Reflection Report

DAT255 - Group TugLife

Åkesson, Jonas
jonasak@student.chalmers.se

Lundgren, Emil
emilundg@student.chalmers.se

Nordenhög, Kevin
nkevin@student.chalmers.se

Nilsson, Sebastian
sebnilss@student.chalmers.se

Mrkonjic, Luka
mrkonjic@student.chalmers.se

Rasku, Kevin
rasku@student.chalmers.se

CHALMERS

Preface

This report was written during the spring of 2017 as a part of the Software Engineering course at Chalmers University of Technology. We would like to thank our supervisor Håkan Burden, the employees at PortCDM and other students for helping us with the work developing the application.

Abstract

This report focuses on the planning and development of a web application aimed towards helping the navigation of tugboats in Gothenburg harbour. It disects each process and method used during the development cycle and critically analyses their advantages as well as their disadvantages. With Scrum being the development framework of choice, the group faces new challenges to overcome trying to adapt to the agile Scrum workflow. The group practiced new as well as old practices such as project management through Git, the use of Trello boards, frequent stand-up meetings and much more.

The end result was a fully functional web application fulfilling all of the user stories presented by the product owner. The final application's functions ranged from continuous displaying of tug boats and towage service requests to updating individual statuses. The group held a good tempo throughout the project and had working builds of the application to present each week, with functionality on both front- and backend. During the life cycle of the project the group has gained knowledge in a wide variety of software engineering practices including group dynamics, Scrum use and problem solving while tackling new areas such as API- and JavaScript programming.

Contents

1 Introduction

1.1 Background

Starting out a project comes with its difficulties, one being planning and structuring. While choosing a development method is a project in itself, there are many good ones oriented towards the software engineering market. A popular method is Scrum, which this project is based on. Scrum is an agile development framework based on frequent task division and short term increments of development called sprints. Besides the use of Scrum the group must also practice communication, meetings and group dynamics to maximize project efficiency.

This report is the result of an eight week long Scrum driven software engineering project focused on creating a web application targeted towards the harbour of Gothenburg, from idea to a final working prototype. It covers the various methods used by the group and the choice made along the way. The report also dives into the problems and solutions encountered during the cycle of the project.

1.2 Goals

This report aims to analyze and discuss each method of choice as well as declaring their pros and cons. It also acts as a reflection and breaks down how well the group executed each process in terms of efficiency and planning. The goal is that the group should gain a deeper understanding of the software engineering work flow and also deepen their communication skills both inside and outside the group.

2 Process

In this chapter the application of Scrum will be explained in more detail. This includes how the group worked together, used practices, time distribution, effort & task breakdown, sprint planning and retrospective.

2.1 Roles, group work and social contract

During the project different roles where held. The group divided the roles down to a scrum master, developers and product owners. The scrum master was focused on dealing with scrum of scrums one time every week but also making sure everything was on track and managing sprint reviews/sprint planning. Early in the project the group contacted the company that manages tugboats in Gothenburg but it was only until the last week that the group actually got contact. Therefore the group viewed personnel from RISE and people working with PortCDM as the product owners. They were also close to the stakeholders which made them suitable.

When working in a project like this it is of utmost importance to have a functioning group dynamics. The group that works well will have a strong efficiency, good delegation and be supportive to each other. If the group is absent of a functioning teamwork it can have mayor consequences for the groups behaviour and in the end the goal that was set out might be missed. A practice for helping the group get on the same page is a social contract. This was written and "signed" of all the members of the group. The contract was used as a spine in the groups group effort. It concluded what behaviour was expected from the members and what was not tolerated. A social contract can include meeting behaviour and consequences for being absent or late, how to listen to each other and where the main communication might be.

Pros & Cons
A benefit of not having the product owners in the group was that we could stay more focused developing. Not having Svitzer, the tugboat company in Gothenburg as the product owner could have resulted in confusion about which features actually were requested.

The initial thought was that we did not need a social contract and that everything mentioned was kind of obvious group behaviour. After actually sitting down and discussing the task ahead and what it should conclude it became obvious quite fast that people had different ideas of how and what good group behaviour is. The whole group then felt it was a big advantage to actually have something written down that everyone could fall back to if they were unsure.

In Practice
The roles could have been used to a greater potential than they were. Because of scrum and agile working being a completely new concept to most of the group the roles were, mostly at the start, not used to its full potential. The group made no clear guidelines for what exactly the role meant which might have led to some confusion.

In practice the social contract was written down and not so much looked at after. The group never felt the need to actually go back and change what had been written before which might have been a sign that everything was already on there but most likely it became forgotten. Most of the group members had the same background from Computer Science. Some had already worked with each other in different assignments but not all.

Future
If a project like this is going to be done again clear guidelines for the roles is a must. To have specific roles is of utmost importance and could easily be realized by at the start of every sprint planning naming sprint master, product owner and responsibility areas for each.

2.2 Used practices

In this section we discuss the the practices we used during the project such as stand-up meetings and chosen KPIs.

2.2.1 Stand-up meetings

Stand-up meetings is a short meeting, typically around 5-15 minutes, at the start of every days work where the group members answers three important questions:

- What did I do yesterday that helped the group reach the set-out goal
- What will I do today towards to help the group reach the set-out goal
- Do I face any challenges that prevents me from reaching the set-out goal

Pros & Cons
The benefits of using Stand-up meetings is that it is easy for everyone to see what they have to do for the project to move forward. Everyone knows their tasks and how to accomplish them. If there are any challenges facing it is mentioned and the rest of the group might be able to help.

In Practice
At the start of the project we did not apply Stand-up meetings. This was mostly because it was a unknown term to all of the group members. Not having these types of meetings resulted in a mayor difference in workload and focus. It was not until a guest lecture on Product Development by Simon Hofverberg at Spotify that the potential of using such meetings became known. The group applied it almost instantaneously after the lecture.

Future
Stand-up meetings have great potential and should have been used since the start of the project. For similar projects in the future, stand-up meetings will definitely be used.

2.2.2 KPI

During the project different Key Performance Indicators (KPIs) were used as a means to help the group keep track of performance. The group chose three different KPIs:

- Stories committed vs stories completed
- Group Velocity (Tasks committed vs tasks completed)
- Retrospective process improvement

The first KPI was keeping track of how many user stories were committed to the backlog versus how many of them were completed. This was chosen as a KPI in order to illustrate how the project is progressing. This was to give the group an overview of the project which could help with adjusting their process.

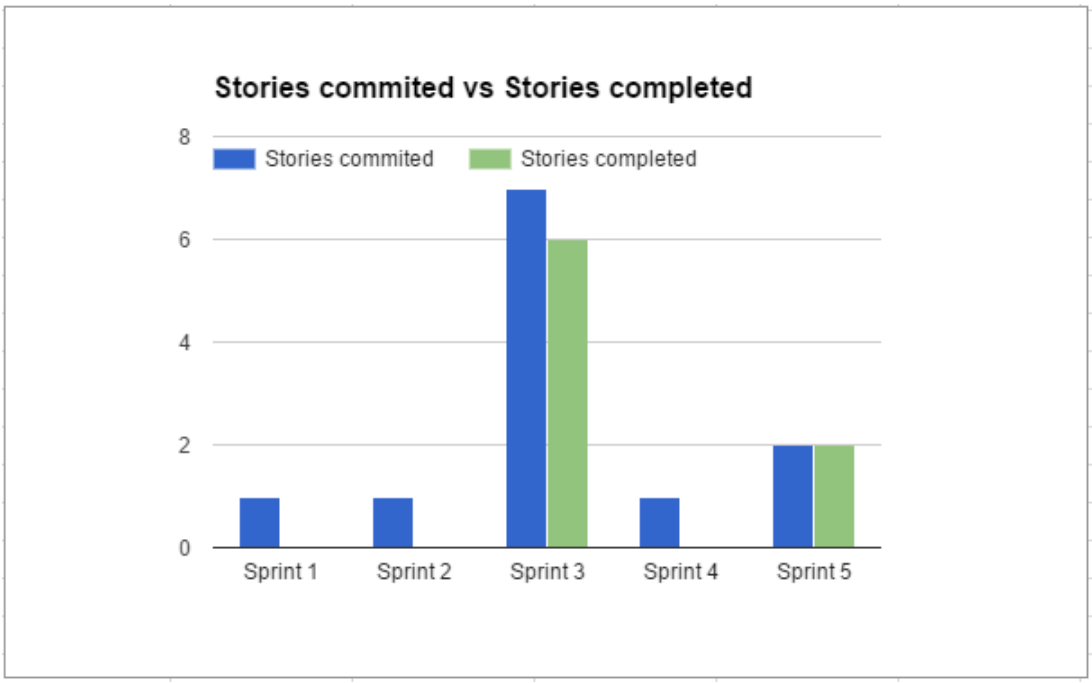

Figure 1: Stories committed vs stories completed during the five sprints.

The second KPI, group velocity, described the rate at which the group was completing tasks. The purpose was to show us how quickly we work and how much we can get done in a sprint. This information was then to be used during sprint planning to help us commit to a fitting amount of work.

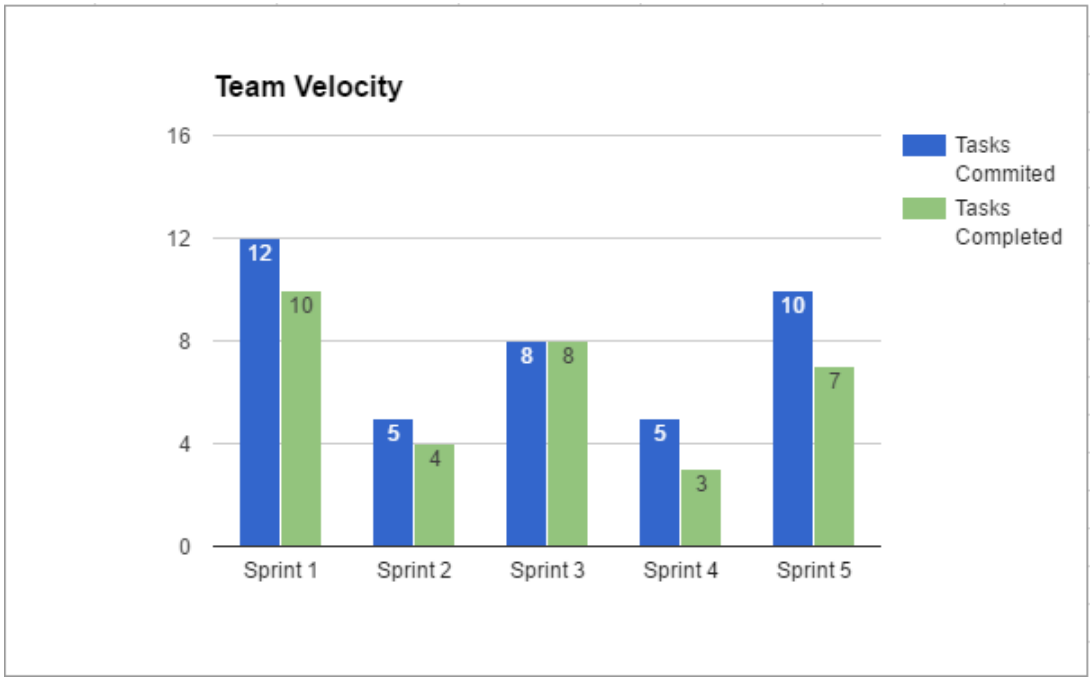

Figure 2: Tasks committed vs tasks completed during the sprints.

The final KPI was using the evaluations of the sprints during the sprint retrospectives at the end of each sprint in order to improve the process. It was to allow the group to see what went well and what didn't during the sprint and give us an opportunity to improve for the next sprint.

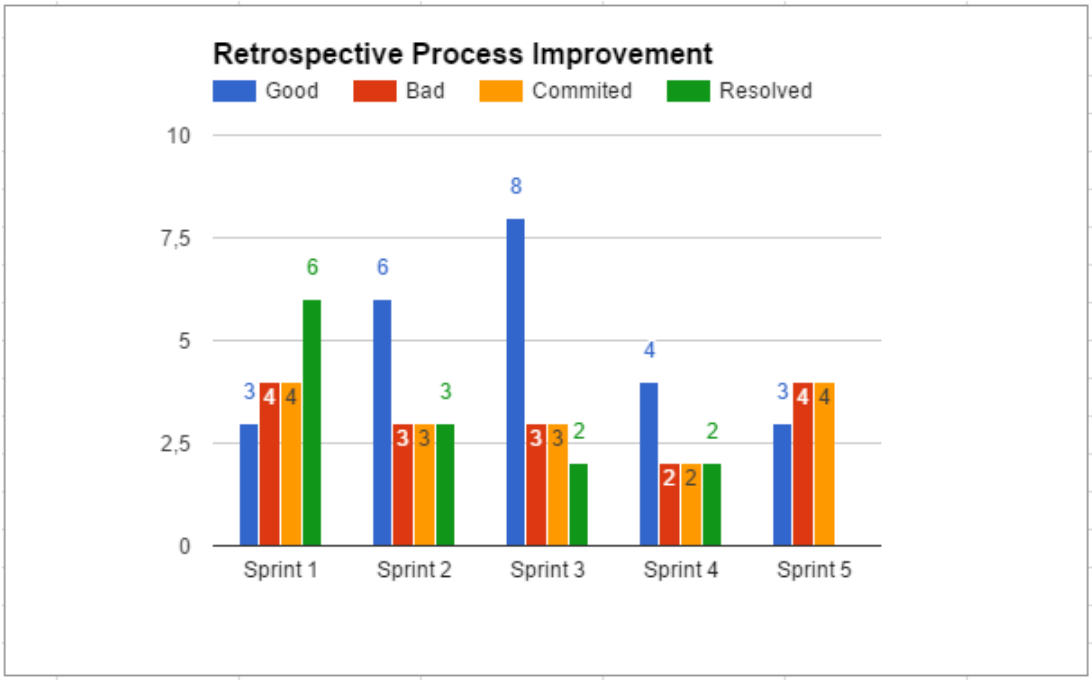

Figure 3: The evaluation of each of the five sprints.

Pros & Cons
These three KPIs were useful in terms of giving us an idea of what we had accomplished after each sprint. The first two KPIs showed us our progress which gave us a sense of where we were in the project and how much work we were actually getting done each sprint. For example the stories committed vs stories completed (1) shows that we made massive progress in the project during the third sprint in terms of user stories whilst the group velocity (2) shows the smaller steps forward in terms of tasks. This information was, together with the third KPI, useful for planning and reviewing sprints. It also helped us improve our process continuously during the project.

A problem with the chosen KPIs was that we only had a small number of sprints in a very short project to make use of them. The purpose of them is to continuously give an overview of the project and to continuously improve the process. The benefits of this are cut short in a short project because there is less time for process improvement and refinement.

In Practice
In practice we noted the stories and tasks that were committed and completed in our Scrum board. We did this in the meeting after each sprint. Together we also evaluated our process during each sprint. This was all done fairly briefly and we could have gone more in depth in terms of evaluating and analyzing our process.

Future
The KPIs would have benefited from having more time. The length of this project was already set but it is clear that having a longer project would have made the KPIs more beneficial. It might have been possible to have even shorter sprints and thus give more opportunities for review and process improvement although

that might take away valuable development time and be difficult to schedule. Having a longer project would have also given a further incentive to evaluate the sprints in greater detail for greater process improvement.

2.2.3 Version control system

The version control system we used was Git with GitHub as host. Git supports working in different branches and the making of pull requests which is a request for merging with the master branch from your own. We tried to use an easy branching model where everyone had to switch from the master branch to a newly created local branch and then make a pull request which would be reviewed by someone else. This keeps the work with Git and Github clean and also avoids merge conflicts in the master branch. Github is very supportive of merge-conflicts and has a intuitive way of dealing with them which makes it easy to fix all the conflicts in the specific branch before pushing the new features or changes to the master branch.

Pros & Cons
The benefits of using the branching and pull request method where someone else review your code is a really good way of dealing with new features being added to the app. In a project of this size it would not have worked if everyone worked towards the master-branch, thus it would have caused a lot of conflicts. One mayor benefit as well is the fact that your code gets reviewed by someone else than you, this helps with finding errors and optimizing the code.

In Practice
Everyone had worked with Git and Github before which made a lot of things easier. But in order to keep everyone at the same page and level a Github "How-To" was made and pushed to the repository. The document was an easy to follow, step by step guide on how to work with Git. The guidelines was in the beginning followed strictly and almost instantaneously after the guidelines where not maintained different problems arouse.

Pull requests was supposed to be a mean for the group to revise and to merge other peoples requests to the master branch. The problem was that it became more and more sloppy and if the pull request was able to merge without conflicts, the posting member of the request would just accept it. This kind of breaks the whole idea of reviewing each others code and might have caused the code not to be optimized and free from errors.

Future
It is clear that working with Git and Github in this way is optimal but in order to actually maintain the standard set out from the beginning the group should be reminded of the guidelines continuously. Since most group members did not have prior experience with branches, there was a few times where the implementation could have worked better. Workflow was better after a few sprints but it could have been avoided if everyone studied how branching in Git works before the development started.

2.2.4 Scrum board

A scrum board is a central part of scrum that enables the team to easily see and keep track of the sprint backlog. Breaking down user stories into smaller tasks and dividing them to group members is easily done with a scrum board. To keep the scrum board easily managed and well-structured, Trello was used. Trello was chosen since one of group member had experience in working with it.

Pros & Cons
The Scrum board gave a good overview of the project. It was easy to see what was left to do in the current sprint and what everyone was working on. It was convenient to use a digital board compared to a physical board, since we could access it from any place.

The alternative would be to have a physical Scrum board which could get more organized than a digital board, but since we didn't have a permanent work area this was not possible.

In Practice
The group started out by setting up a collective Trello where everyone had administrative access. After that the Scrum board was set up with help from an online template for using Scrum with Trello. After modifying it a bit the relevant user stories was initialized into the original product backlog.

The board was used continuously and made all of the work so much easier. At the beginning of the sprint planning the whole board would be revised. The main board consisted of a product backlog, sprint backlog, current sprint tasks and completed (user stories and tasks). The product backlog did not change a lot from the beginning and the backlog was sorted out with the highest priority according to the group, product owner and stakeholders at the top. This made it easy to see what the next steps would be. After the group concluded what product backlog item to manage forthcoming sprint it was moved to the sprint backlog. After breaking down the tasks they were moved to Current Sprint Tasks where they would get an estimate on how much effort in hours they would take and also be delegated to a person/s responsible of said task. In order to structure the sprint planning, the user stories would firstly be sliced to tasks. After the tasks had been created, they were delegated to different group members and a time estimate for each of the tasks was made. After said task was done it would be directly moved to Completed. In figure 4 the Scrum board can be seen for during a sprint.

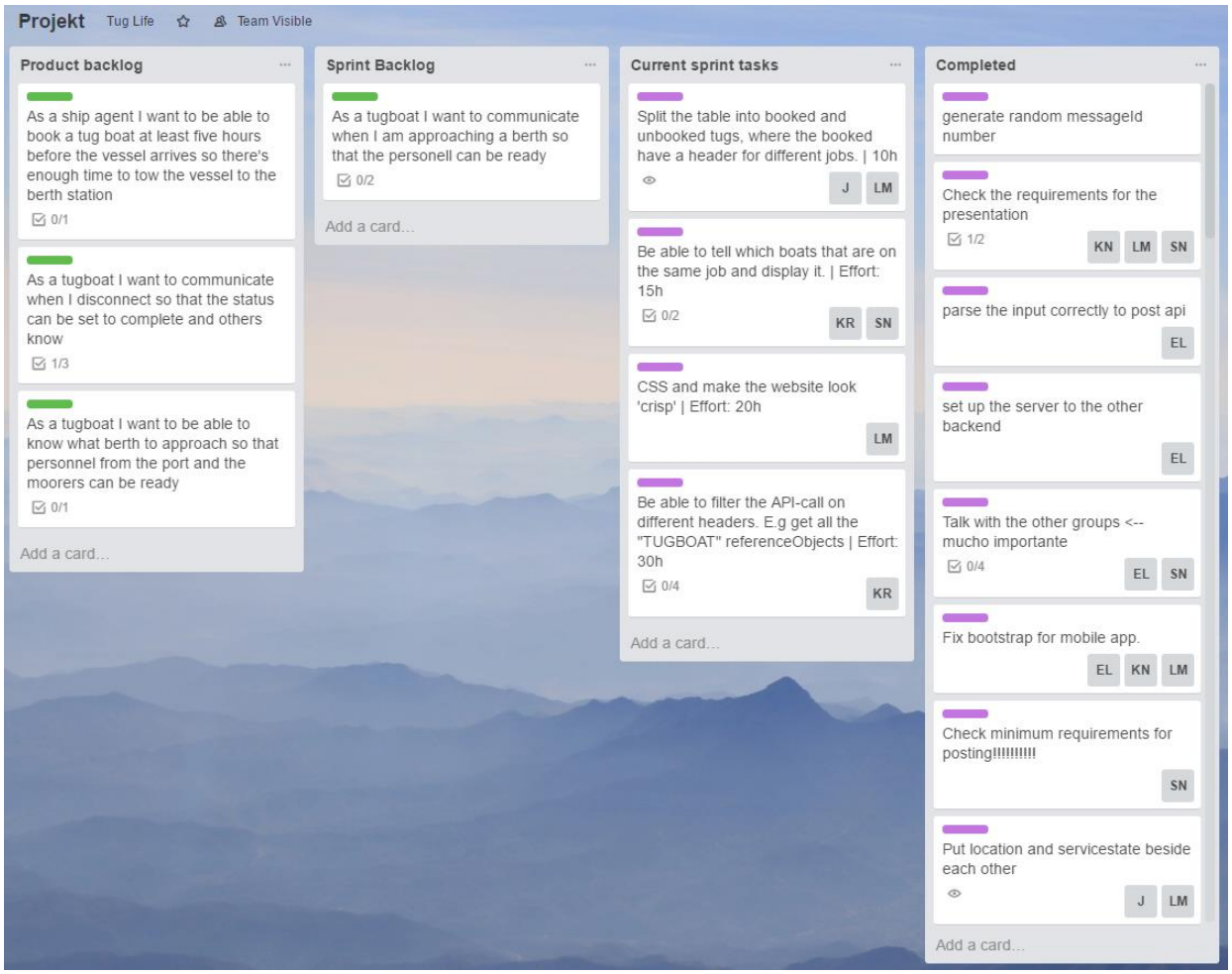

Figure 4: Screenshot of the Trello Scrum board used during the project.

Future
The work with Trello was proven to be very helpful and easy to manage. It seems highly suitable for projects of this type (and many others) and will be used in future projects.

2.2.5 Slack

Slack is a widely used messaging-service for communicating in different projects. As stated in the social contract the main communication was to be held through Slack which is a tool for keeping all communication at one place, mostly used by groups. A Slack group is divided into certain channels e.g "general and backend" where questions and discussions most suitable to the channel is held.

Pros & Cons
The benefits of using Slack is its highly adaptable way of keeping track of different group-communications. In a project where over 50 people have to communicate with each other it is of utmost importance to have a structured way of communication. By using different channels it was easy for the group to communicate with other groups and ask relevant questions and get answers when needed.

In Practice

A participant of the course made a Slack group where PortCDM, all the groups and supervisors could communicate. Different channels where relevant questions to the channel would be asked was made. It was divided into the channels:

- Ask-a-developer
- Ask-a-teacher
- Backend
- General
- Random
- Tech

In addition to this we also created an own Slack group where communication within the group could be held. This was often used during days which we could not have actual meetings to make sure everyone knew what their task was. In figure 5 an example can be seen of the use of Slack where a thread has started of a question that was asked and where both developers of PortCDM and other students answer.

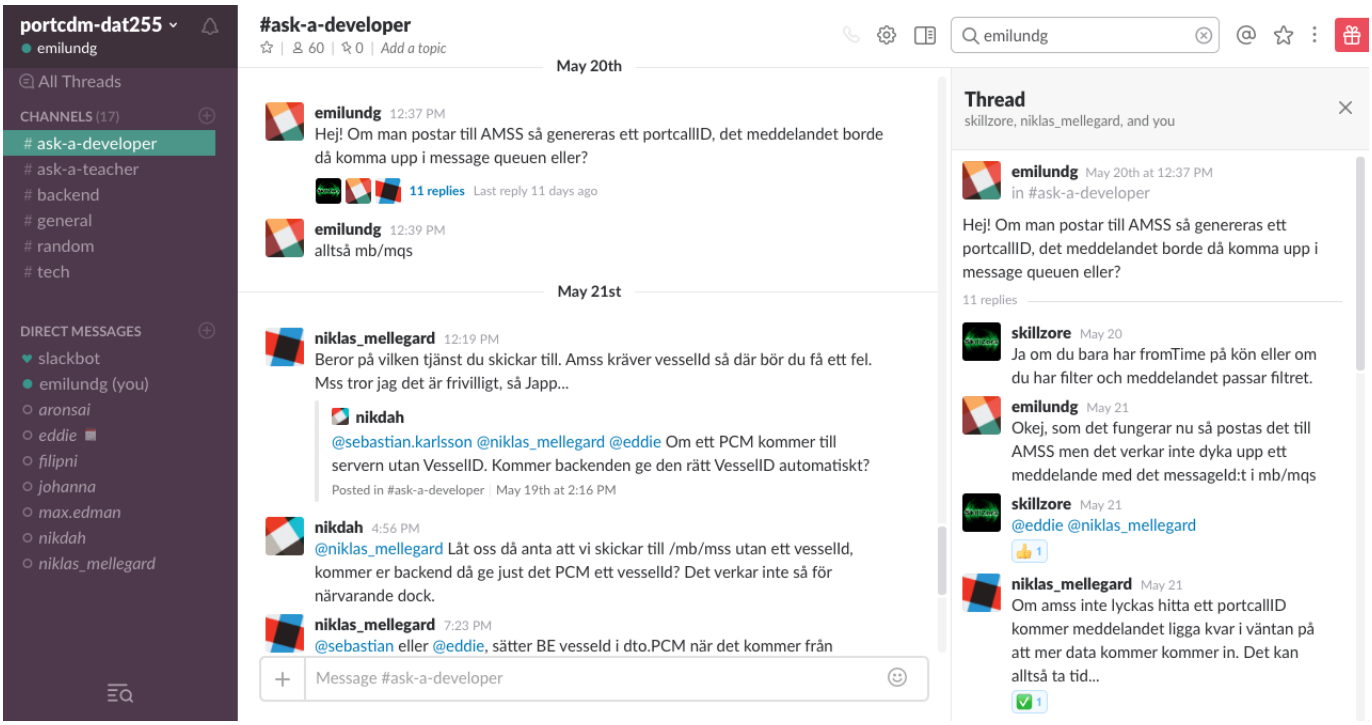

Figure 5: Screenshot of the Course Slack used during the project.

Future
How Slack was used in this course was optimal for projects of this size and would be used in the same way in the future.

2.3 Time distribution

Already from the start it became obvious that a task takes as much time as it gets. The result may vary and what is being prioritized in the task is up to the person performing it (e.g chooses quality over quantity).

Since we had 5 sprints that each were 1 week long with an effort of 20h per person, the project had a total time of 600h. All of the members in the team had the Bachelor Thesis along with the project, which caused one sprint to get behind in time spent, but other than that it worked well with spending 20h/week.

The major part of the time was spent on the product development, e.g coding. The entire group had experience in programming before, but web development was new for some people in the group. This resulted in some team members being able to produce more in the same amount of time. The appliance of scrum took almost a third of our time. This included among other things sprint planning, break down the tasks and standup meetings. The reflection part of the project included to reflect what we had done so far and if it meets the expectations we had before starting the tasks. In figure is a pie chart of the time distribution shown.

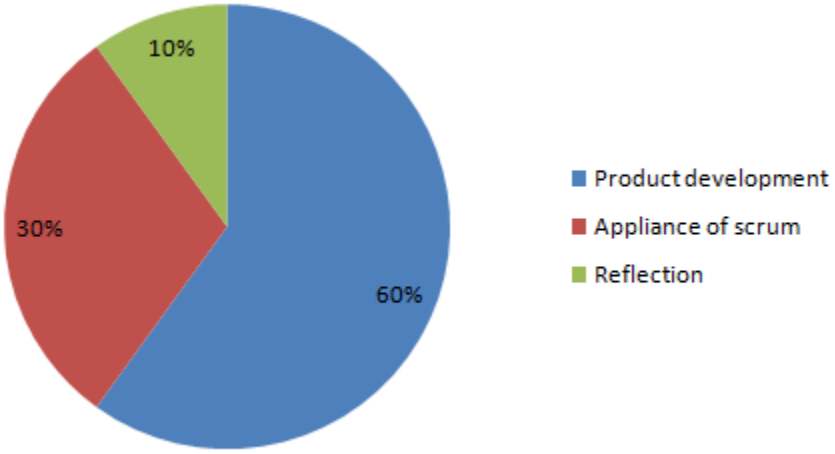

Figure 6: Pie chart of the time distribution.

2.4 Effort, velocity and task breakdown

An essential part of dealing with user stories and getting the correct feedback from the product owner involves applying the correct estimate of an effort. The effort helps the product owner what to prioritize next and is often not measured in time. Instead several other types of scales is more common when estimating scrum e.g Fibonacci sequence (as used in the Lego-Exercise), dog breeds or t-shirt sizes but it all comes down to what the group actually finds is the most comfortable. Planning poker as used in the Lego-Exercise is a common way of performing these estimates. It revolves around the Fibonacci sequence.

Velocity is used as a means for the group to give a correct estimate of the workload that they can handle in a sprint. It is often calculated using user stories and their corresponding story points. After calculating the total amount of story points per sprint an average over the time can be calculated.

Task breakdown is vital when working agile. The necessity comes from making large user stories easy to handle. From the Elephant & Feedback lecture, Backlogs & Task Breakdown held by Jan-Philipp Steghöfer and Håkan Burden it became obvious that in order to maintain an efficient work flow and keep the product owners happy it was crucial to slice the backlog items down as far as the group possibly could. Two factors played a mayor role in order to give the product owner a "vertical slice of the user story cake". At every week the group should be able to present something visual (connected to UI) with a function (connected to back-end).

This results in the product owner always having something useful (in contrast of just e.g presenting a design).

Pros & Cons
It is a good idea to break down tasks as it makes them easier to distribute and understand. In addition, the task's velocity and effort also becomes less problematic to estimate. Estimating effort and calculating velocity was helpful for the planning of the sprints as it gave us clearer picture of how much work we can get done and thus how much we should plan for.

As with the KPIs, a problem was that we only had a small number of sprints in the project. This meant that we only had a few opportunities to learn how to estimate effort, to keep track of the velocity and to break down tasks. It was still useful but the usefulness of these methods ought to become greater in a longer project.

In Practice
The velocity was calculated only with the amount of user stories/tasks completed vs user stories/tasks committed. The result of this being that a clear estimate of a user story's velocity is absent. This because of the size of the user stories may vary.

Breaking down tasks was incredibly useful for the group since it provided an easier way to divide the work to the group's members. In the beginning we found it difficult to break down user stories in smaller tasks. Thankfully this was realized early in the first sprint and it was something we worked to improve in the following sprints. When this was working better, the Trello board was very useful and each person in the group had a clear task to work on.

Towards the end of the project, the Trello board was not as well managed which led to some confusion regarding which task each member had and what their purpose was during the sprint. It should be noted that during this time, the list of tasks to work on was small and this could be the reason the board was not put to use as much as earlier.

Future
Breaking down tasks could have worked better in the beginning of the project, it took some time to learn how to manage the user stories and divide them into smaller components. Looking at the velocity and accurately estimating the effort was also something that became easier with time and would have continued getting easier with more time.

2.5 Sprint planning

At the beginning of every sprint a sprint planning meeting is held. The first part of the meeting is typically about reviewing the product backlog items and contemplate with the product owner which are still of the highest priority. It is during this phase of the meeting that a lot of discussion regarding what the product needs is held, both within the group but also with the product owner.

After prioritizing and discussing the backlog items need to be broken down. The second part of the meeting is about slicing the decided product backlog items to smaller ones. Different techniques might be used for this as explained above with task breakdown but the main goal is to make as clear and small steps as possible and add them to the sprint backlog.

Pros & Cons
A sprint planning meeting is a good way to break down tasks and estimate the workload for the next sprint. The meeting also increase the likelihood of a satisfied product owner, as he is given the opportunity to make changes and look at the progress. The results of a sprint planning meeting are a sprint goal and a sprint backlog, giving the group clear tasks for the sprint.

In Practice
Both Sprint planning and Sprint review was held at a set time every week. They were both held at the end of every sprint at the workshop with the stakeholders. The group got some time with the product owners and stakeholders every week to discuss the next steps of the development which made the planning efficient and easy.

Future
In the future sprint planning would be done in basically the same way, at least in a project of this magnitude where different actors work towards a unified goal. To have the product owner come by at a set time and check in with every group was really helpful. To keep well documented sprint planning meetings is important as well. This would be implemented further in the future and serves the benefit of easily being able to review past meetings. An agenda for every meeting could also be posted the day before to keep everyone in the group focused.

2.6 Sprint retrospective & review

Sprint retrospective is held by the scrum master where the group discusses three important questions:

- What went well during the last sprint?
- What went badly during the last sprint?
- What could/should be improved in the future?

The importance of a Sprint retrospective lies in the way that they keep the group constantly involving and improving. It also serves as a base to where the sprint planning (and project) should head next.
The sprint review is at the end of each sprint when the group meets with stakeholders to review the progress achieved during the past sprint and to get feedback. A burndown chart can be used as a means of visualizing the work done in all sprints.

Pros & Cons
Sprint retrospectives are useful since they give the group an idea of what was done well during the sprint and what was not successful. We also discussed what we could have done better so that we could improve on that for the next sprint. The sprint reviews gave us an opportunity to get feedback from stakeholders on what we ought to do in the next sprint.

In Practice
At the end of every sprint at the workshop with stakeholders a retrospective/review was held to break down the concluded sprint. We talked about which user stories we completed during the sprint and which ones we might not have completed, as well as why these were not completed. The sprint reviews were well documented and was an open-ended discussion where every member could air their opinions. Because of a KPI being Retrospective Process Improvement it was of a certain importance to actually take time and discuss what were good, bad and what the next steps for work improvement should be. The group's sprint review document can be found in the Appendix.

The burndown chart presented below in figure 7, was constructed after the final sprint, to visualize the work done in each sprint.

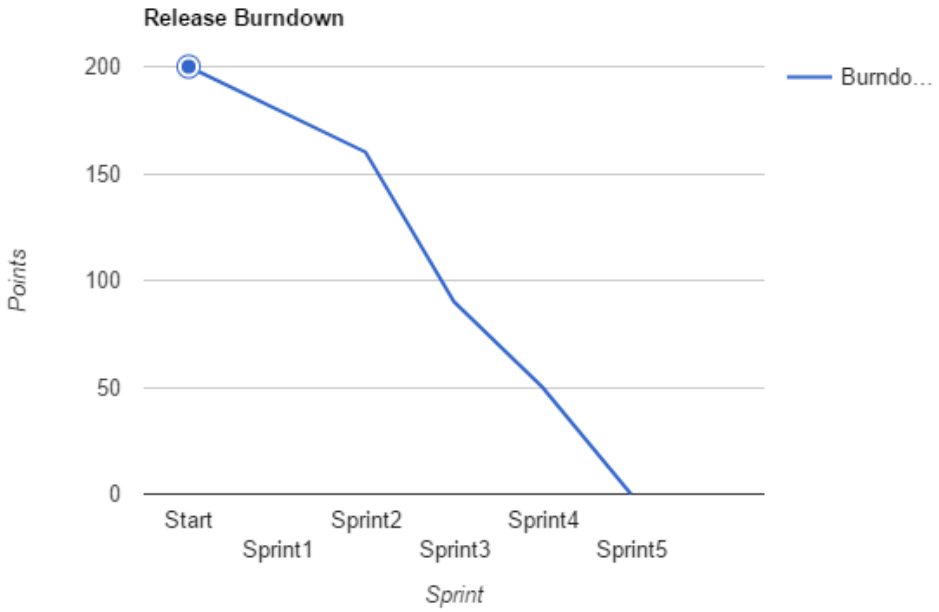

Figure 7: Release burndown chart with the development of approved story points per sprint, with a start of 200 points.

Future
Sprint reviews and retrospectives worked well for the group. To keep the meetings well documented as well as actually taking time and listening to each other without judgment was proven to be useful and the importance of it became clear over the course of the project. The burndown charts would have been more representative of reality if we would have worked on the charts during each sprint review. Instead, we created the burndown chart in the end of the project with the help of the sprint review document, which can be found below in the Appendix.

3 Reflection

In this chapter reflections on the project will be explained in more detail. This includes thoughts about new tools and technologies that we had to master, prototype and process as well as implementation of different requirements.

3.1 New tools and technologies

Over the course of the project we decided to use tools and technologies that were new to some or all of us. Examples of this would be Javascript, NodeJS and VueJS. Learning how to use these tools and technologies took a lot of time during the first sprint and varying amounts of time during consequent sprints. Since the group members had different levels of prior experience with the tools, it meant that some group members had a more difficult time contributing to the project because they needed more time to learn how to use the tools. The choice of tools could have been different but since it was based on what was fitting for the project it made sense to go with these choices regardless of prior experience. Learning new tools is also a part of projects such as this.

Other than having to learn how to use the new tools and technologies, we also encountered other difficulties with them. An example of this would be the use of git where we had the problem of pushing most commits to the master branch which led to git conflicts. To solve this problem we started using separate branches for different parts of the development. These could then be merged together. The short sprints and frequent meetings made it easier to quickly identify and solve problems like this.

Learning the new frameworks took time and effort which slowed down our efficiency overall in the beginning. As the time went by we noticed a spike in our development which was a result of our efficiency going at an increased rate upwards after learning the new tools.

3.2 Prototype, process and stakeholder value

To maximize stakeholder value we needed to know what the prototype should be like and what we were to develop. This information was then used to shape our process in terms of planning, executing and reviewing. In practice this meant that we communicated with stakeholders to figure out what we should develop, and during the sprint reviews we could also show our progress and get feedback on the current version of the prototype. Based on this we then made changes, decisions and plans to adjust the development of the prototype to be in line with what the stakeholders wanted.

Since our project was divided into week-long sprints during which we did not change the sprint backlog, it meant that every new sprint needed to include the updated information and feedback from the stakeholders. A strength of this development method is that changes to the plan only happen for every new sprint and not during a sprint which makes the direction of the development clear for both developers and stakeholders. It can then be adjusted accordingly.

3.3 Evaluation of D1-D4

D1 was a document describing three strategies on how we will implement Scrum in our project based on what we learned in the Lego Scrum exercise. The exercise helped us familiarize ourselves with Scrum, and the potential problems and benefits of it. The three strategies we chose were:

- Involve product owner
- Avoid unnecessary planning
- Communicate with other groups

Involving the product owner was a successful strategy since it allowed us to understand what we needed to develop. There were some difficulties involved with this. Notably, we got put on the wrong track where we developed separate ship agent and tug boat pages accessed through a start page when there was no need to create the ship agent page as that was the responsibility of another group. This wasted valuable development time. The strategy was still helpful overall because without it we would not have known what to develop.

We avoided unnecessary planning by having sprint planning meetings in a defined time slot, and then executing the plan during the sprint without any extra planning. It worked well in the sense that we did not waste time planning in more detail when it was not needed. This allowed us to start working earlier and thus save time. It might have been possible to get value out of planning in more detail to avoid misunderstandings and mistakes but that would have used up extremely valuable development time. In a short project with short sprints such as this, it appears to have been a sensible strategy to avoid unnecessary planning.

The strategy for communication with other groups was executed by talking with other groups in person and through Slack, as well as having the Scrum master attend meetings with representatives from the other groups. Because the project involved ten different groups, it was difficult to know what needed to be communicated and with whom. For example, we could have communicated better with the ship agent group to avoid the development of the previously mentioned ship agent page. While we could have been communicating with the other groups more actively, this strategy still allowed us to get a better idea of what our task was in relation to what the other groups were doing.

D1 also consisted of a social contract in which we specified how we should work in the group. Being on time for meetings, being open minded, asking for help, communicating through Slack and being civil in case of conflict were the parts of the contract. We followed the contract without any notable problems and we did not do any changes to it.

D2 consisted of three Key Performance Indicators (KPIs) which were chosen by us to monitor our implementation of Scrum in the project. They are described in section 2.3.2

D3 consisted of an initial product backlog and an initial business model. The initial product backlog contained the user stories we thought were going to be necessary based on the information we had available at the time. Some of the user stories were changed over the course of the project and some of them were removed, but it was still useful to have an initial idea of what needed to be done. This helped with decision making and planning the project.

We used a business model canvas to create a visual chart of our chosen business model, as seen below in Figure 8.

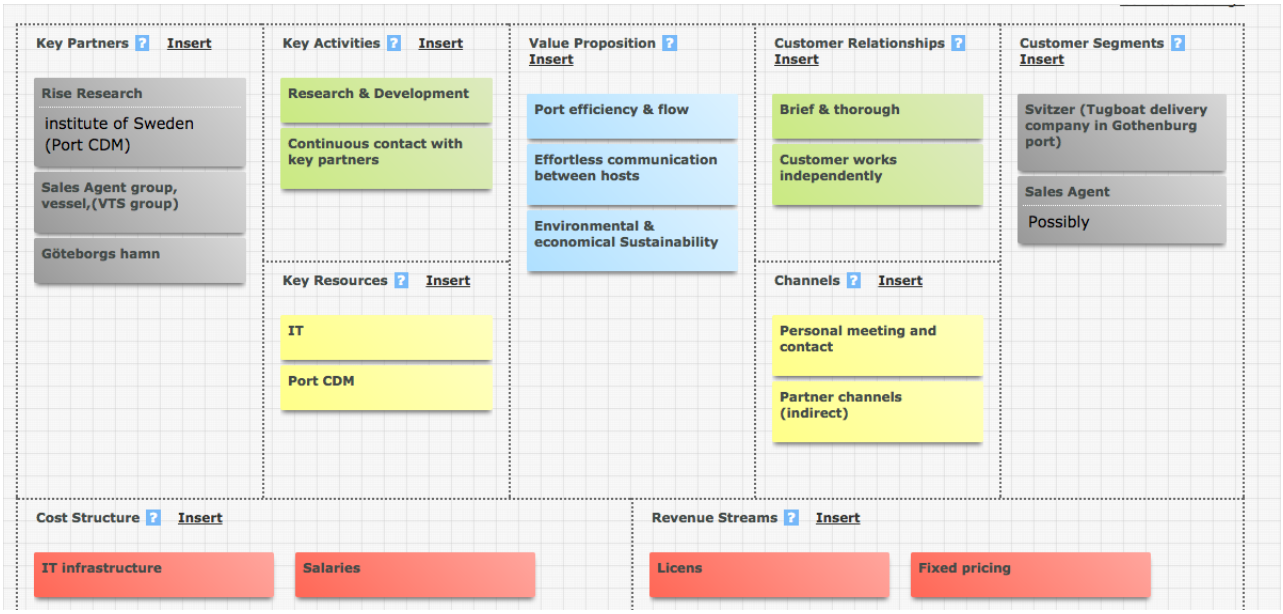

Figure 8: Business model canvas showing our business model

We filled in the different sections of the business model based on the setup of the project. Since this project was not an actual business project the business model was not of utmost importance and we did not pay much attention to it during the rest of the project. It did still provide some insight into what one needs to consider when planning a business project.

D4 was a half-time evaluation of the project. We evaluated the first three sprints by looking at what we had accomplished in the project up until that point and what difficulties we encountered. This provided an overview of what had been done in the first half of the project and some insight into difficulties we had in the early parts of the project.

4 Workshop with stakeholders

On every Wednesday the group assembled for a sprint retrospective as well as talking to the project stakeholders on a workshop. The workshop included an activity called "scrum of scrums", where the scrum master of each group got together to discuss what they've done in the group, what they're going to do, what they're struggling with and also if their work is going to affect the other groups. This way all of the groups had a clear vision of what was coming next and also presented an opportunity for mutual activity planning and discussion. The workshop gave the group a great opportunity to chat with the representants of the harbour and PortCDM, to get a perspective of the project and to know if it was going in the right direction.

As mentioned the workshops also served as a great opportunity for a sprint retrospective as the whole group was together. During the sprint retrospective the group went through the efficiency of the sprint as well as future planning and KPI measurements.

5 Conclusion

Scrum being used as a method of managing the development of this project has been widely successful. Since Scrum development was a new concept to the project group, it was at first challenging to fully take advantage of the agile methods and practices that define scrum. This was evident in the fact that we did not, for the first few sprints, use stand-up meetings at all. Once we realized how useful these meetings could be, we began implementing them in our daily work and saw an improvement in the development process.

Overall working in sprints proved to be successful and it is something we all found very useful. The first sprint was a bit disorganized since no one in the group knew for sure how much work we could handle in a week's time and most of us were inexperienced when it comes to developing a web application.

As mentioned earlier, using a scrum board was a key part of the project and proved to be very effective for dividing work once we realized how it was supposed to be used. As mentioned earlier, the use of Trello diminished towards the end of the project. Ideally, we should have kept using the board in the same way we did in the earlier stages of the project.

Appendix

Sprint 1 Review

- No matter which platform & programming language we chose it'll always have a steep learning curve (which later will settle).

The good:

- Helpful teammates
- Effective scrum use
- We sliced the elephant good

The bad:

- Bad communication between teammates
- Git problems, bad use of branches (Don't push to master branch!)
- Not having clear goals for everyone
- Bad at dividing work

Future:

- Better communication through Slack
- Meet up more often with clear set goals
- Better use of branches and more effective Git use

Sprint 2 Review

The good:

- We've been hacking like crazy peoples
- Accomplished one slice, ish.
- Stand up meetings are wonderful → Everyone knows what to do etc.
- Slack communication has improved
- Better at dividing work
- Met more, which is nice.

The bad:

- Css not working
- Everyone should check the other slack channel
- Git is still chaos → Push too often to master, get develop and master branch?

Future:

- Keep the good up.
- Make clear rules for git.

Sprint 3 Review

The good:

- Found out what problem caused issues with the previous product!
- Best sprint so far
- Communication through slack have been working good.
- Github better, good work with branches
- Everyone had something to do
- Stand up meetings
- Good planning
- Good tempo

The bad:

- Bad contact with authorities.
- External things, PortCDM not working for all and not reliable.
- Not talked to the other groups at all.

Future:

- Contact the other groups more
- Contact switzer (finally received a contact person)
- Heroku app implementation for final presentation.
- Referera till webapp initieringen på deras github.
- Start to think about presenting it to Burden → Deliver working prototype

Sprint 4 Review

The good:

- Vi fick saker gjorda
- Bra kommunikation
- Hade några möten kanske
- Emil MVP

The bad:

- Många timmar gick till kandidaten.
- Kunde fått mer gjort

Future:

- Contact the other groups more
- Heroku app implementation for final presentation.
- Start to think about presenting it to Burden → Deliver working prototype
- Kolla kraven på post

Sprint 5 Review

The good:

- Fick det att funka och scenariot gick fint!
- Kommunicerade med andra grupper/Frågade om hjälp när det behövdes.
- Fick kontakt till Switzer

The bad:

- PortCDM ändrade grejer så vi var tvungna att tänka om lite.
- Kaos-uppgifterna blev lite dåligt delegerade.
- GIT-Kaos kvällen innan redovisningen.
- Missuppfattat en user story, hade löst sig om vi pratat med ship agent gruppen tidigt.

Future:

- Snygga till hemsidan och koden. Fungerande för Burden.
- FindBugs, GitInspector. Skriva dokumentation.
- Report.