their tools, and (3) the systematic review of existing project management tools. Each of the three elements contributed to the creation of our proposed solution: a prototype in the form of a customizable dashboard that facilitates Scrum events, named *Mirror Force.* The subsequent section discusses how each of these elements inspired the prototype and explain the design.

5.1.1. Inspiration

The interviews from the previous section showed that the individuals were mostly satisfied with the existing set of tools. Though similar result are not explicitly mentioned in the literature, it can be identified by other works that, instead of trying to replace an entire toolkit, it is best to extend them (Biehl et al., 2007; Jakobsen et al., 2009; Mateescu et al., 2015). Therefore, the placement of our solution is not one of replacing the teams' current tools as this would neglect its value, but instead, it acts like an extension that quickly connects all those tools together.

Our solution places itself as an integrated and customizable solution for Information Visualization of Scrum projects. It allows users to create custom dashboards and place single purpose visual components, called widgets, virtually anywhere on the screen. These widgets are blocks where the source information comes from an external tool, such as a team's project

management, communication, or code configuration tool. The recommendation on how the dashboard should be visually composed is also part of the contribution of this investigation.

The visual placement of the widgets is inspired by another important element from the interviews, the second most often mentioned category: *offline communication.* Based on this knowledge, we aimed to design a dashboard that would aid this type of communication. To reach this objective the proposed solution, which is the main contribution, follows a storyline to be followed during Scrum ceremonies. In total, four views were created to represent three of the Scrum ceremonies: the Daily Scrum, the Sprint Review, and the Sprint Planning. Each of these ceremonies follows a different agenda and the placement of the widgets come from the guidelines of the Scrum methodology and what was mentioned in the interviews.

Next to that, the solution was highly inspired by Microsoft VSTS dashboards, as the custom dashboard represent the core of the final design. This, in combination with the feature of allowing different tools to be integrated, would provide a level of visualization that fits a team's culture. Finally, the prototype follows design heuristics, which are thoroughly described in the remainder of this section.

5.1.2. Visual Design

The prototype was designed with Sketch, a design toolkit that aids in the creation of visual interfaces from mobile applications to websites. There, the dashboard screens were designed as they would look like if they were a real application. Afterward, we uploaded the screens to InVision, a platform for designing workflows. In InVision, we were able to add interactions that would allow users to navigate through the screens as if they were interacting with a real application. The latest version of the working prototype is available in Appendix E.

Daily Scrum Dashboard

Figure 16 presents the visual composition of a dashboard for the Daily Scrum. Since it is a short meeting, usually with an extension of maximum 15 minutes, the level of information is concise. The objective of this meeting is to raise group awareness, therefore, the Scrum Board stands out as a way to display what team members are doing.

On the right side of the screen are other important elements that can be discussed, the Sprint Goal, for instance, stays on the top right side of the screen, as an important reminder of what

the team aims to complete in the current sprint. The burndown chart is positioned close to the Sprint Goal in order to create a visual connection between the two, establishing an opportunity for the team to visually communicate if they are going to reach the goal or not. The build status and the code quality stands for metadata about the current project status. However, it can be anything that represents the current status of the sprint that can have daily, unpredictable changes. A visual representation that matches the integration tool being used is important, for instance, the build status widget in the example figure is integrated with Jenkins, where the status of the build is represented by a weather forecast (sunny represents that there were no problems building the system lately).

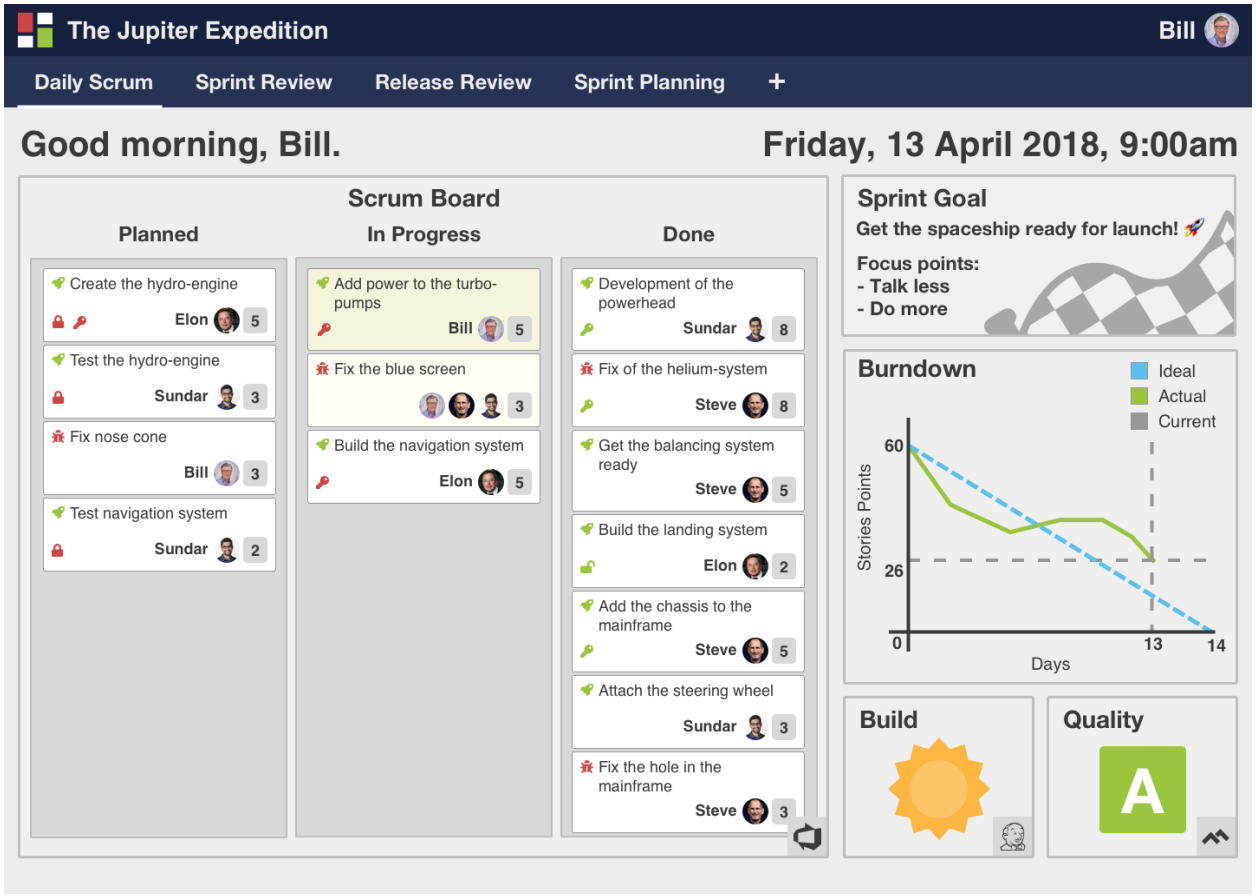


Figure 16. *Mirror Force*: Daily Scrum screen.

This was the first view to be designed and many design heuristics were incorporated here, which served as a guideline for designing the following screens. Using the *Principle on Proximity Compatibility* (Wickens & Carswell, 1995), we surrounded the dashboards under a theme and, within those themes, we grouped the elements together under a block that would represent the

tasks that are supposed to be accomplished in that area. These are represented by the borders around the widgets, facilitating the visualization of the elements.

The *Principle of Salience* (Bertin, 1983; Dent, 1999; Kosslyn, 2006) states that the most important thematic information should be prominent on a display, therefore, the Scrum Board takes the most space on the screen. The same design principle is followed for the creation of the stories that need more attention. These stories, found in the *In Progress* column, mimic the aging process of paper, getting yellower as its development time starts to take longer than expected.

In the same element, the *Principle of Visual* Difference (Kosslyn, 2006) was applied to the Scrum Board. As it can be seen in the *In Progress* column, color differences were used to highlight stories that had been in place for a longer period of time. Multiple shades of beige, inspired by the color changes a piece of paper faces as it gets older, were used to increase awareness of the longer life cycle backlog items might suffer during a Sprint.

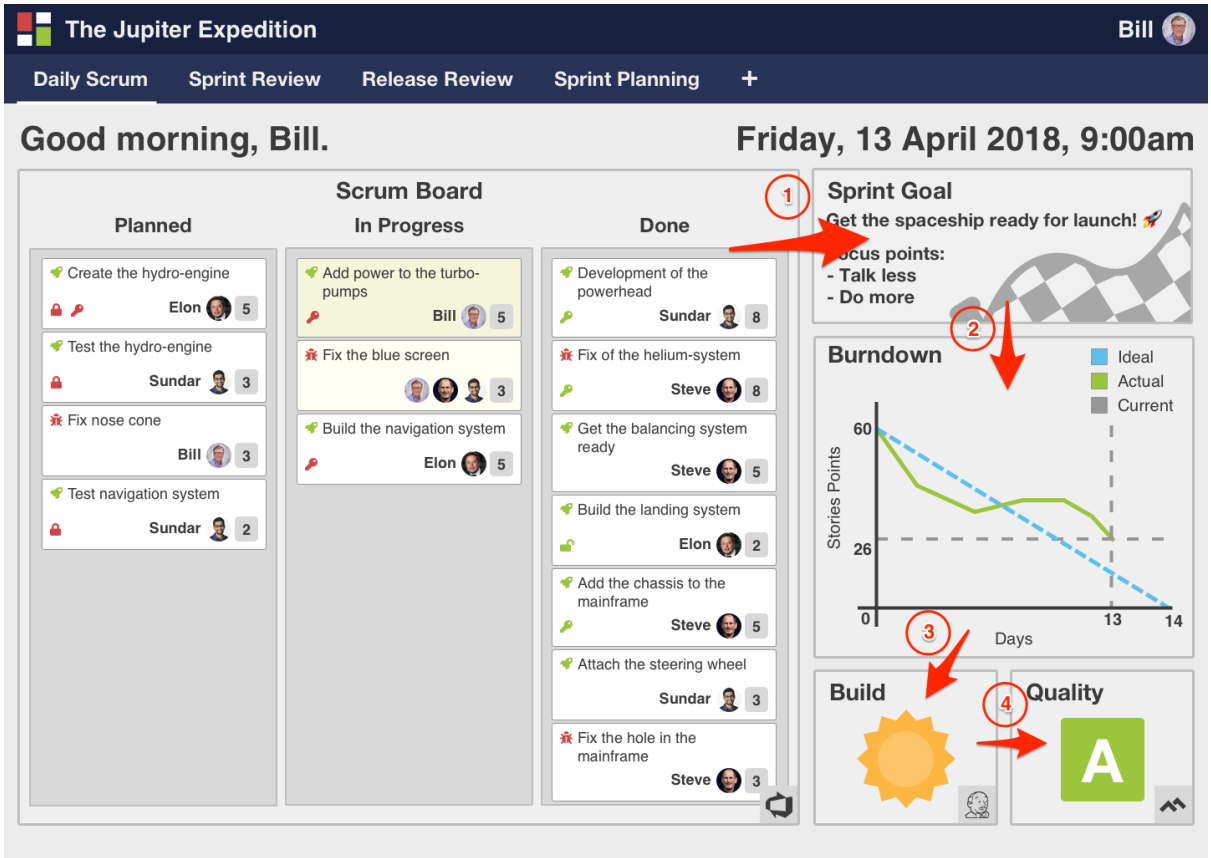


Figure 17. *Mirror Force*: Screen reading flow.

Forsell and Johansson (2010) mention the *Principle of Spatial Organization* as an important design heuristic that aided us on deciding where to place the widgets in such a distribution that would ease the legibility of the screen following an agenda, which was part of the culture of the teams investigated, reading and writing from left to right, top to bottom as discussed in the eye-fixation research by Carpenter and Shah (1998). Figure 17 shows how this principle is applied as a way of reading the Daily Scrum dashboard.

The elements on the dashboard are also part of the daily work of Scrum teams, so the boards, charts, and figures should be easily recognized by them, which follows the *Principle of Appropriate Knowledge* (Kosslyn, 2006). However, teams new to Scrum and the other integrated tools could be confused with the elements being displayed. To make them familiar with the displays, additional knowledge such as overlays and legends are used to provide context to the charts.

Another element that is present in the Daily Scrum is the Sprint dependency resolution, where the tasks that have dependencies have icons in the forms of keys and locks. The tasks that would solve dependencies are tagged with a red key, and when these are done the color of the key would change to green. Tasks that are depending on other tasks to be concluded first are marked with a red closed lock. When that dependent task is solved, then the lock is opened and its color is changed to green.

The heuristics mentioned so far, relate to elements that are visually presented within the design. However, design principles can also impact the design, by excluding visual elements. In our solution, we have left out elements that, after reviewing them, did not contribute to the information represented by the dashboard. This choice links to the *Data-to-ink Ratio*. An example is the decision to exclude a background picture of a spaceship, which was part of the initial solution.

As stated before, the Daily Scrum is a short meeting. For this reason, the *Principle of Minimal Actions* (Forsell & Johansson, 2010) was applied to the design of this dashboard. To ensure this principle, a maximum of one click to show all required was used as a rule of thumb. The *Principle of Minimal Actions* was balanced against the *Principle of Capacity Limitation*. To help this balance, the *Visual Information Seeking Mantra* by Shneiderman (2003) is included in our proposed solution. This heuristic is applied to ensure that the dashboard can provide an overview at first, but allow for zooming in, filters and details on demand whenever this is needed

by the user. An example of this mantra can be seen in Figure 18, which shows the overlay when the user clicks on the story called “fix the blue screen”. Once this overlay is shown, a description is shown, and the user has the possibility to go directly to the integrated systems.

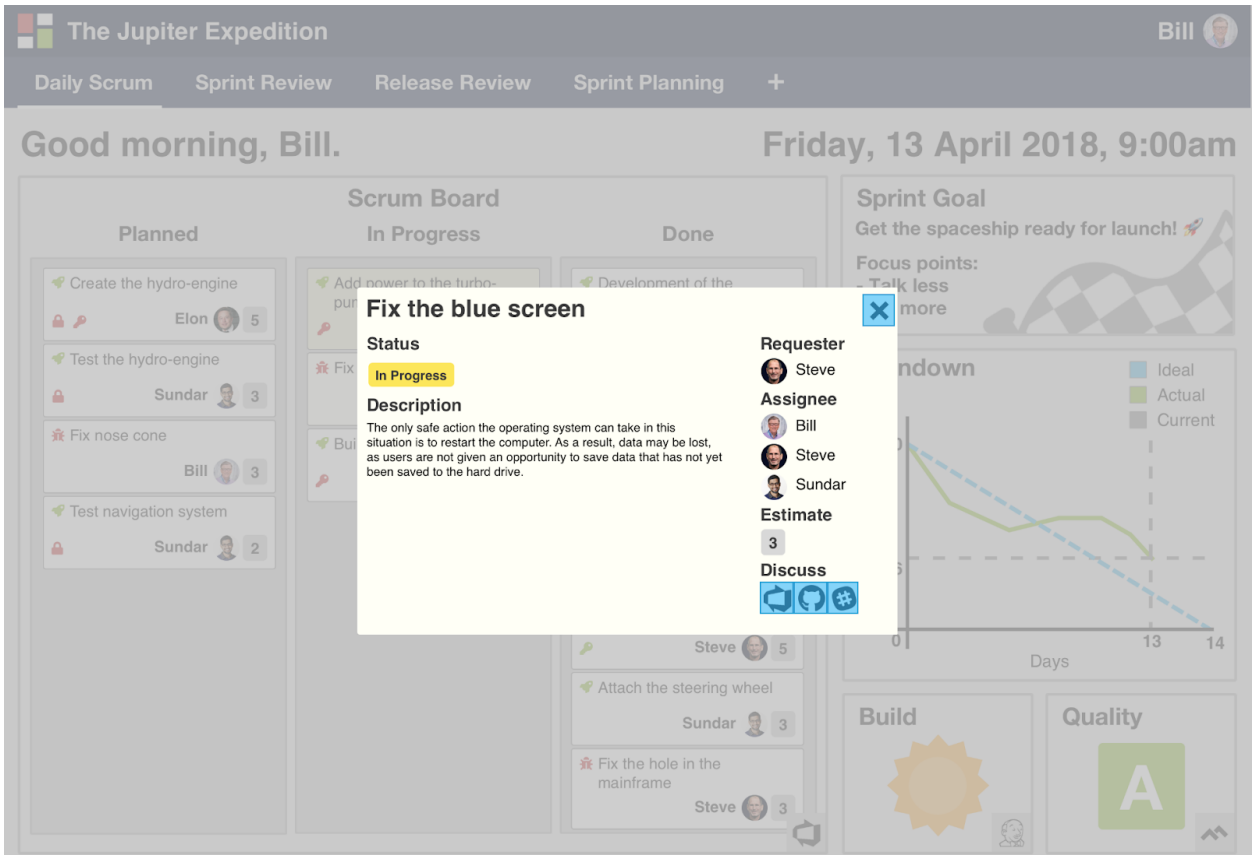


Figure 18. *Mirror Force*: Representation of the Visual Information Seeking Mantra.

Finally, the *Principle of Visual Momentum* (Wickens & Holland, 2000) and the *Principle of Informative Changes* (Kosslyn, 2006) are secured by creating a consistent design over the multiple dashboards. The four displays that were made for this test used the same set of colors, fixed widget sizes, and the same font. Basically, the whole system follows a style guide, so that the users are able to recognize what the visual elements mean, even if no further explanation is given.

Sprint and Release Review Dashboards

The next Scrum event being designed as a dashboard is the Sprint Review. This is a longer meeting that represents not only what has been done in the previous Sprint, but also a review of the timeline of the project. Since it is a longer meeting, recommended to take four hours in a

one-month sprint, this view has been split into the Sprint Review and the Release Review, where the first provides a glance at what happened in the previous sprint, the latter looks into the project as a whole. Figure 19 and Figure 20 present the recommended visualization of the Sprint and Release Review respectively.

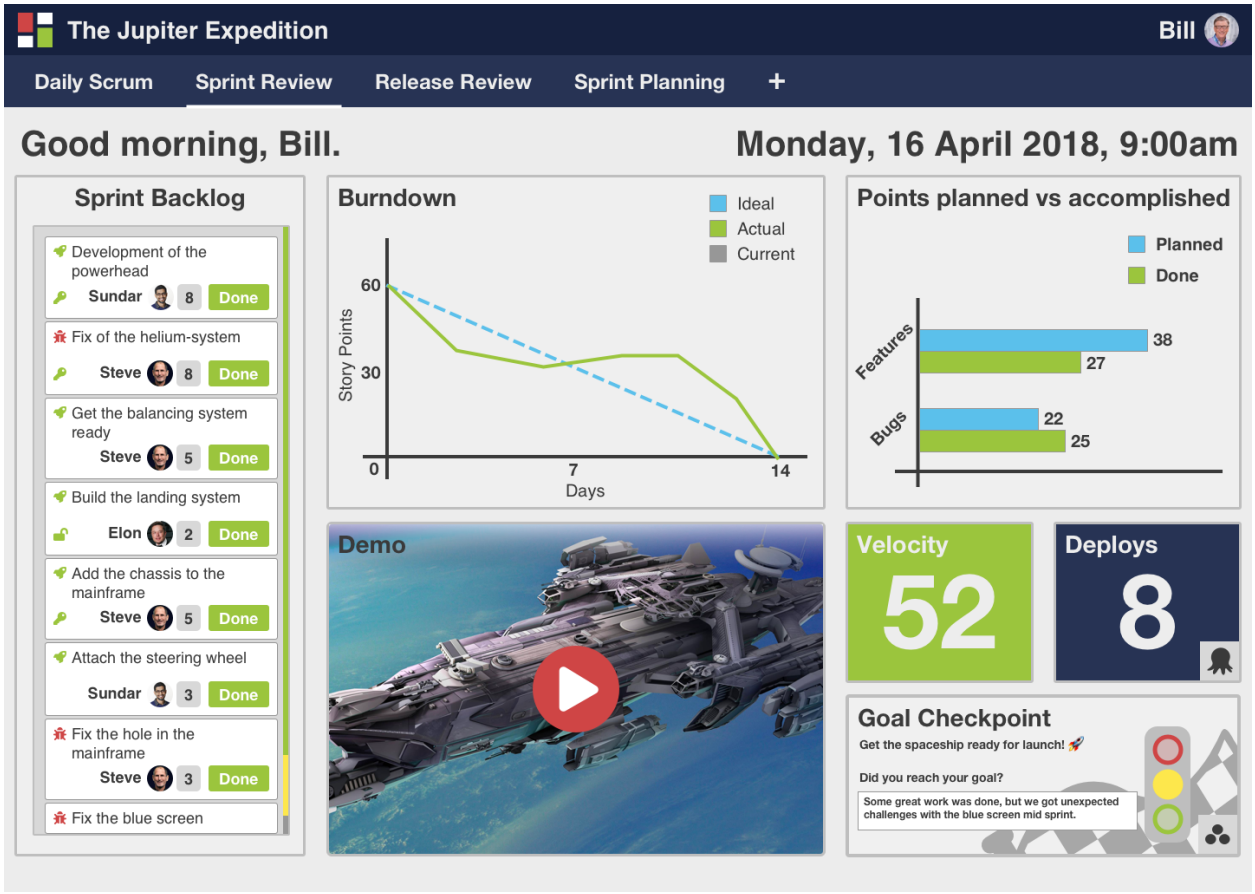


Figure 19. *Mirror Force:* Sprint Review.

The Sprint Review ceremony starts by going through the Sprint backlog, verifying which stories were completed and which ones need to return to the backlog. This is the first element seen in the Sprint Review dashboard and aims to boost the communication on what went right and what went wrong in the Sprint. The “Burndown” and the “Points planned vs accomplished” charts provide an overview of what happened during the sprint, giving a perspective of the unexpected work done and how the team fared in terms of their expected velocity. A demonstration of the work done is provided in the form of a video. It serves as documentation and displays preparation for the meeting, in order to avoid unforeseen issues. Then, some metrics are shown, like the sprint velocity and the number of deployments done during the sprint. These also stand out as metadata about the state of the Sprint and should be used as such. Lastly, the

goal checkpoint is placed on the bottom right side of the screen, as an inspection step to verify if the team has reached its initial goals. When all elements in this view are discussed, users should move on to the Release Review tab.

The release review tab aims to provide a higher level awareness of the project to the team, plotting where the product is on a timeline that broken down by the planned Sprints. The Release Review is where the Product Owner reveals to the team and the stakeholders the overall state of the project. From the investigated project management software, there were no built-in features to really provide information of the releases. It was usually, but easily accessible, that one would need to install additional extensions to build this view.

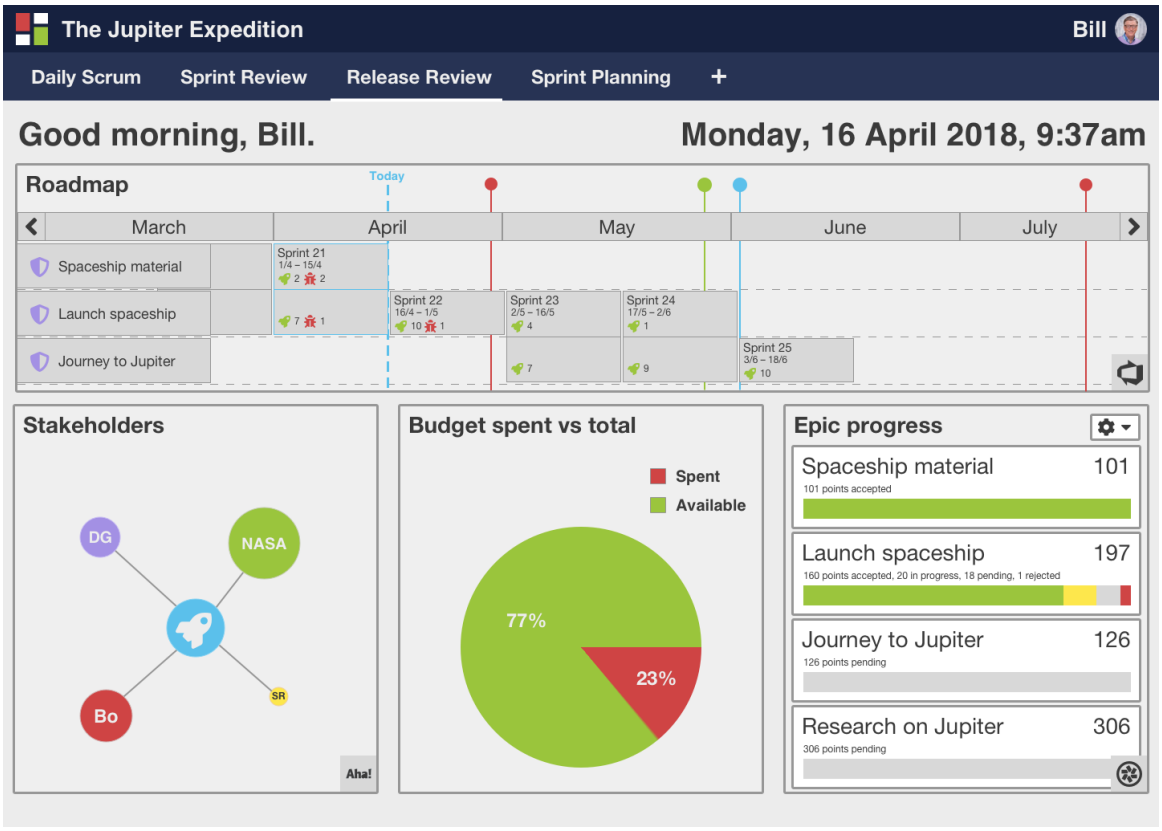


Figure 20. *Mirror Force:* Release Review.

Four widgets were therefore placed in order to support the Release Review meeting. The roadmap, where the product is seen as a series of sprints with an overview of the stories that were completed or were planned to be done. These are also grouped into epics, as a way to see where the focus of the Sprint is being spent at a higher level. This view also connects to the

stakeholder's widget, providing a colored line that represents a milestone that marks an event between the team and the stakeholders, the launch date, for example.

The widgets on the lower part of the screen serve as support information to the upper side of the screen. The stakeholders widget represents a graph of the project stakeholders and their relationship with the project. The bigger the node, the greater the number of tasks needed to be handled with that stakeholder. The budget spent versus total budget is represented in the form of a pie chart, in order to bring transparency to the team. This format was used, as some of the companies used this representation to communicate the budget to stakeholders and the team. Epic progress is shown as the last widget as a way to wrap up the meeting, providing a perspective of how much work is still remaining.

The recommendations on the design of both the Sprint Review and Release Review are related to the principle of visual momentum (Woods, 1984; Wickens & Hollands, 2000). Seeing that, providing all the relevant information in one screen is not possible. These two screens are connected in such a way that would be similar to preparing a slide show, removing the need from the user to query the data and later, take a snapshot of it.

The charts used in the Sprint Review dashboard were in accordance with the recommendations by Few (2004). He states that time series charts (used in the burndown) serve to emphasize an overall pattern, therefore, the *ideal* line draws a perspective of when work is expected to be accomplished, and the *actual* line marks what really happened, creating a fast comparison between what was expected against what really happened. This is also a very familiar chart from Scrum teams. An addendum to that is the crossover line (shown in the burndown chart in the Daily Scrum) to display where the team is at this exact moment.

A ranking chart used to represent the amount of work planned against the amount of work realized is a way to visually represent where is the focus of the team's workload. It also helps teams to identify changes in the plans during a sprint. Other alternatives to replace this chart, is a ranking chart that displays the number of hours expected and the number of hours spent grouped by the type of item (bug or feature).

Sprint Planning Dashboard

The last view, the Sprint Planning is used to prepare the work that is going to be done in the next sprint. These meetings usually start by defining the goal and doing a check on what is the duration of the next Sprint and what would be the available workforce in the following Sprint, also known as team strength. When this is done, the amount of points available is calculated, basically following the simple formula that capacity = velocity * team strength. This would lead to the last part of the Sprint planning, which is filling the Sprint backlog with the items that are prioritized in the Product Backlog. The items here are also displayed in a different way, based on their estimates the items would take more space on the screen. This would give a visual perspective that the items are bigger, therefore enabling users to perceive the size of the stories when choosing then to the next sprint. This would also aid to identify stories that could be broken into smaller stories.

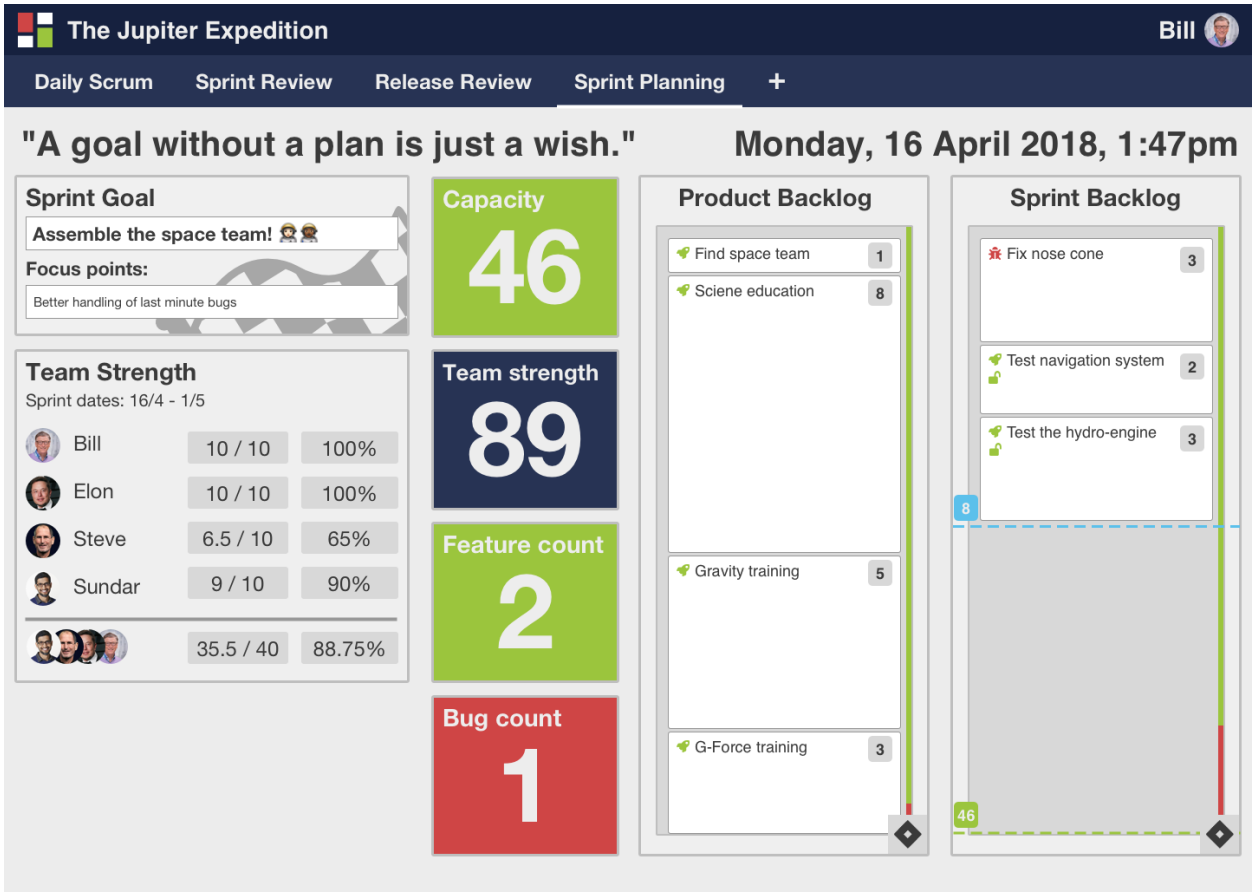


Figure 21. *Mirror Force:* Sprint Planning.

5.1.3. Functional Design

The previous subsection already touched upon some of the functional elements present in *Mirror Force.* This section, however, gives a complete overview of the prototype's functional design. To do so, we use the framework proposed by Storey et al. (2005).

Mirror Force is an integrated, customizable solution that visualizes all essential high-level information for Scrum projects. It includes multiple dashboards, which can be used by all the roles within the Scrum team. However, each dashboard was designed with a specific role in mind. The solution represents information about the recent past, the present, and the (nearby) future, that together should allow for a quick overview.

All four dashboards have a different purpose. However, all of them provide some type of high-level awareness. It shows who is working on what, whether or not the goals will be achieved, and what the plans are. By means of integration, it allows the user to go to information on a deeper level.

A wide range of widgets is used in *Mirror Force.* In relation to the display categories mentioned by Hegarty (2011), all three types are represented. An iconic display, for example, is represented by a profile picture from each of the team members; a relational display is visualized by the burndown chart, as it shows a relation between the story points and the time; the Scrum Board is an illustration of a complex display as it allows for interaction.

Mirror Force is created as a dashboard where simple interaction should be possible. Interactions include drag and drop of stories; hovering on the roadmap for more information; writing and adjusting stories; filtering to allow for different views. Moreover, the idea behind the proposed solution is to offer a high level of customizability, as widgets can be added, removed and placed in different positions. Finally, *Mirror Force* should allow for instant updates. No statement can be made yet about its effectiveness, as it hasn't been evaluated.

5.1.4. Technical Aspects

In order to understand the viability and feasibility of the development of the prototype, one should understand the technical requirements needed to create the product. In the literature, an explanation of the development of mashups was already provided. Now, an explanation on how

mashups relate to the development of *Mirror Force* is given. The technical explanation provides an overall perspective on what technologies are available today, and how they could be used in *Mirror Force*. We don't go deeper to discuss implementation details for the tool. The subjects discussed here encompass the integration and automated updates, important technical aspects considered for the development of the tool.

Integration

In principle, the plans for *Mirror Force* is that is a tool that does not generate any data by itself since all the data shown should be coming from external programs. To facilitate this, integrations with existing tools should be done through application programming interfaces (API). An API is software-to-software communication, which allows developers to use certain technologies to build their own application. In an API a piece of software asks another piece of software to perform a service such as the access to data or execute a function (Orenstein in Nguyen, Kowalczyk & Xhafa, 2013, p. 91).

An API allows a third-party to access a service from another. The power of this can be found in the interoperability and integration of tools since unification is possible regardless of the language used to write the program (Nguyen et al. 2013). Moreover, an API makes development easier as there is no need for a programmer to fully understand a program it is taking its functionalities from (Scheller & Kühn, 2015). All the programs that we have used for the creation of the prototype allow access for API integration.

Consider the following scenario with *Mirror Force*: Initially, the tool is only integrated with Pivotal Tracker and Slack. With the Pivotal Tracker API, we are able to create the Scrum Board based on the status of the items in the Pivotal Tracker backlog. This already shows the power of the APIs, as it is possible to provide a different design than the original one. By combining Pivotal Tracker with Slack through *Mirror Force*, users would be able to have a straight link to discuss an issue that is originally in Pivotal Tracker. This could reduce the time to link these two sources of data.

Through a marketplace, the idea behind the prototype is to allow the user to configure *Mirror Force* based on their current set of tools, in a simple and intuitive way. Ideally, this would allow

users to only integrate with the tools that they are already familiar with and to creatively combine the widgets to facilitate their process.

Automatic Updates

Another key feature is automatic updates. This is needed to increase the usability of the tools, as it was mentioned by one of the interviewees to be an essential functionality for the successful use of a tool, especially for outsourced teams (Morten, personal interview, March 22, 2018). Automatic updates ensure that all users have the same information on the screen in real-time, which prevents miscommunication and confusion. Two technologies are necessary so that Mirror Force can do automatic updates.

An API allows for a connection between tools, however, to continuously update a screen through an API, a technique known as *polling* would have to be used. It means that the original source of data would have to continuously be inquired for new updates. To avoid this costly procedure, a new approach known as webhooks were implemented. A webhook, also called HTTP Callback, allows the original source to push updates into a third party tool if it matches the set criteria (Ruppen, Pasquier, & Hurlimann, 2011). Looking at the same scenario with Pivotal Tracker again, *Mirror Force* could subscribe to updates on the backlog items in Pivotal Tracker through webhooks, so at any given time a task is updated, this information would be pushed to mirror force, allowing it to handle the updates on the Scrum board, for instance.

Nonetheless, only having this push functionality would not be sufficient to do real-time updates, as these changes need to be broadcasted to all users that are accessing *Mirror Force* as well. In other words, webhooks only allow for one way communication, between the source of data and the consumer, the consumer, in this case, is *Mirror Force*. As a final step, a two-way communication is needed between the final user and *Mirror Force* to reflect the changes made throughout the toolkit. Thereby, another technology is needed to solve this issue: websockets. Websockets allows the browser to maintain an asynchronous connection to a server (Wessels, Purvis, Jackson, & Rahman, 2011), in this case, *Mirror Force*. Through websockets, it would be possible to broadcast any changes of data in the original source to all subscribing browsers.

5.2. Data Collection Plan and Analysis

After running the interviews, analyzing the results, and designing a solution, the interviewed companies would be revisited. This time, to test an artifact that summarizes the learnings from the previous data collection step, and its connection with existing literature. The four dashboards were the focal points in this second part of the study. The main objective of this phase is to evaluate the proposed solution.

A survey in the form of a usability test was chosen as the approach towards data collection, as Azizyan et al. (2011) found that the ease of use of a tool is the most important aspect for using a tool. This also relates to the TAM, which states that the intermediate variable ease of use, affects both perceived usefulness and the actual use of the system (Davis, Bagozzi, & Warshaw, 1989). The usability test was followed by a group interview, in which participants could explain their experience in detail. Both the usability test and the group interview acted as a way for us to discover how the proposed solution could be improved, which is leading for future improvements of the tool.

Krueger and Casey (2009) state that three or four group interviews are sufficient, and more interviews lead to data saturation, a point where extra information does not bring additional insights. For our research, however, it was decided to conduct the survey and the group interview with the same companies as the individual interviews, since this served as a way to communicate the possible solution to the users, which we believed was valuable information to them. Including the same companies was a deliberate choice, since it safeguards having participants which are able to contribute to the discussion, and are comfortable in talking to each other. Next to that, it saves time as there is no need to contact other companies (Richardson & Rabiee, 2001).

Though the objective was to conduct this part of the research with the same companies, we were informed beforehand that there was a possibility that other employees could represent the company. A difference in knowledge between those who did, and those who did not participate in the first part of the research, could have an impact on the results. For this reason, a thorough explanation was provided at the beginning of the presentation, to ensure that the participants gathered some knowledge before doing the usability test. The questionnaires were also

conducted when we were present in the organization, another measure to secure the right setting and response rate.

5.2.1. Survey

The test was conducted using the survey research strategy, therefore, a fitting method was chosen: questionnaire. A questionnaire is a data collection technique where participants answer the same set of questions in a predetermined order (Saunders et al., 2016). The questions designed here were, therefore, a reflection of the findings from the first data collection phase. This was done in order to confirm if the designed artifact reduces or even removes the issue that affected them in the first place.

The survey was conducted in the form of a usability test. Usability inspection is either focused on finding usability problems in the design, the severity of the problems, or the overall perception of the entire design (Mack & Montaniz, 1994; Nielsen & Phillips, 1993). This is used to improve the usability of a product (Dumas & Redish, 1999), in our case the prototype. Nielsen (1994) explains that many different methods are suitable for the assessment of an interface. The method we used was mostly related to a *walkthrough*, a detailed procedure to understand the problem-solving process of a user (Nielsen, 1994).

Realization of a high-quality usability test was done by including real users, who complete real tasks, while observations and opinions were recorded (Dumas & Redish, 1999). Real users were included by inviting all who participated in the individual interviews. In order to ensure a high response rate of the questionnaire, the invitation was extended to not only the original interviewees, but the other members of the team as well. The participation of additional team members would also allow us to gather more information from the group interviews.

The activities performed by the participants are inspired by the individual interviews from the first part of this study. In this way, activities are realistic and relevant to the user (Dumas & Redish, 1999). The activities were followed by questions that would grasp the essence of that task:

1. **Stakeholder Engagement**: Which stakeholder are we going to ask for more funds?
2. **Project tracking:** Which story or stories do you think that caused the issues with this sprint?
3. **Integration:** Name one of the services that this tool integrates with.

4. **Project progress:** What was the velocity in the previous sprint?
5. **Project planning:** What would be the ideal number of story points that they could complete in the next sprint?
6. **Goal Oriented:** How likely do you think that you're going to reach this sprint goal?
7. **Dependencies:** Which story depends the most on other stories in the current sprint?
8. **Different roles, different perspectives:** Which screen would help the most with your work?
9. **Project Overview:** When will the spaceship be launched?
10. **Group Awareness:** Which task is Steve doing right now?

Finally, the recording of the test was realized with the support of an online user testing tool named Loop[11]. Although the tool has some usability issues, it contains features such as live statistics, screen recording, and extraction of detailed quantitative data needed to maximize the value of our test. On top of that, participants do not have to switch screens as the tool is in the form of an overlay on the prototype.

In addition to this, we asked the participants to think-out-loud, which is a technique to capture the thoughts of the participants during a test. One advantage described by Rubin and Chisnell (2008) is capturing preference and performance information instantly. However, some participants may find this technique unnatural (Rubin & Chisnell, 2008), therefore, we only recommended, and did not force this technique. If the participants decided not to think out loud, we took notes on their behavior, focusing on cues that revealed difficulties encountered by the participants, during the performance of the activities.

Creating a Task Scenario

With the prepared tasks we followed the recommendations proposed by Dumas and Redish (1999), creating a scenario as a way to present the participants' activities. According to the same authors, a good task scenario is short, told from the users' perspective, unambiguous, gives the participant enough information, and relates to the activities.

We included all these qualifications of a good task scenario by creating a project called *Jupiter Expedition.* The aim of this fictional project is to launch a spaceship to explore Jupiter and it reunites most of the brilliant minds in technology. The project was created to provide context, in the form of a story to the user, that was easy to understand by all participants. Also, the scenario

was important for the activities that the participants had to carry out. The artifacts in *Jupiter* were chosen so they were recognizable and easy for the audience to relate to. Examples include building an engine and fixing the blue screen of death.

To supplement this part with in-depth qualitative data, a group interview was conducted to gather the overall perspective of the final product. However, before starting the group interview, it was decided to share the findings of the individual interviews. To avoid participant bias, it was intentionally decided to not reveal the results until they were done with the usability test as they were related to the activities they would do.

5.2.2. Group Interview

The prototype test was followed by a group interview, in our case, it was a semi-structured interview conducted on a team basis. Saunders et al. (2016) use the term *group interview* to describe all semi-structured and in-depth interviews conducted with two or more participants. Group interviews were chosen since the participants provide rich information (Krueger & Casey, 2009) which is complementary to the data retrieved from the survey. It is rich in the sense that participants can give further explanation of their experience throughout the test. The interview had the following two objectives:

1. How can we improve our prototype?
2. What aspects of the tools do you think are useful?

Contrary to the individual interviews, it was decided not to transcribe the group interviews due to time constraints. This restriction was the consequence of having two sources of data to be collected and analyzed in this phase. With the estimation from Saunders et al. (2016) on interview transcriptions presented in section 4.1, the 5 hours expected of test and interviews would result in 30 to 50 hours of transcribing. Thus, it was decided that at this stage notes would be taken during the group interviews. Capturing essential data from this stage was accomplished by having both of us taking notes.

One possible danger of a group interview is that not all participants contribute to the interview (Saunders et al., 2016). To grasp on the full potential of the group interviews, we made a clear division between roles, one was facilitating the discussion, while the other taking notes and asking for clarifications, such as: *“What was unclear about the Stakeholder widget?”.* By these

means, an attempt was made to balance the contribution of the different participants. Another point of attention, mentioned by the Saunders et al. (2016), was to ensure that all participants understand each other's contribution. This was secured by asking the participants for clarification or summarizing their statement. Another pitfall we tried to avoid was to only receive favorable answers during the group discussion, as we were presenting our own proposed solution. To prevent this, we strongly emphasized that well-argumented critique was more valuable than compliments, in order to improve our prototype and research process.

As it was mentioned, our role during the group interviews was to take notes on the most relevant information mentioned by our participants and facilitate a smooth process. It was the opportunity to explore areas of future research, and pains that were not solved yet by our prototype. To ensure a natural progress, the following interview guide was used:

1. What was your overall impression?
2. Where there elements in the tool that inspired you?
3. Which of these dashboards would have a priority to implement?
    a. Which dashboard has the most positive impact
4. What is information that you might have missed? How can we improve the prototype?
    a. Inform: Dashboard would be customizable
5. What type of information would have the lowest priority for you?
6. Do you think that organizing the dashboard, based on ceremonies, is useful?
7. We have added some visual elements to improve convenience, what was your opinion about it?
8. If you could rearrange the dashboard, how would that be and on which factors would that be based?
9. This was the task that was most time-consuming, why do you think this was the case?
10. Do you have any other comments?

5.2.3. Data Analysis

From the second part of this research, descriptive quantitative and qualitative data was gathered. The descriptive quantitative data was gathered by means of Loop[11]. This tool provides extensive analytical aid, including data on average time spent per questions and number of

screens user clicked before answering, in the case they completed the task. Overall it provided the report that is available in Appendix F.

Next to this, the tool provided survey data, which included measurements on the *ease of use* and the open questions asked to the participants. The data from the open questions did not serve as a means to assess individual participants, but rather as a way for us to relate to the outcome of how easy the components of the prototypes were to use. For example, if the overall outcome from the components was *very easy to use,* the answer given to the open questions served as a way to check this.

To analyze the results from the usability test and assess the ease of use of *Mirror Force*'s components, a scale ranging from -2 (very difficult) to +2 (very easy) was used to represent what were the activities that were most easily done with the prototype. Tasks that have a negative score were considered hard, tasks with a score lower than 7 and equal or greater than 0 were considered neither easy nor difficult, and tasks with a score equal or greater than 7 were considered easy. Table 6 displays the questions and the categories sorted by level of difficulty (from easiest to hardest). The choice of the number seven as a threshold was due to the number of participants, meaning that, to consider a task easy, at least half of the participants found the task easy and the other half considered it neutral. Figure 22 offers an alternative visualization on the degree of difficulty of each of the tasks.

As it was decided beforehand no transcriptions of the group interview would be made, therefore, additional focus was given to the note-taking during this part of the group interviews. Both of us took individual notes on all aspect perceived as relevant. Directly after the first company was visited, notes were compared, differences were discussed, and complementary information was merged, resulting in one document containing all the important information. Next, this information was evaluated to discover overall themes and patterns. After each company visit, this procedure was repeated. By these means, we avoided loss of knowledge and had the opportunity to add or rename categories, based on the new information. After the last company visit, we ended up with the final set of categories. Next, the information was clustered based on the categories, providing us with an overview useful to discuss the most prominent aspects of each category. Components were included in the result if they were mentioned by participants from multiple companies, or if they were strongly emphasized.

5.3 Findings

Altogether, the analysis gave us an understanding of the collected data and highlighted the relevant findings, reflected in this section. Four out of the six companies initially investigated were revisited for the reevaluation of the solution, meaning that one of the original companies did not participate. The members from Exitable were not reachable, and unfortunately, did not participate in the evaluation phase of this research.

In this stage, a total of 14 participants took part in the evaluation process. However, some of the original interviewees were not available in this phase. Santander and Blue Billywig had the same participants, Karnov Group had some additional people with only the Scrum Master missing, and Adapt had a completely new team.

The flow of the evaluation was the same with most companies. First, we presented ourselves, our research process and our solution. At Adapt, the introduction was more comprehensive, since the new participants did not know what the research was about. After the introduction, the solution was presented, but not many details about its features were given. The presentation of the prototype served only to contextualize the tool, and where it would be placed in the current toolkit they already have. Basically, we explained that the prototype would act as an extension of their existing tools, integrating with them, and expanding their features in one visual and customizable dashboard. It was also mentioned that the dashboard starts off with a template that is a recommendation on how to run Scrum events, similar to a meeting agenda.

Later, the usability test was explained. There were not many questions regarding the tools being used and the process went smoothly. We have asked the participants to think out loud while doing the activities, while we wrote notes about their experience with the solution. Lastly, the group interviews were conducted to harbor an overall assessment of the prototype.

The different companies had different postures during the think out loud usability test. While participants from Karnov Group and Blue Billywig actively participate while doing the test, members from Adapt and Santander stayed silent until they were finished with the tasks.

An additional question was ran at the end of the questionnaire in order to identify the Scrum role of the 14 participants. There were four possible options: Team Member, Scrum Master, Product Owner, and other. Only one of the participants replied with other, stating that the role was of a

“frontend developer”. However, frontend developers are also considered as part of the team members in our research, therefore, the responses from this participant were grouped together with the other respondents that stated that they were team members. From the participants, seven were team members, four were Product Owners, and three were Scrum Masters.

5.3.1. Findings Survey

Question	Category	Score
Which task is Steve doing right now?	Group awareness	18
Which screen would help the most with your work?	Different roles, different perspectives	14
How likely do you think that you're going to reach this sprint goal?	Goal oriented	13
When will the spaceship be launched?	Project overview	11
Name one of the services that this tool integrates with.	Integration	7
What was the velocity in the previous sprint?	Epic progress	7
What would be the ideal number of story points that they could complete in the next sprint?	Planning	6
Which story depends the most on other stories in the current sprint?	Dependencies	-3
Which stakeholder are we going to ask for more funds?	Stakeholder engagement	-6
Which story or stories do you think that caused the issues with this sprint?	Sprint tracking	-15

Table 6. Assessment of the questions sorted from easiest to hardest.

Most of the activities were completed in the usability test, only 4% of the activities were abandoned, due to the fact that the participants couldn’t find the elements in question. The activity that had most resignations was related to the stakeholder engagement category, where the participants would have to find which stakeholder would provide the dummy project with

extra funds. Moreover, 79.6% answered wrongly the question associated with this category. It was also classified as the second most difficult task, with only 21.4% stating that accomplishing it was either easy or very easy. One of the participants has also abandoned the tasks that are related to the project progress and dependencies categories.

Six out of the ten tasks were considered easy, while one was seen as neutral and the remaining three were considered difficult. The categories of the tasks considered the most difficult were dependencies, stakeholder engagement, and project tracking. The most difficult task was project tracking, where the users would have to identify which of the story or stories in the Jupiter Expedition sprint had caused the issue in the sprint. When the prototype was designed, we have chosen “Fix the blue screen” as the problematic issue, however, none of the participants reached the same conclusion. Moreover, five participants stated that they couldn’t find what caused the issue.

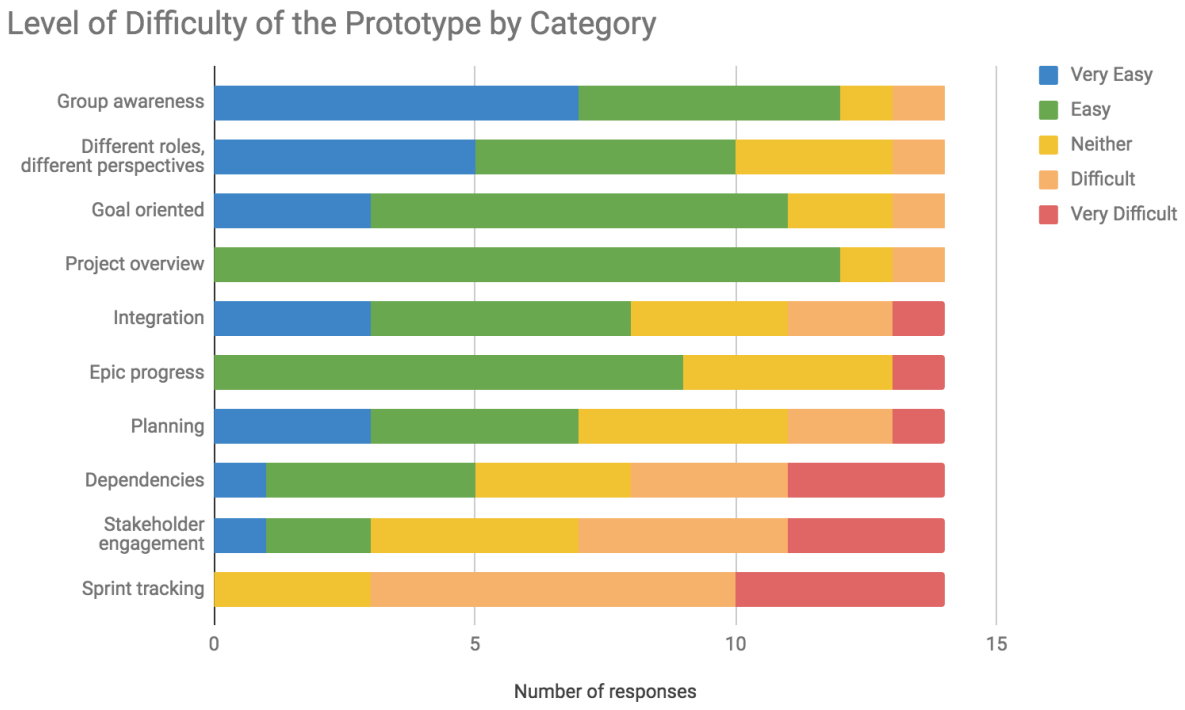


Figure 22. Level of difficulty of the prototype.

Average time spent and page views were also recorded, giving a perspective of which categories were more time-consuming to complete. Table 7 displays what was the average number of page views and time spent to accomplish the tasks in sorted by the order of

occurrence of the task in the questionnaire. The average number of page views done to accomplish a task was 3.02 and the average total time to finish the entire test was of 9.6 minutes.

Category	Average page views	Average time (in seconds)
Stakeholder engagement	3.6	117.8
Sprint tracking	4.6	103.9
Integration	5.6	88.1
Epic progress	2.4	26.9
Planning	3.5	67.4
Goal oriented	1.9	39.7
Dependencies	1.4	41.4
Different roles, different perspectives	2.7	39.9
Project overview	2.9	35.9
Group awareness	1.6	15.9

Table 7. The average number of page views and average time spent to accomplish the tasks for each category in order of occurrence.

Stakeholder engagement was the first task that they were required to do and also the most time-consuming. The activity with the highest number of views is related to integration, where the participants would have to name the tools that can be integrated with our prototype. The integration component is marked on the right bottom corner of the widgets.

The participants were also asked which dashboard they found the most useful. This question was related to the different roles, different perspectives category. The Daily Scrum screen was the option with 76.5% of the responses. Followed by it, the Release Review was the second most useful screen with 17.6%, then the Sprint Planning with 5.8%. The Sprint Review did not receive any votes. When this is broken down into the different roles of the participants all the Scrum Masters responded the Daily Scrum. In addition, 90% of the Development Team also

responded Daily Scrum, with only one participant stating that Sprint Planning was the most useful. Finally, 75% of the Product Owners preferred the Release Review, and the remaining preferred the Daily Scrum. Figure 23 displays these findings in a bar chart.

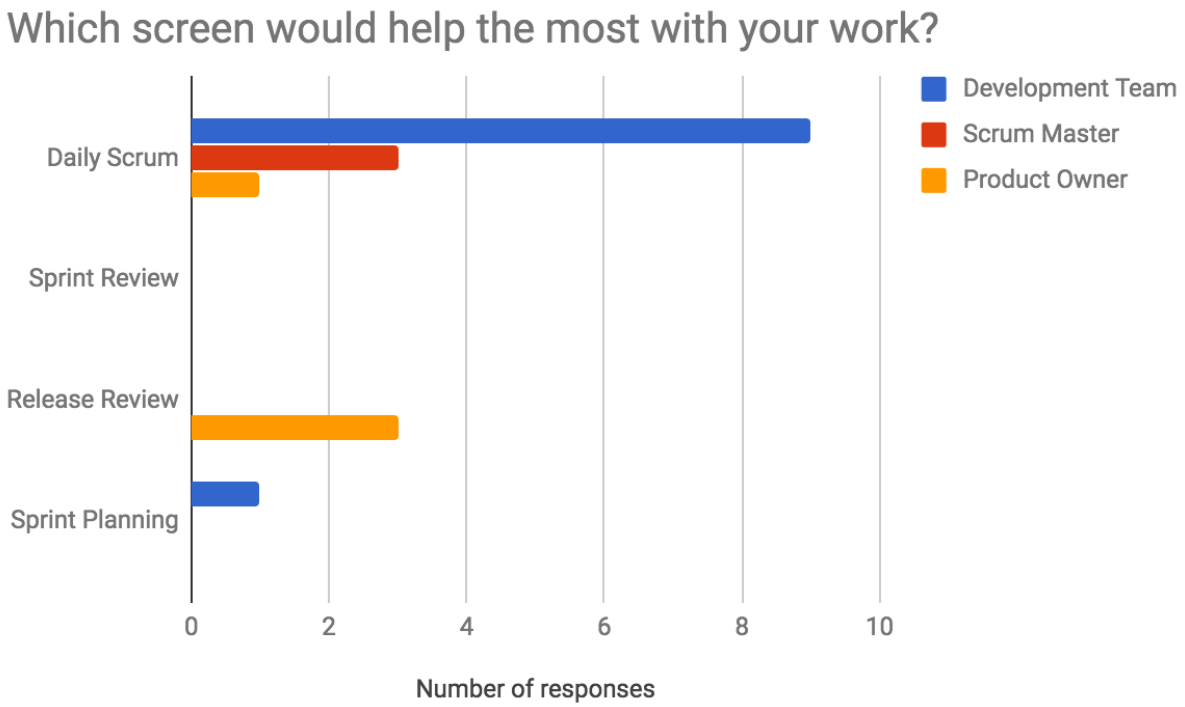


Figure 23. Most helpful screens grouped by role.

5.3.2. Findings Group Interview

Later, the group interviews complemented the findings from the usability test, uncovering the participants' impressions on the prototype, the issues they faced, and what stood out in the tool. These considerations were taken in form of notes by us, which was analyzed and categorized afterward. From our notes, which can be found in Appendix G, the following three high-level categories were deducted from the group interviews:

Category	Definition
Problem	A statement would be categorized as a **problem,** if participants described they were not able to complete a task or encountered any other pressing difficulty during the test, including unclarities and confusion.

Future Improvement	The **future improvement** category relates to two elements: Ideas mentioned to better the prototype for following iterations, and suggestions to advance the conducted research itself.
Positive feedback	**Positive feedback** included the contended perception of the participants towards the prototype, such as the design or the presence of specific widgets.

Table 8. High-level categories from the group interview.

During the group interviews, the participants mentioned 28 times a problem, suggested 58 possibilities for future improvement, and had 28 times positive feedback on *Mirror Force.* The most prominent findings are presented below.

The overall impression of the proposed solution was positive. Participants praised the different approach that the prototype had towards the Scrum events. The suggested structure of the dashboards, formatted in such a way that outlines a meeting agenda for these events, was an approach that was found interesting. It was pointed out that this flow of the dashboards was an intuitive way of storytelling which, by means of face-to-face communication, fit their general needs during a ceremony. Moreover, all the teams were fond of the sprint goals' prominence in most of the screens, especially the Daily Scrum. Though being a simple widget, it reminded the participants what they were doing it for.

The inclusion of stakeholder management was recognized as an important feature. However, at the same time, the Stakeholder widget was the most challenged feature in the prototype. The teams could not understand why the nodes had different sizes, nor why they were plotted as a graph. They stated that it was an interesting and important feature, however, it wasn't a good representation. One team recommended that a simple list displaying the contacts of the different interested parties could suffice their needs.

Overall, the Product Owners from the different teams stated that the tool did not capture the work of a Product Owner yet. The Roadmap widget was commended, however, they would like to see dependencies between sprints and a stronger identification where the team currently is in the project. The Stakeholder widget was commonly mentioned due to confusion on what it was really trying to convey, and the usefulness of the Budget widget was argued in the different teams. Adapt, which offers IT services to its customers in the form of consultancy mentioned

that it was an important feature, whereas the other teams which handle products instead, stated that the budget is not the most relevant thing when doing a review.

Charts were also considered important, with the burndown being the most mentioned. The gray dashed lines helped to identify where exactly in the sprint they were. It was also recommended to use a burnup chart instead of a burndown chart, where the amount of work done would increase over time until it reached its goal. The argument was that this approach sounded more positive. The budget pie chart was criticized, and a different visual approach was suggested. The Product Owner from Blue Billywig mentioned that with the pie chart it is challenging to identify if they are on budget or if they have overspent.

After explaining some of the added visual components like the coloring of a task that had a long lifecycle, and the size of tasks based on their estimates, a general discussion about them was led. Regarding the task lifecycle, where the task gets yellower over time, some participants stated that was an interesting idea, whereas others did not completely understand. They thought they were selected items, instead. The size of the tasks based on their estimates in the Sprint Planning was also considered a good idea, liked mostly by team members, which made easier to identify if the item was really that hard or if it could have it could be re-estimated. However, some concerns were raised on how it would look like if stories had large estimations, for instance, forty. It would take up most of the screen, removing the ability to have an overview of the backlog. It was recommended to have it as an option instead.

The design of the tool was also part of the conversation and also received some feedback from the participants. Participants stated that the borders between the widgets were too thick and that the bars, which offer an idea of the type of work that is being planned seen in the widgets Product and Sprint backlog in the Sprint Planning dashboard, look like scroll bars and might confuse the user. Moreover, participant, in particular Product Owners, stated a visualization in the roadmap was needed to represent where you currently are.

Another element which raised a discussion was the dependencies between tasks. Some of the participants stated that they were confused with the concept of locks and keys. They mentioned that they initially thought that the color of the keys and locks represented a relationship between them, not that when the dependency was unlocked, it meant that it became green. Members from Karnov Group, however, thought that the solution was actually nicer than what they currently had with Pivotal Tracker. Moreover, it was argued that on a higher level, *Mirror Force*

could improve its dependencies, by visualizing a relation between sprints on the roadmap and even between projects. The latter is not possible with our solution right now. However, having the option to see multiple projects simultaneously was missed by some participants, especially design agencies, who run multiple projects at the same time.

Teams and team members disagreed in terms of the need of a personal dashboard. They said that one view was missing which was related to user's personal view, a sort of shortcut for them to see the tasks that were assigned to them. Other teams perceived the tool as a means to communicate as a group and did not see the need of personal dashboard. Another disagreement came from the representation of metadata, while some argued that it did not add any value, others stated they found the information helpful.

One of Santander Scrum Masters said that the solution would be unusable if the ability to add comments and updating items wouldn't be available from the tool. It is common in their meetings that they have to do minor updates in the tasks, and having to go back and forth between different tools can be a hindrance. Another improvement, mentioned by several teams, was to have the solution in the form of a big touch screen, which would allow for interaction. In this way, one screen only has to be used. Finally, to improve *Mirror Force*, it was argued to provide more information, such as legends, to improve the ease of use, and the use of drop-down menus to show additional information when needed.

5.4. Discussion

After identifying the 11 categories that affect Scrum teams, designing a prototype, and conducting a usability test to verify the strengths and weaknesses of the prototype, we reached the stage where we look into the latest findings and reflect upon them. In this section, we ponder over our entire research process, providing answers to our research question and wondering on which other roads this research could have taken.

5.4.1. Test Process

The dashboard was designed based on the data from the individual interviews and combined with design guidelines. From the test that was carried out with the dashboard, it was possible to evaluate its usability, in which each display represents a Scrum ceremony. The process followed during the prototype test was consistent among all participants. The first learning for following

iterations, in relation to this process, is to allow all the participants to get to know the tool before getting into the test. By first getting familiar with the tool, the participants are able to have a general understanding of how it works, which has a higher probability to produce more reliable data.

The representative group of participants deviated from what we expected beforehand. Blue Billywig showed up with only one person. This was out of our power and, due to the lack of time, we would either skip the test or do it with one person. Since we consider every input of new information valuable, we did the test with Blue Billywig, although we had to adapt our test. We have, therefore, to properly apply the think out loud method, by which we could capture the thoughts of our participant.

Adapt, on the other hand, turned up with a completely new group, with no insight about our research. To avoid that this difference in knowledge would affect the results, a more thorough explanation was given to them and they had the opportunity to ask questions to solve unclarities. Finally, we observed a different ambiance between the companies during the tests. While all participants at Karnov Group were very talkative, other companies were more quiet during the test. It was noticed that Karnov Group provided richer data. Hence, lessons learned for subsequent iterations is that we aim to have more consistency in terms of a group size, with a minimum of 3, the same participants throughout the process, and encourage a talkative environment. This should ensure more fairness between the data retrieved from different participants.

5.4.2. Proposed Solution

The design of *Mirror Force,* a Scrum ceremony based dashboard, includes widgets that can give an answer to questions such as *who is doing what* and *which bottlenecks do we have*. Additionally, the prototype was created as a tool that facilitates communication among team members and coordination of tasks in Scrum ceremonies. All these elements contribute to *Mirror Force* as potentially being a tool to gain and maintain high-level awareness.

The general perception of our proposed solution was positive. The participants had the same perception towards the dashboard as a means to facilitate face-to-face communication. This finding is in line with our expectations, as it was found that offline communication was one of the most important aspects affecting Scrum teams in their tool usage. The theory of Agile support

this, as it stresses the importance of human communication in its key principles (Cockburn, 2002).

An argument for this positive perception is that the aim of the dashboard is to provide all the necessary information needed at a certain point throughout the process, at a glance. In general, this structure and specific combination of elements were perceived as intuitive and helpful.

A second explanation is that the design of the dashboards is strongly connected to the core of the process used by our participants, as the design is based on the existing Scrum ceremonies. This can be explained by the fact that the participants were already familiar with ceremonies that we used, and the related artifacts. This conclusion can be derived from the information of the test, combined with the discussion of the first part of the interview. Even though the use of Scrum differs across the teams we interviewed, a core set of elements from the method, including ceremonies as Sprint Planning and Daily Scrum, were shared by all of the participants.

After the tests were conducted, one of the participating companies proactively asked us to share our dashboard and the related findings, as they wanted to see if the dashboard could be implemented in their real-life setting. For us, this was a confirmation of the overall positive attitude on the usability of *Mirror Force.* However, it is not possible yet to say something about the value of the proposed solution. For this, a working solution with real data would be needed.

5.4.3. Dashboard Usability

The findings of our study suggest that the participants found the Daily Scrum dashboard the most useful overall, followed by the Release Review and the Sprint Planning dashboard. None of the participants declared the Sprint Review as most useful. This information does not mean that the Sprint Review dashboard is perceived as useless but solely suggests that the other dashboards contain more relevant information. This information is pertinent as our objective is to provide a dashboard that visualizes the most relevant information. Thereby, for future iterations on the prototype, the focus on developing the features of Mirror Force should be either on the daily operations, represented by the Daily Scrum dashboard, or the Release Review.

Similar views with the same goal as the Daily Scrum are already available throughout many studies (Biehl, 2007; Jakobsen et al., 2008; Mateescu et al., 2015). However, there is very little research on creating visual elements to establish a connection between the technology

department and the rest of the business. Thus, an interesting and relevant approach for future iterations is to work on the Release Review, a screen that is more focused on the Product Owner than the Development Team.

When comparing our results with the Scrum Guide (Schwaber, & Sutherland, 2017), it is noteworthy that an important ceremony as Sprint Review is not mentioned as the most important dashboard by any of the participants. One explanation for the low score of the Sprint Review dashboard comes from Adapt. Here, participants stated clearly that the presented information was confusing as they did not understand the meaning of some of the widgets. Next to that, the given information was not in line with their own process. This draws back on the Principle of Compatibility (Kosslyn, 2006). The elements provided in the form of widget were not compatible with the elements that they are used to see.

On the other hand, the Daily Scrum dashboard was the most useful as it was expected. First, this board is designed with the intention to use it every day, as opposed to all other dashboards. Second, the Daily Scrum dashboard includes a Burndown and Scrum Board, two widely used visualizations in the process of the participants. Another clarification comes from the number of participants and their roles. Almost 60% of our participants were part of the Development Team, employees that mostly care about the daily operations and the product to be built (Sutherland & Schwaber, 2007).

The fact that most of the Development Team valued the Daily Scrum dashboard, where most of the Product Owners found the Release Review mostly useful, partially prove our initial assumption that different roles have different perspectives. We expected that the Scrum Masters would prefer the Sprint Review or the Sprint Planning, nevertheless, they opted for the Daily Scrum. We believe that, since the Scrum Masters are also involved in the daily operation, they were leaning towards the Daily Scrum more than the expected dashboards, which are not often used.

Overall, 6 out of the 10 activities performed during the test were perceived as easy to use. The easiest activity was the group awareness activity, with a score of 18. Being perceived as most easy can be argued in two ways. The first argument relates to it being the last activity to perform. Overall, the amount of time spent per question decreased as the test continued and the participants got familiar with the prototype. Where the first two questions took more than 100 seconds on average to complete, the last three questions were all under 40 seconds and the

last one even in 15.9 seconds. This trend can be seen in the average time spent shown in Table 7. Hereby, the Scrum Board and its tasks can be argued as a successful element of the prototype, being able to show information at a glance. Nevertheless, this comes as no surprise since the board is a familiar tool for these teams. This is the second argument, the application of the *Principle of Appropriate Knowledge* (Kosslyn, 2006). For this reason, they had a sufficient amount of knowledge to quickly solve this issue.

Project overview was ranked fourth in terms of easiness of use, with a score of +11. The notes from the group interview showed that the widget related to this category, the roadmap, was a very useful view, especially for the Product Owners. This positive perception indicates the relevance of this element. However, improvements were desired, as the prototype does not show where the team is in the timeline and relations between stories should become obvious. The importance of dependencies is confirmed by our data, emphasized in the next part.

On the other hand, three out of ten activities were perceived as difficult. The Technology Acceptance Model proves that the ease of use partly explains the acceptance of an information system. Since one of the objectives of the evaluation of *Mirror Force* was to identify areas of improvements, it is interesting to understand why these activities were perceived as not easy to use. With a score of -3, the task related to **dependencies** is the highest ranked activity that is not easy to use. To visualize this, we created locks and keys that should represent dependencies. However, our findings show that this was not always understood. Hence, it can be argued that the proposed visualization did not meet its objective as the participants did not understand it, and, after an explanation, did not see it as the best possible way to represent dependencies.

The same reasoning can be applied to the **stakeholder** activity, which was ranked prior to last in terms of easiness of use. Here, participants did not understand the visual representation. The low score for both can be explained by the *Principle of Compatibility*, which suggests that natural mappings should be used between the visual representation and its objective. Therefore, in a following iteration, a better visual representation should be provided to improve the ease of use for both activities.

Moreover, the *Principle of Appropriate Knowledge*, which states that user needs the necessary information to extract and interpret relevant information (Kosslyn, 2006), also had an impact in these widgets. The participants were not familiar with these visual representations and did not

receive any guidance to use the tool. Reflecting upon our decision on how the design of the Stakeholder widget should behave, we deliberately not provided additional information, as we believed that the visual features of the nodes would become clear as they got used with the component. However, none of the participants gave the right answer to the question related to this element. Though the participants did not always understand the newly implemented elements, their attitude was rather positive after an explanation was provided. As a result, the usability test has shown the importance of the aforementioned design heuristic.

Sprint tracking was the activity with the lowest score for ease of use. Contrary to other the other activities, this score cannot be explained by means of one of the heuristics. In this activity, participants had to identify which of the story or stories in the Jupiter Expedition sprint had caused the issue. After reflecting upon this question, we realized that the answer was open to personal interpretation. This is also reflected in the answers given as none of the participant mentioned *fix the blue screen* as the story. However, one of the participants provided a good explanation for this task, that is related to the use of face-to-face communication. She stated that she would be able to know if she were part of the team, meaning that she could ask her colleagues to find out, which is more in line with what we expected from the tool.

5.4.4. Design Heuristics

During the group interviews, the design of the proposed solution was discussed in detail. Based on this information, we gave an indication how well the design principles were implemented in *Mirror Force* and the general importance of the heuristics.

The group interviews formed a solid foundation for the discussions. First, participants discussed the need for a personal dashboard and the relevance of individual widgets. Though not represented in the prototype, *Mirror Force* is intended as a configurable solution, meaning that widgets and dashboards could be adapted to the personal preference of its user. However, the fact that this was not the case, resulted in a discussion where no concurrence was found, indicating the relevance of the *Principle of Preference* (Wilson & Zigus, 1999) and the statement of Hegarty (2011) that no single display is best suited for all purposes. A practical implication, based on this finding, is that dashboard designers should allow for configurability of the solution as the tool should be tailored to fit the team's process, not vice versa. From the reviewed project management tools in this study, only Microsoft VSTS would comply with this implication.

There was a positive impression of the dashboards based on the ceremonies. It was pointed out that this flow of the dashboards was an intuitive way of storytelling. One argumentation for this finding comes from the *Proximity Compatibility Principle* (Wickens & Carswell, 1995), which state that integrated activities need high proximity displays. Scrum ceremonies, such as Sprint Planning, form an integrated activity as they have the need to combine multiple widgets for an answer.

Interesting were the findings regarding stakeholder management. Though the participants understood the value of its inclusion, the representation of the widget was never fully understood, hence, it was experienced as one of the most challenging elements. This visual confusion can be argued from *Principle of Compatibility* (Kosslyn, 2006), which states that natural mappings should be used between display and meaning. Since multiple participants were confused about the bubble representation in this widget, it can be concluded that the mapping of display and meaning can be improved. Once the meaning of the increased bubble size was explained to the participants, it was understood and found logic, confirming the *Principle of Dimension Representation* (Bertin, 1983).

The inclusion of a pie chart to represent the budget led to noisy discussion. Though inspiration for its inclusion came from what the participants already had been using, based on data from the interviews, its incorporation was strongly disapproved. This rejection is in line with Few's (2010) reasoning that circles should not be used to visualize quantitative data, as it is a mismatch between form and data, which, in turn, relates to the *Principle of Compatibility* (Kosslyn, 2006). From this, the valuable lesson is learned that better representations for budget exist. An alternative, mentioned by one of the participants, is a line chart with time included.

Furthermore, all group interviews confirmed the need for more information increase usability of *Mirror Force,* including the need for legends in graphs and an explanation of new visual elements. An important lesson for future iterations is the relevance of providing more in-depth information about the tool. Moreover, these findings imply the relevance of the *Principle of Appropriate Knowledge* (Kosslyn, 2006) as fundamental for the usability of a system. Another confirmation of this principle is the positive perception of the Burndown Chart. Though the individual interview confirmed that this artifact was used by all of the participants, it was mentioned as being helpful due to the added grid lines.

Another helpful element was the presence of the Sprint Goal in the Daily Scrum, Sprint Review, and Sprint Planning dashboard. It was argued that this widget helped them to reflect upon the objective of the Sprint, indicating the pertinence of the *Principle of Salience.* In fact, this principle was not fully implemented for the Sprint goal, but the goal is more prominent than the investigated tools, thus, the participants noticed and appreciated its visibility.

During the test and the group interviews, the participants expressed the desire to click around in different dashboard, once they understood the general story. Reflecting upon this, there is an indication that, with this type of dashboard, Shneiderman's (2003) *Information Seeking Mantra* is relevant. A lesson learned for a following iteration is to provide more options to zoom and filter.

Finally, we have also received feedback on the visual elements included in the proposed solution, for example, the size of the borders was too big, the colors of the stories in the Daily Scrum dashboard, and the size of the stories in the Sprint Planning dashboard. Feedback in relation to the first two focused on dropping the thick lines around stories and increasing the color difference of the older becoming stories. Thereby, these two ratify the importance of the *Gestalt Principles* (Tversky et al., 2002) and the *Principle of Discriminability* (Kosslyn, 2006). The data on story sizes indicate the importance of *Data-to-Ink Ratio* since the idea was nice, but it took too much space without adding enough additional information.

5.4.5. Future Iterations

The main objective of this study was to create a dashboard that would aid Scrum teams in their process. To reach this objective, it was important to discover elements that could be improved. These improvements can serve both academics, as inspiration for future research, and practitioners, as lessons on how to develop a dashboard for Scrum teams. During the group interviews, many improvements were suggested by the participants. An attempt was made to cluster them, and thereby show which improvements were most relevant and why.

A first improvement, resolved from the data, is the possibility to perform interactions. First by performing simple actions such as a drag-and-drop on the Scrum Board, or adding comments to stories. All the participants of Santander on agreed this to be prerequisite to using the tool in daily life. If this tool is intended to serve as a way to facilitate effortless face-to-face communication, it should be possible to make adjustments without the need to switch between

tools. This finding is in line with the Principle of Minimal Actions. Second, by transforming the proposed solution into an interactive touch screen.

Next to that, the results show room for improvement in relation to the dependencies projected in this tool. Though visualization was used to indicate dependencies between stories, the participants pointed out that dependencies between different Sprints on the roadmap and dependencies with external teams would improve the usability of *Mirror Force*.

Another important improvement, and a lesson for the development of any comparable tool in the future is the ability to make configurations by its users. Despite mentioning this tool should be able to do so, all participants stated they desired to make some adjustments in the tool, varying from dropping a specific widget, to adding a complete personal dashboard. The importance of configurations can be explained by the theory of Scrum, which states that each that Scrum is a framework that should be adapted to the specific needs of a team. Reflecting upon the systematic review of the existing tools, only Visual Team Studio Service from Microsoft can be classified as a highly configurable tool.

The final manner to improve *Mirror Force* in future iterations is by adding essential missing information. The data shows that participants had a need for more information in relation to making fast and well informed decisions. Examples include number of points for each Sprint on the roadmap, and being able to understand how much workload each expertise has in the Sprint Planning dashboard.

5.4.6. General Discussion

Reflecting upon the overall process, it can be stated that finding the right balance was a difficulty throughout this research. A balance that was needed to achieve the main objective of this research: the evaluation of the prototype. To get there, two equally important research questions were used, wherein the second builds on top of the first. It could have been decided to focus only on the first research question, implying that we would have more time, and thus a more thorough analysis of the data, resulting in an even deeper understanding of how Scrum teams are affected by their tools. However, this would not have provided us with insights in line with our objective.

Arguing the other way around, we could also have chosen to spend very little time on the interviews and aimed for (an) additional iteration(s) as a means to further develop the dashboard. However, without a proper understanding of the requirements and constraints of the tools, it would be difficult to determine the benefits for Scrum teams. Therefore, it was chosen to balance the two parts of the research.

This also meant we had to make tough choices as time was limited. Understanding the users' perspective was leading in these decisions. Examples include not analyzing our data as thorough as desired in the first part, and the lack of category evaluation in the second part of the research. The latter can be argued, however, from the perspective of Design Science as we found it more important to focus on improving the existing prototype, than double check the 11 aforementioned categories. Next to that, our population in the survey was too small, therefore, we decided to discuss the categories during the group interviews. This relates to the final argumentation, that our objective was to have a comprehensive discussion to discover the overall opinion of the participants.

During this research, we discovered that the existing set of tools is extensive and most of them fulfill their objectives to a certain extent. Therefore, the purpose of *Mirror Force* is not to replace the toolkits from its users, instead, it acts as an extension of the tools already in use. It augments the way data is visualized in the original sources by integrating it with the other available tools.

6. Conclusion

Before departing on this research journey, we had only one belief: that within Agile teams, different roles needed different perspectives from their tooling. Basically, it means that the systems that are shared by a team should provide different forms of visualization depending on the role of the user. To confirm this assumption, we defined two milestones that mark the research question of this investigation:

How are Scrum teams affected by their tools and how can dashboards aid those teams?

First, we set out to explore the Agile environment. We studied the Agile practices of six different organizations, the specific tasks the different roles realize, their tools, and the strengths and weaknesses of their tools. With a focus on project management systems, we have identified **19 classes of requirements and constraints** that characterize the tools used to support the work of the team members. At that stage, we have learned the importance of **stakeholder engagement** for the Product Owners and the influence that **human interaction** has on the Development Team.

Additionally, we had confirmed indeed that the **different roles, needed different perspectives**. However, the number of references to this sort of statements was not as high as we have initially expected. In fact, the mentions on this category during the interviews were barely enough for it to be considered relevant on the next stage of the research.

A final step was taken prior to navigating towards our second milestone. A casual **systematic review** was done with the project management systems used by the different teams in order to pinpoint the visual elements that the tools provide to solve the issues mentioned by the interviewees. After doing this review, we were able to see that, occasionally, what was said during the interviews about the limitations of the tools was not accurate. Moreover, the systems were able to handle most of the issues mentioned during the interviews.

After gathering, analyzing, and reflecting upon the data, we were able to identify breaches where a new solution could be placed. These **goal-oriented** teams needed a swift way to show the current state of the project in order to enable and boost awareness. We have, therefore,

decided to create a dashboard-based solution that would reflect the current state of the team, a mirror per se. That marked the birth of ***Mirror Force***.

To carefully design our solution we have searched for inspiration on the theory of Information Visualization. As a way to improve human cognition, it provided the foundation for the tool to reach the objective of being a quick method for retrieving and understanding data. Design heuristics and the Dashboard theory allowed the design to be consistent with already proven methods. Moreover, it established the best practices to select which information to show and when.

Thus, to aid Scrum teams, *Mirror Force* turned into a dashboard that provides an **integrated timeline for the Scrum ceremonies**. With a set of widgets placed on four different screens, we tried to assemble the learnings of this research to further test them out. To go along with our original assumption, the screens were designed with a specific role in mind.

One more time, we set out to revisit the companies, present our results, and evaluate the prototype. The usability tests and the group interviews gave us the opportunity of reflecting upon the design of our solution. Although it was a commended solution overall, we were able to see that we have failed to fully grasp what consists the work of a Product Owner.

The evaluation phase provided us with the opportunity to see that the Product Owners indeed needed different perspectives from the system when compared with the rest of the team. Since the Product Owners have a strategic position, they require a long-term overview.

Another implication of the evaluation is the exigence of filterability of the widgets to address the issues that they have. It reflects that the user needs might change over time and they require easy adaptability from the software side.

Our core contribution to this research was given in the form of a prototype. It summarizes our findings and proposes an alternative approach for the maintenance of awareness in Scrum teams. Within the prototype, we have the layout of the dashboards, which are connected with the Design Heuristics and verified by the participants.

Overall, the research journey was as consistent as it could be and provided valuable insights. Nevertheless, future iterations could be more strict in regards to the data collection, by establishing a closer relationship with the participant companies and only negotiate a more

active involvement in the process. Moreover, it should be assured that all interviewees would participate in the usability test and that a conversational environment is created.

6.1. Future Research

The natural path to follow with this investigation is to produce more iterations, developing and testing the tool in such a way that new and more powerful features can be implemented. This would draw a connection with the aforementioned theory on dashboards for Agile processes. Nonetheless, in order to provide an academic contribution to the subject, the approach should focus on the issues found in the evaluation of *Mirror Force*, which was not yet mentioned in the related research. Another academic addition can be made by conducting the research within a different context. Other Agile methods will include different artifacts and processes, which can lead to different needs from a solution.

The concept of *ease of use* was tested during the usability test. Opportunities for future research are found in testing different concepts, such as perceived usefulness. To strengthen the test, a working version, instead of a prototype, of *Mirror Force* should be tested. This would allow to include real data in the tool and thereby provide strengthen the test.

Our recommendation is to reevaluate the Release Review dashboard. It is the display that glues the rest of the business with the development team, but the resulting screen has not yet fulfilled its goal. Thus, a study that, through visual elements, is able to express the work that a Product Owner does, translating the business language to the development language and vice-versa, would have an impact not only in Agile Software Development but also Information Visualization.

Even though it was a secondary contribution, identifying the relationship of the individuals with their tools was a very important step that supported most of our research. Indeed, this step of the study as it is could be transferred to different settings where investigators could use the same categories identified here in a corroboration phase. However, extending this aspect of the research to a global setting would also provide a guideline to many different investigations on the usage of tools in Agile software development.

In fact, the findings from such a research could lay the foundation for a systematic review of Agile tools. The literature in this area is scarce and outdated. The landscape of project

management tools for Agile Software Development has changed and an analysis of which features are best in which scenarios could provide guidelines to teams when choosing a solution to manage their projects.

References

Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2002). Agile software development methods: Review and analysis. VTT publication 478. Espoo, Finland.

Al-Ani, B., Sarma, A., Bortis, G., Almeida da Silva, I., Trainer, E., van der Hoek, A. and Redmiles, D. *Continuous Coordination (CC): A New Collaboration Paradigm. CSCW Workshop on Supporting the Social Side of Large Scale Software Development*, Banff, Canada, November 2006, pages 69-72.

Amer, T. S., & Ravindran, S. (2010). The effect of visual illusions on the graphical display of information. *Journal of Information Systems, 24*(1), 23-42.

Asana. (2018). *Use Asana to Track Your Team's Work & Manage Projects - Asana*. Retrieved from https://asana.com/

Atlassian (2018). *Jira | Issue & Project Tracking | Atlassian*. Retrieved from https://www.atlassian.com/software/jira

Atkinson, R.C. and Shiffrin, R.M. (1968). Human memory: A proposed system and its control processes. In: *K.W. Spence and J.T. Spence (eds.) The Psychology of Learning and Motivation: Advances in Research and Theory. Vol. 2*. New York, NY: Academic Press.

Azizyan, G., Magarian, M. K., & Kajko-mattson, M. (2011). *Survey of Agile Tool Usage and Needs, Agile Conference 2011*. Salt Lake, Utah.

Beck, K., Beedle, M., van Bennekum, A., C. Alistair, Cunningham, W., Fowler, M., Grenning J., Highsmith, J., Hunt A., Jeffries, R., Kern, J., Marick, B., Martin, C. R., Mellor, S., Schwaber, K. Sutherland, J., Thomas, D. (2001). *Agile Manifesto*. Retrieved from http://agilemanifesto.org/

Begel, A., & DeLine, R. (2009, May). Codebook: Social networking over code. In *Software Engineering-Companion Volume, 2009. ICSE-Companion 2009. 31st International Conference on* (pp. 263-266). IEEE.

Berczuk, S. (2007, August). Back to basics: The role of agile principles in success with an distributed scrum team. In *Agile Conference (AGILE), 2007* (pp. 382-388). IEEE.

Bertin, J. (1983). *Semiology of graphics: diagrams, networks, maps*. Madison, Wisconsin: Wisconsin Press.

Biehl, J. T., Czerwinski, M., Smith, G., & Robertson, G. G. (2007). *FASTDash: a visual dashboard for fostering awareness in software teams*. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 1313–1322.

Blichfeldt, B. S., & Andersen, J. R. (2006). *Creating a Wider Audience for Action Research: Learning from Case-Study Research*. Journal of Research Practice, 2(1), 6.

Booth, P., & Schulz, A. K. (1995). The Effects of Presentation Format on The Effectiveness and Efficiency of Auditor's Analytical Review Judgements. *Journal of Accounting and Finance, 12*(3), 107-137.

Burnard, P. (1991). *A method of analysing interview transcripts in qualitative research. Nurse education today*, 11(6), 461-466.

Brinkmann, S. and Kvale, S. (2015) InterViews: Learning the Craft of Qualitative Research Interviewing. London: Sage.

Brown, L. A. (1979). *The story of maps*. New York, NY: Dover Publication.

Card, S. K., Mackinlay, J. D., & Shneiderman, B. (Eds.). (1999). *Readings in information visualization: using vision to think*. Morgan Kaufmann.

Carpenter, P. A., & Shah, P. (1998). A model of the perceptual and conceptual processes in graph comprehension. *Journal of Experimental Psychology: Applied, 4*, 75–100.

Chow, T., & Cao, D. B. (2008). *A survey study of critical success factors in agile software projects. Journal of Systems and Software*, 81(6), 961–971.

Cleveland, W. S., & McGill, R. (1984). Graphical perception: Theory, experimentation, and application to the development of graphical methods. *Journal of the American statistical association*, *79*(387), 531-554.

Coram, M., & Bohner, S. (2005, April). *The impact of agile methods on software project management. In Engineering of Computer-Based Systems, 2005. ECBS'05. 12th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS'05)* (pp. 363-370). IEEE.

Creswell, J. W. (2014) *Research design: Qualitative, quantitative, and mixed methods approaches.* SAGE Publications.

Cockburn, A. (2002). *Agile software development*. Boston: Addison-Wesley.

Davis, F. D. (1986). *A technology acceptance model for empirically testing new end-user information systems: Theory and results* (Doctoral dissertation). Retrieved from DSpace MIT.

Davis, F. D., Bagozzi, R. P., & Warshaw, P. R. (1989). User acceptance of computer technology: a comparison of two theoretical models. *Management science, 35*(8), 982-1003.

Dent, B. D. (1999). *Cartography: Thematic map design*. Boston, MA: McGraw-Hill.

Denzin, N. K., & Lincoln, Y. S. (Eds.). (2011). *The Sage handbook of qualitative research*. SAGE Publications.

Dilla, W., Janvrin, D. J., & Raschke, R. (2010). Interactive data visualization: New directions for accounting information systems research. *Journal of Information Systems, 24*(2), 1-37.

Dingsøyr, T., Nerur, S., Balijepally, V., & Moe, N. B. (2012). A decade of agile methodologies: Towards explaining agile software development. Journal of Systems and Software, 85(6), 1213–1221. https://doi.org/10.1016/j.jss.2012.02.033

Dix, A. (1994). Computer supported cooperative work: A framework. In *Design issues in CSCW* (pp. 9-26). Springer, London.

Dourish, P., & Bellotti, V. (1992, December). Awareness and coordination in shared workspaces. In *Proceedings of the 1992 ACM conference on Computer-supported cooperative work* (pp. 107-114). ACM.

Dubakov, M., & Stevens, P. (2008). *Agile Tools. The Good, the Bad and the Ugly.* Target Process, Inc.

Dumas, J. S., & Redish, J. (1999). *A practical guide to usability testing*. Intellect books.

Eckerson, W. W. (2010). *Performance dashboards: measuring, monitoring, and managing your business*. John Wiley & Sons.

Eick, S. C., Steffen, J. L., & Sumner, E. E. (1992). Seesoft-a tool for visualizing line oriented software statistics. *IEEE Transactions on Software Engineering*, *18*(11), 957-968.

Elmeleegy, H., Ivan, A., Akkiraju, R., & Goodwin, R. (2008, September). Mashup advisor: A recommendation tool for mashup development. In *Web Services, 2008. ICWS'08. IEEE International Conference On* (pp. 337-344). IEEE.

Endsley, M. R. (1995). *Toward a theory of situation awareness in dynamic systems. Human factors*, *37*(1), 32-64.

Eriksson, P., & Kovalainen, A. (2008). *Qualitative methods in business research*. Los Angeles, Calif. London: SAGE.

Esbensen, M., Tell, P., Cholewa, J. B., Pedersen, M. K., & Bardram, J. (2015, November). The dBoard: a digital scrum board for distributed software development. In *Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces* (pp. 161-170). ACM.

Fekete, J. D., Van Wijk, J. J., Stasko, J. T., & North, C. (2008). The value of information visualization. In *Information visualization* (pp. 1-18). Berlin, Heidelberg: Springer.

Few, S. (2004). Eenie, meenie, minie, moe: selecting the right graph for your message. *Intelligent Enterprise, 7*, 14-35.

Few, S. (2006). *Information Dashboard Design: The Effective Visual Communication of Data*. O'Reilly.

Forsell, C., & Johansson, J. (2010, May). An heuristic set for evaluation in information visualization. In *Proceedings of the International Conference on Advanced Visual Interfaces* (pp. 199-206). ACM.

Froehlich, J., & Dourish, P. (2004, May). Unifying artifacts and activities in a visual tool for distributed software development teams. In *Proceedings of the 26th International Conference on Software Engineering* (pp. 387-396). IEEE Computer Society.

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1993, July). Design patterns: Abstraction and reuse of object-oriented design. In *European Conference on Object-Oriented Programming* (pp. 406-431). Berlin, Heidelberg: Springer.

Gillan, D. J., & Richman, E. H. (1994). Minimalism and the syntax of graphs. *Human Factors, 36*(4), 619-644.

Gillan, D. J., Wickens, C. D., Hollands, J. G., & Carswell, C. M. (1998). Guidelines for presenting quantitative data in HFES publications. *Human Factors, 40*(1), 28-41.

Golafshani, N. (2003). *Understanding Reliability and Validity in Qualitative Research*. The Qualitative Report, 8(4), 597–606.

Goldstein, E. B. (Ed.). (2008). *The Blackwell handbook of sensation and perception*. Oxford, UK: John Wiley & Sons.

Gutwin, C., Penner, R., & Schneider, K. (2004). Group awareness in distributed software development*. In Proceedings of the 2004 ACM conference on Computer supported cooperative work* (pp. 72-81). ACM.

Gutwin, C., & Greenberg, S. (2002). A descriptive framework of workspace awareness for real-time groupware*. Computer Supported Cooperative Work (CSCW)*, *11*(3-4), 411-446.

Guzman, E., & Bruegge, B. (2013, August). Towards emotional awareness in software development teams. In *Proceedings of the 2013 9th joint meeting on foundations of software engineering* (pp. 671-674). ACM.

Healey, C., & Enns, J. (2012). Attention and visual memory in visualization and computer graphics. *IEEE transactions on visualization and computer graphics*, 18(7), 1170-1188.

Hegarty, M., Canham, M. S., & Fabrikant, S. I. (2010). Thinking about the weather: How display salience and knowledge affect performance in a graphic inference task. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 36*(1), 37.

Hegarty, M. (2011). The cognitive science of visual−spatial displays: Implications for design. Topics in cognitive science, 3(3), 446-474.

Hevner, A., & Chatterjee, S. (2010). *Design Research in Information Systems: Theory and Practice (Integrated Series in Information Systems 22)*. Boston, MA: Springer US Imprint: Springer.Huang, W., Hong, S. H., & Eades, P. (2006, January). Predicting graph reading performance: a cognitive approach. In Proceedings of the 2006 Asia-Pacific Symposium on Information Visualisation-Volume 60 (pp. 207-216). Australian Computer Society, Inc.

Highsmith, J., & Cockburn, A. (2001). Agile software development: The business of innovation. *Computer, 34*(9), 120-127.

Huang, W., Hong, S. H., & Eades, P. (2006). Predicting graph reading performance: a cognitive approach. In *Proceedings of the 2006 Asia-Pacific Symposium on Information Visualisation-Volume 60* (pp. 207-216). Australian Computer Society, Inc.

Hupfer, S., Cheng, L. T., Ross, S., & Patterson, J. (2004, November). Introducing collaboration into an application development environment. In *Proceedings of the 2004 ACM conference on Computer supported cooperative work* (pp. 21-24). ACM.

Iselin, E. R. (1988). The effects of information load and information diversity on decision quality in a structured decision task. *Accounting, Organizations and Society, 13*(2), 147-164.

Jakobsen, M. R., Fernandez, R., Czerwinski, M., Inkpen, K., Kulyk, O., & Robertson, G. G. (2009). *WIPDash: Work Item and People Dashboard for Software Development Teams,* (1), 1–14.

Köhler, W. (1967). Gestalt psychology. *Psychological research, 31*(1), XVIII-XXX.

Kosslyn, S. M. (2006). *Graph design for the eye and mind*. OUP USA.

Kraut, R. E., & Streeter, L. A. (1995). Coordination in software development. *Communications of the ACM, 38*(3), 69-82.

Krueger, R.A. and Casey, M.A. (2009) *Focus Groups: A Practical Guide for Applied Research*. Thousand Oaks, CA: Sage.

Larkin, J. H., & Simon, H. A. (1987). Why a diagram is (sometimes) worth ten thousand words. *Cognitive science, 11*(1), 65-100

Larman, C., & Basili, V. R. (2003). Iterative and incremental developments. a brief history. *Computer, 36*(6), 47-56.

Lempinen, H. (2013). *Design framework for performance management systems: an ensemble approach*. Helsinki: Aalto University.

Lehtonen, T., Eloranta, V. P., Leppanen, M., & Isohanni, E. (2013, December). Visualizations as a basis for agile software process improvement. In *Software Engineering Conference (APSEC), 2013 20th Asia-Pacific* (Vol. 1, pp. 495-502). IEEE.

Mack, R. L., and Montaniz, F. (1994). Observing, predicting and analyzing usability problems. In *Nielsen, J., and Mack, R. L. (Eds.), Usability Inspection Methods, John Wiley & Sons, New York*, 293–336.

Mackinlay, J. (1986). *Automating the design of graphical presentations of relational information. ACM Transactions On Graphics (Tog)*, *5*(2), 110-141.

Mandel, T. (1997). *Elements of user interface design*. New York, NY: John Wiley & Sons.

Mateescu, M., Kropp, M., Burkhard, R., Zahn, C., & Vischi, D. (2015). *aWall: a socio-cognitive tool for agile team collaboration using large multi-touch wall systems. In Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces* (pp. 361-366). ACM.

Microsoft. (2018). *Plan, Code Together, & Ship Faster | Visual Studio Team Services*. Retrieved from https://www.visualstudio.com/team-services/

Nerur, S., Mahapatra, R., & Mangalaraj, G. (2005). Challenges of migrating to agile methodologies. *Communications of the ACM, 48*(5), 72-78.

Nguyen, N. T., Kowalczyk, R., & Xhafa, F. (Eds.). (2013). Transactions on Computational Collective Intelligence XI. Berlin, Heidelberg: Springer.

Nielsen, J. (1994, April). Usability inspection methods. In *Conference companion on Human factors in computing systems* (pp. 413-414). ACM.

Nielsen, J., & Philips, V. (1993). Estimating Relative Usability ot Two Interfaces: Heuristic, Formal, and Empirical Methods Compared. *ACM 1993*.

Paredes, J., Anslow, C., & Maurer, F. (2014, September). *Information visualization for agile software development. In Software Visualization (VISSOFT), 2014 Second IEEE Working Conference on Software Visualization* (pp. 157-166). IEEE.

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). *A design science research methodology for information systems research. Journal of management information systems, 24*(3), 45-77.

Perera, N., Goodman, A., & Blashki, K. (2007). Preattentive processing: using low-level vision psychology to encode information in visualisations. In *Proceedings of the 2007 ACM symposium on Applied computing* (pp. 1090-1091). ACM.

Pivotal Software. (2018). *Agile Project Management | Pivotal*. Retrieved from https://www.pivotaltracker.com/

Pocock, D. C. D. (1981). Sight and knowledge. *Transactions of the Institute of British Geographers*, 385-393.

Richardson, C. A., & Rabiee, F. (2001). A question of access: an exploration of the factors that influence the health of young males aged 15 to 19 living in Corby and their use of health care services. *Health Education Journal, 60*(1), 3-16.

Rising, L., & Janoff, N. S. (2000). The Scrum software development process for small teams. *IEEE software, 17*(4), 26-32.

Robertson, G. G., Mackinlay, J. D., & Card, S. K. (1991, April). Cone trees: animated 3D visualizations of hierarchical information. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 189-194). ACM.

Robertson, G. G., Card, S. K., & Mackinlay, J. D. (1993). Information visualization using 3D interactive animation. *Communications of the ACM, 36*(4), 57-71.

Rubin, J., & Chisnell, D. (2008). *Handbook of usability testing: how to plan, design, and conduct effective tests*. John Wiley & Sons.

Ruppen, A., Pasquier, J., & Hürlimann, T. (2011). A RESTful architecture for integrating decomposable delayed services within the web of things. *International Journal of Internet Protocol Technology, 6*(4), 247-259.

Santiago Rivera, D., & Shanks, G. (2015). A dashboard to support management of business analytics capabilities. *Journal of Decision Systems*, 24(1), 73-86.

Saunders, M., Lewis, P., & Thornhill, A. (2016). *Research Methods for Business Students.* United Kingdom: Pearson.

Scheier, M. F. (1976). Self–awareness, self–consciousness, and angry aggression. *Journal of Personality*, *44*(4), 627-644.

Scheller, & Kühn. (2015). Automated measurement of API usability: The API Concepts Framework. *Information and Software Technology, 61*, 145-162.

Schwaber, K. (1987). Scrum development process. In *OOPSLA'95 Workshop on Business Object Design and Implementation* (pp. 117-134). London: Springer.

Schwaber, K., & Beedle, M. (2002). *Agile software development with Scrum*. Upper Saddle River: Prentice Hall.

Schwaber, K., & Sutherland, J. (2017). *The Scrum Guide - The Definitive Guide to Scrum: The Rules of the Game*. Retrieved from https://www.scrumguides.org/

Schwandt, T., Lincoln, Y., & Guba, E. (2007). Judging interpretations: But is it rigorous? trustworthiness and authenticity in naturalistic evaluation. *New Directions for Evaluation, 2007*(114), 11-25.

Shneiderman, B. (2003). The eyes have it: A task by data type taxonomy for information visualizations. In *The Craft of Information Visualization* (pp. 364-371).

Smith, P. J., Bennett, K. B., & Stone, R. B. (2006). Representation aiding to support performance on problem-solving tasks. *Reviews of human factors and ergonomics, 2*(1), 74-108.

Storey, M. A. D., Čubranić, D., & German, D. M. (2005, May). On the use of visualization to support awareness of human activities in software development: a survey and a framework. In *Proceedings of the 2005 ACM symposium on Software visualization* (pp. 193-202). ACM.

Sutherland, J., & Schwaber, K. (2007). *The scrum papers: nuts, bolts, and origins of an agile method*. Boston: Scrum, Inc..

de Souza, C. R., Quirk, S., Trainer, E., & Redmiles, D. F. (2007, November). Supporting collaborative software development through the visualization of socio-technical dependencies. In *Proceedings of the 2007 international ACM conference on Supporting group work* (pp. 147-156). ACM.

Takeuchi, H., & Nonaka, I. (1986). The new new product development game. *Harvard Business Review*.

Tam, J., & Greenberg, S. (2006). A framework for asynchronous change awareness in collaborative documents and workspaces. *International Journal of Human-Computer Studies, 64*(7), 583-598.

Tory, M., & Moller, T. (2004). *Human factors in visualization research. IEEE transactions on visualization and computer graphics*, 10(1), 72-84.

Treude, C., & Storey, M. A. (2010, May). *Awareness 2.0: staying aware of projects, developers and tasks using dashboards and feeds*. In *Software Engineering, 2010 ACM/IEEE 32nd International Conference on* (Vol. 1, pp. 365-374). IEEE.

TRINT (2008). *Home - Trint*. Retrieved from https://trint.com/

Tufte, E. (2001). *The visual display of quantitative information*. Cheshire, GT: Graphic Press.

Tversky, B. (2001). *Spatial schemas in depictions. In Spatial schemas and abstract thought* (pp. 79-111).

Tversky, B., Morrison, J. B., & Betrancourt, M. (2002). Animation: can it facilitate?. *International journal of human-computer studies, 57*(4), 247-262.

Vicente, K. J. (2002). Ecological interface design: Progress and challenges. *Human factors*, *44*(1), 62-78.

Vessey, I. (1991). Cognitive fit: A theory‐based analysis of the graphs versus tables literature. *Decision Sciences, 22*(2), 219-240.

Ware, C. (2004). *Information Visualization: Perception for Design* (pp. 1-27). San Francisco, CA: Elsevier.

Wertheimer, M. (1923). A brief introduction to gestalt, identifying key theories and principles. *Psychol Forsch, 4*, 301-350.

Wessels, A., Purvis, M., Jackson, J., & Rahman, S. (2011, April). Remote data visualization through websockets. In Information Technology: New Generations (ITNG), 2011 Eighth International Conference on (pp. 1050-1051). IEEE.

Wickens, C. D., & Carswell, C. M. (1995). The proximity compatibility principle: Its psychological foundation and relevance to display design. *Human factors, 37*(3), 473-494.

Wickens, C. D., & Hollands, J. G. (2000). *Engineering psychology and human performance*. Upper Saddle River, NJ: Prentice Hall Inc.

Wilson, E. V., & Zigurs, I. (1999). Decisional guidance and end-user display choices. *Accounting, Management and Information Technologies, 9*(1), 49-75.

Woods, D. D. (1984). Visual momentum: a concept to improve the cognitive coupling of person and computer. *International Journal of Man-Machine Studies, 21*(3), 229-244.

Yi, J. S., Kang, Y. A., Stasko, J. T., & Jacko, J. A. (2008, April). Understanding and characterizing insights: how do people gain insights using information visualization?. In Proceedings of the 2008 Workshop on BEyond time and errors: novel evaLuation methods for Information Visualization (p. 4). ACM.

Yigitbasioglu, O. M., & Velcu, O. (2012). *A review of dashboards in performance management: Implications for design and research. International Journal of Accounting Information Systems,* 13(1), 41-59.

Yin, R. K. (1994). *Case Study Research: Design and Methods*. Michigan: SAGE Publications.

Yu, J., Benatallah, B., Casati, F., & Daniel, F. (2008). Understanding mashup development. *IEEE Internet computing, 12*(5).

Zhang, J. (1996). A representational analysis of relational information displays. *International Journal of Human-Computer Studies*, *45*(1), 59-74.

Zhang, P., & Zhu, D. (1998). Information visualization in project management and scheduling. *Former Departments, Centers, Institutes and Projects*, 55.

Zuk, T., Schlesier, L., Neumann, P., Hancock, M. S., & Carpendale, S. (2006, May). Heuristics for information visualization evaluation. In *Proceedings of the 2006 AVI workshop on BEyond time and errors: novel evaluation methods for information visualization* (pp. 1-6). ACM.

Appendices

Appendix A: 12 Principles of Agile (Beck et al., 2001)

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity--the art of maximizing the amount of work not done--is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Appendix B: Interview Transcripts

Linda - Blue Billywig

08-03-2018 16:02

Stijn: [00:00:00] So in small introduction. Also the role division I will mainly do the talking and Marco will do some questioning if something is unclear for us. But in terms of the objective of our research as I mentioned before both of us are have worked or are working within the field Agile and within that we experienced many tools that can be used in order to make the process of software development a little bit more smoothly. But what we also experience that each of these tools to have some things that they're lacking or could be improved and that is actually where the objective of our thesis is coming from. That we believe that we can improve or adjust existing tools so that so that is what we would like to talk about. And we have a special focus on data visualization in that sense and actually what we're very much interested in is what your professional experience is with both Agile in general and also a little bit with the tools that you're using so that is the objective and that is what you would like to talk about.

Linda: [00:01:10] Okay.

Stijn: [00:01:11] First a very general question but could you maybe describe the company what you're working for what they do. Oh what's a product or service.

Linda: [00:01:20] Yeah. The company I am currently working for is Blue Billywig I've been here since the fall of 2015. Yeah and before that I worked for NCI they do courses for the private market in the Netherlands I was a Product Owner there too. And there we had like four or five products development teams that I was collaborating with, they worked on an offshore location they were in Romania. So having the correct tools to collaborate was very very important and the current company I work for is a company that has an online video platform and a player that's the two main products developing developing. We are a Business to Business company. And I think that really defines the communication and also we have a team in-house that also changes the way you collaborate. And so I think that should give you an overview.

Stijn: [00:02:28] Yeah definitely. And in terms of organization size right now how many people

are working over there.

Linda: [00:02:35] There are like 30 to 35 people now. Our product development team, which I am a part of is 6 - 8. Okay. Including myself.

Stijn: [00:02:46] And could you tell us a little bit about the composition. So you maybe have some developers some designers some just or some product owners.

Linda: [00:02:53] Yeah we have six developers you know it's a mix of back-end and front-end developers some are more specialized in one of the two but they can actually do both.

Stijn: [00:03:05] Yeah.

Stijn: [00:03:07] One way or another we have one UX designer and me (Product Owner).

Stijn: [00:03:11] Okay.

Linda: [00:03:12] So no Scrum Master.

Stijn: [00:03:13] Okay. Yeah. Very clear. Yeah the role you already described, what your role is within the team. In terms of methodologies. You use some way of Agile. Could you maybe explain a little bit more in detail what type of methodology you are using.

Linda: [00:03:29] Yeah we've kind of looked into the Scrum-methodology and we picked out the one elements that worked for us. That's what we've implemented. So for instance we do a retrospective meeting, but you should know that our team consists of many seniors. So it's a very steady team.

Linda: [00:03:56] So we don't have a lot of people leaving we don't really have a lot of people coming along now that the company is growing. People will be entering, but it's a very solid team. So for instance the frequency of retrospective reading, which usually is after a sprint, after each sprint. We have taken that frequency down a bit because we are because our process is working very efficiently already. So we do it less frequently like once every three sprints once

every 3 weeks makes when do you ever expect that we have a more high level retrospective and actually not so much on an individual sprint.

Stijn: [00:04:36] And how do you guarantee a iterative process if you're not using retrospectives that often. Is there another way in which you try to use the iterative element of it.

Linda: [00:04:48] Yeah well the retrospective, we really as see something to improve our process not so much our are our products. We have a SaaS solution that's the software we make. So are like us and our customers are companies. And that means that we have a lot of liberty in responsibility actually to decide on how we make our features we can't just do what a customer requests. So you have teams that build websites and they do scrum but they do what the customer is asking them. Yeah we're building features in a platform that is being used by many many users. Now we get a big say in how in what they built and we used to have an evaluation in there for customers. So we had a little feedback button in our software so they could tell us what they liked and didn't like it wasn't really actively used. I'm not sure if that's... Most of our customers are our Dutch. I'm not sure if that's like a cultural thing or not.

Linda: [00:05:52] Joining on the community and rather just talk then puts it back into form. But the rest of the process is really based on iteration so we have two weeks sprints. They have daily stand-ups of course and after each sprint we do the demo and before we did the demo for the organization we were show a pre demo with the team. They knew how a feature went, the development went and how we should fund is there anything else we should do. We added a small step in the process and that's what we call the user-story-review. And what that essentially does is that when your stories are finished within an sprint I get called over. The team presents to me what they've done, then I can either ask them to do something else or change something if I think they should change it. Okay create follow up user stories for the next spring.

Stijn: [00:06:49] Okay, clear. It's a lot of elements that you have to using over there.

Linda: [00:06:52] Yeah we are using a lot. I think the main elements that we're not using is this the scrum master role. Why that the case? Yeah I think that also has to do with the seniority of the team. They take a lot of responsibility in just getting the job done and if there are any impediments they just want to handle it. And I do understand that it does take away from their

developments time a bit. One of the things scrum wants to do is make sure that the development team can focus on developing stuff so anything that is not developing self is a distraction.

Linda: [00:07:33] But since our process is already very mature we don't get that many impediments anymore. That influences the process a lot.

Marco: [00:07:42] Okay. Can I jump in? So who actually facilitates the meetings would be you as the Product Owner then kind of schedule the planning.

Linda: [00:07:58] Yeah I actually follow up on the structure and make sure everyone is there and plan the meetings.

Linda: [00:08:05] Okay. You mentioned a little bit but in the process because you're not building it for one client, you are basically building one platform with can be used by many different stakeholders. But in the process, do you talk with your client. And if so how do you talk to them. You mentioned like the feedback mechanism. That is one way of communicating with them and you use any other methods?

Linda: [00:08:30] Well not as much as I'd like. I mean in the ideal world, I would be talking to customers all day. What I try to do is visit our customers regularly. Yup. For instance if someone's not a customer yet they're still in the sales process, I'm joining one on a sales team persons to talk about case and to get a feel of how they could be using our product and how we think they could be using our product but also our existing customers. I talked to on a regular basis and I tried to spread it out in a way that I talk to every type of customer. Yup. And also it helps me balance out what kind of decisions I should make or kind of priorities I should give. Some of our customers are very true to us. They have been a customer for many many years so that gives special type of relation and also a possibility to be very open on a long term perspective on how they see our products on the long term and that's input that I need. Of course there's also a financial input that I need and something that would be nice to have in the future for us is to have a better perspective on UX. So there are a lot of tools that can help you see how the user navigates your Web site. Yeah we have an application which is a bit different from a regular static Web site but still very interesting to see where users click.

Linda: [00:10:15] Yeah.

Linda: [00:10:16] But I think that's something that will take place in the future. But it's not very right right now also because of the seniority on our designer. Who already has a very good conceptual idea on how things should be. In a screen when we should position buttons.

Stijn: [00:10:36] Yeah but but to a certain extent you would think it's useful especially if the team is growing maybe in nearby future just to have a better understanding of what the experiences or how your customers are navigating et cetera et cetera.

Linda: [00:10:49] Yeah well I think you will be especially interesting for like our video-player. Because our video players being used on Web sites and that's a product to our customers of our customers are in touch with us and they're going to be very interesting to see to look at conversion, to have A/B testing. To see if you are if your goal can be reached better if you position things differently and I think the platform itself, so the platform itself is used to manage content to look at general statistics for the video and since it's software we will find their way to do their task. Of course we think it's very important that you don't have to guess where you're going. And it's like you have this theory that 80 20 theory lets you put 20 percent of your effort in 80 percentage results and then look at the remaining extra results of 20 percent with 80 percent of the effort there and try to balance that line. You also make sure you put in. That's why I think it's also very agile to just put in as much effort as you need to get the best result possible within the time you have.

Stijn: [00:12:14] And in terms of communication you say that you have a lot of communication also in face to face. Is there any tool that you're using in order to communicate with them...

Linda: [00:12:27] With customers? Well we have InterCom I think you've heard of that, we have that integrated in our application as a way of communicating of course they call. This enables it from a tools perspective, InterCom is what we use.

Stijn: [00:12:45] OK. OK. Makes a lot of sense. And for how long is it actually that you have been using this current method so you have a lot of seniority within your team is it was it from

the start of the company or has the methods you've been using changed over time.

Linda: [00:13:02] I think that with the growth of the products and the growth of the company things have changed. I have only been there for two and a half years and the company is over 10 years old. And I think it's traditional software companies go through. It's very tech minded at first and then all the team is based on making sure your code is okay.

Linda: [00:13:25] And then we start making the shift from just hand in tech-stuff to actually having software that people want to use. And then the company extends so you get different tooling. And we are eventually going to move up to a more ... to having less tools, but tools that can do more.

Stijn: [00:13:56] Okay that's interesting and you going to put my camera off because sometimes we cannot really understand or like we lose connection a little bit.

Linda: [00:14:02] Is says something you missed just let me know. Yeah this was just a split second but just for the... You mentioned tools already a little bit. And you said you have love tools out there that you're using everything a little bit. If I understand correctly what types of tools are you using right now. Could you give us.. Well for the the product development. When I started this company they were using 'Unfodle'(?) But they were also using a whiteboard with posted notes on it we actually had a digital ticketing system and a physical ticketing system. I don't think I have to explain why that didn't work out that great. Having a double administration is always the base for many many many problems. And the reason why they had the (online) ticketing and the physical ticketing system so that the white board with the post its on it is because 'Unfodle' didn't allow them to have a hierarchy in tickets.

Stijn: [00:15:13] OK.

Linda: [00:15:14] So or at least it did allow for hierarchy, but that didn't visualize it properly.

Linda: [00:15:19] I guess if you would have sprints they were the list of tickets will be very long and it will be difficult to see which tickets belong to which user story. And that makes it difficult to see the priority.

Linda: [00:15:33] Yeah.

Linda: [00:15:35] So I've actually investigated a lot of different ticketing systems to find one that could do that properly. And there aren't that many out there.

Stijn: [00:15:47] OK.

Linda: [00:15:48] So what we've actually ended up with was a Visual Studio. These is a Microsoft. So I got into a lot of trouble for that because all the guys here are Apple fanboys, but when I showed them why it works it has a pretty cool dashboard in there that we actually use in our standup. So we look at the complete overview of the current sprint. And I think that that really did it for them because it gives you a very nice overview. They recently added a feature where you can kind of customize and automate parts of your workflow, when assigning a ticket to a person you can log some other fields or when the state changes you can for instance you have assigned to field and you can imagine you can solve ticket it changes to the state resolved. I can remove your name from the assigned to field. And then login as person who resolve the ticket. So that's something that just recently added. And that. Help us out in the automating workflow bit but that also means that we use that system a lot to just have our project management so I actually have our tickets and use it to organize our code. We have Jenkyns for that and we have Slack to chats.

Linda: [00:17:27] We used to used HipChat, but now up and now Slack because sometimes we work from home and that it's important he could still talk to each other. We already mentioned InterCom of course, I've started an investigation to get CRM system up because I think it's important that we have an overview of who our customers are what they do, the kind of money they bring in how they use systems stuff like that. We already have a financial system which can imagine that. And we use Google for our our personal database with the people with employees: that our emailing. It's a big part of how we share files with each other and so we are kind of the type of company that doesn't want to invent stuff. They always think there's a solution out there and there should be a tool out there for what we want. So we'll find someone else who could do something for them.

Stijn: [00:18:32] Yup yup

Linda: [00:18:35] I think that that's gives you a good picture on the tools. I'm thinking I think this pretty much it this from somewhere to fly around. I know some people just personally use some tools. Generally these are the ones that are used by the whole company.

Linda: [00:18:52] And so there's no general statement. So I hear myself double right now. So there's no general. Would you said requirement within the company that a certain amount of tools are being used and the rest is being done so to say is not allowed to be used.

Linda: [00:19:09] It's a personal preference some people. We look at what the benefit is, if a tool brings us something that we cannot get in any other way then we will probably keep using it.

Linda: [00:19:28] But I do expect, you know with the company growing, that they want to prevent having 30 40 50 60 tools in the end. What it tells you if I had to use a lot of different tools is that when you're using one tool and looking at specific problems from a specific perspective that you have to be be in a different tool to get another insight. And let's never good. You should ideally get a total overview of everything that's going on

Stijn: [00:20:03] That you mentioned before. You said maybe it's interesting to see if there's some type of integration tool,, because according to you right now there's a lot of tools that we are using. Is that something that you are looking into right now.

Linda: [00:20:17] Not active. You know we keep our eyes open if we stumble upon something.

Stijn: [00:20:23] OK. And with the tools that you're using I mean you described I think like five six maybe more tools that you're using. Besides not having the a clear, I would see that maybe just one issue over here. Do you have anything else that pops into your mind that might be a problem or that you encounter personally within the use of the tools.

Linda: [00:20:48] No no not really. It's like I said personally I just think having to be in a lot of different places to get a total overview and to get all the information you need to do what you need to do your job. It's just weird. Yeah you should be able to assemble all the information it

wants funding get what you need when you need it.

Stijn: [00:21:15] I think we lost you again.

Linda: [00:21:20] I'm still where we are as well where we were when we missed like the last ten seconds more or less... Then some questions in terms of visualization, you mentioned for example the tools tool from Microsoft if you are using this tool could you maybe explain to us how if you were to produce a certain output together with a colleague how does this work and how does this help you with that.

Linda: [00:21:53] We used that tool in our process. There's a backlog in the tool of course, and I maintain the backlog. And we use that backlog and what I put in it, in our refinement meeting in which we talk about the most important priorities that we have in the current period as a preparation for the upcoming sprint. And then when we start the sprint we have the sprint planning meeting in which we select the user stories that we think we put in the sprint, and then the team just creates the tasks in the tool. And then during the Sprint of course they update the status and yeah when the sprint is done. You know we would sprint like I said with stand up we use it to get any insight on what we're doing and what the status is and where we think we're just failing, so if the ticket is in progress for a very long time you know... Try to figure out why. Yeah OK. So that's how it's just taking us inside in the process and any progress of our work.

Marco: [00:23:14] The tool you are using for all this is the Visual Studio. Or following up OK.

Linda: [00:23:19] Yeah actually I can send you some screenshots.

Linda: [00:23:23] That would be nice. That would be nice. The backlog and the sprint and how that works. Yeah that would be nice.

Stijn: [00:23:30] So so if I understand correctly this is also the tool and also the way in which you track how artifacts might change over time. Right. That is also being used within this tool.

Linda: [00:23:43] With artifacts I mean like our documentation.

Marco: [00:23:45] Yeah.

Linda: [00:23:47] Yeah kind of our technical documentation is in this tool. So the developers write down what they've changed in the code will not literally but they they write on the technical aspects of what they're doing. And how they resolved it, functional documentation is in our support websites. And that's when I forgot to mention we use GitHub for our code, for our front-end code and any documentation on the front end is written there too and then automatically published to our support website.

Marco: [00:24:24] Ok. Yeah. Do you use any integration than just to keep because yeah so you have like two sources one for the front and the maybe one for a back and so do integration

Marco: [00:24:39] The back end is mainly in Jenkins and then the code is on a server somewhere. Our back-end is very complex. So the thing is usually you have like a database right in the database you have all the records. But that doesn't work for us because a regular database doesn't give us the performance we need because for instance part of our beckoned architecture is our analytics platform anywhere where someone is watching our content and our player, we are loging statistics and to give you a bit of an idea of the volume and why a regular database doesn't work. And is that when we started out you know we'd be very excited is we get a million views per months and now we were not even surprised when we do a million views per hour.

Linda: [00:25:36] That volume means we have to do some asynchronous data storage. And so it's very very complex and actually actual code for developing is to use external services mostly on Amazon to get all that going. So we're not hosting the data anymore. It's all somewhere else and that makes our backend different from a traditional one. And that also means we don't really need a checking check out for the complete application but only for functions within it. Okay. And its very techy,

Stijn: [00:26:22] In terms of handling a test if you need input from more than one team member. So you can for example within one project, you have a designer, UX designer, and a front-end and back-end developer how does that work?

Linda: [00:27:01] Different from a traditional one. And also that means we don't really need code checking check out for the complete application but only for functions within it. Yeah okay a techie yeah yeah yeah.

Stijn: [00:27:18] In terms of handling a task if you need input from more than one team member so you can for example within one project you have a designer who is designing, a front end a back end developer. How does that work within your process.

Linda: [00:27:34] We create tickets for each expertise. We have the expertise design we have expertise backend expertise front end and expertise player and..

Linda: [00:27:48] When we have a user story that meets all of them. We just create tickets for every a task they need to do and we give them the expertise. We already order them up. We visually order them in the in the correct way of handling them. So who ever needs to go first is on top but we've recently started doing and I already see it working. Is that we have assigned team members to use your stories. The reason why we do this is because there's always a risk, and we we've you of course experienced this in practice that people only do the task their expertise. And they forget to look at the bigger picture. So Someone develops code in the front end and forgets to check with the back end guy if he's using the right terms and then stuff doesn't work. Now I think this is a problem that a lot of teams experience, so aside from having to user story review which is more functional.

Linda: [00:28:51] I assign one of the developers to the user story as like not so much that he or she is responsible for getting all the work done so he's not talking to his colleagues saying you still need to do this. But he is there to make sure that the technical implementation goes okay so that we follow a plan and if the technical design we've made.. That that's honored by all expertises to make sure the answers are always okay.

Stijn: [00:29:30] And you mentioned before a little bit about the hierarchy or an order within and ticketing system. But of course you have a situation in which the order might change. For what reason whatsoever. Because you're working on multiple project for example. So would be interesting for you to be aware of what another is doing currently what does he or she is working on. And if so how do you do that.

Linda: [00:29:57] Yeah it is interesting to see who is working on what specifically because like I said it could be a user story needs different types of expertise. Someone starts with a user story that's at the bottom of the priority list and someone else starts on the top. And you know we have a problem. So what we do is within the sprint I order everything based on priority. So you can imagine there's like a list of user stories that are orded based on priority and then within the story it actually kind of like folds. There are tasks and they are also ordered by priority. But then it's the priority in which they should be developed. So it's more technical logical to start with task A and then B and C and D. Yeah and that's the way we handle that. And for me of I've been keeping an eye on it for on track, reaching our Sprint goals. And also if I see someone being done with a task sooner than expected I can always figure out if if it makes sense for that person to start on the next user history or maybe it's already at the end of the sprint and it would be better if starts on something else because other team members can do the rest of the work for that user story.

Stijn: [00:31:24] Now that makes sense. That maybe you should already a little bit. A follow up question would be how do you know where you currently are. Like we did if we didn't a bigger project.

Linda: [00:31:34] I'll send you a screenshot after this meeting. In Visual Studio. So they have a board and the board has columns in it. And the left column is the user story level. There you you see the user stories and then the columns following it are for the tasks and they have the states. So the first column is the to do column. Then there's the in progress column. Then there's the resolved column and then review and then test it and then done. You can customize these if you want but these are states that we needed. So we can actually just immediate the see for a user story if all tasks are done or not and if all of them are done it automatically changes its own state to done and then it folds in. So you don't see all the tasks anymore you just see one line for that userstory being done.

Marco: [00:32:35] Do you do work with Epics.

Linda: [00:32:39] Yes yes yes we do actually have three backlog's. No 5

Marco: [00:32:49] All right.

Linda: [00:32:52] One of the back logs is with Epics and Epics are of course above the level a level above the user stories you the stories within them. But then I also have the user story backlog because there are also stories that don't have an Epic and and apart from these two I have a suggestion backlog in which I put ideas and every once in a while I look at the ideas and sometimes I upgrade them to a user story and sometimes I just throw them out and I hsave bug backlog. So any bugs we find when they're not urgent I put them in the bugs backlog and I prioritize them. And then we also have a maintenance backlog for our technical maintenance. So sometimes you need to upgrade technical parts of the system and that's actually something the team usually maintains and they put new stuff in there whenever we start the sprint. We look at all the backlogs. Or at least I do and I do the suggest. So I tell them I think we should do these user stories and then this maintenance task and then maybe some bugs and that's how we assemble the sprits.

Marco: [00:34:07] OK. Yeah. Following up on that sense. So let's say you have a epic, which I will call here like a project write something that you write. You take more than one spring to complete. Yes. Do you follow one approach to figure out which stage are you when. So let's say I am on 25 percent of the stuff before 50.

Linda: [00:34:32] Yeah well the epic itself has states too. So it starts out being new and then I figure out what we kind of need to do and then we have the user stories. So then I call the epic refined and then we start putting stuff in sprints I put the state to progress and because visually it can also fold in and out and I can see all the user stories with it. I can easily see based on states of the user's stories how far we are. But there's no percentage of potential visualization that

Stijn: [00:35:10] Would be something you would be interested in or is it not of an additional value in terms of what just.

Linda: [00:35:17] It is interesting but it's kind of nice to have. Every story has its own effort so sometimes it takes a lot of effort and sometimes not so much. I always think it's more important to know what goals we've reached which features we have finished then to talk about

percentages because 75 percent doesn't mean anything when it's just about effort.

Stijn: [00:35:43] That's clear. You you talked a little bit before about it but in terms of what do you think is interesting to see. You talked about like where in the process you are whether it's in percentages or whether it's in features that are being done. Are there in other elements from your perspective from your role that you're currently having. That you find interesting to know during a project.

Linda: [00:36:06] What I find interesting that also is based on the type of product do. Who my stakeholders are always creates or design a feature based on having a specific customer in mind. We have customers that are more like publishers, but we also have bigger brands right now. It doesn't matter. Bigger company.

Linda: [00:36:35] And then another product we have is our add services so that's online video adds and they use different sub-parts of our platform. We design our features based on which customers we think will use them. And sometimes it's just for one customer group. But it could also be two or three. And so I always think it's important to keep that in mind when you are creating a software you should always know what the goal is and only your users.

Stijn: [00:37:14] Okay two important elements. We're almost done btw, we have two more questions. But in terms of the frequency that you use how often do you use tools in terms of clarification or being aware of the project status.

Linda: [00:37:32] Daily. I always like to keep an eye on what we're doing and how we're doing. It is just that the task you're doing differs per day so like I said we used to board for the stand-up and the guys that.. The developers of course use it to update the tickets and write their documentation. And I use it daily just to see the status and then I also create and update the user stories but that's not something I would do on a daily basis.

Stijn: [00:38:12] OK. Yeah. And in terms of decision making. Now to what extent do you rely on the data that is being did you have as an output from the tools for making decisions.

Linda: [00:38:26] Well we always do and it's of course the velocity. This is something I think you

probably already know. We look at how many story points we can put into a sprint.

Linda: [00:38:39] But there is a bit of a side note there and that has to do with the specializations that I explained. If we can do 200 points in a sprint. Well I put a 200 points in and it's all front end development and go back and then we have a problem. Because some developers will be doing nothing and some developers will be overloaded. So that's always something we have to look into. And that means that we're almost like a task basis looking if we can or cannot commit to something. So it would help us if we could do commit based or being informed on the effort we need per expertise. So we can get a more balanced sprint, because in the ideal world you know where our priorities are and based on your product and where you mean envision your product to go and what your customers want but then reality hits and you have three front end developers and two back end developers and you need to make sure everyone's doing something valuable. So that is always something that we need to take into account to balance out our work. Yeah makes a lot of sense. Nice.

Marco: [00:39:56] Thanks a lot. One more question. You're talking about their analytics right. You have your software kind of receives a lot of data from Yeah. Does the data actually help you to actually figure out what you are going to be next.

Linda: [00:40:19] It kind of does I always use it to see well it helps itself out. So it kind of gives us an idea on our growth and which areas we're growing.

Linda: [00:40:34] And so it also in a way tells us how it should perform so we have the analytics platform and our analysts platform shows us how many views we have. So we have an idea on what the platform should be capable off in the future. And we get some pretty extensive lobbying so I can see what apart from the fact that I see how many views we have I can actually see if its views on video advertisement or on regular content or on one of our key products is interactive video. So I can actually see how many users we have interactive video. But also on what kind of devices. And that's always interesting to look at as we as everyone does. We see the clear shift to mobile. So that means that we're actually developing new layouts for our Player and of course we're making it mobile first. So it tells us a lot about how our videos are used and that always helps making decisions based on that rating system for that products

Stijn: [00:41:48] Cool. Ok thank you very much for your time. Is there anything based on nice talk we had so far that you would like to add something that maybe we forgot to ask or that pops into your mind like hey this might be of any value to you guys.

Linda: [00:42:06] Yeah I say I can't really think of anything but if I do think if anything I will definitely let you know and I'll be very interested in hearing about your results and to follow up on your research to see what kind of conclusions or ideas or tips you can come up with so keep me posted that would be great.

Stijn: [00:42:28] Yeah most definitely. Just to inform you we're very much at the start of our research. So two and a half months to go.

Linda: [00:42:35] And that's fine yeah. So it will take a while just some expectation management over here. But most definitely. Yeah we will most definitely inform you with our conclusion or maybe we'll short summary that's easier for you. Send over whatever you wanted to share. Sure. Most definitely. Okay.

Marco: [00:42:53] Thank you very much for your time and your effort and your it has moved us both. Because we've already heard that you are kind of hard on time. But we would we would. We would really love to actually talk with one of the developers but probably with a more concise interview but. Yeah. What our research is mainly some visualization that we would actually like to see what the developers see because actually that the company was very very interesting to us actually right after the talk. So if you think there is a possibility for us to talk to one of the developers we'll be out there.

Linda: [00:43:39] I'll definitely ask ask. I need to tell you already upfront that very busy. But I ask them if have some time and I'll let you know.

Stijn: [00:43:50] Okay. And what Marco stated If we need to make it more concise we can fit it in 20 minutes if needed. I mean it's for us the more interesting part is actually the second part because what we would find very interesting is to see like what's the difference between different roles and what is your preferences what what is it that you want to see and is it the tools that you're having right now is it like. Often it is it made for one fits all and we are asking

our successes for the developers.

Linda: [00:44:21] I actually made a dashboard for them in Visual Studio thats actually tell them which tasks to have to do and what the status is. So that's indeed that's a very different level of visualization than my visualization which is more like an overall view. I ask the guys that they had like 20 minutes to to talking about what the are you still there. Yes yeah yeah. I saw so I was saying that I'll ask the guys if they have time. If I find someone that has time I'll let you know. But I'll include a screenshot and a short description of their personal dashboards for you.

Stijn: [00:45:15] OK thank you very much already. That should help you in anyway. Yeah definitely. OK thank you very much and enjoy your day enjoy your weekend. We will talk again. Yet we keep in touch. Bye bye bye. That was cool. This was so much better.

Karnov Group - David (Scrum Master)

13-03-2018 13:01

Stijn: [00:00:01] So just as a starter to introduce myself, my name is Stijn, I'm actually from the Netherlands and currently writing my Master's thesis together with Marco. Subject is actually because I was working in a software development company as well where I had an internship over there and there we had some mutual problems that we encountered and, therefore, the topic of our thesis came together, you know Marco. So as I mentioned before we first start with a couple of general questions and then go to basically the visualization part of the tools. Could you describe what the company is doing a couple of words?

David: [00:00:48] So we're making websites for loyal, hmmm for lawyers and all the legal professionals. So a place where they can find legal documents but also work with them in all various ways like bookmarking, saving, printing, highlighting, and other stuff. The easiest way to describe in a few words is like Google for lawyers in a way, so you can find all types of documents. We also crawl other content in order to provide everything in one place.

Stijn: [00:01:28] And in terms of organization size and team size, could you tell us something about that?

David: [00:01:31] I think there is around 120 people in this office, and we have the development department which is around 30 to 40 people and then they're split into smaller teams so like 5 to 6 people.

Stijn: [00:01:51] And the composition of a team. I don't know if you can have a team or is there one team but what you consist of terms of developers, designers.

David: [00:02:03] So we have mostly developers here, then we have a designer as well embedded in our team and also works with other stuff but not just with us but mainly with us and then we both because we have PM as well project manager it's product manager the actual meaning of that and then we have also the position which is something like me which we call the dev lead which is supposed to be like someone overviews the general technical future of the product. Where are we going. More like the technical side of it. There's a lot of the technical projects which we have to do or infrastructure ones. It's more like my part.

Stijn: [00:03:04] Could you say that's the Scrum Master so to say?

David: [00:03:05] No, no, no. This is more like we're still kind of figuring out the definition of it, but in general, the dev lead is there too. Oh actually I had this nice overview to show what it was but basically you have an overview the technical vision of the project. So what technologies we should use, which direction we are going and, some more from the product side as we are going to have this feature, as we are going to target these kind of customers are projects which are going to work. It's more like the technical side of it.

Stijn: [00:04:00] We're doing our research about Agile I chose a pretty broad concept I would say there are multiple methodologies within Agile. Are you using one specific method of Agile?

David: [00:04:15] What we're trying to do is Scrum. Yeah. And yeah I wouldn't say we're doing exactly as we should we are trying to do our own way.

Stijn: [00:04:31] And then what elements are you using and what elements are you not using.

David: [00:04:37] I would say oh well we have of course from the from the ceremonies we have a planning meetings. We have refinement sessions. We have a demos mostly internals not really with the stakeholders, it's more like internal demos and then we have daily standups and of course retrospectives.

Stijn: [00:05:02] Are there any specific parts that you're not using?

David: [00:05:03] Well I would say that this is not only about doing the actual thing it's about the whole process itself. So of course we're trying to do all the ceremonies from Scrum but not necessarily. There are different things in between which we can do more Scrum oriented.

Stijn: [00:05:23] Could you give an example of one maybe?

David: [00:05:29] Let's say the , I think we can do better at the way we handle our planning and refinement and prioritization, this could be a bit different. Also, from my previous experience from Scrum I would say that it's important to do... How we use Scrum from the first time by having external told who was the first coach in Denmark for Scrum. And the first half a year was the period where we were supposed to do everything by the book with quite a strict criteria. So for example estimating all the tasks, breaking stories into smaller tasks, estimating each of them in a disciplined (inaudible) and actually having a burn down this is one of the examples, we don't do burn down charts for example, this is one of the things but it's also because we don't also really split the stories into smaller tasks which I also think we could be better. And then it's we don't track really good progress right now. So it is hard to say. Are we on track or not of course you can always see from maybe the different tools which we use for the over v iew or so how far are you. Having a burn down chart might help. So back to my thing I think that first half year we should do it we should do everything by the book. And then when you actually really try to do it and then you start tweaking think a lot of people wondering when they do Scrum is that they want to start tweaking from the first day. Okay. And that they basically all we don't need that, we don't need that, we just do this because it's some times hard to say we don't need it but you haven't tried that, in a serious note. So by actually being forced to try something in like say 3 to 4 months, for example doing the burn down precisely, doing the hours as well around eventually forces you to go deeper in it and then in the end maybe you realize don't need needed and why you should skip it rather than just skip it just because you think you don't need that. That was a

interesting experience too. You also see how the team develops over the time from that first first and as you said and you start being more self organized and you eventually transform into what's best for you. I think is was an interesting experience to do that way.

Stijn: [00:08:14] That makes sense. In terms of talking to the customer or including the customer within the process of development. How do you do it? Do you talk to them? Do you include them maybe within the tools that you're using in terms of communication or how does that work?

David: [00:08:33] That's always of course right. The internal users if you make something for the external users. I don't think we are including external users that much. Which could be better. But I also think that it's not always necessary that I think that part of including external should come more in the beginnings when you're trying to scope their requirements or get input for something, of course for testing as well. But during the actual development and during the actual sprint of course is nice to have someone to ask to, but some time it might actually I think it might be that if something is poorly refined if you still have some big questions during the development that's probably that someone did not ask the correct questions. So it was not refined properly. What was the question again?

Stijn: [00:09:31] The questions is like how to include customers in the process of developing a piece or a certain feature, is it only for example, by talking for example, that they are included within a type of communication tool that they can see the status of the project.

David: [00:09:52] No. And I think that of course that could be better. And on the other hand what is the role of the product owner in the process, right? He's the one supposed to be the in contact with the stakeholders that users are. So it comes a bit against this two things if the PO should be the shield for team to the external world and then how are we supposed to include the external users is a bit of a clashing thing in general.

Stijn: [00:10:24] One pretty general question but for how long have you been using this method? That you know of? Like the Scrum or the version of Scrum that you're using.

David: [00:10:34] What do you mean? Here? Let's say from November. That four months.

Stijn: [00:10:40] OK. And if you're talking about developing a piece of software. You have some tools that you can use. We heard Pivotal Tracker before. What types of tools are you using so it's in terms of communications, in terms of tracking, and terms of general project management.

Stijn: [00:10:58] So we're using Pivotal Tracker here and I think that's the same thing when you were using.

Stijn: [00:11:08] Also communication? Maybe Slack?

David: [00:11:11] Oh yes! We have Slack for internal communication.

Stijn: [00:11:16] Any other tools that you're using?

David: [00:11:19] Nothing. Like e-mail. I think that's it.

Stijn: [00:11:26] Why you're using this particular tool and not any other tools. What is that they serve for.

David: [00:11:35] Well I guess we use again because someone started to use them at some point. I think that's the case with Pivotal Tracker. And regarding Slack like we didn't use that when I started. We started to use it, we switched from the previous thing because Slack was better. So there's always a place for change. But the reason was because it was here.

Stijn: [00:12:03] And could you maybe explain how you use Pivotal Ttacker like in a general process of developing a piece of software. What are the elements for using Pivotal Tracker.

David: [00:12:15] Well I'm personally one of the people who doesn't like Pivotal Tracker, but in general we would, we have a ticketing system which has integration with PivotalTracker so we can easily create new stories from there by just pressing a button, which is the PM's work, and then the team would look into that, estimate it, and then it would get prioritized. And then we would be put it somewhere, probably some labels. So what is this related to. That is more flow for the bugs. So when something comes from our customer service, for example, the other way

would be that we are working on a project so it would be a team who sit down and tries to make the epics and then try to split things down into smaller, smaller tasks and stories and then estimate those. And then during the planning would look what we want to do the next that.

Stijn: [00:13:22] You said that you don't like Pivotal Tracker that much. Why?

David: [00:13:26] Because I don't have an overview of the... I don't have good overview of the current sprint there.

Stijn: [00:13:34] What type of information would you like to follow?

David: [00:13:37] My first issue is that the actual Sprint is maybe getting 20 or 30 percent of my screen, right? And the rest I don't care about. Unless I am doing something else. Let's see if we are in a refinement session, then it's nice to have an overview of other stuff, but when I am in a sprint I don't care about the rest. So it's hard for me to see the... It's just a very small part of the actual screen where I could utilize the whole thing. And the second thing is the way you handle state. So let's say I have a story which has, let's say someone started on it, but I will not see that this is started, what I will see is the name of the story and the button which says 'finish'. So like when I look at I think 'Is this finished?' because there is a button called finish. So it's not that it doesn't show what state it is. It shows you what's the next state, rather then in what state you are in. And I find it very confusing regarding that. Also the order of it, is more important than the state. So it's not sorting by the state it's sorted by the order.

Stijn: [00:14:52] What do you mean by that?

David: [00:14:53] So let's say that I have these stories. So this one could be here. So it's ordered by the way I've ordered it not ordered by the state, right? It is not like finished first, reviewed one first and unstarted next. It's the way you ordered it which can be also sometimes confusing.

Stijn: [00:15:13] Ok. That makes sense. Yeah I got this question. What do you think in general about the tools and process or basically the tools that you are using right now. Is there anything that you're missing in general within the tools is there a complete feature that you would like to

see but you're lacking currently? Is there any data additionally on what you have so far would make you work or your life easier?

David: [00:15:42] Well, I think most the things which I said before, right? For me I think it's important to a better overview of the spring. And I think there's a difference between I am in the sprint mode and I'm in the mode of doing something else. Let's say like refining, planning, right? They're different and to different use case. And I think those two should be should be separated and that's why I don't like Pivotal, because it's doing the same both at the same time. Of course you can close the tabs there and stuff like that, but the overview is still not that good.

Stijn: [00:16:18] Do you think that in general, just talking for people within the company. Do you think everybody is using the same type of tools or maybe people getting their own tools because they think they're just as you lacking some features, or?

David: [00:16:32] I think people are trying to use different tools if they need something specific, but I wouldn't say they are using things other than Pivotal. They are using Trello for small stuff like personal or like 1 to 3 people have something going on to keep track of stuff, instead of doing Pivotal, because that's also the other thing of Pivotal is really enforces its own, its own flow on you. How you are supposed to do stuff you have to start, and finish, you have to deliver, you have to accept. There is no other way. True. And one of the other things which I actually is that your freedom in splitting stuff or moving things around because, let's say you don't finish the stories. What happens with that? In Pivotal, you can't just remove it or you can just move it out, because then the calculation of how many points you spent kind of disappears, so it tries to enforce that you use the same story for the whole thing. But then it's not that good. So let's say I do something and spend all the time in and then there is some bug in it, it needs to be redone, right? It also comes maybe a question of the general flow. So should I just reopened the old one? Or should I make a new one as a bug? Right ? So if you went for the flow where you actually reopened the old one and then goes back the points disappear. But you spent time working on it and it was kind of delivered in a way. So then it comes back but then the point disappears from calculation because the story was moved somewhere out. So the little flow of that is not really good either. O ver there.

Stijn: [00:18:29] That makes sense. There's some questions in terms of visualization. The tools that you're using or that you describes before. If you would like to have an output together with a colleague. How does these particular tools above Slack and Pivotal Tracker facilitates you cooperating, collaborating with your colleague.

David: [00:18:59] The good thing about Slack is, I think it is good for notifications. So it's having an integration where I can see: 'Oh my colleagues closed some task'. That's good too for me to have an overview of what happened over time, so more like a feed or flow and like a list of notifications of what have happened. So that gives me a better idea of what my colleagues are doing. Can your repeat the question?

Stijn: [00:19:31] You would like to have a certain output because you are working on a project with a colleague you have one role and he or she has another role. How does the tool that you're using, either Slack or Pivotal Tracker, whatsoever, facilitates the collaboration between you two? So how can you use the tools to create a certain output?

David: [00:19:52] Well, as I said, I think Pivotal has a bit of a limitation on how to display stuff, and for me it is important to see the overall state fast without going spending too much time and going too deep into stuff and see who's working on what. It's also important for me. I would say that's the two most important things and I don't have a good overview in Pivotal in either of those. As I said the state is the issue that the way it displays your current state. And also the names it just shows the, shows very little initials of the person making which I think is a bit hard to see. Yeah hard to see, especially if you have too many people with similar names like M.V. , M.A., M.R. , right?

Stijn: [00:20:49] How would you like to see the change?

David: [00:20:54] People have had pictures, you could use pictures, for example. If you didn't have pictures of course then you would see the initials, but having pictures , would be easier to spot faster.

Stijn: [00:21:10] In terms of tracking changes. Because artifacts can change over time of course. How do you see this? Do you see this in a proper way?

David: [00:21:24] No. But again that is something I don't care too much about. I'm not. Some people like systems like Jira or other things where you can basically have the thing from the beginning till the very end and you can see the whole flow how the things happened. Personally, I don't care about it that much. But again maybe maybe someone like a product owner might care more about it. But for my personal things I don't really care that much what happened with the things before or what happened to it after. It got out of my hands, because it doesn't help me that much. I would rather, I think it is important thing, but on my priorities list is closed to the end of the things.

Stijn: [00:22:13] What is the highest thing on your priorities list?

David: [00:22:13] I would say the two things. See who is working on what and the general overview of where we are. That's the highest.

Stijn: [00:22:26] And if you have a test?

Marco: [00:22:30] What was the last one? See who was working on what?

David: [00:22:34] Then that having an overview of the status fast, but you can have it. By fast I mean, I look at it and I see it, other than spend a minute trying to read the thing. That's why the columns are good, right? If you see OK most of the things is there, you see most of the things are unstarted. Then I can easily say OK it's not going well.

Stijn: [00:22:59] If you have a task that needs input from multiple members within the team. So are you working on a piece of software or solution. It needs input from a designer, from frontend, backend developer. How does this work within the current tool that you are using?

David: [00:23:22] You mean communicate more of us?

Stijn: [00:23:25] Yeah, maybe like you were beforehand you're saying this is the objective. Everybody knows the task that they're doing but it might be the case that the frontend developer and the backend developer need to cooperate. Or he or she can only start if somebody else's

finish. How does this work, like this flow, how does it work within the current tools that you are using?

David: [00:23:49] I would say. You can do it. You can have the notion of blockers and then dependencies between tasks. But in general I don't think you should have too many things which are dependant on each other in a sprint. So in general I am trying to avoid that before going to sprint rather than trying to then solve it during the sprint. Of course it happens once in a while out but it's quite obvious because there are only one or two things that are really depending on each other and then know about it from the beginning. So I don't really see, I would say I would avoid it before rather than using tools to get the many dependencies in and then trying to solve it during the sprint.

Stijn: [00:24:40] You mentioned before as well that one of the things that you're really interested in, is what another team member or what is a person doing. Is it the case that it doesn't provide any information currently, the tools again, is that that they don't provide any information or is it too difficult for you to get the information?

David: [00:25:01] They provide it, is just too difficult.

Stijn: [00:25:05] And in terms of where you are in your project so could you state for example based on the information provided by the tool where you are in the process of completing. So can you say for example right now around 50 percent of where we need to be.

David: [00:25:25] I can say that. That's the epics and shows you like how much does this far, like how many were actually estimated and how many aren't estimated and how many did you finished and how many were closed and stuff like that.

Stijn: [00:25:45] And if you're looking for information like on the state's of a project, what is it specifically that you're looking for.

David: [00:25:56] Well I guess, most how much stuff there is left. How much we completed and how much there is left.

Stijn: [00:26:10] And could you maybe can give us an indication on how often do you use the tool for clarification. Clarification of what you find interesting, so you find interesting who is working on what. You can somehow see it but not so much. How often do you use is that in a daily basis in an hourly basis, is it in a weekly basis.

David: [00:26:36] Usually if I am about to start on something then it is important to know like if someone else started on something or not. In general to know if we're gonna finish or not of if we have any dealings or if we should do something about it. Then I will probably look as well. I don't use it that much, because I think it doesn't give me that great overview. So that's why I'm not looking into it that much.

Stijn: [00:27:05] Would you use it more often if it would give you?

David: [00:27:11] I think I would.

Stijn: [00:27:12] Last question from my side. In terms of data it produces a ton of data. Not only quantitative, but also qualitative. To what extent do you use it to make decisions. Or how do you use it how do you use the data within the tools that you have?

David: [00:27:32] Well the important thing is to know like ok 'what is our velocity?', how much can we do given the specific frame and the team strength and that it has different analytics options, but I don't trust it myself.

Stijn: [00:27:52] Why?

David: [00:27:52] Because I don't know how it works. Of course it tries to show you that it's an average of the last few sprints. But it actually it doesn't gives much sense to me. And then there's the issue of what happens with the stories which I not finish. Like unless you don't press the specific button then all those points will not get counted into that but you finish like maybe morning right away the next day and you pressed then accepted, then the point gets counted to different sprints. I know it will not use the correct amount of the things. And that's why personally before we did it manually in the end of the sprint we really counted we finished this amount and then we just write it down in a spreadsheet and you could have the average there for the last

four sprints. Generally in Scrum I think in Scrum it's OK to do things manually, it's OK to do things like that. It also enhances the team's responsibility to be more self organized and you take care of yourself and it's your responsibilities to your team. It's you guys who have ownership of that. It's not that bad thing which is writing down the number once in a while and make an average of some thing. And then regarding data it's hard to do planning there for next sprint because of the way how you move things to the next sprint is a bit weird. Like how do you set it's more like you have to set the numbers correctly for you to grab the right amount of stories you want next, rather than just, you know, you grabbing something over, I this to be next. This is a lot of calculations going on. It could be because maybe we are not using. We have another team said that they were using it really as it was supposed to be and it work well for then, but again I don't think that they weren't using it in Kanban way or Scrum style so it might have worked well.

Stijn: [00:29:58] OK. Is there any question that you have left remaining? Is there anything you would like to add? I mean we talked a little bit about like tools, the visualization part of it. Usefulness. How are you using. Maybe we miss.

David: [00:31:22] No I don't think so.

Marco: [00:31:23] Yeah, well the question that I do most and you kind of already replied more or less is OK. You're now going for your holidays for two weeks they will come back. What do you actually do to catch up as fast as possible. Actually. If you could have that in one screen what it should actually look for your project.

David: [00:31:36] So I will be here before planning tomorrow and I will come a week after. So I will know what we agreed on then I will come back before the end od the sprint. So for me it's gonna be OK so are we gonna do every thing which we said or not. If not it would be interesting to see what took the most or what was... Why didn't we. And it usually is because something took more time or something was blocking us or something to unexpected. Oh we didn't do because this took more than we expected. We didn't estimate it correctly so. How far did we get and if we didn't why didn't we. If I wasn't here planning tomorrow then would it be interesting to see. OK well. In general what are we actually working on right now. Because I will still be here

for two three days to work on the rest of the sprint. OK so what are we working on and what's left. Finished.

Stijn: [00:32:44] Cool. Thanks. That was it for us. Thank you very much. You.

Marco: [00:32:46] Thank you.

Karnov Group - Ivan (Product Owner)

17-07-2018 14:05

Stijn: [00:00:11] And the situation actually that we're in and we're interested in is it in your personal and professional experience within software development and also with the use of the tools that help you to develop software. First we start with some general questions and then we go a little bit more into visualization but the first general question is excellent. Can you describe the company in just a couple of words.

Ivan: [00:00:42] This company? So Karnov is a publishing. It's a publishing house that lives. Well in its core it's sort of a legal publishing house. So we publish laws but the value that we add, that you can't get from other publishing houses is legal publishing houses that we add commentary. So if you have the. We like to use the horse law as an example then you would have the law itself the law text which in general is gibberish because the laws get amended in every amendment is basically a door that contains the amendment. So it's like in line, in paragraph six line four change is to pass. And that's the law change. Just takes the whole law and just changes the word. Yeah. So what we do is we aggregate that into a whole law that makes sense and then we have experts write commentary. So really OK this is how this law could be interpreted. This is some of the precedents there are, this is how we feel in this legal profession that this should be interpreted. So we had that commentary on top of the law and then on top of that we have references to court cases that we feel are a principle for that law. So that's the content and that's sort of the core of business. That content is then offered to our clients via a software as a service solution I guess. So. You can you can basically access the tool via a browser, so it's a browser based tool and you you can search for whatever law you want you can look into textual and bookmarks and you can do a hundred different things. And

the areas that we cover are legal, like lawyers and anybody in public sector working with law and legal professional. We also cover accounting and financial advisors to sort of help them figure out what can and do what can I not do and then we are currently covering but moving stronger into caseworkers. So that would be if you were. If you were had a kid in school you wanted to apply for assistance with transport to school. Because you live out in the country that would be a caseworker for you city that would handle this that, those are the target group for case worker so these would be public sector caseworkers. Mainly in cities and more citizen close departments in a state like immigration services and stuff like that.

Stijn: [00:03:38] And in terms of the organization what is the size and medium sized organization in general?

Ivan: [00:03:43] Perhaps 400 square metres.

Stijn: [00:03:47] And the amount person that work here?

Ivan: [00:03:48] It's hard to say because we just bought a new company. So that adds it. I have no idea I should know this is embarrassing but I would take within company bought about 400. OK. It's not a giant company it is an ancient company. I think we're like 1870 something a little bit before the Internet. You know it's an old company is owned by an investment fund in France called the Rothschild. It's pretty aggressive growth strategy and it's a lot of investment going on. Pretty aggressive goals for developments as well digging into I.T.

Stijn: [00:04:51] And in terms of the team that you are working, how big is that team?

Ivan: [00:04:55] So well so right now my job here is as Director of Product which means I manage and prioritize all the products. But I also work as the Interim PO for. For PRO which is our main product. So I spend a lot of time doing that right now until we get the heck out of resources to find somebody who. Maybe I will stick around. So that's my job. So I am overall responsible for all the product owners. And then I am also product owner for one of most dynamic teams particularly now that Marco's not here. Things are really happening. Get stuff done.

Stijn: [00:06:09] In terms of Agile development what method are you using because Agile is a pretty big method I would say?

Ivan: [00:06:15] So we're using a classical sprint based process with all the ceremonies that's associated with. Refinement and planning, retrospective. We have our stand up every morning. You know we're using a tool using which is Pivotal which sucks. And then we are using a Scrum board like a whiteboard with post-it notes it's all very classic. Now I would say that's the team itself. Looking at the product. It varies. Our team is using a roadmap based structure. I'm going to do it as sort of a steering group or a product board. Some of the other tools are are sorry products are using a more requirements driven approach. We're looking at looking we need to do this to meet these requirements. Okay so roadmaps are a new thing for the company. That we would try to introduce in. It seems simple if you think about it. It's really not. If you're not in that mindset and if you're working in an organization that's in transition. So that's I mean that's a good point about Karnov, actually. Is that we are we're currently transitioning from being a company that was used to working with agencies and just going out 'I want this'. And the agency goes fine it will be 2 billion Karnov was OK here's your money then build it. Where is we're moving into a Development Department that actually owns the product. Yeah that actually prioritizes. And that actually goes 'Nope'. So I get a lot of tickets and a lot of questions like oh I asked this last week why is it not done because there's not a priority. That that's a whole new mindset that people are getting used to. It also that thing with the road maps which we've already planned what we're going to do for the rest of the year. So if you want to come in with a big project. Unless. It's like a. You know it has to be like a high a high value thing for us to change it. So that mindset is changing. So that's interesting. And this of course affects our approach to Agile.

Stijn: [00:08:34] And in terms you mentioned a couple but what elements are you using from Agile you said like the daily stand up, etc etc.

Ivan: [00:08:46] So the ceremonies just like I mentioned. We have. Well if you're looking at the. The Sprint format because because Agile is like a whole whole big box of things for a long time period so if we're just looking at the development cycle and maybe not focusing on the product cycle which is bigger that we're actually using two week sprint we are using a backlog methodology. Using epics. To sort of chunk our different worker pieces. We have our various

stand ups in the morning. We use our Scrum board up there. We have we have a team leader who also functions as a Scrum master. Ideally I feel that roles should be rotating into the team. So everybody tries it. But our Scrum master is really good and he really enjoys it.

Stijn: [00:09:46] Okay.

Ivan: [00:09:52] And I think the rest of the team appreciates that so they don't have to worry about it. Maybe a little bit different for some companies but quite similar to a lot of other companies. We have an embedded designer in our team which means that we have a UX slash designer person that can answer questions on the fly. That means that we can have a little bit more low fidelity. Specs because he's on hand to answer questions and gives us a little flexibility. Other than that it's pretty damn straightforward.

Stijn: [00:10:25] Okay. And you said it a little bit before but like before people asked you to build something and then you basically did it or you didn't do it. You said like your spend your money. But in terms of including your customer within the process how does it work within the process that you have right now?

Ivan: [00:10:41] Again including the end customer? And end user? Well, to be honest, we've I personally have been too busy to do any real surveys or any sort by surveys I mean getting interviews getting out there and working with them doing some field work. What we do is we have. Our UX guy does some interviews he's been visiting some of. The customers we have and another team in the building that does like focus groups, surveys, collect user data. It's definitely one of the areas that we can do a lot better and I think it's one of the places we sadly down prioritize because of resources.

Stijn: [00:11:29] But within the process itself there's not a. How would you say it a fixed moment in which the customer is constantly included or it may be too...

Ivan: [00:11:40] For example we just developed a new... When are you publishing this?

Marco: [00:11:45] 14th of May.

Ivan: [00:11:49] You're busy!

Stijn: [00:11:49] Yes we are.

Ivan: [00:11:54] [Snippet removed per request of the interviewee]
Ivan: [00:13:21] We would like to actually discuss on a meeting today how to do that. How do you make sort of a customer forum over ideas or something that we can do.

Stijn: [00:13:33] Okay. And could you explain me a little bit maybe what type of tools you're using for like your software development.

Ivan: [00:13:39] Right now we're using a I think a pretty wonky mix of tools. And before explaining that I think it is important to understand that we were sort of coming from a startup mentality. We just picked the tools that worked so there's well not even the tools that's worked tools that we could get to work now. Now the tools that worked with the tools that we could get to go work and we could get to work. So for our whole backlog management and our sprint management we are using a tool called Pivotal integrated integrated with GitHub. So those are like development management tools. And also to some degree our product management tool and I am have my backlog in there. On top of that we are using a tool called Freshdesk for our for our tickets in regard to bugs in regards to customer requests for improvements or new features they would like. So that's separate. It's just the only reason for that and customer facing we are using salesforce to deal with that and we're using. Rollbar to do some verification and testing. So for communication within teams we use Slack, so that's a good point. Sadly we don't really have a common repository for knowledge but we are using a mix of OneDrive for documents we're using Yammer as sort of our work Facebook we're using again Slack in a horrible way where we sort of use it as a repository for knowledge instead of using a Slack channel. Terrible way to use Slack. But I'm the only one who feels that way. Go. Because you can just search on Slack.

Stijn: [00:17:16] You mentioned a little bit before. But why are you using these particular tools?

Ivan: [00:17:23] Trust me I would have picked something else.

Stijn: [00:17:25] Ok.

Ivan: [00:17:25] I would kept Slack and then saying you know only chat don't store it any knowledge there, because it's an ephemeral medium so that it existed and then it's gone so it's good for instant messages and it goes for like clearing out. I have this problem right now. Here's this announcement I'm going to be late today. Marco's sick. Yeah Marco brought cake. Marco made David cry again and so David went home. Something like that I mean it's good for those. Or you know I have this pull request or you know Ivan is blocking it because he hasn't approved it or anything like that. Slack can work. Pivotal is really a poor tool. I think. Because it doesn't visualize very well what we are doing. It does not allow you to customize your view take your users stories. It doesn't allow me as a Scrum master or a Product owner to build or use a road map or even. Manage my epics very efficiently. Worst of all it doesn't provide very much data. And. You wouldn't think that data isn't important but it is really important when you work with these things so you know what to optimize. So I dislike and then the visual interface is not very intuitive. It's hard to see where you are in the process. So one day, that's why we have the Scrum board, you look at the Scrum board you can immediately see what we're working on. What is really working on this is very clear right away. But Pivotal is shite. Sorry not a fan. I don't really have an opinion on Freshdesk. It seems a little dated. A little old School. It doesn't seem to be very customizable but Nina says it is so probably it is.

Marco: [00:19:22] Nina is the CTO.

Ivan: [00:19:26] Yeah. And I just wonder why. It's a horrible interface because there is no naming conventions there is no sort of format convention there's no I mean it's hard to to find a way around. You have to know what you're looking for like not just looking for a road map for this product. You have to know what folder is that in. And even if you search for it if the file is not named the way you want it to be then you're screwed. And then what format is it in. Oh no. This was a Microsoft project. I don't have that on my computer so I'm fucked. So that sucks. And then Yammer is nice for intra work for Facebook. Exactly the same as Facebook's but that's also used for a repository. So we have like medarbejder handbog. I don't know. How you should behave when you work here handbook. Which is a legal requirement for companies in Denmark. That's in there, in our work Facebook it's kinda, but that's it. So we need a common repository of data. An Intranet in real life.

Stijn: [00:20:40] OK. You said something that data is important. What what type of day would you like to see in there?

Ivan: [00:20:45] All of it. That's a big question. You mean particular data points and just the types?

Stijn: [00:20:55] What would your interest be to see if you?

Ivan: [00:20:58] My interest is this for example given to be able to I can see burndown out right now I can see how fast we progress through the sprint and stuff like that but also to be sort of be able to pull out how much of the sprint is bugs. How long does it take to resolve bugs, how long has the bug stayed in the backlog before they get resolved. How is these different epics doing against each other, or how are we progressing with this. I mean how many of our tickets are reopened from Marco and so forth and so on. I get... it's hard for me to sit here and just pull out a mess, but I get a little data from Pivotal but I can't filter for example. I can't do filters by epics and so forth and so on which makes it hard to use. There's also a lot of questions like. How many how many blocks do we get on things of things we have just released.

Stijn: [00:21:49] Okay. And in terms of the tool that you're using I'm not sure if you're aware of it but is everybody using it in the same way everybody using the same tools are there also people who just maybe start using their own tools or...

Ivan: [00:22:01] I know some people using Trello for some different things. But I think in all the product teams we are using Pivotal to manage our backlog and stuff like that. As far as knowledge repository I think those are the tools. When Trello is used again for some stuff for like. Task management, right? Now. I wish we had one tool to rule them all. I'll be fine with the Trello stack which is Atlassian which is Jira, Confluence. If you would like to produce a certain output together with your colleague how does the tool that you're using for example Pivotal Tracker and how does this facilitate this process. I would say it could be of course I mean like any. Like any tool like Pivotal. Would facilitate is just managing the priority of the tasks and the workflow of the task who's who's who's owns the task right now what are you doing with it. What phase it is is it in. This is not particular to Pivotal but any sort of Scrum based, sort of Kanban

based tool. It just how you make sure that a certain production goes to all the right steps specified. It's open it's ready. It's in development. It's in code review. It's a functional review, it's in test phase, it's ready for release. So that's basically what it does for us and then it's a repository for requirements for each user story. So those go in there as well but that's not unique to Pivotal. That's a process we can do data on post it notes on a whiteboard. What the tool could help us with is automating with a lot of tasks into this one what it does for us for example is integration with GitHub of course you couldn't do that with a post it note. But, so that's one thing. The tool could help us with data, but it doesn't. And sorting and managing your tasks and stuff like that. And of course it helps us to chunk tasks so we can always see all these tasks are related to this particular epic in the Kommun Saga. It helps us structure like that. But again. I mean you could do this with Post it notes, it's just easier. Yeah. But when you look at what you need as a Scrum Master and a Product Owner it doesn't facilitate that. So that sucks.

Stijn: [00:25:35] Yeah I can imagine we're here to solve that problem.

Ivan: [00:25:40] One other problem we have which I think is interesting is, when somebody submits a ticket to customer service be it internal or external. They sort of lose track of it. It is submitted to fresh desk. We get a notification and if we decide as Product Owners that it's important enough to go to the backlog or relevant enough. It goes into the backlog. But there is no connection back to Freshdesk. So the original user that submitted it be it internal or external or customer service in case of an external can't follow a ticket they can't follow the user's story generated by the ticket. They can't see if this bug is resolved or not and that gives a lot of the manual overhead.

Stijn: [00:26:20] Ok.

Ivan: [00:26:20] Which sucks. And to that extent I assume that you use again other channels or other tools in order if at all you inform the person who inserted a ticket.

Stijn: [00:26:23] And to that extent I assume that you use again other channels or other tools in order if at all you inform the person who inserted a ticket.

Ivan: [00:26:31] What we are basically trying to do is when we write our release notes we try to look them up in Freshdesk and write a comment on Freshdesk. And say ok I resolved this. Close it. It's a manual process extremely error prone.

Stijn: [00:26:44] Ok. And in terms of tracking changes in the project artifacts I assume Pivotal Tracker can facilitate that as well.

Ivan: [00:26:56] It can facilitate changes but it can't facilitate tracking changes so that's a pain in the ass.

Stijn: [00:27:02] Okay.

Ivan: [00:27:03] So if I can't pull out data saying what is the change in story points. So if I can if I could pull data saying general in every sprint we've added forty story points. During this sprint. We have to do something about that. I can't there's no way to do that. So that's one of the points where I am like it's not good enough.

Stijn: [00:27:23] OK. And if you have more tasks or if you need to do handle more tasks with input from more than one team member how does the tools that you're using facilitate this?

Ivan: [00:28:05] Fragmented? So a good case would be we have a bug. That maybe I am assigned to Marco, Marco looks at and goes. Yeah I built this, I remember that I built the core code at some point and I fix this and then Marco get to a point where he's like. Martin was in here and fucked around and he's seen have a relation to this bit of code and it might have changed and it might have fucked up the CSS over here. So I need to talk to Martin and then my, I don't see these conversation, but based on my experience is that that will generally happen on via Slack. Even though sitting right next to each other, but developers dude. Why talk to each other when I can type to you?

Stijn: [00:28:48] Of course.

Ivan: [00:28:51] Which is I guess it's practical to get shit done but it would be better if we had like a common trail on the tickets or something so we would have the knowledge preserved.

What's going on here. So if Marco decides to elope back to Brazil and to live of [inaudible] again. Forever. It's totally totally possible. Then we have somewhere to pick up on now. So again that would be nice for a repository. I have no idea where to put our software documentation.

Marco: [00:29:23] There's no such a thing.

Ivan: [00:29:26] See, that's Confluence. That's why we need the Atlassian stack.

Stijn: [00:29:45] And in terms of the process of your projects. How do you measure it? Like are you aware of where you are in the process? Would you like to be where first of all?

Ivan: [00:29:54] Yes.

Stijn: [00:29:57] I mean some people...

Ivan: [00:29:58] Respondent nods vigorously. Yes. Of course I want to be aware. There's always a difference between what you track, rigth? So it's hard to track the progress of a bug that's in a sprint because you can see how it moves into a sprint and then it's done. Assuming what you're asking is if when we have bigger projects like the new Kommune we're doing and that's all divided into Epics. It's all been estimated. So I know how many story points are. And then it's just a simple matter of saying what is our velocity per sprint and then I can just break it down and then I can track if we deliver on schedule or not. And if we deliver on schedule they get stake and if they don't deliver on schedule they get whipped.

Stijn: [00:30:38] In terms of the project or project status. What is the most specific information that you're looking at from your goal?

Ivan: [00:30:49] Progress towards [inaudible]. So there's different things I mean if we are tracking from a development perspective that would be burn down to to deliver it. But then there is other products related things like I need to to keep up on what's our go to market strategy. Is marketing really is our content department ready with the content that needs to go in here. Is our sales department trained, ready to go. Is our educational department are they up to speed on

what's going to come. It's our customer support department already. So those are also things I track. What is their progression. That doesn't touch our developers luckily. So that's good. No I would say burn down from a development point of view it is epic burned down.

Marco: [00:31:46] But that's actually a good question how do you keep track with the other departments: marketing and sales...

Ivan: [00:31:52] Meeting. See if we only had like a common repository...

Stijn: [00:32:28] We have two more questions. So you don't know. Next question. How often do you look into these tools. And ask for clarification look into certification is it on a daily basis. Well you try to look for certain information?

Ivan: [00:32:52] Every day. I would say if you mean by clarification if you mean like analysis if I go in if I need to know the answer to one hypothesis started then I would said that about every two weeks right now. Just go and see how are you doing, how was the sprint, are we performing. We do that together. And I would say I handle a lot of quantitative data collection whereas David our product our Scrum master he gives a lot of the qualitative stuff in the team, to make sure that they have and stuff. I couldn't give a shit. They just have to work hard, fast.

Stijn: [00:33:56] In terms of the questions I have over here. I think I'm pretty good. Yeah. You have any additional questions based on all that notes that you make?

Marco: [00:34:09] Yeah one of the things that I was wondering here is actually OK let's say you were going on holidays. And then you come back. And you want to see in one screen. What the hell have you missed in these two-week holidays what would actually look first.

Ivan: [00:34:27] It depends on the sprint, because the last time I was on holidays for example we were sort of in bug crunching sprint. So in that case I would just look at into the release notes and look at what was closed again Pivotal shitty for that, because I can't just go and see what was closed in that sprint, I did a huge workaround to get that data. But normally I would just look it up and say OK what was closed in that sprint. What was what was the left over, what was pushed over from one sprint to the next. And then I'll go why didn't we finish that. What was

this? What was the release notes. Just ask about it. And then look at. Did we meet velocity. I mean hopefully that sprint would be prioritized in the backlog. So even if I was gone for almost 4 weeks or two months the developers with a good backlog should be able to just work. And then of course I would need a proxy to verify and approve the development and it's done. But I mean Tim, our visual design could do a lot of that if the backlog is maintained and nice. It is not right now, by the way.

Karnov Group - Martin (Developer)

13-03-2018 15:02

Stijn: [00:00:11] To start with we'll get some some some general questions could you maybe describe what the company is doing what you're working for? The company.

Martin: [00:00:22] Basically at this point we're publishing company . We publish law in Denmark and Sweden. We annotate and augment content and I'm part of the tech department. We do kind of the digital publishing part specifically the pro platform which is where we actually publish content and where customers consume their content. So that's the kind of that aspect of the business where I focus on, customers consuming the content that Karnov sells basically.

Stijn: [00:00:57] How big is the organization as such. And the team that you're working in? Like to have an idea of the size.

Martin: [00:01:04] So I'm working in and I'm working in the pro team on the platform team and we are six. I think in total. Our tech department of around 40 in Karnov which is in Denmark around 20 people. And now with the acquisition I guess have the same amount in Sweden. Until recently Sweden was much smaller business that Denmark.

Stijn: [00:01:41] And in terms of the composition of the team like you said you were with more or less 6 people.

Martin: [00:01:48] Yes we are technically titles' are always difficult but we are 4 backend developers and 1 frontend but then at least multiple people are fine working on the front end as

well. Both Marco and David have no problem jumping at front end issues and me and Rajesh have also we traditionally were very backend focused. We have also been doing react stuff.

Stijn: [00:02:22] And then there's one person left, I would say.

Martin: [00:02:26] Yeah that's Tim. It's our designer. And has a cross team function. Technically there's also Ivan who is a PM, product owner slash also cross team.

Stijn: [00:02:45] And you talked a little bit about it but what is actually the methodology that you're using because we base it on Agile but that is a fairly broad term. So to say. Is there any type of specific methodology that you're using and if so how?

Martin: [00:03:00] I have no idea what terminology is but I can tell you that we do two weeks sprints at the end slash start of each sprint we do a kind of postmortem celebration type of thing and then we do planning the next sprint. And then in between those we do refinement of tasks. So before the end of a Sprint or the beginning of the next specifically we should have tasks refined.

Stijn: [00:03:33] So you have two times a weeks and in between there is like a day or maybe two days for refinement. Is that how you?

Martin: [00:03:42] Well it's maybe 30 to 60 minutes a refinement to maybe two days during the 14 days.

Stijn: [00:03:48] And in terms of both starting the day like the daily standup is that what you also use.

Martin: [00:03:53] Have this in the morning like 9:50. Everybody is here. Yes we do very short , stand up. Now everybody tell us what they did yesterday and today and specifically it's just keeping tabs on everybody. It's a good place to raise any blockers that you have if you are depending other team members or even external stuff maybe you need the PM maybe you need the designer to pitch in.

Stijn: [00:04:26] And from your perspective as a backend developer what you often encounter is that your requests are coming in from customers for example. How was it that you, do you communicate with the customer at all? And if so how do you communicate? Do you use tools for that? How do you receive information from that? Is it directly, is it in another way?

Martin: [00:04:50] I don't communicate with customers, at all. We have kind of a multi tier process around that I would say. So we have our support team that handles most tasks. So I would say that requests and reports come in from either the trainers who have a lot of access to customers and then get a lot of feedback and from our support team phone, email support so they get a lot of stuff and then I don't know how to categorize the rest but there are probably more. And then that goes into a ticketing system. So they basically feed in, so the customers can't feed into the ticket system not that I am aware of. They feed into our internal ticketing system and the PMs as a group will figure out where to put the tickets. And as of now I don't think we really have a way to prioritize tickets. So that's progress. So I think basically from there from the kind of central tech ticketing system to our backlog there's a bit of prioritization going on. But it's not a biggie.

Stijn: [00:06:20] And it is out of your influence that is somebody else who is arranging that interaction between what the clients or customers would like to see or where would like to have improved. And it's not directed to you. They will never communicate directly to you.

Martin: [00:06:37] No no.

Stijn: [00:06:41] Okay. Could you maybe explain what type of tools you're actually using during the software development. So we heard a couple of tools already which includes Slack and...

Martin: [00:06:52] For planning?

Stijn: [00:06:52] For everything basically for communication, for coordination.

Martin: [00:06:57] So we use the most important one is Pivotal we use for tickets backlogs, icebox, all that stuff. And for comments, my favorite feature for any ticketing system and that is, that in combination with Slack and like tapping people on the shoulder is probably ninety five

percent of my communication. In other places I've used e-mail but I just don't find the email tools that we have in Karnov very good. So I don't want to use it. And also the kind of the spam on Slack is not so bad in Karnov compared to other places. So I enjoy using them more. Slack just hits a wall when you reach maybe 100 200 people and the noise is so heavy that you can't use it for anything. Just turn it off and then you use e-mail instead. I use GitHub a lot. I said 95 percent but it is not entirely true. GitHub and pull requests and comments there is also a very important tool for communication internally in our team. We also use it on externally but it's really kind of the developers and the PM. So before we released it's a fairly new process we create a pull request to production and so we kind of list in a nice way what kind of changes high level that we are deploying to production. And Ivan will aggregate that into customer facing changes and post that to our internal communication tool which is called Yammer. I'm a consumer of Yammer but I don't post anything in there.

Stijn: [00:09:00] Okay. And you mentioned Pivotal a little bit. Could you maybe explain why you're using Pivotal for what purposes it has been used.

Martin: [00:09:10] Yeah we use it for everything to related. Right. So we don't work on anything generally speaking that isn't in there. So for bugs, we also do it for planning chores as well as features and so forth. So bugs come from the ticket system, gets refined by the PM, goes into Pivotal and gets prioritized in our backlog and features and chores, we usually have like some high level goals and we find those into individual tasks. So as a team we have a session, a refinement session where we like decided we wanna work on this feature we want improve this feature built this new feature. And so there are some high level requirements for that and we will refine that into what do we actually need to do to make that happen. So all that stuff goes into our backlog. Yeah and there's one backlog prioritized.

Stijn: [00:10:15] That makes sense. What do you think in general about the tools slash processes that you're using.

Martin: [00:10:22] The commenting system works. So I'm happy. That the commenting system works well

Stijn: [00:10:30] Why?

Martin: [00:10:31] So it's it's it's for documentation basically. So you've spent a lot of time investigating problems and if you don't write it down the next time you have to investigate the same thing once more because you can't remember what you did three months ago. Is possible. So I like that. Even just like between days when I go home , I'll add a comment to a ticket saying what am I doing because then the kind of the overhead of starting up the next morning is much lower. Now so between that and statuses, sure I shouldn't get into what I don't like, right? So we basically have the tickets there. So the time we actually in combination of Pivotal we also use a whiteboard. Yeah. So there's a bit of overhead making that obsolete and I don't really use it that much.

Stijn: [00:11:34] The white boards you mean? Yeah.

Martin: [00:11:35] But I think it makes it more visible what we're doing or that we're doing something now. I think that's the main purpose of having that. But other than that Pivotal for me it's pretty straightforward. Like you have a ticket has a title, has a status, and has comments and they have some tools for like for dependencies like this ticket depends on this one and that can definitely be useful. But I don't think it's quite powerful enough. I've used Jira in the past and there you can have like the problem is that a dependency is kind of a hard dependency, right? You can't solve this ticket until this one is done. But there are lots of soft dependencies like. So this is related to this one. I really miss that in Pivotal. So that's the kind of the primary features also. But that was annoying for me the last time I used Jira as well. But the kind of the statuses is like every software likes to have like different workflows for different types of tickets. I think it's useless I think it should just be the same. Like we go through the same kind of steps regardless of what type of task it is. And if for one instance we don't want to go through that step then we click twice on the bottom. There's no reason to force me because then I don't have the tools available that would otherwise have. So I have a lot of tickets I will change from chore to feature in Pivotal, because chores have unstarted, started, and finished, right? And they don't have kind of the, it doesn't fit our workflow because our workflow is always like you make a change then you make a pull request then that gets viewed and then that goes in and then that gets deployed right. So that process is true for every every change that we made, right? So that's me. So I don't know if that was the question.

Stijn: [00:13:39] And you said something about the whiteboard that it provides some type of overview was that something that you... Could you state that that's something you're missing within the set of tools that you're using right now? Or maybe it's not their function. You can also.

Martin: [00:13:59] Yeah actually I think I prefer it's also kind of a mess by default in Jira , right? But basically in Jira I would just have I set up a filter for myself. So my own tasks that I need to close. And that's kind of the view that I want. Right now in Pivotal I have to every time, I go in there I have to control find my letters, my abbreviation. So MV, I search for that then I can find my tickets. It's the same thing I do every time. So I don't really understand why that is kind of the default view. So it's good for picking up new tasks. That is a thing I do really like once or twice a day you know where like finding that ticket might be something I do 10 times a day because I close the tab finally again or maybe I'm have two three tests open maybe I context switch between those then I need to be able to quickly find those and there is my tickets in there somewhere. So it's also a matter of personal process. Yeah. I don't really use the whiteboard for that.

Stijn: [00:15:18] Good. Do you think actually that everybody is using the tools the set of tools that you're having in the same way or is it maybe that you for example have not added one tool because you think the tool sets that is being provided by the company is not sufficient.

Martin: [00:15:39] I think it's I think it's sufficient. I think people use those tools in many different ways. So a lot of people don't use the comments at all. For example that wouldn't work for me. Yeah that doesn't mean that other people can't do that. But then again then people have other requirements of those tools even though the developers are maybe now I'm thinking about David our team lead. He has some specific scenarios that he wants support with the tool and he doesn't think Pivotal support those but I don't really have those requirements so I don't know if it's in the position of team lead that you would have those requirements or if it's like a personal preference. And then obviously if you go beyond that into PM and stuff and you will probably have completely different requirements. Where I am coming from? If we use... if the tools that we have are sufficient enough? I guess, yeah. But I really would prefer a kind of better linking between tickets and would really like having that especially like for just tickets that are related.

Stijn: [00:17:13] You mean the linking between tickets?

Martin: [00:17:15] If this ticket relates to this ticket. So not like a child parent type thing more like this, this has something to do with this. The other one in this particular context I mean.

Stijn: [00:17:29] OK. And now we have a couple of questions in terms of visualization. You mentioned a couple of tools. I was wondering how does how does these tools help you to produce a certain output if you have to work with a colleague. So you're working with a colleague on one project one front end and one backend and how does it work? How does the process look like and how do the tools that you're using help?

Martin: [00:18:03] So one aspect of that would be a lot of our features are prepared by our designer Tim. So he will do some footwork and gathering requirements maybe even talking to customers if that's possible. And he will basically condense that into a design in some tool called InVision so basically mockups or maybe a bit of interactions which is a good foundation. I think. My old company we also did something more static but also did mockups which help just convey the idea of what are we actually doing here. So everybody's on the same page between me and the front ender. It's actually kind of muddy here in Karnov because it's not so segregated between front ender and back ender and because Johaness would do what might be considered back end work or at least plumbing. And the back enders would also go into front end related tasks. But we pair a lot, so most of the time when I need Johaness for something or he needs me for something I'm kind of pointing him out because he's the kind of dedicated front ender, right? So there is a distinction at least in the title and... And we will pair on it, because that's the quickest way to get feedback I think. And then through pull requests I think, someone just reviews on those and anyone could comment on the pull request. So that would be like Tim the other day requested to be assigned on you UI related things so he can kind of get a change because it's kind of the final step before going to the integration environment. So being put into the loop there where he can kind of comment on the actual implementation of his vision of the design. And the same goes for when the front ender turned us back end stuff and the back ender does front end stuff or there is a requirement but usually I think the kind of an ideal world I would do the back end and I would maybe stub out some front end related components like maybe the mark up without any styling and then I would pass it on to the front end developer to finish that implementation or vice versa. The other way around also works fine. So basically

getting a styled mock up in the app and then wiring that up to data services or actions, whatever. So in terms of tools it's mostly actually GitHub and face to face maybe a bit of slack.

Stijn: [00:21:28] And where is Pivotal in this instance if you work together in a project? That I did not follow.

Martin: [00:21:35] Most of the time we would actually have a separate task for Pivotal, so because usually we split up from like strictly front end stuff from strictly back end stuff so they have so they're not dependent on each other. So you can finish one task without finishing the other. You can assign one task to one of the other tests with the other and then there will of course likely be some kind of coordination, communication which would happen through the other means. Right? Like Slack or something like that, to make sure that you're not working on the same things or doing the wrong thing basically.

Stijn: [00:22:20] Is it somewhat annoying that you have to have multiple tools communicate this with another or I mean you assume that it's normal but if you think about it you have multiple channels through you can communicate and coordinate. Does it provide any disturbance within the process of developing it?

Martin: [00:22:53] It does like to spend some time just linking things together like Pivotal and GitHub for example. So I think I'm not a particular fan of Pivotal so I don't hate it either but I'm used to use external or other tools than GitHub for managing tickets, right? Basically never used GitHub ticketing system for anything. So I'm kind of used to having that kind of what you call that? Siloed experience. But it does takes something that includes some overhead like every every pull requests, every commit basically, I would point to a ticket somewhere and in the ticket I would usually place that linked to the pull requests as weel. And there is integration. But then the integration will change the status of the ticket and I don't want that which is annoying. So but I think Marco find a syntax that actually works to do something without it breaking everything. Yes there's some overhead in that. In communication I don't know if, because that would assume that you could have like realtime communication from inside. Like let's say GitHub or some other application like the kind of have these individual applications be the best in every discipline. And that's kind of why I think that's why there are so many different products. They serve different purposes or Slack can be good instant messaging app basically a decent one, I

won't call it good and GitHub can be good because it is truly good doing source code management and change management, basically. Pivotal is what it is. It serves it's purpose I think and it has low barrier of entry. Basically just get started. That's why people use it.

Stijn: [00:25:14] If there's any changes into artifact during the project how how do you know that. How do you how do you how are you aware of that change?

Martin: [00:25:26] Change to what?

Stijn: [00:25:26] Her like an artifact within the projects or something is changing. I don't know maybe you can explain a little bit more into detail.

Marco: [00:25:36] Yeah it's actually any piece of documentation, right? When something changes here in Karnov. How do you get to know that? Here I mean in the pro let's say, right?

Martin: [00:25:54] So I have a very specific example which from today many other days is that the designed document in InVision changes in the way you like because there you can create projects in there but then every time that there is a new iteration or something we create a new project and then I'm not add into that. So that is a very nice disconnected process for me and you can actually it's not so bad for. Because I usually don't need to reference the read like I referenced the functionality of the actual layout. So if and usually that's what goes in the last right so that's usually the parts I'm missing because I have an old, specially it feels like passing around Word documents in the late 90s that you have to steal copy of the thing which makes no sense. Terms of documentation, it's not a problem. We don't have any documentation at all. Actually we have another product project called Document Service. They have started documenting a lot about our data model like in the company not specifically to my product. But there are a lot of shared content that many different applications use. And that has been at least it seems that way has been not documented at all or it's been very hidden. Basically in code or in XSLT stylesheets somewhere in an app that you need to know was there. So I think that process has started. Now we also have a dedicated team to solve all that stuff. So I'm hoping that this will be basically be a Wiki style web based resources with a static URL like so you can put in references to that every where it's relevance and that content will be updated. The reference will need to be updated because that's usually like you link to some document with

some random generated ID and then that document is no longer the right document then you have to go to another one. All right. So I, I hope we can use something like GitHub Wiki pages for most of that aspect of it.

Stijn: [00:28:26] And if you would like to be aware of what one of your colleagues is doing how do you do that yourself?

Martin: [00:28:34] If I want to be aware of what they are doing?

Stijn: [00:28:36] Yeah. So for example you need to work with a front end developer and it has strong dependencies on each other. How were you aware of the fact that he's done he's finished or he's working on something right now.

Martin: [00:28:54] Basically I would say I mostly to rely on kind of ambient awareness. So I follow everybody can and should look at a ready pull request that comes into the platform. So through that I know of the changes that are happening now, that happened in terms of what people are working on in this instance. I would say that I don't know and I don't care and if I need them then I will try to ping them on Slack, even though they are next to me because like people might have headphones on they might be in the zone, working on some specific problem. So I think out of respect I would prefer that. But that being said half the time I would just poke them on the shoulder bother them saying I do need help. Yeah. So I try to be respectful of their attention. At the same time sometimes it's also human contact, right? That's also a good in and of itself.

Stijn: [00:31:35] And in terms of knowing where within the process you're currently in. So you have a big project it runs for like half a year. Are you aware of where in the process you are right now like you on 50 percent or 70 percent? Are you almost there to provide the deliverable. Are you aware of it and are you actually interested in it?

Martin: [00:33:06] I I think I am but probably a very high level. Like I think probably I will come at it from bottom up. So how are we in terms of this sprint right. And my tasks and our team's tasks and where can I help. There's also usually a problem right like steal tasks somebody is working on something and you're not working on something. Should I jump in or should I not jump in and

then to answer your question I guess. I feel it's very important to know how our team is doing. So from that perspective yes I wanna know how we are progressing towards kind of the goals that we have now but not not in further detail than that. Like not in a gamification aspect of it like I don't care how many percent of a given thing that we're like just are we roughly are we doing well are we doing bad. I think that's that's the perspective that I would like to have while keeping my head low. Having an idea of how things are overall. Okay.

Stijn: [00:34:30] And if you're looking for a status what is the information that you would like to know what is it what confirms to you like we're on the right track yes or no?

Martin: [00:34:44] It's a good question. I think what would traditionally happen wouldn't satisfy me in that respect would be that we get the... We sit down with our PM we walk through the roadmap and compare it to our backlog which is usually split into, split into epics and tasks. And so we kind of compare the two in X number of weeks or months we expect to be here. These are the tasks that we have. These are the tasks that we have completed because history is actually the most important factor in the velocity basically objects of subjects of like one is the number internal for example like what is, what is called Burn rate or velocity. But also if you look back you can probably get an overview like. Because every sprint stuff sneak in and like how much of our time are we actually working on the. Like the roadmap will have one thing on it like we work on this thing. But our backlog will have at least three things, right? It will have bugs, maintenance, and chores and then it would also have something related to the features that we want to get some. So I'm saying subjective because like maybe you can look at that. Look look at the kind of the overview and then you can say feels like we have spent most of our time doing this stuff and maybe you can back that up with actual numbers split in a sensible way. I think it feels like that is important. We don't really have much visibility on that right now. So like where are we spending it, like banks basically right where they start like tracking monthly purchases. You can kind of see how I'm spending all my money on takeaway instead of the other thing, right? Where are we actually spending our time. And then what does that mean in terms of our progress and how you map that too. How we call it? Lost track... In terms of like the overall planning. We don't have that in Pivotal I think at all. I think that's one of the reasons Ivan is pushing for Jira because you can do the roadmaps in there and you can kind of plot them against your team's backlog velocity and stuff like that to kind of get a sense of how the two things fit together.

Stijn: [00:37:44] Okay that very much relates to the next question that we had is that in what way or how do you rely on data that you currently have but feeling from this. It's less than desired.

Stijn: [00:37:57] Actually don't use the tools very much. Yeah I think that's been there it's very important especially because I think I just had some revelation lately about how much time has been spent on that. So Karnov has some time dedicated to maintenance or care which is very reasonable but actually pretty words and numbers all that stuff actually means a little bit. And plus when you extract that. So just the maintenance part how much of the remaining time are we actually spending working towards the main goal compared to all the sidetracks that you run into. Like you also need this like we had a project come in. People were aware of it but it ended up probably spending 2 3 sprints which is a very long time on something that was actually on the roadmap labeled for something else. So of course we didn't and we felt like we did some good in the context of what we were like in reality. But it could have we could have basically had some context is necessary. We had a kind of maintenance period where we're trying to clean up some technical debts, basically, and we had some time for that in the roadmap and we also managed to dedicate a fair amount of time to that technical debt. But as we ran into it all as immediately as we went into that another project came in and took off 50 percent of our time which was not on the roadmap, right? So on the roadmap and to our communications stakeholders we will have been spent. We should have spent a lot of time on tech debt. So basically we should be tech debt free right now. But in reality was much less. So for our own sake but also in terms of keeping the road map transparent and accurate it doesn't have to be detailed but it should represent reality. So I think it goes both ways.

Stijn: [00:40:25] OK. Last question that I actually have is a fairly straightforward one but how often do you use tools in order to be updated or ask some clarification about requirements.

Martin: [00:40:39] All the time. Comments in Pivotal all the time. Yeah. So the communication you often goes beyond Pivotal comments because not everybody uses it the way that I'd like to use it. Now in my previous company we used Jira and everybody used comments. So all requirement clarification basically was on the ticket to implement that feature. So even if you had to go back to it or maybe if the feature got blocked somehow you could get two months. So

you come back to it and all the refinements is in the context that it should be, it's not some private Slack conversation somewhere that you can't link to. So that's my preference. What I do to compensate is that I copy paste stuff, right? I link to Slack conversations which is an awesome feature, I copy paste conversations from private Slacks into the actual tickets so I have in the context that I want. I guess that's mostly it.

Stijn: [00:41:55] I think one last question that I don't have here but it's about the holidays. That is. Imagine that you went on a holiday for like a month or maybe two months. And everybody continued working how do you make sure that you're updated again about what is being done so far, what they are currently doing and how you can smooth into that process.

Marco: [00:42:23] It's kind of what would be the first thing to look for, right?

Marco: [00:42:28] Yeah I have done it a couple times. Actually, I went on a vacation for five months. Yes. So coming back there was that's maybe a little extreme case maybe not the best reference but that that took awhile took like a week or something like that to get my mind to the game again. But basically coming back from vacation I would read up on e-mail to catch up on basically just to filter out all the crap and then get into if there are any new things that you need to worry about then in terms of how the code has changed I doubt I would have liked. I don't think I would go through all pull requests and review stuff to get up to date. I think I would basically look. I think I would ignore all the stuff that happened while I was on vacation at least until I have to deal with it. And I think I would focus on what the tasks in the sprint which I probably didn't plan maybe didn't even account for me in the velocity. So I would probably look at that and then I would maybe find something maybe not. And then I would pick from the backlog and I would just take it from there. So and then of course that would probably bring a bunch of stuff that I don't know anything about. And I think usually that would require some kind of verbal introduction to the changes. I think that would be the most efficient way to do that. At least for me maybe not for the person that has to explain it.

Marco: [00:44:11] It's like being rehired, right?

Stijn: [00:44:18] Okay that sounds good from my side. I don't know if you have any additional. Is there anything that you would like to add based on what we just discussed so far. Maybe that's something that pops into your mind like 'I totally forgot'.

Martin: [00:44:39] Well my initial thought was like listing out all the problems that you run into on a daily basis so that would be requirements, missing requirements specifically. So I know you are more of a process that goes through throughout the whole thing, right? So missing requirements is a big thing for us especially for bugs as well. Yeah specially for bugs and features just the same. We don't have all the stuff that we need. Sometimes that goes to the customer sometimes it goes to the designer, it might even go to the front ender, there there is context switching which is a big issue as well.

Stijn: [00:45:30] What do you mean by that?

Martin: [00:45:31] I mean you work on one thing that you work like then you're now I'm saying interrupted but I don't want to focus on that because you can get somebody can poke you on the shoulder and that's fine but just

Marco: [00:45:47] Someone come desperate and say we have a bug.

Martin: [00:45:50] Yeah yeah just a just when the backlog just something changes priority, something in the team chat that makes you like that requires your attention in some way, that will waste a lot of time, there is no probably no silver bullet for that. But it's a big issue. So that was requirements and context switching and then I guess dependencies. But you touched on that, right? So if you depending on the front ender or the designer. But also test dependencies but usually there pretty good tools for that. If you are good at mapping it out and trigger if you speed stuff into reasonable parts and you make sure that you have the correct dependencies mapped there then you don't start working on one thing until you have all the requirements covered. I think that's it.

Stijn: [00:46:54] OK. Thanks. Yes that's good.

Martin: [00:47:02] Good luck with everything.

Marco: [00:47:02] Thank you. I wrote three pages of stuff as they come and go.

Stijn: [00:47:12] But nice to meet you too.

Exitable - Maikel (Scrum Master)

03-16-2018 9:02

Maikel: [00:00:00] ... And everything that works in line and strategic planning goals targets building of the platforms and maintaining them. OK. And we work together with Ketchapp. That's an app development company. And so we have a lot of things in common share a lot of knowledge and use the same principles for managing a project.

Stijn: [00:00:27] Is that like one company that is split up into is like two separate companies that work a lot together.

Maikel: [00:00:33] It's two separate companies that work a lot together but they're both from the same owners.

Stijn: [00:00:38] OK. OK. Like that makes sense. And in terms of your organization what is the size of the organization and the teams that you're working in. Could you tell us a little bit about that.

Maikel: [00:00:49] Exitable has 14 people. Yeah we have two main teams. Both teams are made a made up of a project manager and also does the account managing so the client side and three team members a designer front end- developer and a back end developers. We have two of those they have their own planning and own projects and clients. Then we have marketing team on the site that is step in when needed. And then we have the app development team. So that's a third development team. And Ketchapp is four people big. So that's the... In total it is 18 people that work in the same way and work together.

Stijn: [00:01:43] And it is split up in two teams you said? And within this team what is what is your specific role?

Maikel: [00:01:48] My specific role was back-end in the team. And now I'm out of the team and someone took my place and I'm busy guiding all four teams and working better and automating things that now takes a lot of time just helping them grow and be better at what they do.

Stijn: [00:02:12] Okay a little bit on a higher level. Okay. And in terms because our thesis our research is about Agile. First of all, do you use agile.

Maikel: [00:02:25] We do.

Stijn: [00:02:26] Okay. It's all a broad concept. Could you maybe explain which methodologies you're using or what concepts are using from the theory.

Maikel: [00:02:37] We're not using just one,. The one that fits the most is scrum I think. We started with getting all planning for every team to get it set up and sprints of two weeks. I think one and half year ago we were doing everything and not doing it really efficient. We had the standard what is it ... A waterfall method of doing projects. Everybody knew their role. We just did their part of the project and threw it over the fence. And then the next that team picked it up and started building. It wasn't really OK for how so say.. Commitment and getting people to know what they were actually doing and what they were doing it for and an attitude to getting get a good result. So we started looking into agile and the chances that come with it. And we looked at our company our people and how we could find a way to implement it without getting too hard on all the principles and be too how do you say it we would we wanted to implement that agile and not be not made a manifesto and get the rules to everybody and just this is how we do it now. Yeah. So we started small we started working in sprints in two weeks and fitted every project in it. Yeah. So everybody's doing that now and the last period is getting all the clients with it and using and really building every project and everything we do with them and sprints of two weeks, get them to know the team and get them to work with us on projects.

Stijn: [00:04:33] Yeah. And is there, are there any other specific elements such as the stand-up, such as retro's such as...

Maikel: [00:04:41] Sure every team has fixed two week sprints. It starts with a kickoff that's Monday morning. That is always blocked. There are no client meetings or whatever. Everybody's here. Yeah. We do the retro of the the previous sprint and we list the three things we're going to work on the next sprint. Then we get to see what resources are available for which project. And the teams make up what is doable in that Sprint or in the resources they get for each project because we have a lot of projects and then they make up the sprint planning for a week, we have two really big sprint boards. This is every day shown they use sticky notes and that sort of stuff to get to show to their project manager and to each other what are we doing. When are we doing it. Yeah. At the end of the morning everybody gets together again. Check if there are any impediments or stuff that can hold them back from completing their jobs for those two weeks. They get cleared and then starting Monday afternoon and the rest of the week they start building or making sure that project completes in time. And every morning they get together as a stand up. It's like five minutes sometimes it is 10 to check who's doing what. And is there anything holding us back, that goes on for of 1,5 week. And then on Friday sort of like demo day and everything that's built gets demo'd. And the project managers take it up. And the next day take it to their clients and of course. We'll go live on that Thursday. But we try and bring everything to Friday so now everybody has their own responsibility.

Stijn: [00:06:48] That makes sense you talked a little bit before about customers or the clients to be included within the process. How do you do it and... I can imagine that you use some online tools for your development.

Maikel: [00:07:05] True.

Stijn: [00:07:06] How do you include clients. And do you also include them within the tools for example with communication.

Maikel: [00:07:13] The first thing we try and do is get them to understand the way we work. Building a big project we did. There are no big projects we do now that clients don't go into the Agile process with us so we explain. First and foremost we explain how the process works. According to us they get to understand a little bit. They assign a product owner. They tell us who the stakeholders are and then we scoped project with them we are sort of like a phase 0 or the discovery phase we do. We make the big scope then we build the MVP, or we decide what the

MVP is, and we start building that. Clients that do that are here once every two weeks. We update them on the status we demo them what we have made and we decide what's going to be built the next sprint and then we have clients where we do maintenance work for or keep innovating on that platform. Then we do the same. We invite them once every two weeks they tell us what's going on in their company or on their platform or in a strategy we make, we fill the backlog. We decide what sort of stuff is important for the upcoming Sprint upcoming sprints and start building that. That's most of the bigger clients and how we get them into the agile process. And then we have some some smaller projects where the client doesn't even know the way we work. But they still fit into our way of thinking and managing our products and projects. They just.. The only thing they know is if I ask you something now I'll get it at the end of the first Sprint or maybe at the end of the second sprint so I have to wait two weeks or four weeks and then I get my stuff. Yeah.

Stijn: [00:09:13] That makes sense. And in terms of the tools that you're using we've interviewed a couple of teams. We notified a difference between smaller teams and bigger teams in terms of communication. So you Slack, some use...

Marco: [00:09:29] Stijn, Can I ask one question before we move forward. so we are talking about the team in general. So who actually does the prioritization? Is it the customer or is it someone from...

Maikel: [00:09:49] Normally the product owner the customer brings and the team that's in the room with them. And try and get everybody from the team in the same room with the P.O. to prioritize the backlog.

Marco: [00:10:01] Okay but it is necessarily the customer. It can be someone from the inside...

Stijn: [00:10:07] It could be someone from the inside if they trust us enough with it. Okay. But normally the product owner is from the client side. So it's the customer.

Marco: [00:10:17] Okay that's cool.

Maikel: [00:10:18] But he gets. We try and steer him to the right choices for us. And the tools we use to build a backlog track it and communicate with the client for the most projects we now use asana. We are really active in asana for just getting an oversight of the project. What tasks are open and what kind of stuff you need from them and what kind of stuff they need from us. During sprints they just fill the backlog and when they're here we review it at Ketchapp we are now implementing Jira those there are. And they get a lot more feedback in the form of issues and technical issues and that works a lot better because it's easier to integrate with our other systems where we store software do our deployments and sort of stuff.

Stijn: [00:11:18] And is there any specific reason why. I mean there's an overload of tools these days. Is there any specific reason why you use these two. Or maybe also integration with the other tools that you have. Do they offer something other tools tools do not.

Maikel: [00:11:34] Asana offers simplicity. That's what we first looked at Jira it was so big and complex it was too much for what we needed at that time. Asana is really clear and simple and we just started working with it at first because you know a lot of people Wunderlist or a to-do list to do that sort of stuff but it's a little... That's OK for your personal stuff. But it is not okay if you're working in teams. But Asana is. We look at a lot of other stuff. They're also great. But that's the one we chose at that moment. That's still working for us. But now we get. But Jira was thought of because there are 2 guys here now that worked with it before. They showed us a version that can be really stripped and there's so many buttons and things you can click and so many options you can select. We were, we thought that's. It's going to be really hard to maintain. Make sure it works if people ask questions you have to have someone that can help them and manage the system. The sort of stripped down version we thought not if we can get to that level of simplicity in Jira. We would like to try and see how that works. Now now it's there and it's connecting the issues to Bitbucket so you can complete issues, you can do commits. Everything works a lot better. Okay that's that's that's one thing Asana really misses, sticky ticket numbers. Issue number.

Stijn: [00:13:19] And is there any other way that you tried to solve this.

Maikel: [00:13:22] At the moment we don't. There is a way to do it. But have maintained the mix. You have to have your own custom code that assigns the ticket numbers from bitbucket to

Asana. We did that once but it's it's good, it kept breaking. We are all for having a system that just works and doesn't need a lot of maintaining so we can just focus on the work we're doing and not on the tools we use for doing the work. So for now this is okay.

Stijn: [00:14:01] Simplicity is key in this instance.

Maikel: [00:14:02] I hope Jira is going to solve the problem for us.

Stijn: [00:14:07] And what is your overall opinion about the tools / the process that you're using right now. You said it's missing some elements which is an opinion of course about it but your overall opinion about it would be positive and negative.

Maikel: [00:14:21] Positive. Yeah. It takes a lot. That the main reason is we don't send e-mails anymore. And when you need to find an attachment it's just in one place you can give comments and everybody can find that comment and before it was just someone emailed me an attachment and I'm like colleague needed it and then I CC'd and then there were two versions of the document et etc.. That's now. That doesn't happen anymore.

Stijn: [00:14:53] And you cannot speak for everybody within the company. But do you think that everybody is using it in the same way other people using maybe additional tools on top of the existing set of tools within the company.

Maikel: [00:15:10] Everybody has their own additional tools that others are maybe not using. For the main workflow everybody uses the same just to communicate without having your clients to use Slack. Yeah there that that works perfectly. Then there are Asana for the main project flow, bitbucket for the code. Now everybody knows that. There are are are people that use Google Drive to track the time they spent that sort of stuff. But the main workflow everybody use the same.

Stijn: [00:15:42] Yeah then we we had a couple of questions about which focus a little bit more about on visualization. Our first question is actually the name of the tool that you're using or the tools that you're using: if you would like to produce a certain outcome especially I can imagine as a developer with your colleagues: how does the tools that you're using so also maybe

including the whiteboard that you were talking about how does it help you to produce a common output.

Maikel: [00:16:15] It doesn't really think of it. That's one reason we're also trying. I guess you mean in managing the project or

Stijn: [00:16:27] In the broader sense like if you working on one project together and you have multiple stakeholders within that project I would like to come to one final product all together. Like how does tools that you're using facilitate that everybody's working and smooth integrated way so to say.

Maikel: [00:16:48] ...and working to the same goal? Right..

Marco: [00:16:58] Maybe I can give an example. So let's say that they ... the customer request a feature. So that the designer needs to come with something then. We have to the front end the back end developers come afterwards with with a solution. Right. So how do these three actually collaborate. Right. In order to. All right. Yes. Yeah. And how actually the toolkits that you use help with that right.

Maikel: [00:17:30] We have the big board to make sure everybody plans in the right order. They do that based on the kickoff we have in Monday morning so they know what's expected of them and the front end developer knows to ask the assets or whatever. So together they plan their week and the designer says OK if you are going to work on it on Thursday I'll make sure I have it on Wednesday and then they put our stickies on the board. So they know OK. Not gonna forget about me and I'm gonna be ready to build the stuff because he's gonna build assets on Wednesday. That helps. And we have Asana to store the actual design and the client can follow up on that give comments, the team or front end find them used them. So we have the board for the overall project. to see what he's doing in the right sequence. We have Asana to track what's happening and collect any comments and then we have the team meeting to make sure everybody's finished. And when something's really going south or it's it's not going well then there's a stand up to make sure if there's any impediment or ever someone forgot something and they tell each other: fix the stuff.

Stijn: [00:18:59] And this is also the way to check potential changes throughout the project. For example if you had to stand up and do the day feature request is coming in for what ever reason... How is it then that you are aware of what is going on. Like the fact that it changed that there was a request for example.

Maikel: [00:19:21] Normally we retry and get new feature requests to the backlog and reviewed every two weeks and we try and keep our planning or Sprint are as clean as possible. There's always something or there's always feedback and that's just got by the team and they make sure nobody misses out by just changing the design and time or letting the front end developer know. We also believe we are in one room together. So if if there's a really big thing or something's wrong we just tell it to each other or if I'm here and there over there I just call him what I've done for bugs or for their project is down or whatever are we ever sort of like support to more people log support issues our clients know to lock them there and they get sold within 24 hours. Again that gets that works too and I hope it makes sense or answers your question.

Stijn: [00:20:39] Are you aware of what other team members are doing and do you find it interesting to know what other team members are doing what they're working on right now. You mentioned that you have like multiple project at the same time. I can imagine that some type of complexity or you're working on project A somebody else working on project B maybe right now. Are you aware of the fact of who is working on what.

Maikel: [00:21:04] The big lines like I can follow it on the big board. Yeah we have: to do, doing and done everyday so you can see what projects someone is working on not you cannot see what feature is building or what line of code he's in. You can follow along with the project if you want to follow along on the smaller tasks you can see in Asana you can see how the project is going what people are doing. And then if you want to follow along over teams that's that's a little bit harder because teams are working on different projects and different stuff. No. Most of the time you don't even know what the other one is doing except for the the back end developers they always talk to each other and help each other, the front end developers talk to each other and help each other. Normally are not aware of what someone else and the other team is doing. So every Thursday. We have a small intervention where every team presents what they're doing in 10 minutes what they're working on and and not every project but just the things that they think is interest are interesting for the rest of the team to follow along. So like a weekly stand-up

we call it wherever Team presents the cool stuff they're doing and remind us of what happened last week. It's notable that sort of stuff because we know just three or four months ago when we started working in separate teams. One and a half year ago we started it separate teams and a few months ago we noticed we weren't really up to speed with what the other team was doing was a lot of stuff was just flying fast and you didn't even know they were building it. Now everybody knows what everybody's doing.

Stijn: [00:23:00] That that makes a lot of sense. And if you are talking about where you are in the process is that weekly stand up is it also a way to share where you are in the process of fulfilling that project as in that you can know these guys. But this specific specific projects are like 50 percent or they are almost to deliver. Is that something that's being discussed as well.

Maikel: [00:23:27] No that is just what the teams do with themselves. You know I'm almost finished or I needed a little help or we always reach a goal we want to reach but we're not big on burn down charts and burn up charts and whatever. We mostly focus on just building the project within the resources we have.

Stijn: [00:23:52] You think you don't need that type of information or that's not the relevant for

Maikel: [00:23:57] I think we it could help. Yeah. I actually don't know how we do it without where we we we need it. But we always we always reach our goal of any time we need to do it sort of like we do we have a different system to track time spent per project so we can always see we are on 80 percent of our budget. So we still have 20 percent left. Now how are we going to spend it. We involve the team in that. But it's outside of the project management tools we use a lot of like a finance tool we use

Marco: [00:24:45] So it doesn't... is it a spread-sheet.

Maikel: [00:24:49] It's it's Streamtime. I don't know if you know it?

Marco: [00:24:51] No, but we can take a look at it too.

Maikel: [00:24:57] Streamtime is built for creative teams not really for software development. It's really clean and it works okay. We use it to store quotes on a project goes on we create tasks so everybody can track the time no one has to send or invoices via that software.

Stijn: [00:25:27] And if you're looking for information I can imagine from your position we're almost done by way.

Maikel: [00:25:33] Sure.

Stijn: [00:25:34] But if you're looking for information in terms of the project status what does it what you're looking at. Are you using any of the tools and what is it that you specifically looking for.

Maikel: [00:25:47] I'm interested in how a project is going. I start with how far are we. And what has to be done so that is Asana. If I think something's wrong or everything everything's okay not perfect. If it's not I think we need more resources or we are going out of budget. I get back to the Streamtime to see if that's really the case and if so. Check if we can get the client sign for a sprint or whatever.

Stijn: [00:26:30] Ok that makes sense. And one of the things that we are actually doing research upon, that we encountered during other interviews is that due to the overload of information there is sometimes difficulty with keeping an overview of just keeping a general overview of the status of a project. Is that something that you think you run into because I hear you say that you go from one place to another place and then to another place again to find information.

Maikel: [00:27:00] Now you have three three or four different tools that track a part of the project. Yeah. So there's not one dashboard I can see how the project is going. So well ... But but I don't know if I miss it of it. Maybe it's because... I don't know what a good answer to that question is. Because but we don't have on overview and maybe it will come in handy. But I would also think that we'd be focused more on are you ... We try to go for a lot of quality. Yeah you don't mind spending more hours to build a better product. Also I don't mind spending less to build a good project. But the main focus is building a good product. My project managers are a

little bit more into how to time budget and the product going. But I personally am not personally I'm not any. Moreover thing in my vision and what vision should be I think it helps the team build stuff. If they... I'm not tracked on.. You still have one hour and you have to finish now and more on. It doesn't really matter I just finished the product and make it look good. Well now that I'm telling you telling you. Maybe one overview would be bad. But that would come in handy. But I would only use it if it didn't it was just to see how is it going and not to make people account for the actions you know.

Stijn: [00:28:55] Yeah exactly. That makes sense. What we are actually interested in is is like if we have such a dashboard then for different roles within a team.

Maikel: [00:29:01] It would come in handy. But I would only use it if it didn't it was just to see how is it going and not to make people account for the actions.

Stijn: [00:29:14] Yeah exactly. That makes sense. Now what we are actually interested in is is like if we have such a dashboard then for different roles within a team with different information might be useful to be shown so a Product Owner like. Where are we or within the project for developer maybe on the lower level like we are the problems right now. So a really a dashboard that is configurable for every individual team.

Maikel: [00:29:44] That would be cool.

Stijn: [00:29:46] So that is one of the things that we have grasped so far from the data that we've collected. So my question is not that you have to like it. But I can imagine. Yeah sure.

Maikel: [00:30:00] Yeah that would be nice because you want to focus on the right thing at the right moment.

Stijn: [00:30:10] The last question actually is how do you rely on data within the current toolkit that you have. So you have data on all types of data can be ticketing system can be chats et cetera et cetera. That produces a certain amount of data. How do you make decisions based on that.

Maikel: [00:30:34] Data we use in the end is it does the project reach its goals. Yeah that's analytics software to track if everybody if everything's okay and we have a plan scope and budget. That sort we get from streamtime. See how the project was built in the end. We do that over all projects within a certain size. So we decide that we have three sizes of projects. Based on them and then try and get better quotes, better plannins or whatever, based on size. And then we steer on making more or better estimates or promising less. So most ... the data we use the most is the goals and project management level: how is plan scope and budget going based on three different levels of projects.

Stijn: [00:02:46] Okay it makes sense. From my side that was the questions I had. I don't know if you Marco have any additional questions.

Marco : [00:02:55] Yeah I think you missed the typical one we kind of end up with. That is just to kind of know Michael what when you go on holidays let's say a long one for one month and they come back. How do you actually keep up with the work. What is actually the first things that you look for.

Maikel: [00:32:18] My colleagues. And when I get back the things I look for is what's what kind of projects are still going on. Who's doing what. And I find that on the big board and what's going to be my role in this. I find that in Asana on a deeper level. And if I want to how see the company in general is doing. I get to streamtime to see if everything's still on plan budget and scope.

Maikel: [00:32:53] If I would have been a team member. So I was out as a back end developer and I was going out for four weeks and I got back I just had to look at my own projects and a Asana I think. The rest is not really interesting then. Just look at what what what is my team doing and how far in the process and I can find that I can find them on the board and Asana. That's what we used the most I think.

Stijn: [00:33:25] OK. Nice. Thanks. Thank you very much for your for your time and also your answers. Is there anything based on like you know a little bit in which direction we would like to go with that research. Is there anything that we might have missed. Is there anything you would like to contribute that you may personally encounter that we didn't touch upon during this talk.

Maikel: [00:33:46] Not really. I am interested in what's going on here. End conclusion is going to be. So if you're willing to share when you finish.

Stijn: [00:33:56] Yeah most definitely did. The process that we're going to use actually is that last week next week we have our less interviews. And based on those interviews we'll make our first prototype. The thing is that we believe there's only going to be one prototype unfortunately. We would like to test it further but we have to hand in one and half month already. So that's yeah that's pretty quick. So it will most likely stay within one test but if you would be up for it we can send you the prototype. We're not 100 percent sure yet how we're going to test it and what the exact way will be and what type of that data we would like to collect. But if you are if you're up for it to tested first prototype then we would love that. Of course.

Maikel: [00:34:52] Yes sure. Yeah nice. Thanks.

Stijn: [00:34:54] We will make sure that will happen.

Maikel: [00:34:58] Cool I'm curious because for me.. You notice that in the way we work we're not really into using a lot of tools a lot of insights were more building or trying to build cool products without focusing on the rest now. But then the process has to be okay... We're so close to each other. All teams sit at the same table work together. That flow is good, so we don't need a lot of tools. We mostly need them to get the client up to speed.

Stijn: [00:35:35] Yeah yeah that's that's what we encountered as well like that the smaller the companies and the more they're concentrated on one place or the less they make use of tools we actually interviewed one and he didn't use anything and the like in terms of our our research that wasn't really of any help. But it makes a lot of sense that the closer you are to do the more concentrated you are on one physical location the less to lesser extent you use tools. But I think you personally you still like I hear at least five tools that you're using so it's not that you actually in my opinion.

Maikel: [00:36:15] No true that it wouldn't be bad to integrate more.

Stijn: [00:36:21] And then it's always also of course a tradeoff between like is. Any additional tools such as an overview is that of any added value in this instance. You doubted it yourself as well to that fact. Which which makes sense but that is what we're going to figure out.

Maikel: [00:36:40] And then there's a lot of tools maybe that you will probably know about them already. But then there's a lot of stuff that already integrates nicely with each other. So you have that Jira we're looking at that integrates with every everything form from Atlassian they have a support desk. An issue tracker for just building projects that you can use as a project management tool then they go to repositories they have and you have Intellige(?) Has all that stuff too. And they also do continuous integration. But I'm really curious what you guys are going to make or find out. Yeah we as well. Okay thank you for your time Michael.

Santander - Daniel (Developer)

23-03-2018 9:30

Stijn: [00:00:03] We start with some very general questions but could you maybe describe the company in a couple of words first? Your company right now.

Daniel: [00:00:11] Santander is a company that offers loans and credit cards and so on to customers, and car loans too. And we have approximately 30 developers doing a lot of internal process support and a lot of, what is it called yeah portals to like customers and partners around the products. So I'm working on it a lot of some of the internal stuff. Right now ready to the GDPR stuff. Okay.

Stijn: [00:00:56] And could you tell us a little bit about the team size and the composition of the organization that you are working. In so you mentioned 40 ish developers?

Daniel: [00:01:03] We are 30 sort of. We are 7, 7 teams of 3 in each right now, might differ a little bit but that's kind of a plan to have small teams and the ability to move a team between the product areas. So if we decide that GDPR is not important now. You can take a team from GDPR move into for example their consumer business team. Okay. So to structure like that again.

Stijn: [00:01:35] And in terms of the composition of the team what roles do you have in within?

Daniel: [00:01:40] That that's the developers when I say we were 3. So we have those two Agile coaches you met and they are shared by the teams.

Stijn: [00:01:48] Okay.

Daniel: [00:01:50] And Product Owners they are on kind of more the product area. So I'm my team is one of two teams in the product area called Tick to Play and and there is a Product Owner for Tick to Play that actually has another product there called Shared Services and show me has one team right now. So there are some kind of I guess you could say shared product and shared Agile coaches.

Stijn: [00:02:18] Yeah okay. And within the team what type of developers are?

Daniel: [00:02:24] .NET stack and that's developers mainly focused on developers so few teams that has specialized testers. But primarily taking...

Marco: [00:02:40] Frontend, backend?

Daniel: [00:02:41] I mean it's it's not developers hired specifically for backend or frontend as it is right now. I think that it's more like the nature in the team that some teams do a lot of back e nd stuff and some teams do a lot of frontend stuff.

Stijn: [00:03:01] That make sense. And so we are taking a look into the context of a job but that's a pretty broad concept that there are a lot of methodologies within Agile. Is there one specific methodology that you are aware of that you're using?

Daniel: [00:03:21] We are using Scrum or at least trying to use Scrum as best as we can.

Stijn: [00:03:24] Okay. And you say training how are you using it if you could explain some components?

Daniel: [00:03:29] Yeah we are doing all the Scrum events, you know the daily standup and the planning, we have a fixed iteration of, my team is one week sprint, many other teams use two weeks. And so we plan on Thursday. We have daily Scrum all the week and have review and retrospective actually, review first and then retrospective on Wednesday. So what we struggle a little bit with in terms of having that kind of short sprint is that it kind of eats half a day some times in each of the end. If you understand.

Stijn: [00:04:14] Yeah.

Daniel: [00:04:15] So if you plan too late on Thursday. What did we actually produce half of that day. And the other way around on after... If you do the review two early on Wednesday. Then what are we doing the rest of the day. Because there's a gap there in between before planning comes so there is some kind of something we need to work on in terms of that.

Stijn: [00:04:38] And how is that different with the teams that are two of the two week sprints. Do you think it's more balanced?

Daniel: [00:04:45] Yes it's more balanced and I think people because I mean that's two weeks. So it's only every other week at least they take that cost. So at least it's more important to to to to push those start and end events toward the real start and end of the time we have when we are doing short sprint. Imagine if we were doing sprints of one day I imagine depending on what you are producing you could actually do that. Maybe it's not development, maybe it was something or other stuff that you actually did, I don't know. I mean an hour in each end would be a lot.

Stijn: [00:05:27] Yeah true. That's that's a lot that that adds up. And throughout the process like throughout a week do you in some way communicate with customers from your position directly or not at all.

Daniel: [00:05:43] No we do very much backend and buried internal stuff.

Stijn: [00:05:47] Okay. So.

Daniel: [00:05:49] So the people we interact with is internal people and they are not really customers. I mean because it's some kind of a support process that we are building more than something somebody will actually interact with the data that would be no UI for it. It's very much back e nd. So the customer is kind of the product owner and maybe some some people in compliance maybe in the data warehouse team. But it's not customers as such. So they would set the requirements so kind of, so the product owner very much sets the requirements in this case for our team.

Stijn: [00:06:30] OK. And from the from the maybe I'm not sure if this is your position to answer it, but from the product owner's perspective who is the internal customer then, who is the one who comes up with ideas to the product owner and...

Daniel: [00:06:44] He reports... There's a program in the business and in the PMO office, there is a guy sitting there running this project under business scale. So our PO reports to him and interacts of course with the legal and compliance function also, I guess.

Stijn: [00:07:08] And in terms of support like the support that you receive from tools what tools are you as a team or your team plus Scrum Master plus Product Owner are using throughout this process?

Daniel: [00:07:23] Well. I have a picture that I can show on the screen.

Stijn: [00:07:28] And just to be clear like a white board would also be like Scrumboard for example would also be a type of tool so everything that can be visualized.

Daniel: [00:07:40] So we have been using a on premise TFF, TFS installation until like a few months ago. And moving out to the cloud version VSTS. That works very well even. That's kind of our how our main collaboration tool. Of course we interact in the team on white boards all the time doing some design or something, right? So we have a lot of physical tools to white boards and phones for taking the pictures and all that stuff. That's the thing. That's how it works everywhere. We haven't bought that what it's called Surface H ub or something from Microsoft which would be pretty neat but the boss didn't buy it yet. So this is this is kind of our tools or most of them. So we use VSTS. So you have some areas like work where we do the planning.

So work items and break down to tasks and features and epics and all that kind of structure which kinda does all the trace traceability to w ards the pull requests and the and the code itself. So so so so it works very well. So we use VisualStudio too well and we push against Git in VSTS. We have Git repositories for a lot of stuff and we have Pull Requests build so we actually before we merge it to master we do validate something makes some unit test and stuff going on. SonarQube for code quality and I guess the plan is that it will become a gate at some point so could have some kind of measures out there to say well ok you drop below 80 percent unit test coverage, whatever. That means in terms of quality I don't know but maybe you're breaking some rules. And then you cannot merge to master. Of course we have some integration build. So when we are merged into m aster we do a build. That creates the actual artifact. It's not released in here. It's only kinda of produced. So it satys with VSTS. And then we use the release management feature to run like you know deploy this package out to QA environment and run some integration tests and then if that goes well then it goes to pre production and then into production and maybe there are some approval steps in depending on the maturity of the product. We want to control it all. So we have ProGet which is a new get package management tool that we have internally so. So when we when when we push when we need to get something out to the environments we push packages to these. So this Octopus piece which is very much just operated from release telling it do it now it will fetch the package from here and put it out on the environment. We have start up using some AppDynamics stuff for monitoring. And that's also interesting for developers to see under QA environment for example if there is any API calls that takes too much time or whatever.

Marco: [00:11:14] Is that external service?

Daniel: [00:11:16] We have an internal installation of it. I guess you can buy it in the cloud. But I think it need some agents out on the boxes to run.

Stijn: [00:11:30] But what is not really clear to me right now is are these all separate tools in terms of separate screens and separate databases that you use or?

Daniel: [00:11:41] Well VSTS that's one tool for us, right? So we don't. We have this in the cloud. So Microsoft hold that in Azure in whatever way they do that. Yeah. And SonarQube is a tool that we have installed. So we have a SQL database backing that. ProGet is I guess is

mainly just file share whatever stuffed out in a box. Octopus I suppose there's a SQL backend for it.

Marco: [00:12:12] And is it like Jenkins the Octopus?

Daniel: [00:12:16] Yeah it solves the same problem. Yeah I guess. But it's it's that's very interesting product. And the UI is even getting good. It's always the problem in terms of change management. That's I mean but it really crossed off to production. It needs to be registered on a some kind of Sharepoint cap list to support change management so like does an incident. Service desk can look into this and say ok somebody got deployed this evening. Maybe we should be interests, interested in that. And of course there are requirements to handle that kind of stuff. We have Santapedia which is our name for the confluence, where we put out our documentation and even some collaboration stuff OK.

Marco: [00:13:14] But the backlog you. You keep at the VSTS.

Daniel: [00:13:17] Yes yes. That's up here that's the backlog. So the product owner puts stuff in here and does what he should. Doesn't work that well for our teams because it's pretty special in terms of you know when we plan... We don't have the luxury of just you know turning on the what it's called. Yeah. You know just to take the top 10 PBI and then just put it in. It's more like doing the next five this time. So so we kind of know what it is of need to just need to size them based on what system it is based on this time. OK. So but but it's

Marco: [00:14:10] But it is more for planning, right? When you actually have the sprint set up than you already have the items in the backlog that you want.

Daniel: [00:14:19] Yeah yeah. That's how it should be, right? OK. But we actually use a lot of time on planning to figure out what are we going to do and estimate it.

Stijn: [00:14:29] Why does it takes so much time?

Daniel: [00:14:33] Because the phase we are in right now is where we're starting to to to to do some kind of figuring out how it should all tie together. We have been using some sprints like

eight sprints or something to be some kind of back end stuff to support all this and that at that sprint. So it worked very well because each team, the two teams kind of had a defined product. They needed to evolve based on some specifications. Right. And and now it's started to be more like gluing it all together. And that's a little bit more tough to refine upfront.

Stijn: [00:15:22] OK. That makes sense.

Daniel: [00:15:24] So if let's let's compare to if you just had that which I always say web shop, right? It's it's kind of more easy to refine that and say OK we need to you. You can look at it, it is tangible you can look at that UI you already have. Wouldn't it be cool if we could do like this and that for our customers. That's much more tangible, our stuff that's very little tangible in that sense. What we're trying to of course still after when planning we want to have some kind of an estimate on what do we take like 80 or 70 percent of the capacity and stuff. So we try to get as close as possible to the Scrum process.

Stijn: [00:16:10] Okay. And in terms of using the tools there are a lot of tools out there. But why are you using particularly this one?

Daniel: [00:16:24] I mean a lot a lot of this is because we have had it like you know for years. So this is just part of our things. SonarQube came in lately, like an year ago. That was one of the POs which has a technical background as a developer, right? And he got this in. Got that approved. And I mean, we had TFS before we still run the TFS, we just don't upgrade it. And try to move all the teams over to VSTS and force them to use Git, because it works better. And I guess we didn't really have an option because we already kind of had the TFS stuff tied to Octopus and shifting to so something else will be a tough one. It's not that you cannot do it. I think it's because you bind, you bind your company to that kind of things. It's really really expensive to get out of it. Even though the marketing guys would say, well it's just Git and you know, but but you still have all your configuration of the builds and the configuration of the releases and the way you work and collaborate up here, right? And shifting that around is just expensive. So I think you don't want you to do that every year at least.

Stijn: [00:17:52] That makes sense. And you said that for example said the media is a way to communicate. If I was correctly.

Daniel: [00:17:59] Well it's it's it's it's just kind of wiki with some cool features.

Stijn: [00:18:06] And are there any other tools that maybe you are using outside of this to communicate or to collaborate with?

Daniel: [00:18:15] PowerPoint a bit and Excel a bit. The office tool set we use a bit.

Marco: [00:18:22] For internal chatting for instance?

Daniel: [00:18:28] We try to use Slack and Skype, sometimes. But mostly Slack. Of course. Yes now I understand. Now my machines turned off but doesn't really matter. Guess that you saw the items.

Stijn: [00:18:44] What is your general perception about like all of that you've just shown to us like in terms of other processes like how the tools help you to do whatever you need to do?

Daniel: [00:18:56] My biggest concern in terms of all the structure is that the time from pushing some I don't know minor change and getting that through the pipeline and heating the QA environment it is sometimes it takes a while, right? And when I'm saying a while I mean that could be 5 or 10 minutes sometimes, right? But it's still a long time you want to just get it out and see it work and it and at some point when we get to that ultimate idea of continuous deployment directly into production because we have... That's actually one thing we lack, we lack a policy called four eyes, which ties into the Git pull request stuff. So there is requirement for that, there is another guy looking into your code before it can be merged in, right? Before it can be merged in. Then that's a big responsibility, right? Because what you should look for. What kind of tests is relevant here. It's easy if it's just a minor thing. But but you could easily forget to check if the sufficient task courage. It's very it can be very hard especially if it is a big pull request. At some point the company wants to get to a place where when the developer has made a change and all the quality gates has being set approval and all the manual approvals it's being set that it flies directly into production. That's kind of the dream and it's kind of it's that's it. It's it's a tough road.

Stijn: [00:20:42] Yeah. And do you think that everybody's using the tools in the same way or maybe use as like extra tools additional tools because this is not sufficient if you if you take just your direct colleagues into your mind?

Stijn: [00:21:00] Yeah I mean in those two teams that we sit together on we we use it in the same way. But it's like this you know some of the developers have kind of have to to get hit by woh! You cannot do that in the builds anymore. You need to take those things and have a release instead. So the structure of how you use VSTS and what you can what you should do in the pull request build and the CI build and in the actually release phases. That's kind of you need to just learn that way and it's kind of tough to configure and there's a lot of nitty greedy in your seat and use time. Okay. Now there seems like that the test just doesn't actually run, then you need to okay. I have some string here in the configuration of VSTS and you said okay I forgot a dot or whatever. That's a lot of waste going on setting up that stuff. If you're not very precise

Stijn: [00:22:18] And if you think you and you mentioned one example already before, but if you think about one main issue that could be improved but what would it be for you within working within the specific?

Daniel: [00:22:31] For the tooling? I think that you know the idea. I guess we VSTS is coming with the support for building releases as code. Those I think it's called Yaml. The file format. I think I don't know. You mentioned some deployment tool or build tool before.

Marco: [00:22:56] I think Jenkins.

Daniel: [00:22:57] Jenkins probably is like that. So it's part of the code base to actually define how to build and how to release. I like the idea of that. I think it will solve a lot of problems.

Stijn: [00:23:11] What type of problems?

Daniel: [00:23:11] What I mean is the inconsistency problem when you first have to do some kind of you know naming in the repository you name your stuff, right? You name your assemblies your classes, everything is named. When you have to reference that in a

configuration that actually builds this stuff. You have to to remember. Sometimes those names. Whereas if it was the pipeline as code I guess you call it. You would do the pipeline aside with the actual stuff. So it would be much easier to make sure that you didn't have those inconsistencies in terms of names. I guess what you lack is kind of some kind of static analysis on on on on the on the pipeline, right? So so that because you can set up the pipeline. But you can not validate it if it would work whereas with code you can actually write some code and if you have a typing mistake where static code analysis. This is what it will take you well, it doesn't exist. And that would be cool to kind of spread that out to the pipeline so that you can much easier maybe even you know reference it to have statically checked. If the tooling for that can works like that. But I like the idea at least and I see what can solve.

Stijn: [00:24:38] Anything else?

Daniel: [00:24:40] I would like to do some closure programming. It's not going to happen. Mean me there has been produced a lot of code and the quality is at least in my eyes, not always that good and I've done a lot of that code even. And I think there's some opportunities for for taking some of that code and actually do it in some different languages. Because it is just too easy to code in C# you can you can do it in 10 million different wrong ways. And then there are those nice good ways to do it. And that is just too easy to do it in a bad way too easy not to to think about immutability for example. I mean it's it's. But you can always argue that if that unit test doesn't fail and the software works but that's not true because unit test only kind of proves if there's something that doesn't work. And in my eyes because you cannot test all of your software in all, I mean.

Stijn: [00:26:03] Yeah, I get what you mean. So questions about visualization you've shown us a couple of tools. If you have to produce a certain output together with a colleague you will work together. How does this tool facilitate this? Like what do we do the working together and you need to like you two or three need to work on one product. How does it work?

Daniel: [00:26:28] I mean the last if you go a few sprints back then because I have like 5 sprints where me and my team very isolated produced one product. And we actually did Scrum very well in that period because we could do some refinements through the week and then kind of describe the next features you want to build. So we we could actually break down the stuff and

estimate it pretty precise. And do some planning on it. So I guess that if you need to to work more than one guy on something you need to be able to break it down and you need to be able to communicate easily that I'm doing that and I'm doing that.

Stijn: [00:27:19] And how do you communicate?

Daniel: [00:27:22] The work feature you. Pretty much facilitates that. Because when you go in and pick a task and put your name on it and say in progress and then you create a branch when you try to create a pull request you tie the pull request and the work item together and and and then you kind of have that full traceability off of this stuff.

Marco: [00:27:50] And so you would first when you're agreeing when you are planning you would break the task in the work part if it's if it was just kind of two people related, right? So I need to do this part and you can do these part two. First break it and then you do it, right?

Daniel: [00:28:08] Yeah. Ultimately we would you know break it down so that. I mean we are three developers. And right now we are actually only two in that team but we were three. And everybody had the same skill sets, right? Developers C#, .NET stack, all that. It was the same product. So we just needed to have the list of stuff to do broken down to something where we could easily estimate it. And then ultimately you should just pick from the top and not based on what you are mostly interested in doing. Sometimes you fall down into that one, right? And then say oh that's cool task and is down on the bottom. So you should pick it up on maybe Tuesday, but I want to do it. Yeah you should not do it. The funny ones, right. Yeah but it's about you and it's it's not easy right because we are human so sometimes you forget to start up on something. Ah okay I'll take that one. But you don't register that you do it. But we sit very close together and we do daily everyday and we actually synchronize over the day too right because we sit desk side by side. So you can actually see what he's doing. You probably get some pull request even, right? You probably what he's trying or saying.

Stijn: [00:29:24] So you think it's a precondition to sit in the same room to be as efficient as...

Daniel: [00:29:29] Ahm I think it helps him. I think there is a cost at least to sit in different rooms. And for the. And then the further you sit away from each other in terms of I would say distance,

language, culture, timezone. Time even all of this kind of add up and gives a cost maybe some are more expensive than others, I guess culture can be a different one, right? Or a difficult one to handle. And then some time actually culture implies time zone so so so so then you add up again, right? Because you can't have daily, together. I've never been on a strong team where we're sitting in distributed location, but I guess it can be pretty tough.

Stijn: [00:32:00] And if there are any or how do you track changes actually if there's any changes in the project artifacts. So if we figure out that something we build like last sprint doesn't really work, then we register a bug in VSTS in Work and probably put it in the current sprint.

Marco: [00:32:16] And do you kind of make the association between the actual original user story?

Daniel: [00:32:25] Yeah that's the thing because we don't really have those user stories. Okay. But but we do have to work items them and no we don't do that. I guess it would be cool if you could tie it to the test definition or test case that were supposed to show that it did work, maybe. Yeah. Because sometimes the test are cool but who tests the tests.

Stijn: [00:33:11] Yeah. What is the difference between uses stories and what you mentioned before?

Daniel: [00:33:18] Work items. I guess user stories is some kind of a format invented somewhere. I haven't really used it very much but it's some kind of you know that you have to to specify like this and that and that and that and then you have a user story, whereas when you do some kind of you know backend programming. Sometimes it doesn't really like that because user stories sounds like you have a user has something has something it needs to do but if you have a system that needs to do something. It's sometimes the format is just easier by just writing down what is supposed to solve and not in any formal anyway. So I guess you should use user stories in other kind of work item formats with care and select the ones that actually fits what you are solving them. If you just say well we want to do use story and everybody should do user stories and stuff then you would you would actually use time on what the hell I'm writing

that column or whatever. I mean that's that's not what we want. We want to get the product out. I want to be Agile in the format we use.

Stijn: [00:34:24] And if you're like if your task needs input from other team members so that's not really collaborate towards one but from other team members, how does that work. Maybe from a design or frontend. I'm not sure if that is applicable here. How would that work?

Daniel: [00:34:37] Well we really we go get those guys. Okay. So we have, we are in the same building. We have one or two teams sitting in Poland. They work on very specific stuff. But we we we have been you know putting in guys from other teams to try to get stuff solved. And and and so so we we have some kind of budget model in this business around the teams so that if you if you need to do some kind of support for another business area that has its own budget that you are not normally linked to, then if it goes beyond some like an hour or whatever then you should actually you should actually report your time on that budget. So so so you are allowed to spend time helping other teams on their business area. As long as it doesn't grow too big. That's how it is. And that's no route for it, I guess. Then the product owner should actually because they kind of have the product. They they they should actually interact them and say okay okay you can use Daniel to solve the issue, you can have him for a week. But Daniel is burning your money. This is kind of how it works. So developers are not afraid of helping the other teams. In that way.

Stijn: [00:36:14] And in terms of process or your, how do you say it. Like the progress that was the word I was looking for. Are you aware of the progress that you as a team are making. And if so how are you aware of how you're progressing or proceeding towards your goal.

Daniel: [00:36:37] So you're talking inside one sprint? Yeah and that's sort of two approaches. But that first we try to use a burned down to so that everybody can do and say okay we probably need to look out there and check if something should be taken out or whatever we need to take stuff in. Yeah I guess that only happens if refinement, and estimation and planning failed. And so and of course on daily Scrum we try to talk to the goal. At least that's our plan because that's the action point out of some of the latest retrospectives said that we need to focus more on have a clear goal, which I guess is...

Stijn: [00:37:26] And that's not being provided by a burn down chart for example.

Daniel: [00:37:28] No. That burn down chart is only you know the work items and the whatever remaining [inaudible] and those others is going down. So it's kind of forecast. Are we driving fast enough? Now where it was the goal is more like did we actually managed to get it out to the environment. Maybe it could be different stuff than just because sometimes you forget to have you know the work you need to do get your stuff out to run in the in the pipeline. Out to the to the environments if you forget to get that kind of stuff in then the goal at least should talk about. Okay. Our goal is not to code this. Our goal is to see it running QA and being able to to to observe the logs and and and and maybe look in the database manually or whatever it is and come backend stuff. In terms of the more the program that we are kind off working for, that's more like the product owner and the the PO that that that kind of does that because they're very varied you know in terms are we actually going to meet that deadline and all that stuff. And it's I would say it's it's it's complicated. Up can we have had Yeah. [snippet removed per request of the interviewee]

Stijn: [00:40:56] Would be relevant for your position to see the progress or two like maybe talking percentages and maybe percentage is not a good way to go. But would it be interesting for you?

Daniel: [00:41:09] For a developer as being you know kind of just a flawed man producing stuff. I guess it's ultimately I wouldn't want to know about with the business was going in the right way because the owner would be very very certain that the developers are doing what you know, produces the best return of investment or whatever you use should call it, right? That the developers should not decide whether it's the right thing to do. The developers should probably decide how to do it.

Stijn: [00:41:49] One of the things in this is not in our questionnaire but one of the things within Agile is that developers or the team should be protected from all the outside noise and like everything should facilitate that they can develop as much as possible. Do you have the feeling that that is applicable to you in this case?

Daniel: [00:42:14] Well I don't think we produce as much as we could do. I mean we have just seen one of the teams producing void for some weeks. Don't tell that to anybody. That's my opinion. We haven't delete that repository yet, but that could easily happen. And I think I think we use a lot of time also discussing stuff. Because we kind of too small teams and small teams are great because you have few opinions. So it's easy to get consensus on something and at least if it's people that work well together. But but if you grow the team for say okay you are two teams but you work together now a n d get one sprint backlog and you actually work on the same goal which is what has happened and which might be good but but it's starts to get harder to agree on like architecture the right way to do to solve this problem. It just starts to get harder and we can flag some kind of lead on on on on that. But it's kind of like maybe not I mean it's always bad to have somebody that leads something because then you might miss that guy. Right. It's just better to have six ants solving the complex task and having like six different animals solving the problem. Because they have different skills it's just to have that completely multi tool developers they can solve a problem other than just the first task and then do it. But it's always comes to architecture and opinions about what is good architecture is the code decoupled enough. Should this reside in a different repository because it's a different thing. This over here changes on a different basis than this so let's split it because we can finish this often we can test is. And we know it will work for the next 10 years but this over here you're probably going to change this area of every sprint. Right. But but but let's not take the risk of changing that by accident. Let's not introduce the need for reviewing a big repository and making sure we haven't screwed stuff out. So does those kind of things. It's it's just hard because many developers just wants to go. They don't bother too much about the architect basically as I see it. They think about frameworks they want to use ASP .NET core and it sounds cool and all that stuff but that's not architecture. Architecture is more about how you make sure that your code is rock solid and and it's structured so that for example stuff that changes on different reasons maybe shouldn't be the same repository, at least it should be split apart and isolated from each other. That that kind of architecture stuff more than like Okay you use C# and .NET framework and ASP .NET core for some developers that's architecture but it isn't. That kind of problems. I guess we are struggle a little bit with.

Stijn: [00:45:27] And when you're looking for information about the project or project status what is it that you would be looking for?

Daniel: [00:45:35] So project status what. We don't really bother about the project status. Except for that once in a while we talk about that deadline right. But we shouldn't I say that all the time. Okay that's not bother about the deadline. Let's make sure that we produce what is in the Sprint because the process should facilitate that reproduce what is needed in towards getting to that goal in the end basically. But of course you cannot completely ignore it. But but the PO obviously should be very concerned about reaching the deadline, but if you have a deadline then basically you should think about some waterfall stuff, right? Because you need to plan ahead. Right? And that and that doesn't work well in Scrum. You need to have some kind of knowledge about what is going to be used in the coming iterations and sprints. But you know you just need to know enough you don't need to know more about what's happening in four weeks because then you start to lose focus on what's happened this week. You want to have the important stuff in the top of your ming. So planning the whole thing and doing the design first and then the analysis and all that and that's just the opposite.

Stijn: [00:47:06] Last no not last question actually. In terms of relying on data that is being produced by the system so it can be chats that can be a ticketing system that can be everything how do you do it and to what degree do you depend on the data that's being produced by the system?

Daniel: [00:47:31] So to our system the data that it produces? Well the thing is that our system deletes data. So it's kind of a different guy. It's so we solve the problem of of making sure that we don't keep data that we don't have any legal basis to keep. It is the problem it solve. So but I guess I mean we look we do produce some data. I mean we have some support systems to facilitate that actually do this, right? But but it's it's not really we don't we don't have to look into like the data for the incoming applications to get an idea of wether the customers are using the UI correctly out there if we need to do some change to make sure that he fills out those fields or stuff like that. We don't have that kind of problems because our what feeds our system is data. Kind of.

Stijn: [00:48:41] Thanks. One last question. I think the question we always end with. Imagine that you went on a vacation for like two or three weeks and you come back and like you are within your team. What is the first thing that you look into? What is it? If you want to go along with the team what is it that you do? What are the first things that you look at?

Daniel: [00:49:06] Yes. How much got fucked up these three weeks?

Stijn: [00:49:09] Yeah but but how do you do it where do you look? What is...

Daniel: [00:49:14] I don't know. I mean we're pretty new. Those two were formed like two months ago. Three. So we pretty new. But what would I do. I mean I would do I would probably ask a lot of questions. So what have we done. I don't know if I would maybe go in and do in the last sprints. What work items were there. I would ask one of my colleagues for a summary what has then happened. Because sometimes the most important things are not in the you know in the work planned but more like in what actually happened on retrospectives and then that is the PO [inaudible] or whatever, right? I mean all the sensitive stuff. No but it will be. I mean it will be perfect to be back from vacation on the day of review. And then not participating in the retrospective probably. But but but. Yeah. But that I would probably prefer that because then you would know something of what has been produced based on that review.

Stijn: [00:50:31] OK. Thank you very much. Is there anything based on the talk that we had so far that you would like to share with me. We forgot to talk about. I mean there's a lot to talk about I know but based on what we have so far. Maybe we missed something. Could be just if something popped into your mind.

Daniel: [00:50:48] No no. I've never been pretty pretty abroad. I think one of the key things for me is that is it's cool to have some kind of a process framework for to work. But they should make sure they fix the product. Because, I mean I have worked on other products where Scrum were just like in the way. And so Scrum is not perfect for all products that's from my teeth. OK. OK. Thank you very much.

Santander - Michael & Stine (Scrum Masters)

23-03-2018 10:15

Stijn: [00:00:00] OK. OK. First in a few words could you describe what the company is doing just as an introduction.

Stine: [00:00:08] Yeah I mean there are a bank. I think...

Michael: [00:00:13] Consumer bank they are doing loans and credit cards. The primary focuses on loans, sales, leasing. That kind of stuff.

Stijn: [00:00:28] An within the organization. What does that what does the size of the organization in Denmark and the team composition over here

Stine: [00:00:41] The organization in Denmark. Two hundred fifty Yeah. And we are connected to Nordic. And then it's up to to...

Michael: [00:00:53] 250 is in Santander Denmark connected to Santander Nordic, and Nordic is connected to Madrid or Spain.

Stijn: [00:01:08] And in terms of the team's size or could you maybe explain little bit how it works over here, we heard before that there are 30 developers. Yeah how is there that organizational structure so to say from an agile perspective.

Stine: [00:01:23] Yeah they say that the teams of three developers. We have around 9 development teams. A lot of teams actually work together. 2 on 2 teams, so there are around 6 people.

Michael: [00:01:39] You just talked to Daniel and he is at the moment in a team that join forces with another team. So they actually they are actually only five at the moment because one just up before they start. But there will come another in and that will be 6 again. I think that's also two teams in Poland within the 9 teams. And they're talking about also starting a new team in Poland. So we come around 10 teams. Okay. And we are two scrum masters, Agile coaches, so we'll have to be all over the place. Okay.

Stijn: [00:02:25] And then maybe a little bit early to to answer but how does that work with multiple teams and having two Scrum Master / agile coaches.

Michael: [00:02:34] Scrum-masters have a very busy Wednesday. Because you have like three daily stand-ups in a row. I have three. And then you know.. The teams I have only run one

sprints. So every Wednesday we have review, retrospective and then on Thursday we have planning. So those two days really are intense. And maybe also too intense.

Stine: [00:03:05] I think the idea of having more than one team as a scrum-master is very good. I have had you know previous company as well I think it's a very good idea.

Stijn: [00:03:13] Why

Stine: [00:03:14] You can be.. Both, you get a little distant from the team, so you can more likely see things from above. They are being part of it and doing the tasks that the other team mates.

Michael: [00:03:31] And knowledge sharing. Yeah. Yeah. Yeah. Kind of having an overview of what's what are the teams doing and making them collaborate better. If you can see something working on one team, you you can try to implement the other teams. Other way around. Easier to push them around somehow.

Stijn: [00:03:54] And you mentioned agile coach agile is a pretty broad concept. Like what method within Agile are you are using and how are you using it?

Stine: [00:04:04] We are using scrum.

Stijn: [00:04:07] And how are you using it. Are you using all elements of it or is it how can that be seen in this particular context.

Michael: [00:04:14] I think we're using all the elements. We of course like. I think all others, we can get better than we are today, but we are having all the events, we have all the artifacts. We talk a lot about it and then of course practice doing it the that's the most difficult part. OK. The reason why we got hired as, like 100 percent scrum master, that previous they selected one of the developers in the team's to say you are the scrum-master. As well as you are the developer. And they saw a lot of conflicts in that set up because at a developer is a developer and sometimes it's very difficult to change hats. They say "o now I am the scrum-master" and you have to do this and that. They would skip daily scrums and suddenly they didn't stand up they just were sitting down talking over their laptops. So they just need someone who is persistent

and keeps reminding people every day that they have to live within the framework.

Stijn: [00:05:39] And throughout the process of developing or within the week. Do you in some way communicate with the client or the customer. If that is applicable to this context.

Michael: [00:05:53] I thing not yet, because we so new. The two of us? No no. We are, of course close dialogue with the product owners. Yeah. They should represent and try to represent business, but we still need to get known into the business and work on how we can get the business closer to I.T. That's still a problem.

Stine: [00:06:19] I think I mean if you see the teams, I think that they if they are in doubt of something. Then they ask the product owner, perhaps he could sometimes just you can ask this guy. So the teams are in contact with customers. Yeah they use the product owner as a point of contact.

Stijn: [00:06:41] But they're not in some way included within tools that you might be using for communication. No no no.

Stine: [00:06:49] I don't think... I mean, then we would need to have a tool that we could use. And I guess that's perhaps also the way around. That we don't have a tool. We can talk to the business throughout. I mean that's incidence or handling incidents, but that's not really a tool.

Michael: [00:07:10] They have that backlog. Did you see the backlog. We didn't see the backlog itself. Just the tool-kit. They have a backlog and use Microsoft Visual Studio and VSTS. Yes yes. So that's the tools they are communicating through which you can say.

Michael: [00:07:36] Also they use Slack to communicate. That's actually one thing they use. And the business uses that as well. They talk to the business a lot through Slack. So it's Slack and VSTS. Yes. Yes.

Stine: [00:07:52] I don't see the business using VSTS. That's just what I have seen.

Michael: [00:07:57] No as we talked about earlier, I don't think the business understand the

backlog because at the moment that maybe, for my teams. They are very technical because of this GDPR, they are only doing this at the moment. So that's a lot of technical stuff.

Stijn: [00:08:18] Okay so the backlog is not really something that the business can use and Slack is something that they use.

Stine: [00:08:23] It's depending on what team. You have some very technical teams. My teams re not that technical and their business are described in the backlog. They can understand what's in there but it's it's written in a different way. Like with accept criteria's. I mean it's just not business wise. They don't use it. Okay.

Stijn: [00:08:47] Is is that is that maybe something that is lacking right now.

Stijn: [00:08:51] Yes like the communication between client or business in this instance.

Michael: [00:08:54] That is one of the reasons we are here. This is one of succeed criteria. We need somehow to find out how we can get the business and IT get even closer. We need the business to understand agile and how to think agile.

Marco: [00:09:18] One of the things how long it's been over how long has the team be using Agile at Santander?

Michael: [00:09:23] As long as David. He is the head of the IT department. Overall 4 years.

Stijn: [00:09:39] And you mention two tools right now. Are there any additional tools. Well we saw it like a bunch of tools but mainly in terms of communication coordination and are there any other tools. And if not why is it that these tools these two tools are out there right now.

Stine: [00:09:56] If they are using a tool for incidents like bug-fixing and stuff like that. But that's also kind of a communication with the business and the teams.

Stijn: [00:10:15] Could you maybe show us the backlog is that possible.

Stine: [00:10:23] Yeah. If I can get that to work.

Marco: [00:10:32] But do you interact with these two tools to, like the VSTS.

Michael: [00:10:35] Yes yes and yes all right. Yeah. Okay. From my perspective I have introduced sprint goals, and there's dashboards in VSTS. So there we write the sprint goals. And also when we have a retrospective we will identify some actions. We also write these actions. So the next retrospective we look at the dashboard and just follow up. Are these fixed or should we work with them also the next sprint. So that's the way I use that dashboard and also try to use, follow the progress during the sprint.

Stijn: [00:11:26] Tools that are being used that is shared by everybody within this company. OK. This is allowed to have personal tools on top of that.

Michael: [00:11:39] Yes. I think maybe you have to get a talk to legal or risk of someone. Get a permission from David but I think there are really... And so this is a backlog tool, VSTS. You know the tools.

Michael: [00:12:06] I haven't seen. It but he has shown like the overall framework. How different tools interact but not other than detailed level.

Stine: [00:12:14] I haven't been working that much in this tool, the only thing is that I know that. Here's a view of the backlog or the items in the backlog here I have the sprints. So this is the current sprint for one of my teams. And then there's future sprint lying ready. And yeah I wish I could. I mean the P.O. had a view, I think was quite nice. I'm not sure how he got in there.

Stine: [00:12:48] You need to go back and column up the right. Well there's a lot of things in here. Well I can explain instead. I think it's faster than trying to show it in here. Well he has, were you can see the backlog and then the sprints. And than he has a kind of visual in a board, where you can see the sprints and then a headline of all the stories and then you can see the next sprints coming up. And that's I think it's quite nice for the team to have an overview of of what's coming in the next sprints. And they have two weeks sprints. So are, I think they have a plan for the four next sprint or so. So they're quite far ahead in the schedule. Yeah.

Stine: [00:13:41] And then you can see team velocity and everything. So they do estimate the stories, before they pull them in. And then we have daily in the middle of the sprint we have that check point where we say: "How's it look? Does it look like you can complete everything? Is there anything we need to adjust in the sprint? Yes. So we actually did that yesterday and then we pulled in and out, based on we have some vacation in Denmark now, next week. There were some adjusting we have to do in the sprint.

Michael: [00:14:14] And that's a problem for us. My teams are only running one week sprints. So that's it's very intense. But we haven't this checkpoint because it goes really fast. Also they put planning and refinement in one session. Because it is going so fast. So many of the user stories are only headliners. They are not described very well. Everybody seems.. It seems to land. Every time. Together with the product owner. But it's also very technical.

Stijn: [00:15:00] But what do you think in general of the tools that you use and so far. Like if you're talking about these two.

Michael: [00:15:07] The problem is that we haven't worked with this tool before. So we were a little shaky about it. I'm used to Atlassian, Jira. Yeah but they are very positive about this one because it's apparently very good to combine with some of the other tools that we have been using. Put it in different environments and so on. It's very good to tracking and so on. So developers are satisfied with this product.

Stijn: [00:15:45] Is there anything based on what you've seen so far. Maybe in your previous work as well that you would like to see improved or that you're missing yourself. I know it's only two weeks but you may have encountered something.

Stine: [00:15:55] I think that when were having a daily stand-up. Then are only looking at the board in this team. There is three of developers in Poland. So we only have there's only one way of doing it. It's digital board. We don't have a map on the wall. It's not working that easy. I can't get an overview of the stories at one time. We just scroll down and see the tasks and then you can say oh I'm working on this one and I've written down the hours so. I'm lacking a bit of overview of the sprint when having a daily.

Michael: [00:16:30] Also lack the overview of the progress right? I think sometimes it's difficult. This is one of my teams. Yeah first I can show you the dashboard. See we have to sprint goal and we have these action points down there from the last retrospective. There is also something about I haven't tried figuring out what it is. But the user story they have committed the user story, that they're working on it and there are some... Lets say there are four tasks, here below. And all four tasks can('t?) be done. But then the user story, but the user story isn't set to done somehow. And it's both difficult for me to see how far are they. It's only at the end, then the user story will get done even if... They have to manually change the status on this story. It doesn't just think. I, I can go in here and I look half an hour before there's a review or not.

Michael: [00:18:12] And that's a lot of thing things that are not done. And suddenly boom boom boom everything is done, because the tasks are done. Because it's a manual process. That's something that's there. I think that's.

Michael: [00:18:24] Yeah I have used to scrum-wise before. I think that was very good tool for teams. But the product owner was not happy about that tool. It was not that easy to get an overview. Looking forward okay. But it was very nice for the team to have an overview of the tasks and the stories and everything in the sprint. And it also kept track of releases and I just think it was quite nice tool, but it wasn't related to development so they couldn't use it.. Also when you deploy. It connected somehow to the developmental tools.

Michael: [00:19:01] Also they have epics and then the breaking down to features and then break them down to they call them PDI's. The backlog items or user stories. And then they'll break them down to the task. You have to see your description except this is also sometimes difficult for me to find out where is the task.

Stijn: [00:19:28] What is the first thing that you're looking for. If you're go into this tool. If you

Michael: [00:19:35] For me on a daily basis as a scrum-master I need to understand how far are they on schedule somehow and. How can I see if there's a problem because they estimate differently. The teams estimate differently the product / backlog items. Some use man-days, some use story points, some use something different, hours. But when they estimate the task,

then they estimate that in hours. It is also different from team to team.

Michael: [00:20:12] So we have some it's difficult you know for. There's no consistency with... And it's very difficult for me sometimes: where are the hours and how can I see where they burn down the hours. And see the progress from the day before.

Stine: [00:20:31] But I don't think there should be consistency between the teams. I mean just as they use relatively estimation.

Stijn: [00:20:39] So not one hour the other one points.

Michael: [00:20:41] What works for the team, works for me. As with previous teams I have discussed a lot about this. Oh we should calibrate the hours or how we estimate... And I was like: why it's banana you're estimating. It doesn't matter. Just as long as you get an overview and you somehow can predict how much you can do.

Michael: [00:21:04] Everybody ends at the same. One story point on this equal one developer one day. Somehow that's always the mission point. Maybe it's correct and maybe it's not. If you asked the teams it's all about, one developer 6 hours. Then it is not relative anymore.

Stine: [00:21:34] But I think it's because it's easy. Yeah when you see the velocity and everything it's not high figures being low figures and then you can

Stijn: [00:21:47] We have some questions on visualization as well. You mentioned some tools. We're talking about a team right now as in a team with a scrum-master, a P.O. and developer. That is what we name 'a team'. If you have to produce within the team a certain output how does this to help you. How does it work.

Michael: [00:22:12] Yeah I guess you're thinking of a review or demo. I haven't seen them use this tool for that. They do a powerpoint and that's how they do this review. Use that in a review. Okay. You're not using this to. And they are actually not. I was use from my previous previous job that we actually went through the sprint backlog. PO, do you agree that this has been fixed this and can fix this. Yes looking into the backlog and then yes it's completed. It's not.

Michael: [00:23:00] They don't do it.

Stijn: [00:23:02] And other ways to communicate throughout the process like between not only get all the goals that are out there.

Stine: [00:23:11] Like it could also be between product owners, I have need or I need to have this in my sprint so you have to prioritize this story on your backlog high. I don't know... Are they using that? I don't think so. They have a meeting. To coordinate between teams. We haven't done that meeting yet.

Stijn: [00:23:33] It's a physical meeting. The tools are not being used to collaborate between different teams.

Michael: [00:23:40] No I don't think so. They have this. We have this room up on 6 floor, they have a big wall. They use a visual, to try to see what are connection to the different projects. They don't use the tools. I don't think that's... I haven't seen a tool yet that connects the different dependencies from different projects.

Stine: [00:24:14] Well, Rallyday, in my previous job, was used for that. It was more like the head of the I.T. department. He said this is the tool we are going to use, because this can be used across teams and features and epics. But it was very hard to use it.

Marco: [00:24:33] How is it called?

Stine: [00:24:37] Rallyday. I mean that's just what we call it.

Michael: [00:24:43] I also think because the dependency can change from week to week because.. The other team: do they deliver or they behind schedule? I think it so difficult to have an updated view that you can trust.

Stine: [00:25:03] Yeah and in this case I'm also thinking, because they're not that many product owners. I think they are, they are four and one of them is very technical. He's like the

architecture team. It's called shared services. So they are supporting the other teams so there are three teams of 3 product owners, working with the business features. It is three people trying to coordinate their backlogs. Perhaps they don't need the tools for that.

Stijn: [00:25:38] And a follow up question is if there are any changes within the project artifacts, how is that being tracked.

Stine: [00:25:53] If it is during a sprint updated? Then it is just updated in the tool. Yeah yeah. OK.

Stijn: [00:25:57] And are you actually aware of what other team member is doing? How does how does it work? Like you have to give three teams you have that you need a lot of overview how does it work. If you want to be aware of what another is doing?

Stine: [00:26:12] I am not sure if I can get the view. I mean this is the board of the current sprint of this team. Now they are unassigned but here there is a task that's assigned to Herman. So in this case I can see that's kind of a scrum-board. So when the developer picks up a task it's assigned to him and then you can see the progress. You'll put it in in progress and then when he is done he's put it in done.

Stine: [00:26:48] But they use it, that's what I have seen. OK. They have tasks and they move those tasks. OK so let's just a scrum board but then an online version of it. Yeah yeah. Yeah. I haven't seen an physical scrum-board.

Michael: [00:27:08] And of course, they talk about it every day, on the daily stand-up. Who's doing what.

Stijn: [00:27:14] And then they use just one I assume as well.

Michael: [00:27:18] Not all team thought. But they also are very good to say I'm finished with this and I'm not sure what I should start on and then they'll say OK everybody will finish here. We'll find out together what you can do and what I have been doing. They go back, use this board

[00:27:41] And that's why. That's your way of saying I don't get that much of an overview. I think it's hard to get the overview. I mean, then I can collapse and then it looks like this and I can extend and then I can see the task. But then I can only two stories at a time and then perhaps there is a story, there are stories below this where there's a lot in to do. But it is hard to get the overview. No matter what tool I'm using, it's hard to get the entire view in one board. I haven't seen the tool that can do that.

Stijn: [00:28:18] OK. Maybe we should change it.

Michael: [00:28:26] We will be the first to buy it. The closest I've seen is scrum-wise.

Stijn: [00:28:38] And we touched upon it a little bit already but if you're in the process you would like to know where you currently are. I know it's sometimes very hard to estimate it but how do you go about it. What is the steps. What is it that you're looking at if you want to be aware of that. And also within the sprint but also like we did a project so within multiple sprints.

Stine: [00:29:03] Within the sprint I think they use the burndown.

Stijn: [00:29:16] And we touched upon it a little bit already but if, if in the process you would like to know where you currently are. I know it's sometimes very hard to estimate, but how do you go about it. What are the steps. What is it that you're looking at if you want to be aware of that. Within the sprint but also like within a project so in multiple sprints.

Michael: [00:29:42] Within the sprint I think the burndown not every team but.. I don't think they do because. I think the problem is, maybe it's because I have teams that works only in week sprints, they haven't time for this checkpoint. So they're not very good communicating with the product owner. If they can see they are behind. Because in my opinion when product owner and the business comes to review, the things that are in the backlog. What do you say... That's what they think they will have. So the backlog must... When you go to a review I think everything must be done. There cannot be anything that's still the commuting and that's not done. So the team needs to communicate to the product owner, so we are behind schedule, we need to put this out. So when they go to the review, everybody knows what they are going to see. They don't do

that.

Michael: [00:31:05] I've seen a few times now: the product owner enter the review and half of the things the sprint backlog is still not yet done.

Stijn: [00:31:14] OK.

Michael: [00:31:15] So they need to communicate this better to the product owner. We need the product owner to take this to the next sprint, or to put it back down in the backlog.

Stine: [00:31:31] Yeah it depends from team to teams some teams do it and discuss at daily. How's it going? Are we on track. Yeah that doesn't look like we're not able to do it? I have started to have this check question as you can say.

Michael: [00:31:47] We just introduced the sprintboard so I'm looking at the sprint board and ask the team: What do you believe? That you can reach the sprint goal. And if they say yes that's good but if some of them start to say no I think we have to have that discussion should... Or what can we do. Is the Product Owner part of your daily? No, he is not. Yeah it makes a difference. Because it's a communication is a present. Yeah but that's why we introduced the sprint goal again. OK. We don't have that much time.

Stijn: [00:32:50] When you're looking for information on the status of a project what is the first thing that you will look for.

Michael: [00:33:01] That's actually a tricky question. For the project of for the sprint? The product owner has some time schedule. So they know if they're behind on our own schedule. But I have had talk with one product owner last week. And I was talking to her about: are you on schedule? "Oh, I don't know." And then we have to talk in colors. Do you think it's red, yellow or green. Then she was, its yellow. But yellow is closer to green or closer to red? Than it is orange and we need to take some actions. But they're not using status reports.

Stine: [00:34:12] So I think it's a lot, is up in their head of the product owner. But I think we need to ask. I think also it depends on what type of projects or features the teams are doing. Some

teams are doing small features, like I solve this one. They have a lot of different features in one sprint. And some are having bigger projects. Most of the teams actually have small items they are doing in the sprint. And then they just bunch it into one sprint and they do that.

Stine: [00:34:50] And then it's. Yeah. How are we doing, as long as we are just doing some, some of these items in the sprint, we are happy. I think my product owner at the moment is very clear on the status of this project but that's also because it is a very hard deadline. The GDPR, the 25th of May right.

Stine: [00:35:13] But have you seen an overall picture of what they're going to do?

Michael: [00:35:16] I've seen the backlog. I know they are schedule, I think every company in Europe is behind schedule at the moment. The thing is also about that project, that you get wiser with GDPR. Well we need to clean up enough data and then as you go along you figure out we have got to think about how this part of the data. So how to do that.

Stijn: [00:35:42] Okay. And then the final question that we always asked and that covers more or less what we what our interest is. Is if you went on a holiday or you just came in actually.. Now you were in the project already you went on a holiday and you come back like how do you make sure you are up and running again. What is it that you look into what is it that you need to know in order to get it back on track.

Michael: [00:36:12] As a member of the team?

Marco: [00:36:14] Just the scrum-master position.

Stine: [00:36:17] I would look in some kind of a backlog tool see what's in the sprint. When I'm a bit more experience here. And I think would know the headlines you would understand that. So I could get some kind of idea of what we're doing at the moment. What kind of sprint is it. And how are we doing? Looking at the burndown.

Michael: [00:36:41] Talk to the team..

Stine: [00:36:43] Yeah but you are interested in.

Stijn: [00:36:42] I mean it, a tool can be something that facilitates certain actions. So

Michael: [00:36:49] Looks to the backlog and talk to the team.

Stijn: [00:36:51] So there are two things that you would..

Michael: [00:36:54] Because, I think the tool can show you know statistics. But it could show you if you're behind or on schedule. But if you are behind or ahead, it cannot tell you why. The team can tell you why it's behind. Has anyone been sick or what's going on down there. Has there been an impediment? What kind of impediment. A tool cannot tell those things.

Stine: [00:37:36] I think I would also look for how the previous sprint ended, like are the stories is continued into next sprint. How does it end? That also gives an idea how it went. You see the past springt, couple of sprints, also the velocity so...

Stijn: [00:37:56] Based on the talk that we had so far. Are there any additional thoughts that we maybe didn't touch upon that you would like to share with us. Because these were the questions the most important questions that we had something that maybe popped into your mind already before when you when we contacted you. OK maybe this is something that we're going to discuss with this is something that I would like to talk about. You don't have to.

Michael: [00:38:24] I think one of the most difficult parts, I haven't seen a tool yet. A tool that support different teams different projects that have dependencies within these projects. Because we have this tool, somehow get very much a silo for every team project. Because I don't think that the business-to-consumer unit looks into [name other team] team and look their backlog and see how they're going. And, but in the dependency is they're just talk to the product owner and then he most... But you cannot, I haven't seen the tool that is very good to see how the other team is doing.

Stine: [00:39:13] No, that's true but I think that that's that's not only the tool. That is the limits for the team. I think it's also just it's easy to just stay within the team and then ask the product

owner, because I have seen.. You know we have this tool and if you want to know where the story is that could go into the team's backlog and they never do and they don't understand how the other teams are using the backlog...

Santander - Morten (Product Owner)

23-03-2018 11:00

Morten: [00:00:02] Absolutely. Stine and I worked together at Top-Denmark and we were using two different tools. I remember. Right. Have you heard about that.

Stijn: [00:00:14] She mentioned it.

[00:00:17] It's huge program for Agile developing. And it's I mean it's really huge because it can do everything and you can configure it in many different ways. So it's it's it's really it's really so big that that you you get a bit lost sometimes in this incredible big program. But it really has some wonderful tools for Product Owners. I wouldn't say it was so nice and easy to use for for the teams. So developers I think would find it hard to use because it's based on simple screens. And you have to push a lot of buttons and stuff like that. So we use more intuitive tools and I'm not sure I remember.. Maybe Stine mentioned it. It's really a visual tool and it's grate for teams that are far away from each other.

Stijn: [00:01:33] She mentioned two tools but I cannot.

Marco: [00:01:36] But I think that was that was...

Morten: [00:01:39] [Rallydev] and but anyway we can we can find...

Marco: [00:01:46] I think it is Scrum-wise

Morten: [00:01:49] Exactly. Scrum wise is really cool because it updates every screen instantly. So if you make change to a story then everyone will see that that you've made this change and it's visually. It pops up and you can see that that actually if you drag a story from from one lane to another it will drag on every screen instantly. It's really cool. Nice but it completely lacks tools

for the product owner. So so so that's that makes it a little incomplete. And and I found myself building a lot of different sheets in Excel just to get an overview of the stuff we were dealing with a Scrum-wise

Stijn: [00:02:43] We are going through it, a little bit different than normal but okay. You mentioned tool and it says it's not really useful for Product Owners.

Morten: [00:02:54] Scrum-wise? Yes

Stijn: [00:02:55] Why?

Morten: [00:02:56] Because it has only this scrum board. So so you can track your stories and you can somehow group them. But you don't have the epic or the feature level. So it's only stories and tasks.

Morten: [00:03:15] A current sprint overview. Yes basically you can plan a little ahead so you can put stories in the comming sprints. And you can watch how you performed in the previous sprints and stuff like that. So it tracks velocity and things like that and it really doesn't give you the overview from a feature or epic view. So I thought that was problematic. Compared to Rallyday. It was just a piece of [00:03:52] Rallyday.

Stijn: [00:03:55] So the ones that was visual but to specific and one was...

Morten: [00:03:58] It was it was really just a visual tool for for the team. But that was that was really nice because you could have the screen of your team. For us it was in Kuala Lumpur they were ahead of many hours. And so when we were together they were just about to leave and we were just arrived. So it had to be very effective and efficient to get the right things done. So it was pretty cool that we had the picture of the team in Kuala Lumpur. And then we had on our own screens [00:04:37] Rallyday [0.5] And we could just drag stories and everyone could see what was happening. Here using TFS or VSTS. We have to bring the board on the screen share it and then we have a little picture of the person's of the team in Poland. So we are having stand up we're doing it in front of a big television like this one and then we have a little corner with the pictures of the guys in Poland and we have most of the screen covered with the scrum board.

That would be pretty cool if if that could be not something we had to share but something that will be updated. So when we say: Oh this story and then we drag it then yeah. And wave it a little it would wave on the screen. OK. That's the story you are talking about. Or we could mark it and it was marked on every screen. So...

Stijn: [00:05:36] And in the current process either within this job or any previous job you had contact with the client or an internal client. How does it work. How are they included..

Morten: [00:05:49] The clients a stakeholder or do you think..?

Stijn: [00:05:52] Everybody who can come up with a feature or something for the back log.

Morten: [00:05:57] Yeah. I had the contact. Yes.

Stijn: [00:05:59] Yeah and how does it work over here? How is the contact. How was it communicated. How do you know what other people might want to have as a feature.

Morten: [00:06:13] I think what was it was pretty similar to how it's done here. It's it's based on meetings and requests I've... Like stated on a meeting and we discuss it. And after that I used to make stories that somehow gathers up

Morten: [00:06:38] Are they somehow included in the tools.

Morten: [00:06:42] No no no

Stijn: [00:06:43] Why.

Morten: [00:06:47] Well I can make a story. But it's not something that the stakeholder or the business or the client would do. I think it's too technical for them. And you know writing a story is it's a little difficult for most people. So there's a structure and we want to do it in a specific way and we want to do it in English because we have the Polish guys at the other end... So so that's a small vary(?) [0.3] For some people. So I try to to grab the stories before someone actually enters it into the system because they can do it. But I think it's it's like a double job if I have to go

through stories afterwards and I'm punnish them.

Stijn: [00:07:46] And but but just to be clear the main way of communication with the business for example would be through meetings.

Morten: [00:07:52] Yes. OK. Yeah that works pretty well. You know sometimes they make a mock up of what they like to get coded. Sometimes I just describe it a little different but it really depends on what kind of job we are going to build or to make.

Marco: [00:08:15] Who would be the stakeholders?

Morten: [00:08:17] Actually a lot of people could be, but it would try to centralize it. And right now we have one person that both prioritize and has the contact with me. So but there are a lot of departments here that could actually have a say in what part of the program we're doing. So there's a risk department, there is a back office department, insurance department, even a legal department, marketing, the business and so on. So so there's a lot of the populace that might influence the way the product or the program should be built.

Stijn: [00:09:14] What are the main tools that you are currently using right now.

Morten: [00:09:17] VSTS. We were using TFS, but that is an on site application. And it is not updated in the same way as VSTS, which is a cloud solution. So so it's pretty nice to using the VSTS. Yeah it's it's there's a lot of new features compared to TFS as it seems to be more modern. Even though that it lacks a lot of nice things that I really would like.

Stijn: [00:09:53] And we also heard a couple of other tools such as Slack and other coordination tools. Are you using them?

Morten: [00:10:00] OK yeah. Slack is a very important tool for us but it it's basically just a communication tool. So when we use that for incident handling on the fly. We want some structured incidents communication using another tool since it's a little tease to work with. It's easier to have the communication, just instantly like chatting it's nice and then we use it as deployments warning feature. Communication tool, just basically a way for business to be in

contact with the teams

Stijn: [00:10:53] And beside two tools, are there any other tools that you use on a regular basis to communicate with other team members when you coordinate.

Morten: [00:11:05] No not really. We have we do a lot of budgeting track tracking with some excess applications someone made, which is very simple but it tracks the hours then export them into an excel sheet and I have my own control sheet. So that's something I use when I meet with the business it for prioritizing and budget follow up stuff.

Stijn: [00:11:45] What do you think in general of the tools that you're using right now. Do do they facilitate whatever you need or are they lacking maybe some parts maybe compare to what you've used before.

Morten: [00:11:57] I think Slack is very strong. But it's it's it's basically just a chatting feature or facility. So when it comes to scrum.. Control, then I think TFS is a little heavy like something between scrum-wise and TFS. I think that would be really nice. I really miss the instant features from this scrum-wise. And TFS has some kind of board view. So you can see the stories in the nice arranged order but it still is. It's like you have to press F5 for something to refresh the screen. It's so old school.

Stijn: [00:12:54] Yeah yeah it's true. Do you think, is everybody using the tools in the same way that you think of or are people also allowed to use some additional tools if they think that the tools that are being provided are lacking some type of features? Talking for yourself would you..

Morten: [00:13:25] I think we are we are we are pretty much, what you say... We have to use the same tools, we cannot just invent something we need to stick to it. I think that's OK actually. Because this kind of self-control is maybe a little too crazy right. So they took different kinds of programs and installed.

Stijn: [00:13:57] So we heard some weird things over here, so don't blame us for asking...

Morten: [00:14:07] Our I.T. department globally is very strict. What you can install. So that's just

the company rule that you would you have to use tools that you're allowed to use. I think that's pretty OK. I can see that TFS is a great tool for a company like Santander... But I must say lacks some cool features.

Marco: [00:14:32] OK can you say one of the cool features I think that it actually misses

Morten: [00:14:38] The instant updates. Yeah. So if you were able to to watch the same board scrum board, then it would be very nice if if everyone would have the same picture instantly of what's going on on the board and then generally there are some really stupid lacks of updates. So if you add something, if you copy something, than you have to press F5 to update the screen I don't know why. I mean it's like common stuff. If you add something it will end up in the list, but doesn't show. It is just empty so if should press F5 than it is there, I don't know why. I think it's you know they write in the top there's a warning saying "Your list is not complete you want complete view of the list you have to press F5". Why do I have to do that, when you know it's not complete whether it's just update it. So I don't know. I don't get it. It's really crazy what someone must have decided that this is a feature this is really needed. Nothing changes on the list, unless you want it to change even though you added some nice stuff. So we usually when we're talking at the stand-up and we say oh someone maybe you could you could reduce the number of hours left on this task and then they say well I had just just before we started the meeting. Oh yeah. Refresh Oh yes. Now it's now reduced. I mean it's so stupid now. So his screen is reduced, but our screen and the screen we're showing it's not. I think that's really something that should just work. And you know it's not a big issue but it's annoying it could be so much easier if it just worked out that way.

Marco: [00:16:42] We are here to make life easy. Some questions about on visualization. You named a couple of tools. But if you want to produce a certain output with your team how does the tool that you're using facilitate this? It could be a feature. Right. So you're developing a feature. How does that tool, like VSTS help people to collaborate together. So you you can actually see what they're doing

Morten: [00:17:22] I can track how the progress is in the sprint. So I can see where they, I can see its burn down chart. And I can see how the different stories are moving from one state to another. Interestingly enough I can't see it. You know when all tasks are done the story

continues to be in progress.

Morten: [00:17:54] So that's another strange feature from from VSTS. And somehow it's it's it's unknown an issue perhaps in the States or other places.. That you might want to have sum of or the story points in a sprint. There is no place in VSTS where I can get a sum of these stories in any given sprint. So we have to add them all together. In our heads, every time we go to plan or look how much is left or stuff like that. Of course you could look at the burn down chart and then you say: "That is approximately something between 30 and 40 points left". Maybe it's thirty five or six. We cannot see but it looks like something like that. And you know the most story-points you're having in a sprint the larger the steps will be and the harder it will be to know exactly how many story points are left. So I don't know. But there's no sum, nowhere. So when we when we were planning we're just putting stories in the next print and then we are saying how much is this sum up to. Then it's like... Oh, 35. Well we have space for a couple more and then we can track something into the sprint and then we are talking talking when someone says OK how much do we have now. Oh yes we can... We have to do it all over again. It's really crazy. I got no idea. Why is there no sum. This is this is really something that I would like to to have easy access to. They have a great plan tool. You can see each sprint and you can see the calendar line. So you can even see when the sprints are done and then you can put stories into these sprints. You can see if you fit the capacity or the velocity or something it's just like "oh you put some stories in the sprint". And then you have to figure out later if it matches something or it's totally way out of the bounds.

Stijn: [00:20:12] So that's that is one element that you are interested in as a product owner. The any other elements that you would like to have. Just just in general without taking in mind...

Morten: [00:20:23] My features stories are usually pretty short. So mosts things can be done within two or three sprints even one sprint so. So it's it's it's rare that I really have a need for a long period of sprints and long planning ahead. So normally we would just break up a job into a lot of stories and then we estimate them and they would just put them into the sprints. Some order like most valuable stories first and then depending on how many story points maybe a little small one in the end of the sprint and then it's just goes sprint by sprint. And usually 2 to 3 sprint thats the most we ended up with. OK.

Stijn: [00:21:21] So but that is you're interested to see that whole process..

Morten: [00:21:27] Yeah sure. And I think it's fairly okay with a plan tooling in VSTS. We didn't have that TFS. It's the new feature VSTS and it works pretty nice except that I can't see how many story points I added up to here. So we drag the stories into the plan then I have to guess. "This is maybe good enough. So by and...

Marco: [00:21:56] Changing artifact within the project. How do you check this. Let's say that feature has had a slight change. Let's say okay we may need to change the color from the red to blue how we get to keep track of that.

Morten: [00:22:19] We would just make a new task if it's something that we missed or if something changes then we make a new task. Probably the old one has already been done or if not it will just be removed. Yeah. We're not using the sprint backlog as documentation for what is being changed or done during the sprint. It's simply time controlling tool. We use it more and more now for mid-sprint evaluation. So we do this mid-sprint so we look at the sprint. The burn down chart shows something crazy like flat line or very... Or how do you say that. Not a slow but more like a plateau, then you then you might want to drag something out of the sprint. So I usually, together with the scrum master ask well guys "you think you can manage to do everything that you had to do from the beginning now know at this point" and normally they say no it looks terrible. And then we've pull out something from the bottom and get get us on track so being midway in the sprint will normally have a burn down line that is midway. So we'll just have to reduce the sprint or the backlog with this number of points that we're missing. So that's a great way to it's just a just a sprint. Many things can happen and normally the velocity is a good point of what you can do in a sprint. Sometimes we have a lot of incidents in the beginning and then we just simply don't have the time for doing a normal sprint.

Stijn: [00:24:18] And are you as a P.O. Interested in being aware of what team members are doing currently? What are they working on. And if so how were you aware of that.

Morten: [00:24:35] I sometimes I'm interested in it. I mean I can actually see it because we put names on task and stories. So part of the stand stand-up. Okay so let's get it gets a brief presentation every day. Okay.

Stijn: [00:24:54] And in terms of how you are processing towards the final goal you touch upon it already a little bit both within the sprint and over a bigger project. Yeah how does that work.

Morten: [00:25:07] The monitoring. I don't think we have any monitoring of the goal, so so it's basically a planning thing that we put together stories that end up to bring us to the goal that is chosen for the sprint. But of course if we change midway then we won't make that goal exactly. There's no really tracking of how we're doing. Unless you could say that whenever our story is done it's closer to goal. But there's really no connection between stores and goals.

Marco: [00:25:51] So we kind of yeah. And the other talks to me is we are talking about a GDPR project that it needs to comply. So how do you actually keep track on okay. We are kind of 50 or 70 percent done with the set. Do you have overview on that...

Morten: [00:26:09] That's simply based on the story points of their complete feature or task we have to do. So if that say I have I have a feature that goes for three sprints, then that would be like 120 story points in general and if I've done 60 story-points them I'm halfway.

Marco: [00:26:41] But you need to sum that because it doesn't provide...

Morten: [00:26:52] Exactly, that is so true. That's that's actually the way I'm trying. And you know. Saying that rather I actually could make a very nice horizontal bar of your of your feature or epic and show you how far you are of the total amount of story points or stories, so you could choose: "Do I want to show how many stories of the total I've done or do I want to show how many story-points I'm done with. I know those two figures can be quite interesting. Sometimes you may not be that far ahead in the story points but that's because you have just picked all the big ones in the beginning. So there will go some time before you you actually can produce the right number of stories.

Stijn: [00:27:47] We're almost there by the way. OK. But in terms of information about the products that state is what is the first thing that you look into.

Morten: [00:27:57] Products stage?

Stijn: [00:27:57] Status. What is the most important for you.

Morten: [00:28:06] Could I say deployment. Is that part of the status or. Yes. I mean everything that we do in sprint leads to some kind of deployments and you know that even though it's not really a part of the stories then we have this deployment after each sprint. So if things are done they might not be available for deployment yet. So. So it's it's always you know most important thing is to to be able to deploy. So if we can say this one this feature of this story will go to production Wednesday morning or Thursday morning.

Morten: [00:01:31] And it's really like a thumbs up. This is something that the business will be able to to use for something. So so so really the deployment of a feature or story is absolute..

Marco: [00:01:48] Yeah. So you have some access to the release notes so how do you become aware of something as deploy.

Marco: [00:01:56] That's a question how do I do that. I actually talk with a team. Maybe I could watch something of VSTS. I'm sure there may be some tools. It's really integrated with our deployment system Octopus, checking repository thing but you know it's a part of VSTS that I am not using. Maybe it could be used, at least there are some nice controls for the dashboard that shows what went well what went wrong for deployment and committing and stuff like that. I think I could see it what I just talked with the team say: "Okay guys what kind of stories are we putting into production at the end of the sprint". And then they tell me.

Stijn: [00:02:53] And this actually the last question that we have. Imagine that you go on a holiday for like three or four weeks and you come back. Yeah. What is the first thing that you do? Besides saying hi to you colleagues and..

Morten: [00:03:10] I think it would be just having a chat with the guys and know did everything went well. Are you sad about something. If they say everything went well. And I'll just go..

Stijn: [00:03:25] On a holiday again.

Morten: [00:03:30] I would rather start thinking of of the things that we are going to deal with. You know when I come back from holiday there must be some kind of low level in the backlog. So I need to put some more stuff into the backlog and getting prepared for the coming sprints. So I'm usually thinking one or two sprints ahead of the team. So either I have to to make that ready before the holiday or just must do it in a hurry when I get back from the holiday.

Morten: [00:04:08] So but fortunately I would say I'm having a holiday in July, so August is normally a very easy time at Santander So. So it's not that terrible to get back from. Are they OK. I can easily get some stuff and get ready for work. But yeah when I get back from holiday there are definitely some people wanting things to do so. So that would be my first priority to get that ready. No but that could be something that we didn't make during the holiday which was planned. And then I'll just try to realize what was a problem and do we have to try again or is it simply just deploying all eyes it's really necessary is it just not needed anymore. You could..

Stijn: [00:05:05] But that general overview of what happened in the meantime is it only by talking to you did you get it or.

Morten: [00:05:12] Yeah I think so and you could wonder why but I think it's basically because a tool is not interesting enough to be used for that. I mean I can see a lot of stories but it really doesn't say what happened or what went wrong or what was like Yeah. So but I don't I don't find need for digging into the backlog unless I mean I know what what we planned. So just knowing that everything went well enough to say OK fine. Then it's it's a sprints that it's gone now. Yet counts up in the velocity but it's it's not interesting watching which stories were made. We are tracking stories. It's like it's done. It's always a piece of paper or you were shopping and there was a list of things you should buy and you bought them and then you just drop the shopping list. Maybe you save the amount of money you spend at the shopping but you don't actually keep the short list unless. You know that this is what you're going to buy every Monday from now.

Stijn: [00:06:36] Yeah based on the talk that we had so far because these were the main questions that we had or did I miss... I think there was it. Anything that maybe popped into your mind beforehand or during the talk that we might have missed covered something in relationship to what we talked about. You don't have to.

Morten: [00:07:00] No no sure. You, when it comes to big projects like a lot of teams and product owners, maybe teams that don't have a product or maybe teams that don't even work agile. When it comes to organizing those projects then we have a project manager. We have actually two that do this kind of stuff. So they have some different tasks to do during the year. Sometimes it will be a specific new product that we want to make which really includes everyone in the house to contribute. And we seem to have a little trouble with just fitting the agile way of working into the non-agile and then again we're using scrum tool and with describing things and stories and other teams like Risk are not agile yet. So so they're kind of jobs like writing things in a spreadsheet and communicating using email and stuff. So something to to help these product managers

Stijn: [00:08:13] Like onboarding..

Morten: [00:08:14] Kind of yeah perhaps. And something that could visualize how far are we... How far are we with the progress of the total projects. When do we have to bring in people from risk.

Morten: [00:08:31] When do I have things ready for testing from other departments and all that kind of connection between the participants. This kind of project and seem to be too hard for us to to manage. So we're using different tools. We tried spreadsheets to try Microsoft in the cloud and some different tools. So it all seems to be inefficient. So. So it would be great to have some some tool that could track on a higher level. Maybe VSTS can do that but I can't figure that out. And it definitely lacks the sums. There must be a product manager that thinks: "Where are all the sums" and the tracking exactly the tracking is a piece of information. I think it's hard to get from VSTS

Stijn: [00:09:34] Ok thanks.

Morten: [00:09:36] You're welcome. Thanks a lot. I hope you got all the stuff.

Marco: [00:09:41] Most definitely yes. Yeah. Actually one actual thing since we have a couple of seconds. Yeah when you actually have some backlog items and the developers come with a

need to for a clarification. How do they do that with. They know your door. Or are they.

Morten: [00:10:08] Well if if they are here in Copenhagen they knock my door or we're seeing just besides each other. So they just talk to me right now but if one of the Polish guys needing some information the write on Slack. And if the business is uncertain about things they might want to look at the stories as well and then they might think "Oh are you sure this is the right way to do it all. Are you sure you've got it right? There could be some. Not exactly how to code it but that could be some specifications that they.. They do not agree about. So then they just grabbed me. Write me an e-mail or call me or something. No it's not. It's just something we could do in VSTS. There's no like process for this kind of refinement of stories. No. And I wouldn't expect to use that kind of process. I think it will be too soon too technical. I mean it's simple just talking to people. When it comes to that is simple just a talk, how I would really like to have processes fall for the scrum sprint. I mean that seems to lag a little in the VSTS. Give me the sums please. Yeah.

Stijn: [00:11:42] He said that 14 time 'sums'

Morten: [00:11:47] Hopefully some Microsoft official is listening to this at some point. Thank you very much. You're welcome. I hope you get a great.

Adapt - Niklas (Product Owner)

28-02-2018 13:00

Stijn: [00:00:00] First question would be what does the company do? Just to have a general idea of what you're doing and the company is doing.

Niklas: [00:00:10] My title is UX designer and digital strategist. And we have, we have that role in each of our teams. Right now we have four teams with inter what we call inter disciplinary areas. We have back end and front end signed us. And the idea is that I should have the overall view of the clients and all the different projects we do on a strategic level level and as well as in the end product, the quality of that, and then we that's the structure of our teams. And then we have the Scrum roles which we give to each other in relation to each project. So for example

right now we have a project where we are building an order system and in that project I have the product owner role.

Stijn: [00:01:20] Okay so within different projects you can have different roles, so to say?

Niklas: [00:01:25] So we have another company where we are just maintaining their websites. And one of the other UX designer is the product owner and I am just the overall person that talk to the client sometimes if the product owner. If it's not like a specific thing thing in the project , like they're prioritizing in relation to their business. But it's more on an overall level than I talk to them. But else he is the product owner of that project. So it's yeah it's a role that we give to each other in relation to what project it is and what client.

Stijn: [00:02:05] In terms of just within from your different goals. And you also have the team. As if you take the role of UX. Are you then within the team or is that not...

Niklas: [00:02:16] That's still within the team. So the scenario was that from two years ago there was not design and UX design in the teams. So it was more like the project manager that took the project owner role as well. And then we figured out, okay makes more sense to have the UX try to take some of that roles and in some cases there are some teamwork between the product manager and the UX designer and splitting the product owner role up between these two persons. But again it's something we try to discuss what is the pros and cons for these projects and who's should take the role.

Stijn: [00:03:02] Okay. And in shaping the team. How does... Can you give us maybe an example on how a normal team would look like if you have?

Niklas: [00:03:11] Actually we have we don't have project teams we have client teams so it's the same people I sit together with every day and then in the client teams we have different project from different clients or different just a big client that have they want to build like an order system they want to update their website. They also want an app. And then we create smaller projects within the teams. Yeah. Yeah.

Stijn: [00:03:41] No no. My my my question was like the composition of a particular team...

Niklas: [00:03:44] So a particular team is front end, back end, UX designer the design, product manager.

Stijn: [00:03:52] Okay. You mentioned a little bit but what we are what our frame is within our which is Agile which you mentioned Scrum already bit. Could you play a little bit what elements you're using from Scrum. Are you using all of it or just some parts.

Niklas: [00:04:11] I don't think we're using all right but I'm not that theoretically involved but we have, we have the concepts of daily stand up, planning Poker, Sprint planning. And then we have created a concept called storytime which I don't know if this is part of Scrum or something we created in relation to is primarily UX or front end or UX and backend or the product owner and front end, product owner and backend discussing how can we solve this in different ways and in trying to break the task down before we all sit together at a sprint planning so try to prepare how the solution could be before we sit a lot of people together in the meeting and try to plan the sprint and hopefully have a decision on how do we solve this task. Let's use the story.

Marco: [00:05:11] Just like a refinement session?

Stijn: [00:05:15] A typical Sprint is two weeks or?

Niklas: [00:05:19] Two to three weeks. But it depends on the project. And yeah, and if the daily, the daily stand ups we run through the user stories in the sprint and, or we run through the tasks of the user stories that is in progress and take a status. Is there any issues? How is the progress here? Anything that is blocking to get further? Or yeah. So that everybody get a hold of what each is working on and how the situation is...

Marco: [00:06:07] Do you also run demos and retrospective?

Niklas: [00:06:10] Yeah yeah yeah we do as well depending on the projects we we almost run like an internal demo so everybody can see what have we built in this sprint and then it's, it comes back to the project and the client. How often we do a demo for them as well or some. Sometimes we have like half halfway demo. So let's say project that runs for three or four

months after two months we have a halfway demo with the client and they can see. Now we are that far, we're missing the last part. Yeah. Yeah but the internal demo we tried to do after each Sprint.

Stijn: [00:06:52] Okay good. Could you maybe explain a little bit better, like how the general communications with the customer that you have, so you have a demo that is a way of communicating. Is there any any other point in time that you have a fixed appointment with them or maybe at the beginning or the end or maybe in the meantime as well that you...

Niklas: [00:07:14] Right now with this order system we are building, where I am a product owner right now. I have a weekly meeting with them to discuss stuff that comes up if it's like an urgent matter then I just call them, or else we have this meeting to like catch up on stuff, besides that we have like specific meetings for clarifying specific areas. So like it's a bigger chunk of the system we need to get clarified or refined and then we have a meeting for that, then we sit together and only talk about that. Yeah. And then in this project we have difficulties getting the client down into the sprint and help us testing. So we're doing all the QA ourselves and not having the client in that process. But normally with other clients we do that. So we have like a lane in the sprint in Jira which we are working in. So after it's in QA we put it in customer test and then they get a notification that this user story is ready for testing. And then the client can go in and test it and reject or approve.

Stijn: [00:08:42] Exactly. And you mentioned that you somehow communicate with them through a tool which is Jira, are there any other ways in which you use tools to communicate with the customer or is it more the physical appointments or appointments...

Niklas: [00:08:59] It's more the physicals appointments. Phone and email. But it really depends on the clients. Some clients say Okay to Jira and start commenting and using Jira a lot and others is just like OK it's your program. It's probably fine for you but for us is very confusing and we can't see any sense of using it. So all communication is outside of Jira and then we have to go in and update on the decisions made.

Stijn: [00:09:34] And next to Jira what other type of tools are you using that support your software development process. It can be anything it can be in terms of Slack in terms of communication.

Niklas: [00:09:45] Right now we have HipChat. I know there is a process of maybe shifting to Slack instead of HipChat, then we use our Google calendar where we use Google Hangout I think for for remote meetings. We have developers in Lithuania and Boston , so that we use for, that we all use for stand up for meetings where Lithuania is it's yeah yeah. Don't thing we actually have then it might be more technical tools that I don't know.

Stijn: [00:10:41] It's more of what you use on a daily basis. That's what we are interested in.

Marco: [00:10:44] In the position of designer do you use something like...

Niklas: [00:10:50] We have we use Sketch and InVision for building prototypes. Sometimes we also just build an HTML clickable prototype. That depends on the project. In some scenarios where the designs or the elements it's very defined. We can also have user stories where it's just the acceptance criteria and maybe hand drawing or sketch related to. Depends on the complexity of the user stories, what is needed to align with the acceptance criteria. To figure out what is we, what are we trying to achieve here.

Marco: [00:11:41] Do the customers actually getting involved with InVision prototypes like they actually comment on that?

Niklas: [00:11:47] We try to use the product we don't use the comment element. We more use, use the prototype in InVision to see this is how it probably it will look like, this could be the interactions and then we use the we still run through the acceptance criteria of user stories to see this is what you can do when the task is done because when you go into this detail talks about the acceptance criteria, always some scenario that we haven't talked about comes up or they. Yes that comes up with good dialogue. If we are missing something or if that's not important at all. So it makes sense to have that dialogue with the client before going into development.

Stijn: [00:12:41] And if you include developers in Lithuania or in Boston how does that work? Is it that you have them in a big screen and have Hangouts in there and that continues or is there anything else involved?

Niklas: [00:12:56] We have down in the team area we have Chromecast and some kind of audio device which actually works pretty well in relation to hardware we have before where the sound was pure or I don't know if it was the Wi-Fi or the hardware but the quality was difficult to to talk to each other. But yeah, if it's in a meeting here, we normally have like like some [inaudible] where the audio and then of course the video as well so we can see each other and hear what each other says. And then we in the Google Hangout you can also share your screen. So it's typically something that you share the screen in the meeting room. And then we also cast it up here so people here can see what we are talking about and people in another room can see through the Google Hangout. What we are pointing at and what we're looking at.

Stijn: [00:14:05] What do you think in general of the process and the process of or its tools that you used within a specific process?

Niklas: [00:14:14] Like the Scrum or.

Stijn: [00:14:16] No no no more the tools that you're using. What is your like general positive experience that you have with where there are many flaws that you might encounter?

Marco: [00:14:29] Now again it's more the in the role of a product owner, right?

Niklas: [00:14:36] Yeah it's hard for me to compare with like other tools because in my old job as well we were using Jira and Sketch as well. So that's what I have been working with. On an overall level I think it works for the scenarios where we need to clarify stuff and agree on what it was. What it is it we are trying to build in relation to the team in relation to clarify the stuff with their client. Typically the client is more visual and it's hard for them to figure out what we building if we don't show them something. And when we talk about a task in the team it's easier just to talk about the acceptance criteria because you know this stuff and we can reuse this, then it's easier just to use the user stories and in relation to use Jira as a tool. I think it works sometimes it's very slow it can irritate people. And then there is as well. It's also in Jira like we, what's called

log our hours for. That's the way the business is handled, we have to log like every 15 minutes. Yeah. And that we do in Jira as well. So there is a lot of like maintains maintaining of the task and the hours you use on different tasks especially for other developers but also as a project owner because it's difficult to the client say. Okay why have you use so many hours? If I only have one log that says backlog grooming. Then is not very precise is it. It's just the whole backlog I have groomed. Or is it a specific area we have tried to go into detail and clarify so. So I as a product owner need as well different areas to see Okay how many hours did I try to figure out service A and B and C. Yeah. Okay that makes sense.

Stijn: [00:17:08] Yeah definitely. We have some questions as well about visualization. If you want to I'm asking you right now come take this from both designer perspective and a product owner perspective.

Marco: [00:17:26] Yeah

Stijn: [00:17:26] Yeah because if you would like to produce a certain output, you want to build something together with a colleague. How does it work and how the tools that you use facilitate this so you cooperating with three or four people on one specific tool, how does the communication, cooperation, coordination work. Could you maybe explain or give an example of how do you do such a thing.

Niklas: [00:17:48] If we start with the designer. Yes. Maybe then we have agreed in adapt to use Sketch. So Sketch we use it, it's just a very low fidelity wire frames or it's a high fidelity prototype or design. Are you a digital designer and goes into details. So it's the same software we use which makes it easier to just have different art boards. So let's say the UX designer could start in Sketch and create some wireframes of low fidelity wireframes and then the digital designer takes over with the same file and just update that now. And besides that, in relation to the teams, mainly of the time we all sit together. So it's it's easy to just go to each other's table and look at it at the same time and discuss a solution or way or approach. And you said the product owner...

Stijn: [00:19:09] Also from a perspective of a product owner. And with that maybe with a focus or like what the goal would be of the tools that you're using within this if you want to produce a

certain output again together. So I can imagine that you have on Jira, you have a list and then a back ender or front ender needs to do some test from there. Like how does it work. How does the process look like?

Niklas: [00:19:35] Like is it, are you thinking of preparing Sprints when they take the task and start developing? Or is it another scenario you...

Stijn: [00:19:46] No it would be more the first scenario that you described.

Niklas: [00:19:48] Okay. So from the start we have these planning poker sessions where we talk about the task very high level and as a product owner then I can get a feeling of is this task very complex or is it not that complex. We tried to give the story points in these meetings and if something is very high tried to state out why is it high. Is it because there's a lot of uncertainty about how do we gonna solve this or is it, because we have a specific acceptance criteria that's that's the complex one and then I can take for example that acceptance criteria out and put it in a task for itself. So it's easier to prioritize the different elements in... We can an Epic we use that as well. From there on then I can take take the task. Let's say do it with the top the backlog we have the planning poker before then can take these tasks to the client and discuss the different task scenarios and see what is important for them or what is not important for them. And then when we our next meeting could then be Sprint planning unless we put in these story times for further refinement before the sprint planning. And then it's the sprint plannings. That is the whole team back end and front end can break the user stories down to some task and yeah and state what, what do we need to do to accomplish this user story. OK. So it's a lot of our talk is about the user stories and the acceptance criteria there. And when the sprints begin then I'm only tagged in the stories or I only get involved if if acceptance criteria like explodes, something we thought we could do it this way we can't do it this way after all. What to do. And if that doesn't happen then it's more the team and the Scrum master try to finish up the Sprint. So is only if we need to talk to the client about this acceptance. OK. We do this instead of the original idea or yeah

Stijn: [00:22:47] What would happen if an artifact would change over time while during a Sprint. So for example you have color blue needs to be changed in color red during the sprint. How does that work? If such change occurs?

Niklas: [00:23:04] Like like something I'm going to do it in another way

Stijn: [00:23:08] You need to do something within another way but you do it like within a sprint already.

Niklas: [00:23:14] If they want to do. Normally it depends on what the different approaches are. So let's say developers say we can't do this but we can do this within a time frame that makes sense in relation to the sprint and I could go back to the client and say. Is that okay. Can that still accomplish this and this, will that be fine. They can say yes then we try to finish it up in the sprint. If they go, don't go along with that and want to do the original the other maybe now takes 20 hours instead of five hours then we will normally take that acceptance criteria out and place it in their own user stories and they need to prioritize it in for the next sprint. Or else we will end up having a lot of stuff we can't close within the sprint.

Stijn: [00:24:13] Okay. Does actually the process that you're using right now does it provide any type of awareness what a colleague is doing at this particular point.

Niklas: [00:24:26] No I don't think I think that's actually a difficult part.

Stijn: [00:24:29] Okay. Would you be interested in?

Niklas: [00:24:31] Yeah. Because it's only up the days of where you get an idea of how is how is it going for the colleagues and are we on track are not on track. And sometimes it's not even that clear stand up because it depends on what the developer or person say in relation to that task. If they get a lot of knowledge you get some ideas and if they just say it's fine everything is expected so far.

Stijn: [00:25:01] Okay so that's kind of a pain point right now. Yeah.

Stijn: [00:25:09] Or else we have this in HipChat we you can have one to one communication you can have these groups and then we normally have a group for each project. So we tried to have a lot of the communication in the group so everybody can see what like two people are

discussing even though it's not related to you then you still have a place where you can see what they talked about. In relation to this task you're working.

Stijn: [00:25:39] Okay okay. Okay.

Niklas: [00:25:41] So that helps a little. I think when it's a big project with this order system we are six developers on it. Is I think it's difficult to know if everything is. Yeah. What is people doing it and how is it progressing.

Stijn: [00:26:05] OK. Not a question in terms of where you are actually within a project relates a little bit to what you said before but if you want to know where you are within the project.

Niklas: [00:26:19] Like the whole...

Stijn: [00:26:21] So stating, I know it's maybe not the correct manner but stating I'm halfway there. Yes. How do you know that. How how how do you get the information from.

Niklas: [00:26:32] And that's that's maybe the part where we took some elements of the product owner and moved to the product manager. So in this case she's the one trying to have the overview of how is the whole project progressing. Are we behind, are we do we have an extra time and we use the story points from the planning poker sessions. So all the users stories in the backlog that we think or a bad expectation will be part of a version one of what we need to have ready for for the deadline have a story point. And then when we have finished up a Sprint we have the exact estimate and then she do some kind of calculation and get what you call it velocity Yeah. And to have an idea how fast are we finishing the task that you then have these different story points. Yup. And then she create a forecast. Yeah. And then we have an idea of are we behind or.

Stijn: [00:27:55] Yeah. Okay.

Niklas: [00:28:00] We actually, we tried to use it on some of the standups to see do we need to try to be more like cutting, cutting... Yeah. Did get more... Is this acceptance criteria really important or can we solve this in another way that doesn't take 10 hours but five hours and

maybe we can't do this then. But it's still okay in relation to the business or the project we are building and if we have extra then there's maybe room for us see can we do this in another way or even better.

Stijn: [00:28:44] Make it a bit cooler maybe. Yeah yeah.

Niklas: [00:28:49] Yeah maybe once a week on the daily you bring this forecast.

Stijn: [00:28:56] From your position provide enough information for would you would you rather be more up to date or is it the amount of information that you get.

Niklas: [00:29:06] No arms I think it's actually okay because if you were like were behind if you didn't have the mindset of you're behind all the time then I think you would be lost in the focus in the daily work you still need to build a project or product that is accepted or acceptable for the business and ourselves so. So I think it's actually fine with a week right now and then and then there also been some day so you can actually see we improve the forecast now haven't we made it even worse or if it is day to day then it's maybe just small changes that you don't even see.

Stijn: [00:29:28] And if you are looking for information about the project itself, project status what is the first thing that you would look at.

Niklas: [00:29:39] Like the status of the whole...

Stijn: [00:29:41] The status of the project. So you walk in, you take a look at your computer and you want to see a stage of the project. What is it that you would look at first. Is it just a philosophy or there are maybe any additional sources of information that you would go to to...

Niklas: [00:30:01] I think it would be only the velocity, of course the product manager who creates the velocity. Like try to get to say you just take all the task in the user stories in the backlog and just calculate the velocity she also goes through some of them and see OK why was this especially why did this task explode. Was that because it was in the beginning where we didn't know anything or was it because the client couldn't decide and change their mind three

times. So she try to figure out is this something that we need to take out and out of the calculation because that was just blurring the picture. So in that perspective I as a product owner on this case, just rely on the stuff she tries to calculate. Is that forecast real or not real. Try to be skeptic, or reflective about the forecast.

Stijn: [00:31:12] Yes exactly.

Niklas: [00:31:18] And else is also when we talk about the forecast it's also like there always be some kind of gut feeling in the team that is do we also think we're behind or the in relation to that we know that we have to build all this or or are we actually really worried, even though the numbers say that we are behind, but do we have some other explanation why the numbers are that bad and we still have a good feeling about it. And then we'll will be a talk at the the standup where we will bring in the forecast.

Stijn: [00:32:01] That makes sense. It's actually my follow up question would be like how do you rely on data but you make a very clear combination between data to make a decision but also your gut feeling.

Stijn: [00:32:11] And I forgot to mention we also try to use some time this burn down chart that we have in Jira. And that's the same if we suddenly see a bump or something. Does anyone have an explanation. And when we find the explanation is that okay or is it something we need to take action so try to discuss and be critical about the data we rely on. And last question is if we are going on a holiday or for a long period I say three weeks and you would come back and you can answer this from both from the perspective of the designer and the product owner. What is the first thing that you would look into if you were coming out... Except for say hi to your colleagues...

Niklas: [00:33:01] And ask them how is it going... That would probably be Jira and see what task are we finished what have been removed from the top of the backlog. What are we now. What do we now have left of the backlog. And then probably also asked to the forecast and probably also look in the sprint to see that, how is it going with that one.

Stijn: [00:33:34] Yeah sounds good. Do you have any additional questions. Okay. Do you maybe have something that we covered, most of the questions are of questions that we want to based on the talk that we had so far you'd think that we forgot something that we thought of that we didn't ask about could be a case yeah.

Niklas: [00:33:59] No not really. But in relation to us you talked about it in the beginning. Now you do these interviews and you would like to do some kind of prototype. Is that off what it would be. Is it something you will try to go out with in a team or...

Stijn: [00:34:18] What do you mean by that?

Niklas: [00:34:19] The prototype that you would you try to test that somewhere or was it just to maintain the idea as you get from the interviews.

Marco: [00:34:29] Yeah our plan is to do a presentation for you as a team and show what we came up with in terms of data visualization. So our idea is to actually build some sort of dashboard where we can actually it's basically related to our last question. Right. So you come back here and you come back after a while and then you want to be updated on the status of the project and in one screen we are trying to convey the most information for you in terms of what has happened since you've been gone. Yeah that's our.

Niklas: [00:35:04] And that's actually a difficult part in the Jira that we use right now. Plus when that task is done then it just disappear out and you can still find the task, but there's no like the place where you have a good overview of all the task you have done. Or sometimes we create relations between the user uses stores that need to finish this before that one then we finish this one in Sprint two and we will start this one in Sprint five maybe. But then there could be some information on the old one that makes sense just to get a short view of when we start when we started this one in sprint five. And then if we haven't made the tech relation that we can easily open it now then it's a hard process of finding the old one and getting that information.

Stijn: [00:36:01] So basically goes into being and...

Niklas: [00:36:03] Yeah you can search it or try to search by name or number. But it's not an easy way. And even though it's done in relation to the opinions of the Sprint ended up in QA'd and everything's fine when you have a long project it could still make sense to have them somewhere somewhere because you're not holed up yet. Even though the specific task is done so it makes sense.

Stijn: [00:36:33] Yeah. Can I say from this that if you go back you would like to have some type of historical overview even though it's...

Niklas: [00:36:40] Yeah I think so. I know.

Niklas: [00:36:46] Okay. I think that was it. Yeah thank you very much for your time and have a great day.

Adapt - Ivan (Developer)

28-03-2018 13:45

Stijn: [00:00:01] Could you maybe shouldn't try but the company is doing.

Ivan: [00:00:04] Yeah well we are a digital agency and we are developing Web sites and stuff like that. A lot of companies. We have offices in Boston and Lithuania. We have a lot of big plans. Yeah I was a developer. That's that's I will do a lot of other stuff but that's the core. That's the core and that's what I'm mainly involved in.

Stijn: [00:00:37] Okay. And how big is the company in having.

Ivan: [00:00:41] Well I think we just got together and we were 130 you know in all the different departments. I think we have around 8 in the market and how many developers are we I think around 12 15 something like that. That's both front end and back end developers. We have this split in all we are around 50 I think, in Denmark.

Stijn: [00:01:10] And how does it work in terms of team size teams. Yeah yeah yeah

Ivan: [00:01:16] We have different teams with different clients located in my team. We have one major client which is inter flora that. In all the different teams we have developers backing front and we have UX us we have project leader project managers. Yeah. That said we run in these self empowered teams so we each team is actually responsible for everything in regards to the clients which like invoicing and sales and stuff like that. Yeah.

Marco: [00:02:01] So it would be around six people in a team? UX, backend, frontend.

Ivan: [00:02:09] Yeah let me just count. I think we are a bit more we have 2 UX in my team we have I think 4 developers who have developers in my team which are sitting in Lithuania as well. So the team size differs. But I think we're closer to 10 in my team 12 maybe I should probably know. Do you want a more exact?

Stijn: [00:02:45] It's just an indication. Just to be clear you are a team member you are developer are you front end, back end or is it not.

Ivan: [00:02:52] I'm I'm what you'd call back end. I started developing before you would be talking about full stack and stuff like that. But I mainly due back end I don't do front end too.

Stijn: [00:03:06] And in terms of because of the framing which our research is within Agile you are using over here. Could you explain to us a little bit how you are using it, what elements of it you're using.

Ivan: [00:03:19] Yeah well it's not my it's not my main interest. But we're using standups daily standups, Planning Poker, but what else is we have a project owner, we have a Scrum Master kind of it's not always that strict.

Stijn: [00:03:56] Is that a developer or is that like a separate role within.

Ivan: [00:04:00] Well actually it's not really something we do that like strict. We have a developer, he is on paternity leave right now some but he is usually the one with the Scrum Master cap on. But it's sometimes it's it's it's a project manager. So from time to time it hasn't been that formal so it's just someone picks up the ball in the daily standup and goes with it. So

Stijn: [00:04:26] Okay. And from your own professional experience do you in any way talk with the customers while you're doing your work.

Ivan: [00:04:38] Yeah yeah.

Stijn: [00:04:41] And if so, how does that work?

Ivan: [00:04:43] Oh it's it's often when we have a new project or new development project for their site. We visit them and talk to them about their needs. I think is usually a good idea to have a developer because we know all the technical stuff what's possible. So then we just participate in a meeting.

Stijn: [00:05:16] So that would be like a one time physical meeting but besides that are you maybe maybe the customers included in any tools that you're using?

Ivan: [00:05:25] Well as they are, we are using Jira and the clients, the customers do have access but it's kind of like some of them use it some of them don't bother reading what we write in there. But Inter flora for example they're not using it, they have access but I think it's mainly by e-mail and I don't have that much contact with them, more goes through the project manager. So yeah did I answer your question.

Stijn: [00:06:04] It's more more like yeah you have you have some contact with the customer but it stays limited it's

Ivan: [00:06:11] It's definitely limited yes. But that that's also different from team to team and from clients to client.

Stijn: [00:06:16] Yes but you can only speak for yourself. What type of tools are you actually using? The entire set. So where are you coding in, it was you you mentioned Jira before.

Ivan: [00:06:29] Are you talking or what.

Marco: [00:06:32] Yeah you can't go crazy talk about Jenkins.

Stijn: [00:06:35] He understands everything I don't.

Ivan: [00:06:38] I would start with emacs. I think I'm the last using emacs it is certainly lonely but we're using GitHub for our code repositories and we're using something called Circle C.I. for running our unit test. We [inaudible] tests as well. And we also use it to deploy some projects and then we are using something called Code Climate which is used for checking code style and different good practices in our code. And what else are we using? I did say Jira. I don't know. These are the things I have most to do with my daily work, I think.

Marco: [00:07:55] Do you work with one repo per day. Let's say so you have this product with the floor so you have just one GitHub repo for it.

Ivan: [00:08:08] No no because on Inter flora we have a lot of different subsystems we have the webshop which is what which is what I'm on mainly and then we have the we have a dashboard for their accountants so they can do marketing so they can see the track the daily sales. Then we have an order management system which we're working on for them as well. So we have like this one client which has around I think four or five different code bases and then. Today I'm helping with patching these websites later so that's like I don't know 10 or 20 different repositories but well.

Stijn: [00:08:58] What do you think in general. We heard some developers with strong opinions, whether they like it or not like.

Ivan: [00:09:05] Yeah well I think Jira sucks. It's slow and I prefer GitLab and not GitHUb. I think that's more...

Stijn: [00:09:17] Why would you prefer that over...

Ivan: [00:09:19] I think well it's more like a personal preference I've worked more with GitLab. I just think they do stuff a lot better. I'd like to get host it myself, I can do that as well. They have a lot of awesome continuous integration. I think. And they keep improving it constantly. And

emacs is the best editor. But but I also use VI. I think I mean, I don't have any complaints in general. Jira is slow it's tedious but maybe it's not. I think we could we could definitely if we had a simpler system which could which had all the features we're using from Jira. I think that would be better but I don't think such a system necessarily exists. I have been in different places where Jira wasn't used. I don't think something called Harvest maybe as well for time registry. I guess it

Ivan: [00:10:42] Just a simpler version. Yeah I think sometimes maybe it's also because we are not posting it ourselves so it's just posting which makes it very slow sometimes and sometimes it doesn't work so we can't log out, work on it. I don't know how often but it's at least once a month there is issues with it. Maybe that's not that often but it's when it happens. It's annoying.

Stijn: [00:11:08] When you say simpler version what what what elements would you like to have in there. Or is that too big that I am asking right now.

Ivan: [00:11:16] Actually I when, when they change the UI I don't know where to click because it's not really to me it's I mean I understand why we use the tool and I appreciate it but it's more of an annoyance in general than all the different configurations with [inaudible] and how to look at a project with different the different teams do it in different ways. And if you help on another project the things just look completely different. I don't know what where to look and I don't I'm not really interested in spending a lot of time on getting to know.

Stijn: [00:11:56] And the tools that you use. If you have to produce something better in front of for example how does the tool set of tools facilitate this process.

Ivan: [00:12:07] Are you talking about front end in particular?

Stijn: [00:12:10] Or somebody else within the team. That can also be.

Ivan: [00:12:14] Well. We have. We are using SASS or SCSS for compiling our css so you can I have the question again. I'm not sure what you want.

Marco: [00:12:34] Yeah let's say what you're mentioning right you have a front end you as a back end want to work together on a certain feature. How do you collaborate? Does Jira actually help you out on doing something together.

Ivan: [00:12:54] Maybe a bit. I mean we usually if we have a user story we have added some subtasks some for the back end some for the front end. So I think usually and of course it differs, but it's most often I think it's the back end task needs to be done before the front end task. So back ender does the task marks it as done or puts it in peer review or whatever happens and then tells the front ender who's assigned to the front end task or makes a comment on it. So the next front ender working on it can see what has been done saying say you need to pass on some information we just log in to the front end task or the user story and then yeah I think we also talk together, of course. I don't see Jira doing a lot in helping that specific case I think. Other than we can mark the task for front end and back end.

Stijn: [00:14:09] Any the other tools that potentially help you within this process or is it... I'm not a developer so I don't know if you have to cooperate with each other I have seriously idea no clue how.

Ivan: [00:14:28] So of course we use Git for it, but I mean that's just a big part of our daily work that maybe just I think it's just implied that

Stijn: [00:14:43] If something like the artifact changes throughout the project how would it work? How would you... What kind of impact would it have? So for example a color or like a color needs to be changed from red to blue like throughout the project. How would you know if that has happened throughout the process?

Ivan: [00:15:11] Well I hope that will have a variable now satisfied so that it's only has to be changed one place asking me how we would know it how you would know. I mean we someone would have to check. Is that what you're asking?

Marco: [00:15:37] Yeah. Let's say for the day for instance the customer would reach your product owner and say, okay the webshop here I actually need to to have added taxes or something like that. So I need to have a system that I can manage the taxes of the products that

I'm selling in. And you already kind of have build the total sales or something like that. So it's kind of a new feature. But that you already have been working in some way. So the product owner from there. How how will they actually reach you, right? In order for you to collaborate?

Ivan: [00:16:20] Are you talking about change requests?

Marco: [00:16:21] Yeah a change in the requirement, basically.

Ivan: [00:16:30] Well sometimes it's just a comment on the user's story. Sometimes, the acceptance criteria is changed.

Marco: [00:16:44] But how do you get to know that. If the acceptance criteria changed for instance.

Ivan: [00:16:49] Well I think that sometimes I get an email if it's changed and it might only be if I'm assigned to a task in Jira, I am not sure how it works. But it it's definitely possible to change it without me knowing it if I had if I'd read it at some point before. I'm not sure whether it's the assignment that does it but sometimes you just get a mail. If the description of the acceptance criteria has changed.

Stijn: [00:17:24] Are you actually aware of what your colleagues are or doing throughout the project in some way.

Ivan: [00:17:32] Yes I think in general I've tried to do it. I'm usually the guy who does the merging and creating the release so I tend to know what's going in the next release and tend to know what other people are doing. That's just I don't have a I don't know we don't have a tool for doing that. We do use Jira to mark to make release versions so we tag our issues on those stories so we know what to deploy with. It's more like, good memory and talking with the team. So we know what to put out and it's not that formalized.

Stijn: [00:18:10] And that's possible because you of course have not such a big team. And how would it work if you were cooperating with the developers abroad?

Ivan: [00:18:20] Well yeah we use HipChat as well and we have a room for each group, for each project. We do talk a lot in there. We also. On our daily stand ups. We usually have a big screen so we have been dealing to them as well. But I think, maybe it's because we are such a small team I think that's that's definitely part of it. And then so we don't have that big releases. We just we're good at knowing what's going on in the team.

Marco: [00:19:02] In Jira do you mark, do you assign to yourself a certain task or yeah. To say that you're taking on a certain task then you assign your name there?

Ivan: [00:19:15] Yes we use that for who's going to work on it who's going to maybe create the subtask or something like that. We definitely use the assign for who's working on it. And...

Stijn: [00:19:36] Do you know both within a sprint and throughout the project where you are like if I know you can't really talk about percentage, but how far you are within the project.

Ivan: [00:19:47] I don't think we are really that good at. I mean our sprint changes a lot. We put stuff in we, at the moment we're in a place where I'm mainly the only available which is working on the web shop. So and we kind of like in between sprints because we are waiting for clients who accept some projects, but we have used the burn down charts and stuff like that. I don't think it's it's not. It's not something we do that much on the projects I'm working on right now at least, because it's mostly me and a front ender and maybe another developer.

Stijn: [00:20:36] So would it be of any interest of you to either use a burndown chart and maybe velocity. Or would it be of interest to you of knowing it?

Ivan: [00:20:46] I think it would be nice to have to to see our performance. I definitely think that I'm.

Stijn: [00:20:58] OK. If you're looking for information on the project's status what is the first thing that you would look for?

Ivan: [00:21:13] I think I would look at the. I think I'd just look through the user stories and see how far they were. We have a workflow where we have I think in progress and then

development testing it on our development environment and if that's okay you make a pull request for getting it peer reviewed and once it's peer reviewed it's a to a test environment where it's QA'd and then if it passes QA it goes to another state which means it's ready to be merged for production. Or merged for the next release. So if I look at it I think if if it's it's either in it's either in progress or incubated, because the other steps all kind of like that fast. In peer review someone can easily grab it and move it further. I'm looking at how many issues are in progress for example.

Stijn: [00:22:27] To what extent do you actually rely on data to inform your decisions so data can be every information that you can get out of a tool.

Ivan: [00:22:39] On which decisions?

Marco: [00:22:46] Well I could say here for instance one in our case as developers is for instance to estimate tasks or the data is basically okay. Previous experience or or actually the tickets that you have already estimated how does actually the data that you produce in your everyday world actually helped you out to make decisions in general?

Ivan: [00:23:26] Well I think we sometimes we do have very similar tasks so we can draw from earlier, for how long time they took to develop. So we don't have to rethink the entire thing to just give a fast estimate. We have for example we have we have some landing pages for the site with different smaller blocks that are build up from, and these blocks sometimes they want a new block. And we just say this is like five hours front end and five hours back end, because we know we can deal with that. We don't spend time on estimating that. It's it's it's a rough estimate and it usually works. So that's like it that's that's not like at the rate I've put from the data we produce and that's just something we remember. But yeah I don't know how we are using that, I think other people are probably using it. I'm not looking that much at the data we produce.

Stijn: [00:24:29] Final question would be if you come back from holidays 2-3 weeks you have been out there, you come back. What's the first thing besides greeting your colleagues. What's the first thing you would look into in order to understand where you are within.

Ivan: [00:24:51] I would probably look at the current sprint in Jira or something like that. I would I would probably already know what to do or it would be planned or I would be allocated to some other project which , maybe I couldn't just jump into a current sprint or something like that. But if I think I would look in Jira and the current sprint to see what's going on. What are we working on . Any things blocked or stuff like that.

Stijn: [00:25:36] Besides that I have no further questions.

Marco: [00:25:39] No I want to ask actually that for how long have you used kind of the methodology that you use right now have. The set of tools if you know the Jira to manage the project and the process itself.

Ivan: [00:25:39] Yes it's a good question. I know I've been employed 10 months here so but it's definitely longer than that. Are you asking about my personal experience?

Marco: [00:25:40] No no. The process and the methodology that you use here right now. .

Ivan: [00:25:41] I don't know. But I think it's quite a while. Yeah but I didn't and I don't know what the company is 19 years old. And so it's probably not from the start. I know, we have a guy who's our Jira expert, he hasn't been employed that long. 2 or 3 years. So... but I can't answer you that.

Stijn: [00:26:31] Yeah sure. OK. Is there anything based on the talk we had so far that we might have forgot or that popped into your mind that we didn't touch upon. I don't have to. No I don't. I didn't really know this what you were asking, so. Okay. Yeah. Thank you very much for your time.

Interview Sandra (Scrum Master)

28-03-2018 14:34

Stijn: [00:00:00] Thanks. First some general question could you maybe shortly describe what the with the company is doing just in a couple of words?

Sandra: [00:00:07] What Adapt is doing in general? Adapt is a company with a lot of different competences within the digital field. To sum it up we develop a lot of different websites, apps. And a lot of UX and design work within the digital... I don't know what to call the digital industry

Stijn: [00:00:37] Okay. And in terms of organization size teams and team composition could you could you explain it in a couple of words.

Sandra: [00:00:46] Had we the entire Adapt group is around 120 maybe 130 people in all. And here in Denmark I think 70 people maybe around that.

Sandra: [00:01:02] And I myself in one of the client teams. Everyone in Adapt is divided into a client team, means that we have our own client portfolio and we actually manage almost everything for those clients within the client team and the client team sizes vary from team to team. I believe the smallest one is maybe five or six. And the larger ones are about 15 people. Our own is 14 people at the moment.

Stijn: [00:01:38] And in terms of composition what was normally included which roles are normally included within a team.

Sandra: [00:01:45] There's usually always a project manager and they even though we do a scrum practices. And that's also, almost you know. No I think an all team a couple of UX designers there is back-end developers, front-end developers sometimes if we're lucky that's also a full-stack developers. And I think is it.

Sandra: [00:02:14] We also, we just added a digital designer to our team.

Stijn: [00:02:19] And you mentioned that you use Scrum. Could you shortly explain what elements of Scrum you're using are using all of it or are there some specific elements such as stand ups that you're using

Sandra: [00:02:34] Now. It's handpicking also from project to project. What are we able to fit into the project, but we almost on all projects use for example stand ups. Then we use retrospective

a I'm not sure in terms of scrum where sprint planning goes, but we also have the sprint planning. And we have back-log groomings [?], story times.

Sandra: [00:03:07] Of course we use the product back log, the sprint backlog philosophy as well. And I think we're using a lot of it.

Stijn: [00:03:19] It sounds like you're using a lot indeed.

Sandra: [00:03:23] I think it's more the roles that we are trying to be better at.

Stijn: [00:03:29] How does it work right now with having a project manager for example which is mostly not included within Scrum. I would say what is the role that he or she is taking.

Sandra: [00:03:43] That varies a lot. I am a project manager and my role has changed a lot since I started at Adapt. I started out as being a project manager, product owner and scrum master.

Sandra: [00:03:58] And now I'm most of the time I'm only project managing. And then I have the accounting responsibility for the team which means that I don't really have that scrum role. It's just the team is my responsibility. But on one of their current projects I also have the role of scrum master.

Stijn: [00:04:27] Yeah. And throughout the process. How do you communicate with the customers.

Sandra: [00:04:37] On a daily basis

Stijn: [00:04:39] How does it how does that work? Is it physical meetings. Is it. Is it through phone. Is it through e-mail. Is it.

Sandra: [00:04:46] I think we have like two or three physical meetings this week. We are trying to get the client as much involved as possible. And then they we talk with the client. I talk with the client almost on a daily basis over the phone or Skype or whatever is available.

Stijn: [00:05:08] And do you also include them maybe within a tool that you're using.

Stijn: [00:05:15] Many clients also use Jira. The specifically clients I'm working. Right now they don't get they don't have a lot of people within their organization which means that we have to be a bit more flexible with how we communicate with them. But usually we want the client to communicate in the same system that we use. It's an internal process.

Stijn: [00:05:42] And how does it work if they do not. If they're not included within that system is it that you talked more with them and take their role so to say or.

Sandra: [00:05:52] Yeah we basically take a lot of responsibility and that would usually be on the clients, away from them and they use us more as their own employees. So to say and especially they... Me the product owner. Do a lot of the ways that other clients would potentially do that by themselves. Yup I've makes a lot more administration hours from our side.

Stijn: [00:06:22] And what types of tools are you actually using throughout the process and that can be any type of tool. So it can be as scrum-board it can be a communication tool, it can be Jira for example...

Sandra: [00:06:33] We almost only use Jira, that and we at the moment we use HipChat, but I believe we're moving into Slack..

Stijn: [00:06:43] You know why?

Sandra: [00:06:45] We both use Kanban and Scrumboards. But mostly scrum.

Stijn: [00:06:50] Do you know why you're moving to Slack? Over HipChat

Sandra: [00:06:56] To be honest I haven't followed that discussion. Its something about the developers. They want this Slack. And we used to only have HipChat and basically I don't care I'm just having my communication when they asked me to

Stijn: [00:07:13] Okay. And what do you think in your overall process of using Jira, using Slack / HipChat. Does it facilitate whatever you need in your role.

Stijn: [00:07:28] It can. But they will have some... It needs the specialized skills to really get the full out of Jira. Okay. Jira can be quite complex if you have no idea what you're doing. It has definitely helped that we have a dedicated guide to look after that.

Stijn: [00:07:49] So he trains or what.

Sandra: [00:07:51] He trains and they constantly look into if we can do something in a smart way. So that's basically his role. It's also his role to make sure that we follow the scrum methodology that we have been put out to do.

Stijn: [00:08:14] Is there if you could point out one thing maybe that you would like to see improved into to current set of tools that you currently use. What would that be.

Sandra: [00:08:25] That would be the project management tool. Because I know there's a lot of stuff to do with the data that you have in Jira, but a lot of manual working hours to actually sum it up nicely way. Some Excel kind of form. We are doing some experiments on the project that I'm working on right now which seems to be very promising. Yeah well that's definitely something that I have been focusing on.

Stijn: [00:09:01] I don't know what could you give an example of the things that you would like to do within a project management tool but you're currently doing in Excel or like yeah.

Sandra: [00:09:13] Show you quickly?

Stijn: [00:09:14] Sure.

Sandra: [00:09:16] I'll have to present something. Can you see this. This is .. Now it's a slide that summed up from this sheet is it's connected to with Jira with scripts. Basically and then there's a lot of [???] on top of it and it's basically connected to the project I'm working on it from

the Jira key. And then I use this for that they stand up to show the team how far are we? Behind on version 1? What does the velocity?

Stijn: [00:10:04] How many is in progress and stuff like that. Oh good. Good. Thanks for showing.

Sandra: [00:10:12] Yeah that is something that I've been missing for a long time in Jira and it does take some more technical skills to set up those kind of sheets right. In Jira you mean?

Sandra: [00:10:29] Getting Jira to talk to Google Sheets.

Stijn: [00:10:32] Okay. Yep and is that thing that you're missing because velocity... I've never used Jira, but I assume that you can also get velocity out of Jira right? Yeah. You can.

Sandra: [00:10:45] It is from Jira and that velocity I just showed you.

Stijn: [00:10:48] Yeah but I mean like within Jira itself it's not possible to get that overview that you are having just to just to be clear.

Sandra: [00:10:55] No

Sandra: [00:10:56] Okay well I'll have to calculate it manually. Okay.

Stijn: [00:11:03] Then you have a couple of questions on visualization and the tools that you're using. Imagine that you you as a project manager and you have front end developers, back-end developers and UXers. How do the set of tools that you're using facilitates that you produce certain output to together. How would that work. Maybe it's the easiest if you could explain if you get a new project like what the process would be like.

Sandra: [00:11:30] You want me to explain what the process is..

Stijn: [00:11:33] Like if you have a new project or a new Sprint. And like especially what rolls is of the tools that you're using in.

Sandra: [00:11:49] As a scrum master? That's a tough one since I usually start out being project manager.

Marco: [00:11:55] And well but the perspective of a project manager also counts.

Sandra: [00:12:05] Yeah I was thinking that's the toughest work in the beginning is the also, is always to get the project broken down adding story points. Yeah. You do different tasks so you can actually do some kind of forecast. Yeah. Well there's always that as a typical task, but its a time consuming task which involves the whole team. So it usually start out with that. But that would also mean that the project would have been somehow described before I get it. Does that me sense?

Sandra: [00:12:40] Is that also your part. Like describing the project. Or is that the rule of the product. For example it's more the product owner again.

Stijn: [00:12:49] And once you have cut down the project into different tasks, how would it work from there

[00:12:59] Then we would have our product backlog into Jira. And then we would do is planning poker-sessions. Do a lot.. Depending on the project size of course. But do a lot of planning poker sessions, until we have story points on everything. That is if the forecast is very important because if it isn't then we would rather stop out doing some of the tasks if we continue doing the planning poker. But usually since we work on the consulting side and we have clients they... they always want to know what is the forecast. So we would yeah we would mainly start out with putting story points on everything and calculating velocity.

Stijn: [00:13:52] Do you know within within the project who's working on what

Sandra: [00:13:59] On what tasks? No. No usually not. Usually we decide that sprint plannings and we would always have some kind of idea of who could do which parts of the software that we are developing that it's not a something that's I or anyone else decides on beforehand. It's something that the team decides together,

Sandra: [00:14:27] But would you be like... What I can imagine is that if you have multiple project or multiple sprints that maybe some person is not that busy and you need some additional manpower on another project. Would you happen to know who's working on what and what load they currently have. Is that something..

Sandra: [00:14:49] Yes that's mainly PM role that uses that, we have in Jira, we have something called Tempo. It's a planning tool where we put all resources. It's called Temple Yeah.

Sandra: [00:15:03] And from there we can see who has which workload and what are they working on. Within the Adapt group, we know which competences people have. We have some... Google Sheets documents where we list all the day developer competences. For example, we know which level they are on, on the different technologies that

Sandra: [00:15:32] So from that I can see who to, maybe pick out and then I'll go and talk to the other teams. We have a weekly meeting where we talk about resources in general.

Stijn: [00:15:44] I just have a question in between. I hear you talk like actually quite a lot of tools so really does it bother you or are you okay with the fact that you have so many tools that you potentially have to switch also quite often between all of those tools.

Sandra: [00:16:04] When it comes to tools we only use Jira and different add ons into Jira. And to different issues that a Google Drive. It's not that many tools it's just different parts of the tools. So no, that's an issue.

Stijn: [00:16:23] OK. And in terms of if you would like to know where you are in a project like in terms of progress how does this work. Maybe you mentioned a little bit with the sheet you already that change

Sandra: [00:16:40] For a quick overview I would look at how many hours that has been spent according to the forecast that we have made. How many story pointes have been solved there. Well I can see it from a percentage: where are we?From a more feeling good in my stomach or something. I would go and look at the product.

Sandra: [00:17:07] Both as scrum master and project manager I would know what we're trying to build and then I see for myself if we are on track. No.

Sandra: [00:17:26] And when you're looking for information on the project the project status was the first thing that you would be looking for. What type of information.

Sandra: [00:17:36] Of course I'm very focused on keeping up with estimates. We use that information for the daily. It's project in general progressing as planned because the focus will always be deadline and budget. So I would mostly be looking at numbers. Okay.

Stijn: [00:18:09] The next question more or less answered, but I still to ask it in what terms do you rely on data to to make those decisions? Maybe it's more interesting that data versus gut feeling in your case.

Sandra: [00:18:32] The first approach would always be data and then it would be the gut feeling from there. Because I can panic over small things I see the data and then I go and see what the product actually looks like and then that's always a solution or finding a way. I think

Stijn: [00:18:54] Actually the last question would be. This is a question that we often use to sum it up but imagine that you go on a holiday for like two or three weeks and you come back. What is the first thing that you would look into besides it's like saying hi to colleagues.

Stijn: [00:19:10] What is the first thing that you would do to keep up or to be involved and get back on track again within the projects that are going on.

Sandra: [00:19:18] I would take a status with the product owner. Yeah. Probably also the person that has been my holiday cover. And get an overview from there. And then go to my usual tools and have a look for myself.

Stijn: [00:19:36] And within those tools what is the most like the most important one for you?

Sandra: [00:19:41] That would be the forecast sheet.

Stijn: [00:19:47] Okay thank you very much based on the nice talk we had so far is there anything that you thought maybe we would cover but we didn't or we didn't touch upon yet or that you would like to share with us. You don't have to. You don't have to.

Marco: [00:20:10] Yeah I have a couple of questions just to get one of the things is the process that you have established here. You know for how long it exists? Us. I mean with the Jira and these team set up.

Marco: [00:20:26] Yeah I think it's three years now. So yes it's been a since I started to have started in the beginning of 2015.

Stijn: [00:20:39] Do you know what they did before.

Sandra: [00:20:42] They've basically had an excel sheet where they wrote down who will work on what which weeks and then all of them developers a set together and all the PM sat together and I guess that was it. I heard it was horrible very difficult to scale. So from hearing that I'm glad that we had Jira. Even though Jira, can be difficult sometimes to work with.

Marco: [00:21:16] Another one is actually that we're talking about we're looking to the present status and looking into the estimates. How will they help you. Like the estimates when you kind of OK. Do you compare that with the estimate o r what has been done or..

Sandra: [00:21:35] Yeah I mostly I come down the story points with what have been done and I know from an early point, where the velocity hopefully should be around. So I will use those two numbers. To get an overview.. To see how much new have been created. Because usually a lot of new stuff will be created during the project.

Sandra: [00:22:08] So actually forget your question was it a number.

Sandra: [00:22:14] No. Yeah. If you rely on the estimates. I just want to know how actually there value for for you when you are looking into the project status. Just to know if it's for a budgetting purpose or..

Sandra: [00:22:29] It is both for process and budget purpose. I'm probably the one in the team that is most focused on both budgets. So that will usually be my focus.

Stijn: [00:22:51] Nice. Okay thank you very much for your time. It was very valuable and it was very kind of you to give us some time

Appendix C: List of Tools Being and Their Organizations

Name	Function
AppDynamics	Software Configuration
Asana	Project Management
Code Climate	Automated Code Review
Confluence	Team Collaboration
Facebook	Social Media
Freshdesk	Ticketing System
GitHub	Version Control Hosting
GitLab	Repository Management
Google Drive	Data Storage
Google Hangouts	Communication
HipChat	Communication
IntelliJ	Code Editor
InterCom	Customer Messaging
InVision	Prototyping Platform
Jenkins	Open Source Automation
Jira	Issue & Project Tracking
Octopus	Release Management
OneDrive	File Hosting
Pivotal Tracker	Project Management
ProGet	Package Management
RallyDev	Project Management
Rollbar	Error Tracking
ScrumWise	Project Management
Sketch	Graphic Editor
Slack	Communication
SonarQube	Code Quality
Trello	Project Management
Visual Studio Team Services	Project Management
Yammer	Social Media (for business)
EMacs	Text Editor

VIM	Text Editor

Appendix D: Systematic Comparison of the PM Tools

Asana

Stakeholder engagement - Only people within the company to be assigned, but guests can be invited.

Face to face communication -

Integration - Seems to have integration with many different tools

Goal oriented - Does provide a clean interface with simple to do list that can provide focus. Easy to see what you should be working on (assigned to you).

Project tracking -

Project progress -

Dashboard - Lack the ability to follow up on what is doing with the project right now, like statistics and stuff like that. You can have a burn up chart, but this is the only overview that the dashboard offers.

Project planning

Dependencies - Does provide connection between tasks but they should know the name of the task.

Different roles, different perspectives - As a Scrum Master, I could look into the dashboard, but lacks information. As a product owner I could create issues, but it's harder to backtrack the work done.

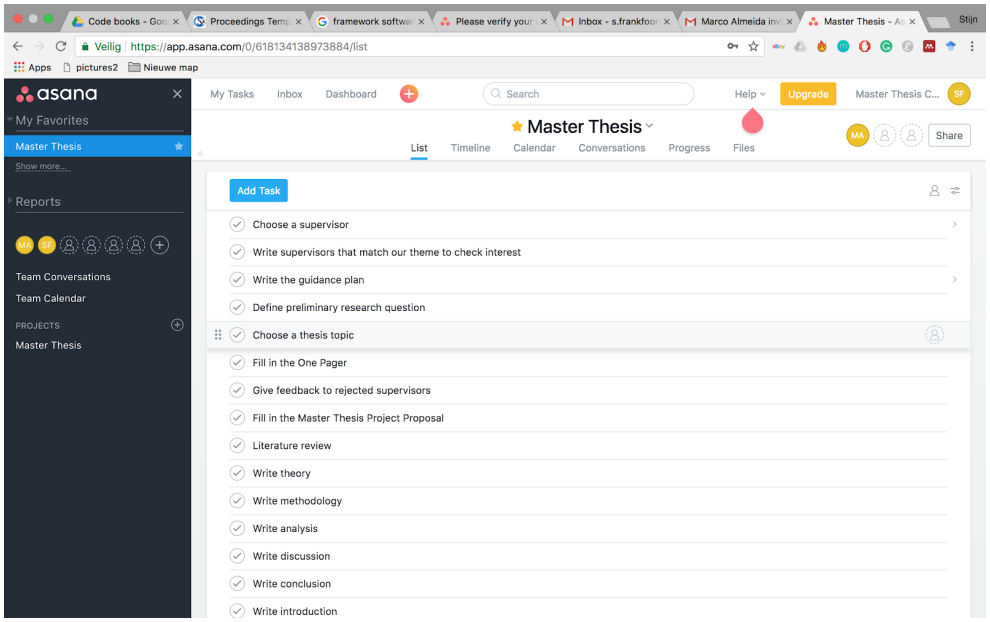
Group awareness - You can see what people are supposed to do, but not what they are working on right now. Unable on knowing what time was spent on.

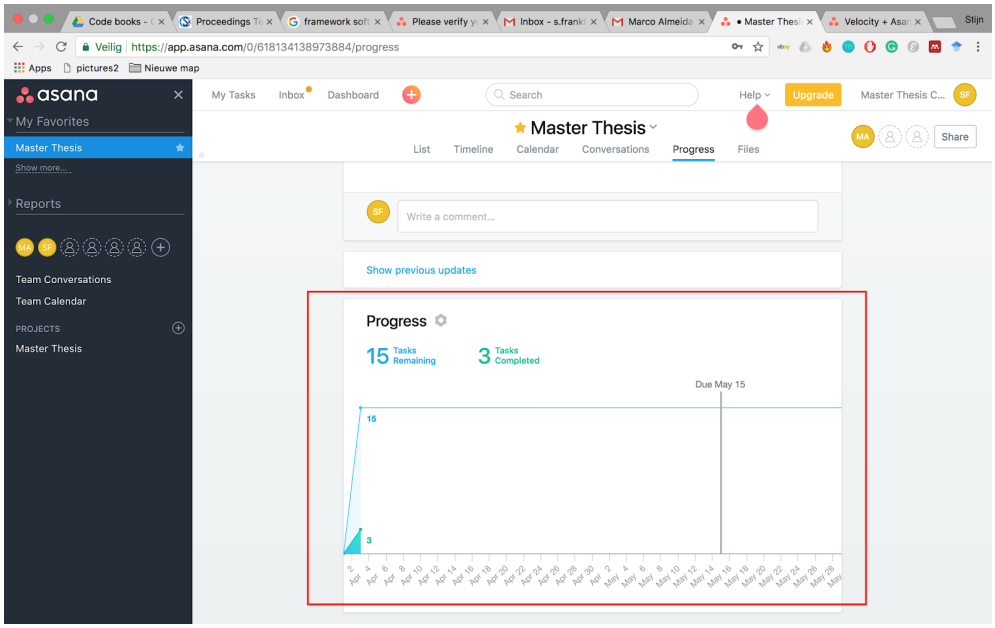
General notes on Asana

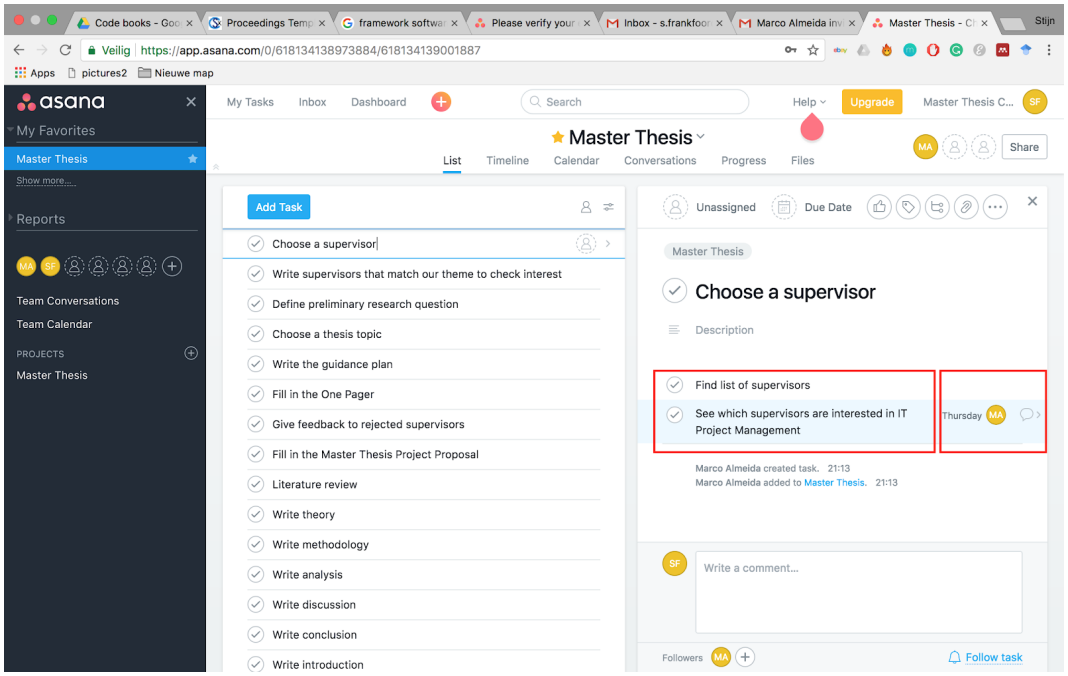
- Seems to be more of a generic tool and not specific for Agile teams. It's highly customizable, but it lacks default functionality and need some setup to run in some sort of Agile mode. It lacks integration with development tools.
- Clean visual. Real time updates. The project status serves as a good reminder on where the project is right now, and offers reasoning.
- Within a project you cannot change the views.
- It also lacks support for a more thorough review of the sprint.
- Nice notification system.

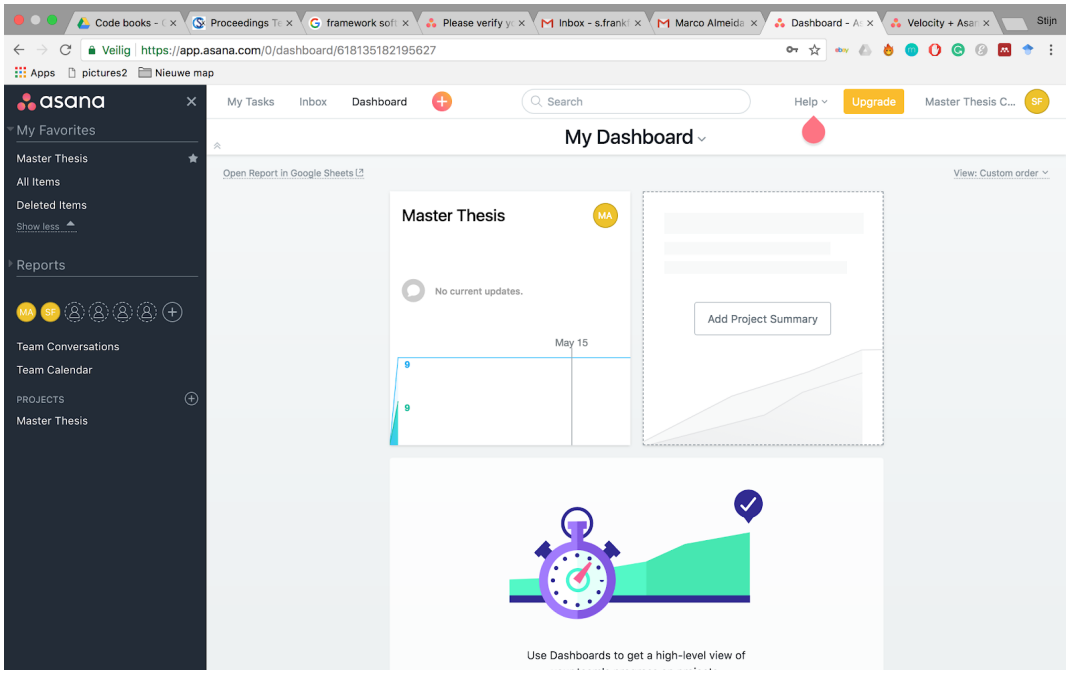
- Calendar gives a nice view of what has been done by whom. Would be good to warn on missing tasks.

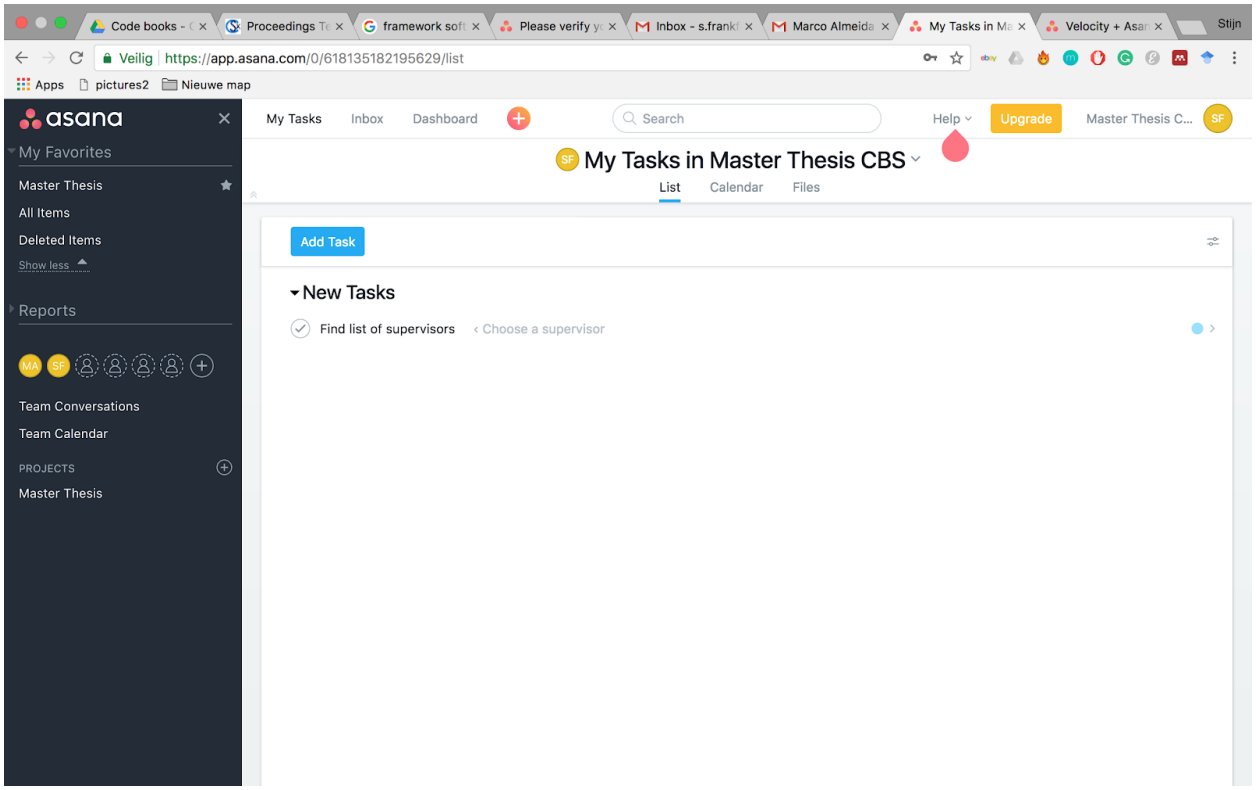
Screenshot Asana

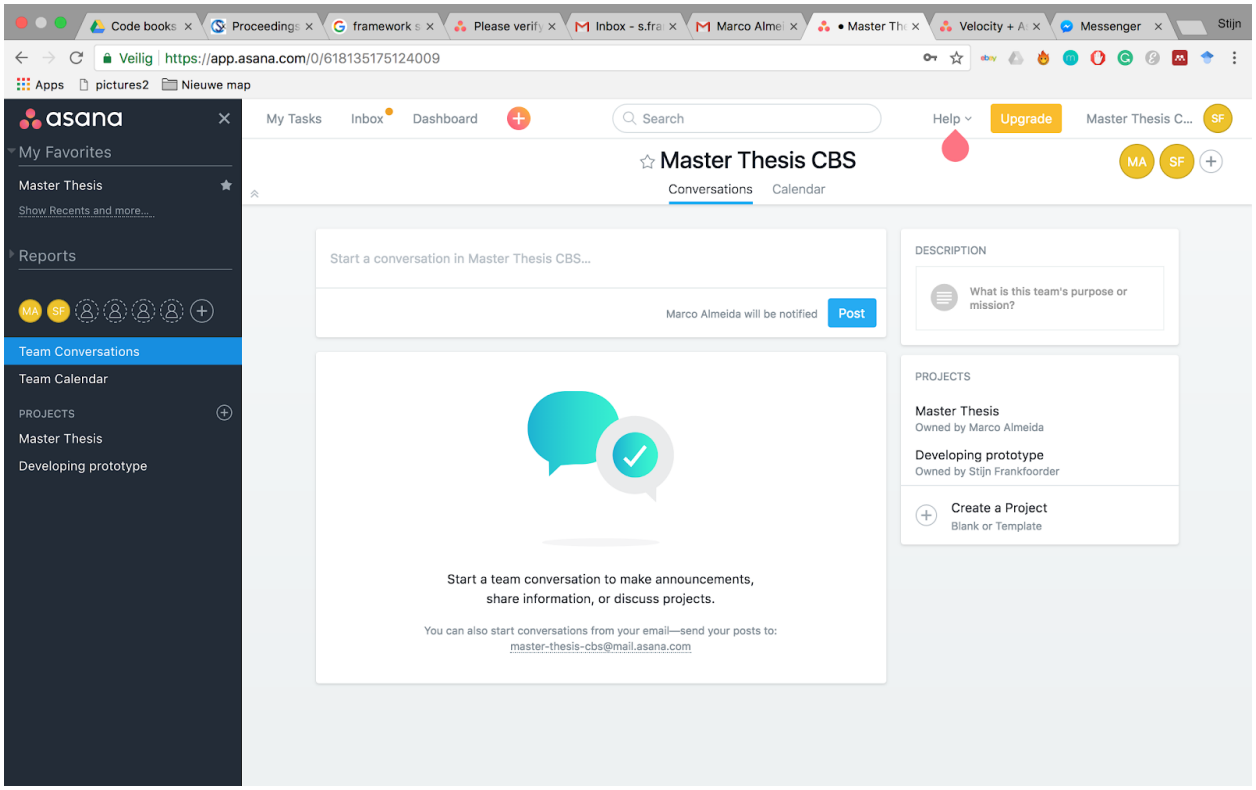





asana
My Favorites
Master Thesis
Show more...
Reports
Team Conversations
Team Calendar
PROJECTS
Master Thesis
My Tasks
Inbox
Dashboard
Search
Help
Upgrade
Master Thesis C...
Master Thesis
List
Timeline
Calendar
Conversations
Progress
Files
Share
Add Task
Choose a supervisor
Write supervisors that match our theme to check interest
Define preliminary research question
Choose a thesis topic
Write the guidance plan
Fill in the One Pager
Give feedback to rejected supervisors
Fill in the Master Thesis Project Proposal
Literature review
Write theory
Write methodology
Write analysis
Write discussion
Write conclusion
Write introduction
Unassigned
Due Date
Master Thesis
Choose a supervisor
Description
Find list of supervisors
See which supervisors are interested in IT Project Management
Thursday
Marco Almeida created task. 21:13
Marco Almeida added to Master Thesis. 21:13
Write a comment...
Followers
Follow task


asana
My Favorites
Master Thesis
All Items
Deleted Items
Show less
Reports
Team Conversations
Team Calendar
PROJECTS
Master Thesis
My Tasks
Inbox
Dashboard
Search
Help
Upgrade
Master Thesis C...
My Dashboard
Open Report in Google Sheets
View: Custom order
Master Thesis
No current updates.
May 15
Add Project Summary
Use Dashboards to get a high-level view of


Code books
Proceedings
framework soft
Please verify
Inbox - s.frankf
Marco Almeida
My Tasks in Ma
Velocity + Asan
Stijn
Veilig https://app.asana.com/0/618135182195629/list
Apps
pictures2
Nieuwe map
asana
My Favorites
Master Thesis
All Items
Deleted Items
Show less
Reports
Team Conversations
Team Calendar
PROJECTS
Master Thesis
My Tasks
Inbox
Dashboard
Search
Help
Upgrade
Master Thesis C...
My Tasks in Master Thesis CBS
List
Calendar
Files
Add Task
New Tasks
Find list of supervisors
Choose a supervisor


Code books
Proceedings
framework s
Please verify
Inbox - s.fra
Marco Almei
• Master Th
Velocity + A
Messenger
Stijn
Veilig https://app.asana.com/0/618135175124009
Apps
pictures2
Nieuwe map
asana
My Favorites
Master Thesis
Show Recents and more...
Reports
Team Conversations
Team Calendar
PROJECTS
Master Thesis
Developing prototype
My Tasks
Inbox
Dashboard
Search
Help
Upgrade
Master Thesis C...
Master Thesis CBS
Conversations
Calendar
Start a conversation in Master Thesis CBS...
Marco Almeida will be notified
Post
DESCRIPTION
What is this team's purpose or mission?
PROJECTS
Master Thesis
Owned by Marco Almeida
Developing prototype
Owned by Stijn Frankfoorder
Create a Project
Blank or Template
Start a team conversation to make announcements, share information, or discuss projects.
You can also start conversations from your email—send your posts to:
master-thesis-cbs@mail.asana.com

Pivotal Tracker

Stakeholder engagement - Guests can be added and be set as viewers, but they can't add comments to platform.

Face to face communication -

Integration - They have a marketplace to handle different types of integration. GitHub is available as the only source control platform, right now.

Goal oriented - The flexibility of having different views at the same time allow to focus on the task at hand. The analytics doesn't have the same view, but it could provide cross referencing of data. They don't have place to put sprint goals in there, which would make it harder to follow within the tool.

Project tracking - You can click on other people names and see what they are doing, have done, and so on. It's possible to extract data from that, but it is harder to find deeper connections.

Project progress - We have the burnup chart which also adds a trend. Velocity is also available.

Dashboard - The dashboard is very flexible, similar to goal oriented. What is need right now can be the only things shown on the screen. It is possible to save workspace to save dashboards that would manage specific cases. Again, the statistics are missing in this screen.

Project planning - Pivotal Tracker is missing a good calendar, which would be useful to get an overview of the tasks that need to be fulfilled in the future.

Dependencies - They have the concept of blockers that could be used within the project and also other projects that are also in Pivotal.

Different roles, different perspectives - Workspaces can help different members to see different views where the dashboard could be interesting.

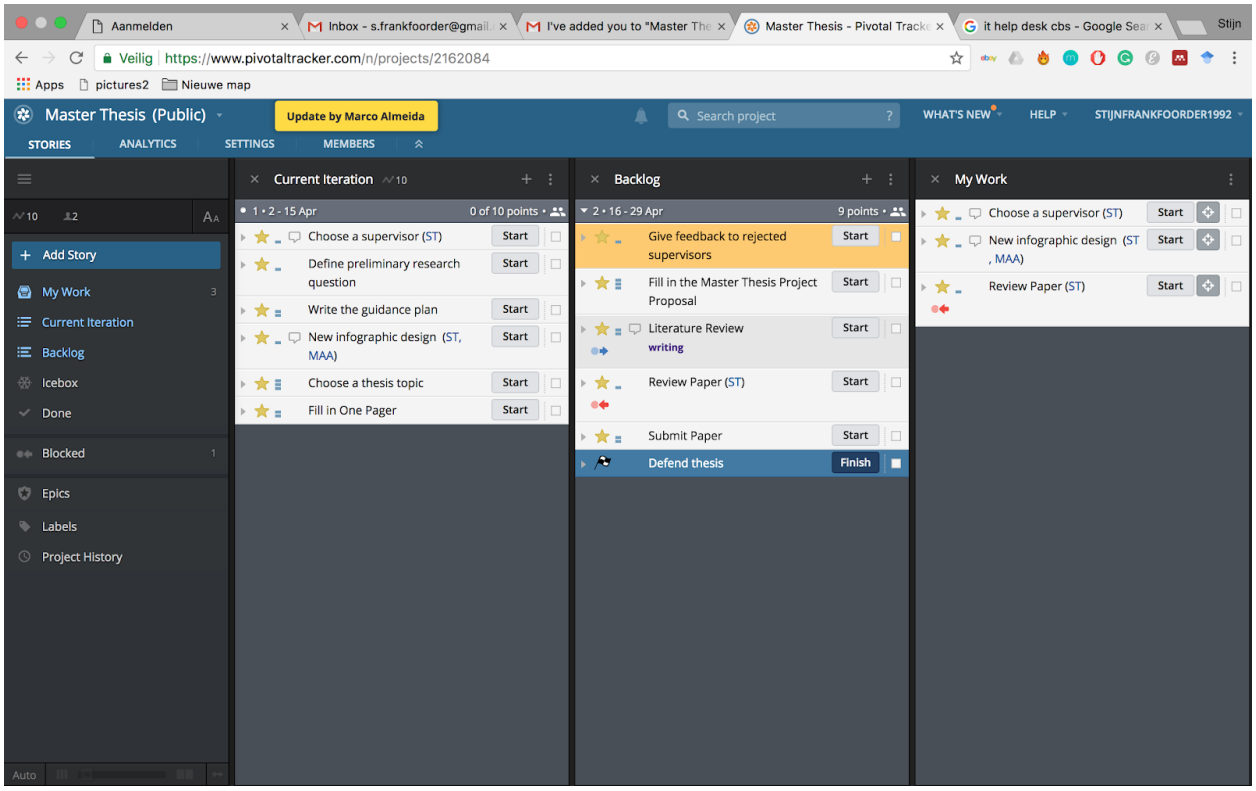
Group awareness - By clicking on the name of another user, you can see his/her personal backlog (in a specific iteration). However, it is not possible to see what you are currently working on.

General notes on Pivotal Tracker:

- The new tasks are added from the top. It's custom made for agile project management offering out of the box features such as (sprint iteration, team velocity and strength, categories of stories, epic support)
- Integration with GitHub (can be good or bad, because there are other source code management tools)
- Easy way to order the tasks
- Real time updates with a visual notification

- There is a pool of integration services
- Better reporting - It states where the time was being spent on (velocity), story cycle time not that interesting. Burnup chart also helpful. Stories accepted are very insightful.

Screenshot Pivotal Tracker



Jira

Stakeholder engagement - Different roles. Custom dashboards that can be made available.

Face to face communication -

Integration - Have strong integration with the other Atlassian tools, but also integrates with other tools. Also has a marketplace with integrations with more than 1660 different tools.

Goal oriented - The current sprint goal is very visible throughout the platform. However no tracking of it is well defined.

Project tracking - Tasks can be plotted against different reports like the burndown chart, to make sure that the project is going on the right direction. There are many different way of handling that too.

Project progress - The reports and the kanban provide a way to see the project progress. The releases and epics also provide a certain degree of understanding on the current state of the project within a specific roadmap.

Dashboard - Stijn likes it. There are many options to improve project's visibility. A user can also have many different dashboards.

Project planning - In general the flow of project planning is nice. To start your planning in Jira, there is a special button. → From here you can estimate points → press the ‘start button’ and you’re good to go. It is still possible to add / remove stories later on.

Dependencies - Dependencies are possible. However, there are not easily recognizable, and you do not get any type of warning if you move a story that is blocked. This comes down to a bad dependency

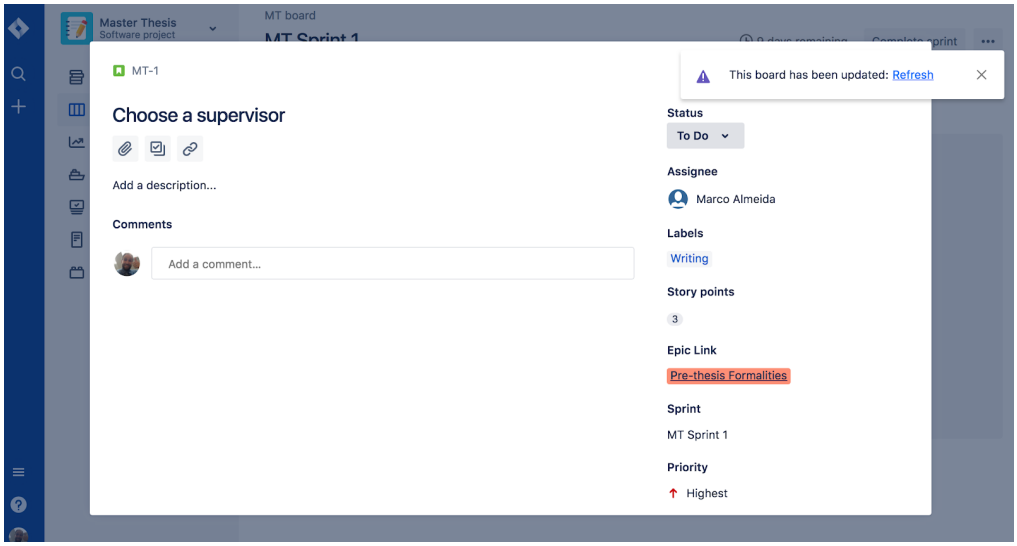
Different roles, different perspectives - It is possible to only look at your individual issues. As a developer you can have only at hand what you’d like to solve. It is also possible to make your customized dashboard, which means that you can tailor it to your personal needs.

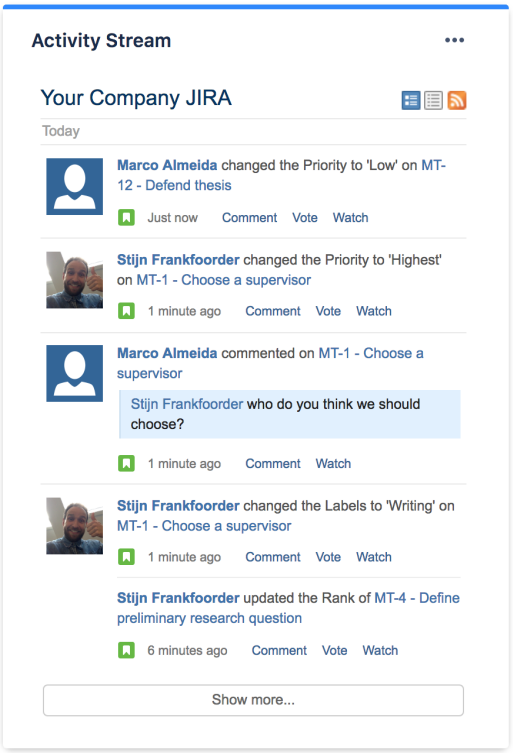
Group awareness - It is possible to see who is doing what within a sprint very easily.

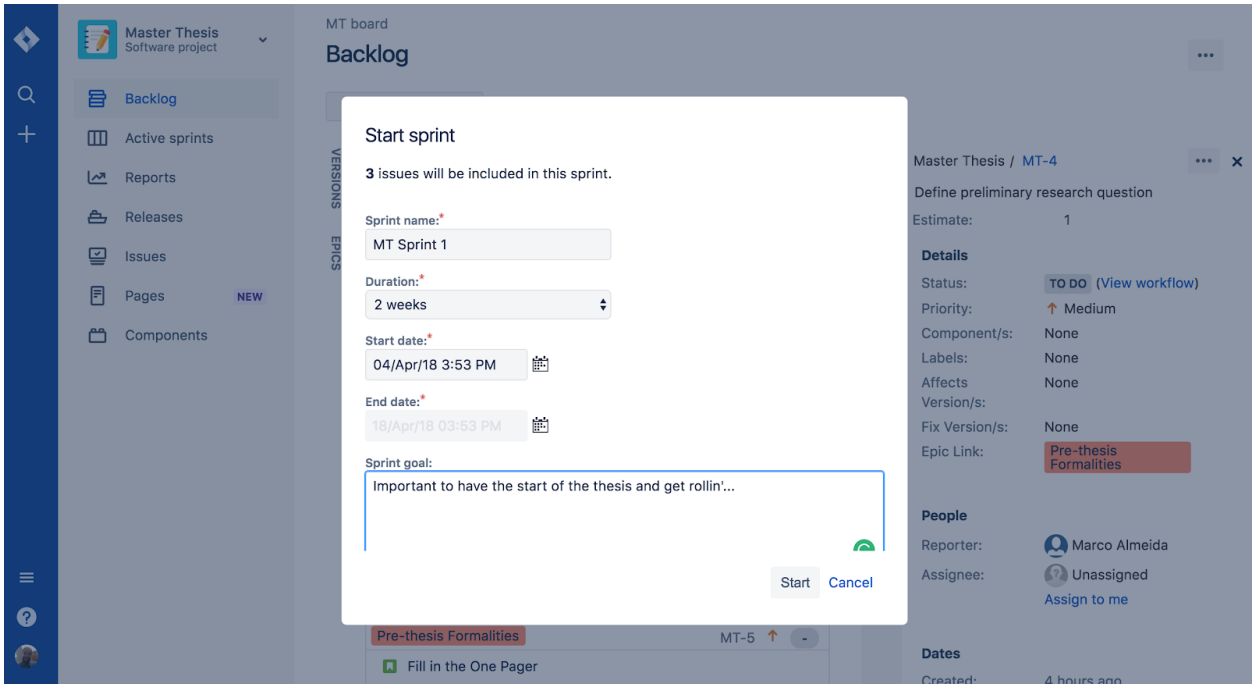
General notes on Jira:

- Does not really support real time activities so everyone can estimate on their own computer. It provides a sprint goal. Lack of in software notification, they are generally sent by email. You're able to see the goal all the time. There is a huge report.
- The planning mode is good. It is very clear that you have a very clear starting point of your sprint.
- It seems to be the most customizable tool so far.
- Notification cannot be seen on screen. -- You receive an email, but only when you are being mentioned. This makes it difficult to get clear when you are tagged into some task. There is a newsfeed (see screenshot), but here you can easily lose overview.
- First impression: possibilities are a bit overwhelming
- Difference between Pivotal and Jira:
    - Pivotal automates your sprint. Based on history you can estimate a future sprint. Which is not possible in Jira. → Takes out the essence of velocity in Jira.

Screenshot Jira:


Master Thesis
Software project
MT board
MT-1
Choose a supervisor
Add a description...
Comments
Add a comment...
This board has been updated: Refresh
Status
To Do
Assignee
Marco Almeida
Labels
Writing
Story points
3
Epic Link
Pre-thesis Formalities
Sprint
MT Sprint 1
Priority
Highest


Activity Stream
Your Company JIRA
Today
Marco Almeida changed the Priority to 'Low' on MT-12 - Defend thesis
Just now Comment Vote Watch
Stijn Frankfoorder changed the Priority to 'Highest' on MT-1 - Choose a supervisor
1 minute ago Comment Vote Watch
Marco Almeida commented on MT-1 - Choose a supervisor
Stijn Frankfoorder who do you think we should choose?
1 minute ago Comment Watch
Stijn Frankfoorder changed the Labels to 'Writing' on MT-1 - Choose a supervisor
1 minute ago Comment Vote Watch
Stijn Frankfoorder updated the Rank of MT-4 - Define preliminary research question
6 minutes ago Comment Vote Watch
Show more...



VSTS

Stakeholder engagement - You are able to see dashboards from a ‘viewer’ perspective. Overall impression is that it is definitely possible to have stakeholder management with this tool, but it is not the focus of VSTS

Face to face communication -

Integration - In comparison to the other tools, VSTS seems to have the strongest integration when speaking about development. Meaning that the code can be (within the application itself) be coupled with project management. On top of this they have a marketplace.

Goal oriented - A nice feature, that increases goal orientation within VSTS is the possibility to go on *full screen* mode, which allows you to only look at what you are currently doing (see screenshot).

Project tracking - A nice feature related to project tracking is that you’re able to see the workload other team members currently have on their plate.

Project progress - VSTS includes all tracking data such as velocity, burndown and many other possibilities. All of these provide a good indication of where you are in the project and how you’re heading towards your goal.

Dashboard - There are many different dashboards that can be created through the tool. These dashboards are shared and can have different widgets that can give a greater picture of what is going on in the project.

Project planning - It is possible to simulate the velocity to create some sort of expectation for what should be planned in the sprint. It is also possible to plan ahead in the future, adding more sprints in the list.

Dependencies - It is possible to set up the relationship between tasks, but it is hard to visualize them without an extension.

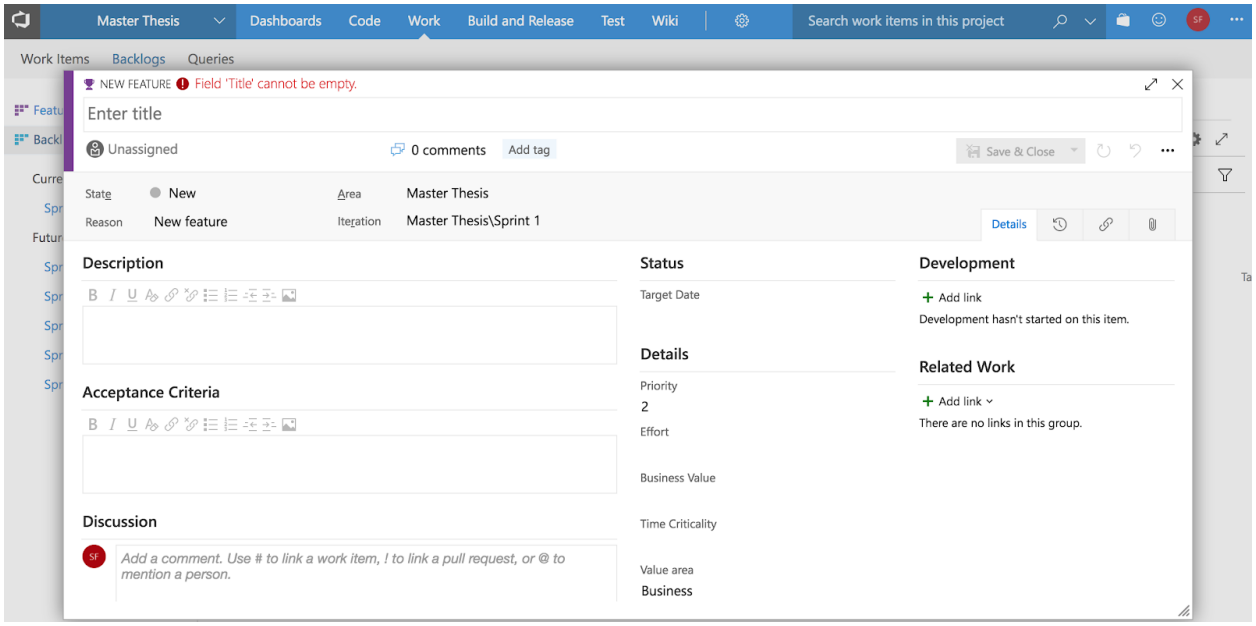
Different roles, different perspectives - There is a possibility to design dashboards that are different for each one of the roles in the team, so they can actually have a different perspective.

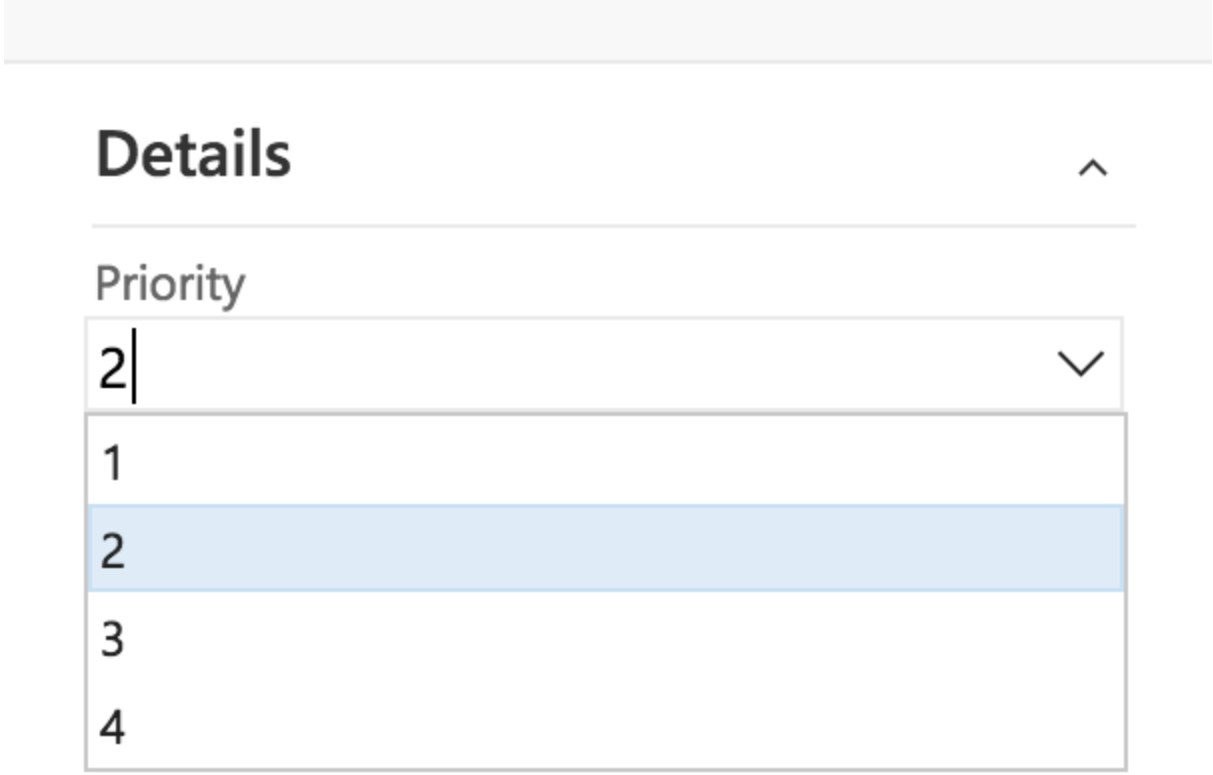
Group awareness - Able to see through the sprint view. Users can be assigned to the tasks they are in.

General notes on VSTS:

- The existing project dashboard is highly customizable, but still very clean.
- The workflow is different from the other tools -- difficult to start with at the beginning.
- Comparing this tool to all the others, it has by far the most features.

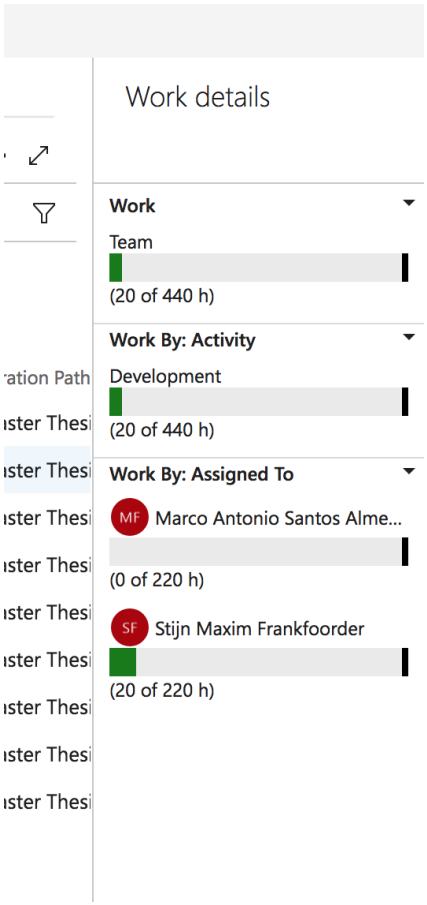
Screenshot VSTS:

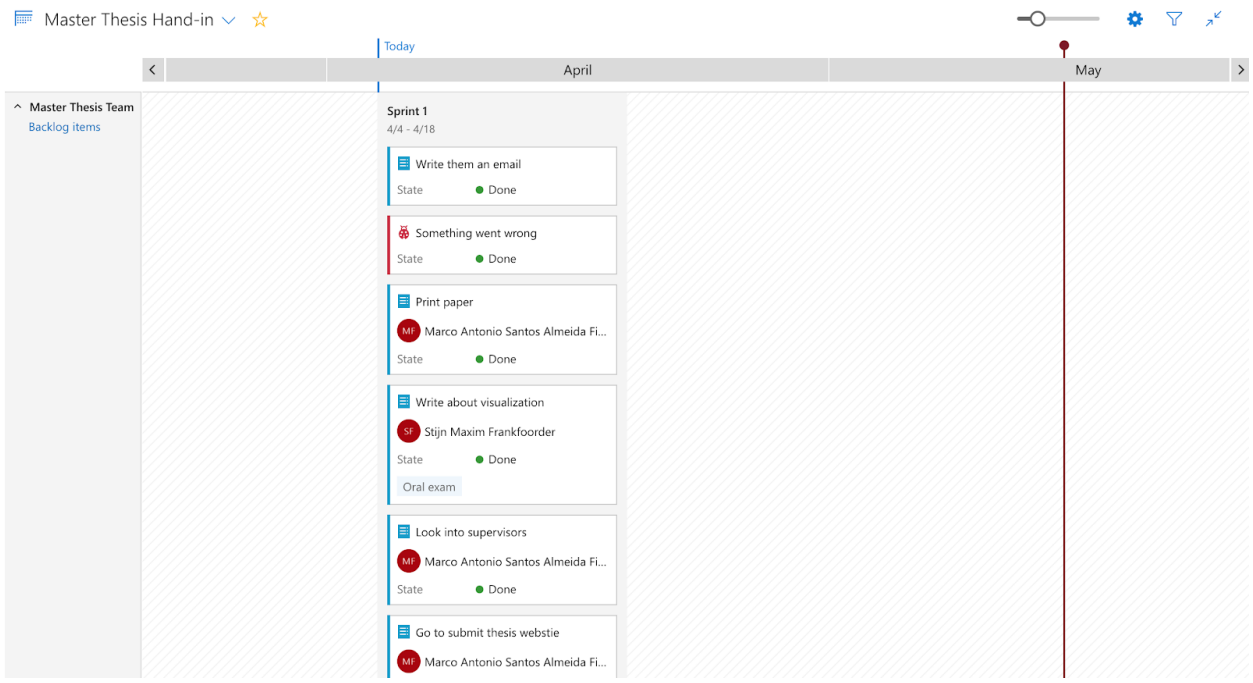


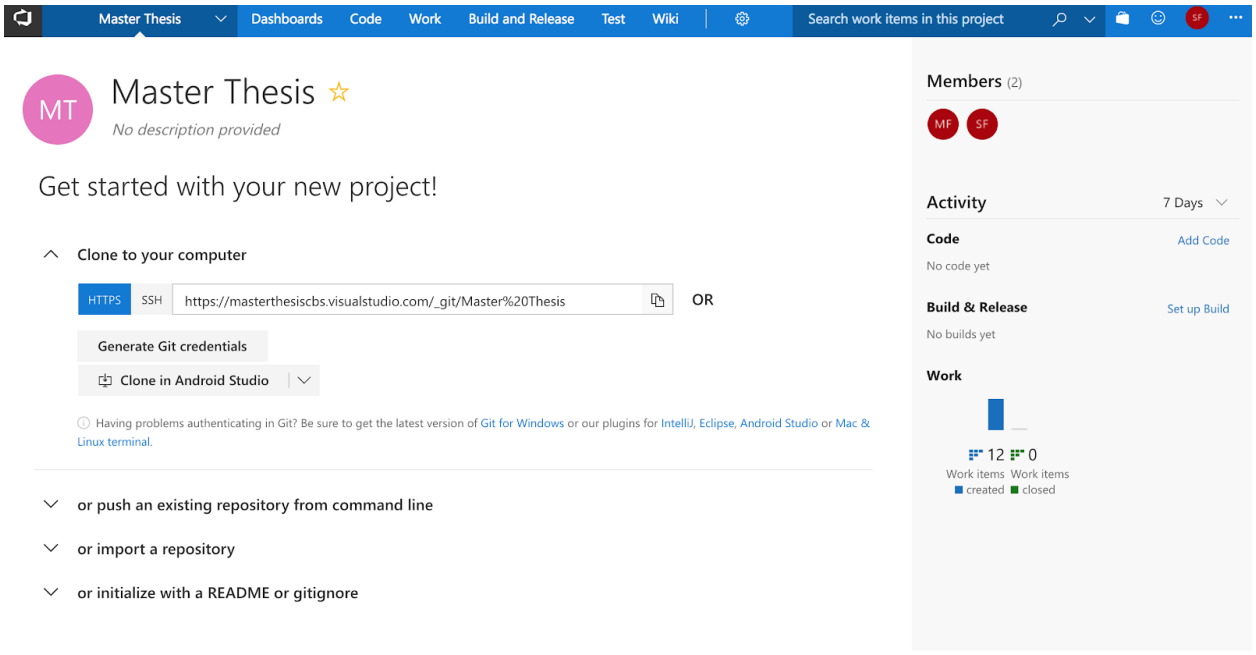


Description

Acceptance Criteria




Master Thesis Hand-in
Today
April
May
Master Thesis Team
Backlog items
Sprint 1
4/4 - 4/18
Write them an email
State Done
Something went wrong
State Done
Print paper
Marco Antonio Santos Almeida Fi...
State Done
Write about visualization
Stijn Maxim Frankfoorder
State Done
Oral exam
Look into supervisors
Marco Antonio Santos Almeida Fi...
State Done
Go to submit thesis webstie
Marco Antonio Santos Almeida Fi...


Master Thesis
Dashboards
Code
Work
Build and Release
Test
Wiki
Search work items in this project
MT
Master Thesis
No description provided
Get started with your new project!
Clone to your computer
HTTPS
SSH
https://masterthesiscbs.visualstudio.com/_git/Master%20Thesis
OR
Generate Git credentials
Clone in Android Studio
Having problems authenticating in Git? Be sure to get the latest version of Git for Windows or our plugins for IntelliJ, Eclipse, Android Studio or Mac & Linux terminal.
or push an existing repository from command line
or import a repository
or initialize with a README or gitignore
Members (2)
MF
SF
Activity
7 Days
Code
Add Code
No code yet
Build & Release
Set up Build
No builds yet
Work
12
0
Work items created
Work items closed

Appendix E: Link to Prototype

The latest versions of the prototype can be accessed in the link below:
https://projects.invisionapp.com/share/DVGO92I942J#/screens/290719868_Daily_Scrum

Appendix F: Loop[11] Report

The complete Loop[11] report (46 pages) can be accessed in the link below:
https://drive.google.com/open?id=1pKRH-iKlLfJUFnEjw27sf02IRXdur0Ck

Appendix G: Notes from the Group Interview

Adapt

Dashboard	Element	Category	Description	Comment to self
All	Structure	Positive feedback	Intuitive - the combination of the elements is very nice and makes in good, the structure	
All	Structure	Positive feedback	Test was limited, but focussed	
All	Customizatio n	Future improvements	Potential in dashboard idea --> we would like to have it configurable	
All	Sprint goal	Positive feedback	Nice that the sprint goal was very clear	
Release Review	Roadmap	Positive feedback	Like: the roadmap to communicate with the stakholders	
Release Review	Roadmap	Positive feedback	Roadmap is nice to have	
Daily Scrum	Sprint goal	Positive feedback	Sprint goal is nice to have as defined	
Sprint Planning	Backlogs	Future improvements	The capacity to plan -- it should provide more features	
Sprint Planning	Dashboard	Positive feedback	Sprint planning is not good in Jira, this thinking is very nice	
Sprint Planning	Capacity	Problem	They didn't understand the capacity --	
Sprint Planning	Team Strength	Problem	They didn't understand the number of workdays in team strength	Add the type of units --> need of additional information to onboard them
All	Structure	Problem	Major problem: the board should be able to look at all projects (one project is not realistic)	New research opportunity

Sprint Review	Deploy	Unclear	One of the participants couldn't understand why the number of deploys were bigger than one for the sprint review	
Release Review	Stakeholder engagement	Problem	They didn't understand how it worked	
All	Charts and Tiles	Future improvements	Make a legend	
Release Review	Stakeholder engagement	Problem	It is not clear how you would connect the task of stakeholder management	
All	Customization	Future improvements	Making the Daily static + add your own individual boards (which is possible right now)	
All	Customization	Future improvements	Add templates	
Sprint Review	Screen	Future improvements	Sprint review (it didn't match with how they do it)	
All	Customization	Future improvements	There is no board for me as an individual (after the meeting, than what??)	
All	Widgets	Future improvements	Time tracking needs to be included	
All	Task color	Future improvements	More yellow = longer task --> did you notice this? They kind of did..	
All	Integration	Problem	The icon show there is a link, but how are they linked??? Not really clear	
Sprint planning	Task size	Future improvements	Fill up a lot of room, however it is easy to understand!	
Sprint planning	Backlogs	Problem	Question: product backlog vs. sprint backlog	

Blue Billywig

Dashboard	Element	Category	Description	Comment to self
				Went to the release review. I would have expected something different, but I was expecting pictures or a list. Didn't find the tasks.
Daily Scrum	Burndown Chart	Positive feedback	Like the burndown. Provides a good visualization	Can't say what the problem of the issue.
Release Review	Structure	Future improvements	Would like to have a project overview. A higher level further.	
All	Integrations	Future improvements	A list of the different integrations would be fine	
Release Review	Roadmap	Future improvements	The indication of where you are today could have a better style to outstand from the others	
All	Sprint dates	Problem	I'm not sure where I am right now. A clear separation of the previous and incoming sprint	If I am part of the team I would know
Sprint Planning	Structure/Capacity	Positive feedback	I like the overview and the idea of capacity	
Sprint Planning	Tiles	Positive feedback	Like the tiles from visual studio and it's very complex which visualization you would choose	
Daily Scrum	Dependencies	Problem	Relationship between the tasks. I was wondering and I am still wonder. It doesn't visually show me the relation.between the tasks.	
Daily	Dependencies	Future	Maybe over thought the	

Scrum		improvements	dependencies of tasks	
Daily Scrum	Dependencies	Future improvements	Give a unique number to represent the locks	
Release Review	Structure	Positive feedback	Release review gives a nice high level for me	
Release Review	Sprint dashboards	Problem	Confused about the sprint planning and the sprint review	
All	Idea	Positive feedback	Overall it's a very nice idea	
All	Gap between smaller and bigger picture	Future improvements	There is a gap between the daily stuff and looking down the line and get them all together, and provide information to stakeholders. Produce something more transparent	
All	Risk management	Suggestion	Where could the conflicts come in to meet their attention	
All	Idea	Positive feedback	That idea of facilitating the process is very nice	
All	Process similarities	Positive feedback	Liked the visualizations and the possibility to look further, but it was really inviting, and recognizable from the process	
All	Stakeholders	Positive feedback	Bringing the stakeholders it's nice	
All	Daily use	Problem	How can you use this on a daily basis? But I would have to think of all the tasks to figure out	
Daily Scrum	Management features	Future improvements	People usually seem to forget where they are on their daily Scrum, so they would want to update stuff on the process	
All	Risk management	Future improvements	Some red flags would be nice, what blockers can we foresee (risk	

			assessment)	
All	Prototype	Positive feedback	This is the tool: What I am doing now, and what are we doing in the future	
All	Prototype	Future improvements	Keep it clean!	
Sprint Planning	Task size	Problem	Size of tasks it's a nice idea to fit still, but I think it's difficult because you're sacrificing the screen	
All	Task size	Future improvements	A folding of the tasks in planning would be nice	
All	Task colouring	Problem	Thought that the paper colouring was selected or something like that	

Karnov Group

Dashboard	Element	Category	Description	Comment to self
-		Problem	The long link is pretty annoying	
-		Problem	The test is not responsive	
Release Review	Stakeholder	Problem	What does the bubble of the different size mean?	
Release Review	Budget	Future improvement	Participants liked all of the charts, accept for the pie chart. No-one was a fan of it, most suggested to replace it.	
Release Review	Stakeholder	Problem	Some couldn't find the stakeholders in time	Look this up on the data that we have from the tool
All		Problem	What type of metadata is coming from icons	
Daily Scrum & Sprint Review	Burndown	Future improvement	Burndown looks negative because the better you perform, the lower it gets → turn it into burn-up;	
Daily Scrum	Scrum board	Problem	How does it work with the key, the colors and the unlocking	
All	Scrum board	Future improvement	Border around the blocks, remove the lines around the cards, try to keep as few lines as possible and group on proximity	
Daily Scrum	Widgets	Future improvement	Remove border, play with the grey intonations;	
		Future improvement	Sprint planning vs. product backlog (there is only one backlog) → re-name it, is the suggestion as it causes confusion;	

Release review	Roadmap	Future improvement	Click on sprint it would be helpful if a dropdown was shown with all the stories in the sprint	
Release review	Roadmap	Future improvement	Naming of the sprint instead of sprint 22	
Release review	Roadmap	Future improvement	Make a suggestion in the roadmap if one sprint is related (dependent) to another one;	
All	Widgets	Future improvement	Click in deeper on the big boxes, so additional information is shown.	
All		Problem / missing	Data / Performance of team over time / What are the long term effects on the team / Long term planning	
All		Positive feedback	Sprint goals, which are prominent	
		Positive feedback	Burndown, they liked it a lot	
		Positive feedback	Overall: really nice tool!!!	
		Positive feedback	Right now we have to do the math in the head.	Not sure what this relates to, it might be the Burndown
			Noise interface: the lines are the problem;	
Release review & Daily	Multiple	Future improvement	Daily scrum: Story points in each of the columns	
		Future improvement	Make it touch screen friendly;	A nice suggestion for future research
		Positive feedback	Scrum in the morning is important → face to face communication is EXTREMELY IMPORTANT;	
		Future	Downsize the pie chart	

		improvements		
Release review	Stakeholder map	Problem	Size of the bubble - related to the budget they thought	
Release review	Stakeholder map	Future improvements	Stakeholder shouldn't be there?? (for consultancy it would be useful)	
Release review	Stakeholder map	Future improvements	Who is the stakeholder for who? → this depends on the role that you take (request of someone within the company; steering group)	
Release review	Stakeholder map	Future improvements	We should include: who is the requester, who is the invested party in here?	
Release review	Stakeholder map	Future improvements	The way it is visualized → rethink how this should be done!!	
Release review	Stakeholder map	Future improvements	Have a list (who is the most active stakeholder)	
Release review	Stakeholder map	Future improvements	Put it in the user story / epics / …	
	Scrum board	Future improvements	Suggestion: scrum board -- have a bar instead of	
	Personal board	Future improvements	Suggestion: No need for a personal board over here	
All		Problem	Big boxes with numbers - very unclear what the context is in;	
		Future improvements (next iteration)	Provide more context when onboarding	We should give a better context, when is which dashboard, include this in the questions
		Future improvements	Turn down on the big numbers	

		Suggestion	You want the feeling, not the exact numbers!! (the numbers seems to be a by-product)	
		Future improvements	Suggestion: what is the total point number / label (design;)	
Sprint planning	Backlog	Problem	Progress bar can be confused with a scroll-bar;	
Daily Scrum		Positive feedback	Overall they very much liked the Daily Dashboard	
All			They liked the ceremony based, they didn’t have an alternative;	
All	Sprint goal	Positive feedback	Positive: the sprint goal!!! Again	
All	Ceremony	Positive feedback	Overall: the participants love the way we combine different views; → truly adds to the ceremonies;	
All	Small icons	Future improvements	Overall: the main take away is that the small information doesn't add a lot	
All	Small icons	Future improvements	Keep it focussed on the tasks → loose the small information;	

Santander

Dashboard	Element	Category	Description	Comment to self
All	Integration	Problem	Challenges on understanding how the integration works	I guess this was also related to the lack of knowledge in other tools
All	Structure	Positive feedback	Very cool the fact that is based on the ceremonies	
Sprint review	Structure	Positive feedback	Sprint review site. You can see what's done and everything	
Sprint review	Sprint goal	Positive feedback	The sprint goal is visible all the time; Then you need to go to a different dashboard on different tools	
All	Comments	Future improvements	Discuss straight from the tool	
All	Comments	Problem	Update the task/story straight from the tool	
Release review	Stakeholder engagement	Problem	Stakeholder thing?	
Release review	Stakeholder engagement	Future improvements	Maybe show in a different aspect; Some sort of budget management	
Release review	Budget	Problem	Budget could be on track; Improve the way with the data	
Release review	Budget	Future improvements	PO, budget could not be needed	
Release review	Budget	Future improvements	Map stories relations to epics	
Release review	Budget	Future improvements	Map Relation between stakeholders and budget	
Release review	Budget	Future improvements	Budget I wouldn't expect from a team overview	

Sprint planning	Size of tasks	Future improvements	Size of the sprint planning/The stories can take the space; Maybe won't use for the screen size. Maybe should exist on the daily scrum	
All	Structure	Future improvements	Depends on the role; PO only wants the results	
Release review	Structure	Positive feedback	Release review give the most overview on how the team is doing	
Release review	Dependencies	Future improvements	Dependencies between different sprints in the sprint	
Release review	Dependencies	Future improvements	Dependencies with external teams	
All	Management features	Future improvements	Dynamic visual; Create tasks;	
Release review	Product owner	Problem	The PO part is not there yet	