

# Reflection report - Runda Bordet

Software Engineering Project DAT255/DIT543 - Spring 2017

Johanna Torbjörnsson  
Robin Bengtsson  
Olle Persson  
Magnus Rönnberg  
Aleksandar Babunovic  
Ken Bäcklund

---

# 1. Introduction

This is a reflection report for the Njord project by team "Runda Bordet". It includes our reflection on how the situation is, how we would have liked it to be or other paths we could have taken, and any realistic way on how we could have gotten there.

---

## 2. Application of scrum

### 2.1 Roles, teamwork and social contract

We started our project by writing the social contract, see Appendix A, and in retrospective we followed it fairly well. I don't think we would have changed anything in it based on our experience of working together. We had two scrum masters throughout the project.

### 2.2 Used practices

We used Trello to help us to manage and break down our project in user stories, and then further into tasks before each sprint. In retrospective, this was a good choice and easy to use, and a valuable tool we would also use in any future project.

### 2.3 Time distribution

We tried to divide the time evenly and each member took on tasks they wanted to work on. Due to that members had different time commitments and some planned absences, some members did put in more work in the project than others. But we all put in effort into the project to each of our abilities, at least 20 hours/week each member.

### 2.4 Effort and velocity and task breakdown

After some discussion we decided to use a velocity measured in hours/person. A 6-person team then gave us a weekly velocity of 120, minus the hours lost due to scheduled activities and any planned absence. This usually gave us a real velocity of around 70-90.

We put some efforts into making the tasks small and clear enough to be able to estimate their velocity better. This didn't go as well as we hoped in the beginning, but we improved a bit during course of the project. If during a sprint we found out that it was still unclear, we adjusted the descriptions but without affecting the overall weekly sprint goal.

---

### 3. Reflection on the sprint retrospectives

In the sprint retrospectives each week we answered the following questions:

- What have done?
- What has been going well?
- What has been going less well?
- What do we want to do different during next sprint?

These questions helped us develop our process and made us aware of how we could do things differently to improve the work.

What could have been different is that we could have included more detailed questions, and recurring topics. Example of those topics could be git, co-working, stand-ups etc. For each topic we could have answered what had been going well and bad and how we could do differently. We could also have discussed the retrospectives themselves during the retrospectives.

---

### 4. Documentation of sprint retrospectives

See appendix B.

---

### 5. Reflection on the sprint reviews

In the sprint reviews we had our opportunity to show our product to the product owners of PortCDM and get some feedback from them. When we showed them our product, they also told us what they would like to see in the application. In that way we knew more what they needed and focused more on to deliver that for each sprint. We got good reviews for each sprint review, but sometimes they also mentioned that we should do some things different.

---

## 6. Best practices for new tools and technologies

In our project we used C# ASP.NET as a tool to construct the application. We took the opportunity to learn a new language since many of us hadn't worked on ASP.NET before. In our first sprint we had one user story where we learned the tools needed. The decision to use ASP.NET unfortunately led to some problems since half the group used Apple computers, and because of this they couldn't use the IDE Visual Studio. They had to use Xamarin instead, which worked fine but had some problems with Visual Studio extensions.

Throughout the project we used the agile technique pair programming. Since some of us had problems with Visual Studio we took the opportunity to use this technique. While this resulted in better code quality, it also made it seem like some didn't make as many commits as others on the project. Therefore, the output from it made it hard for us to measure how much each member actually spent on the project, as seen in the output from Git-inspector, appendix F.

The usage of Git worked smoothly since most of the members of the group had used it before. The more experienced ones helped the ones with less experience. During the project we had one branch each and every Friday we merged the branches so all the members got the last version of the program. Trello is a tool some of us had used before and it was pretty straightforward and easy to learn for those of us who hadn't used it. The good thing with Trello is that you get a good overview of the status of the project, and you can clearly see what is finished and what needs to be done. To make the project available for all we made an azure account and made it to a website.

Code Climate is a tool we used as a KPI to check the quality of the code during the entire project. At the end of the project we used FxCop as a code analysis tool. The recommended code analysis tool for the course was Findbugs, but since it's made for Java projects and our project was in C#, we had to use another tool. Looking back, it would have been better to use FxCop as a KPI instead of Code Climate, since it was made for our kind of project.

FxCop categorizes its issues in five categories; Critical Error, Error, Critical Warning, Warning, and Informational. It seems like the issues were a bit inconsistent though. Most of the issues were complaining about our naming convention, since FxCop compares the naming with the Microsoft standard. We used the convention we are used to, a naming convention that resembles that of Java. One of the things FxCop complained about was that Enums all used capital letters, without using this convention the Enums sent to PortCDM wouldn't have worked, so this was something we could not control. Therefore we chose not to adhere to it's standards and ignore most of the naming convention issues.

---

## 7. Reflection on the relationship between prototype, process and stakeholder value

During the start of the project we built an initial backlog with features that we felt the end product would need. Although we knew that the initial backlog was subject to change, our whole picture of the project was changed as soon as the first sprint was over and we got to meet the product owner (PortCDM). Since we were given the role of the ship agent (and a lot of the ship agent's responsibilities are handled by PortCDM) there were many of features in our backlog that the product owner felt weren't really needed. This made us rethink the whole project and instead of handling the responsibilities that the ship agent handled in the given scenario we decided to focus more on the different side tasks that the ship agent could be given by an incoming ship. This was more inline with the product owner's views. Thus we had to change a lot of our initial backlog and we fell into the habit of adding tasks to our sprint backlog after every retrospective and meeting with PortCDM.

We learned quite early in the project that we had to keep an open mind, many things changed from week to week and the product owner had a lot of requests that we hadn't necessarily thought of. This was not a bad thing, however. Quite the opposite. The more we got to speak with the project owner the more we felt that the final quality of the product increased.

During the last week, after the final project demo we felt that PortCDM liked the prototype that we managed to deliver and that we had (mostly) passed the acceptance test. One has to remember that the process of software engineering (or customer service as a whole) is not really about the product. In the end it is all about making the customer happy.

---

## 8. Relating your own process to literature and guest lectures.

The book “Scrum and XP from the Trenches - 2nd Edition” [1] was an interesting read, especially on how the author used a causal language and how we reflected over his own work eight years later. It was a good insight on how our work methods can change.

Relating this to our own process, as a novice team we might get caught up on following too many techniques and strategies that other teams used before us. As we became more adept in using Scrum, we realized that some methodology might be more of guidelines than rigid rules written in stone.

The “hamburger method” [2] by Gojko Adzic wasn’t something that we did ourselves, we did however use a lot of similar techniques in how we broke down user stories in estimable tasks. It might be similar to the cake method that we learned of during the lectures which influenced us a lot more in how we planned each sprint.

We didn’t focus more on other literatures. In retrospect, perhaps the article about Scrum of Scrums [3] could have been a valuable read. We didn’t read any of the Git-related books referenced on the course website [4] since we all had basic knowledge of using Git from before which we felt were sufficient enough to work on our project.

We attended two guest lectures during the project, and while they were interesting to hear about, many of the techniques described were better fitted for bigger projects with hundreds of developers. Therefore, we didn’t apply any of the techniques on our project.

---

## 9. Evaluation of D1 - D4

### D1: Scrum strategies, appendix C

This document includes three scrum strategies, which we came up with during a lego exercise. We used strategy one and three during the project, and the outcomes were the same as we predicted in this document. This implies that the strategies were valuable.

We didn't have the need to use strategy two. Nevertheless, we think it's a good strategy if you are in a situation where the customer has a specific request and you think there could be a better solution to the problem.

### D2: Key performance indices, appendix D

This deliverable clearly describes what KPIs we were going to use, why we choose them and how we were going to execute the use of them. A reflection of the KPIs can be found in section 10.

### D3A: An initial backlog

The initial backlog was a great tool to get an overview of the project as a whole. We also learned pretty quickly how different our view of the project was from the product owner's. As the project proceeded we continued to use less and less of the features we had initially thought of in our backlog and exchanged them for features that the product owner wanted. All in all it was a great experience to see how quickly a project can change.

### D3B: The business model canvas, appendix E

We feel that the business model canvas (BMC) that was created early in the course was a bit unnecessary. It was created as a hand-in but neither we nor the product owners used it during the development of the product. Although we feel like it was an unnecessary part of this project we can still see the need for it in actually monetized project. As with the initial project backlog you can easily get a good overview of the project from a business point of view with the BMC, with can help with priorities of the different features when developing the product.

### D4: Half time evaluation, appendix F

The halftime evaluation provided some good insight into the progress of the development of the product while also giving us a chance to compare our process with another group.

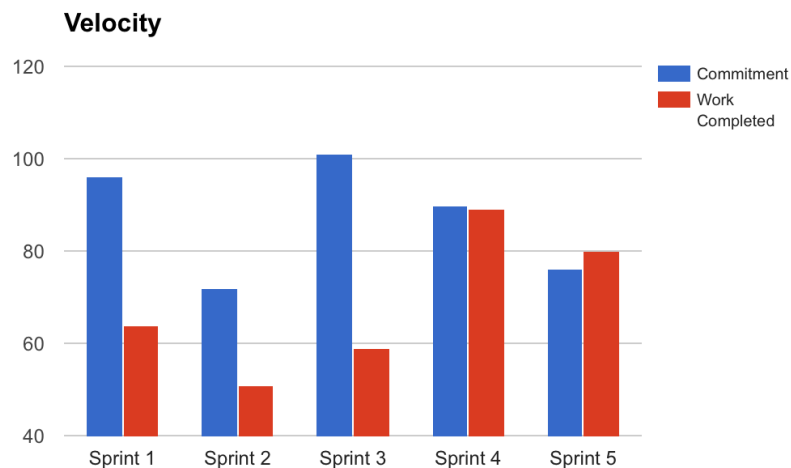
---

## 10. KPI charts

We used three different KPIs, which we thought would compliment each other in a good way since they focus on different things.

### 10.1 Velocity

The x-axis represents the sprints while the y-axis represents story points, and two bars at each sprint shows the estimated and completed story points. See image 1. In the beginning of the project there was a big gap between estimated and completed story points. Most likely, this was because we estimated some of our user stories far too low. The cause of that was often technical difficulties and that the API was much harder to implement than we expected. During the last two sprints the amount of estimated and completed story points are nearly identical. To summarize, this KPI shows that we improved our ability to estimate story points during the project.



*Image 1*



## 10.2 Burn up chart

The x-axis represents the sprints while the y-axis represents story points. Two curves is shown, one representing completed story points, and the other the forecast. See image 2. While we unfortunately didn't reach the forecast, this KPI shows that we managed to keep a continuous speed throughout the whole project.

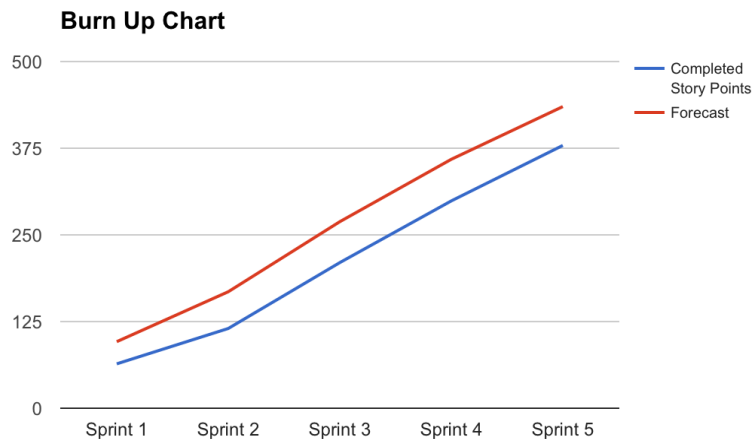


Image 2

## 10.3 Code quality

We used the tool Code Climate to evaluate the quality of our code. The blue curve represents how many issues our code had, and the red represents the GPA of the code, which is some kind of grade. See image 3. We can see that the number of issues increased almost exponentially. We didn't have the time or effort to fix these issues, since we thought they were only minor. It might not be apparent from this image, but the GPA decreased slightly towards the end. We didn't have much use of this KPI, mostly because we didn't know how the GPA was calculated, and therefore didn't know how to improve it. If we had put time into understanding the GPA it could have been more valuable, but that was not something we prioritized.

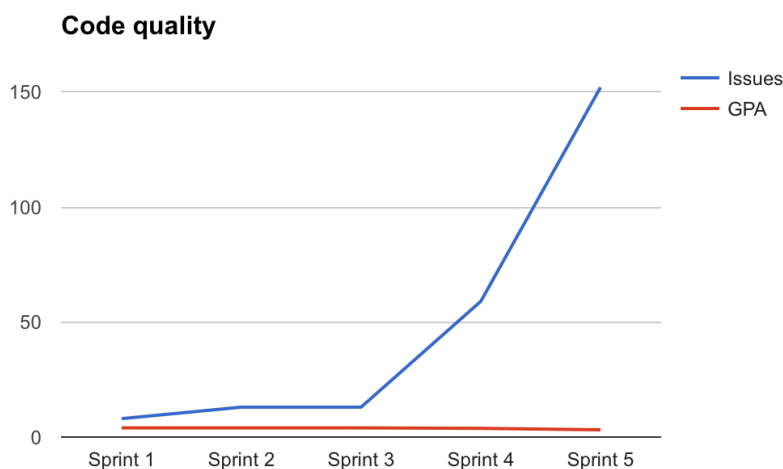


Image 3

---

## 11. References

1. Scrum and XP from the Trenches - 2nd Edition, Henrik Kniberg
2. Splitting user stories -- the hamburger method, 2012-01-23, Gojko Adzic, <https://gojko.net/2012/01/23/splitting-user-stories-the-hamburger-method/> (Acc 2017-05-30)
3. Advice on Conducting the Scrum of Scrums Meeting, 2007-05-07, Mike Cohn, <https://www.scrumalliance.org/community/articles/2007/may/advice-on-conducting-the-scrum-of-scrums-meeting> (Acc 2017-05-30)
4. "Course PM for Software Engineering Project (DAT255/DIT543) 7.5 HEC, Spring 2017", <https://github.com/hburden/DAT255/blob/master/README.md> (Acc 2017-05-30)

---

## Appendix A

# Team social contract

Standup meetings twice a week; Thursdays at 13:00 and Mondays at 12:30.

We try to not book any meetings on weekends.

If you are late for the meeting, please send a message to all the members of the group.

We have zero tolerance for bullying

Don't dismiss other people's opinions, instead try to listen and see their point of view

Ask for help

Between meetings our primary channel for communication is our Facebook Messenger group.

Signed:

Johanna Torbjörnsson

Robin Bengtsson

Olle Persson

Magnus Rönnberg

Aleksandar Babunovic

Ken Bäcklund

---

## Appendix B

# Retrospective 26/4

### VAD HAR VI GJORT:

Flera i gruppen hade stora problem med att få API:n att fungera, därav kom vi inte särskilt långt mot våra user stories. Vi underskattade vår egna story att kunna använda systemet. Vi föll spontant in i våra roller i och med att vissa inte hade fått igång en API att jobba emot. Vi lyckades göra en första "Hello World" app som hämtar meddelanden, och en pappersprototyp för en initial design. Flera storys från sprinten kommer vara in progress även nästa sprint.

### VAD HAR GÅTT BRA:

Grupparbetet har fungerat bra trots många tekniska problem, vi hjälper varandra och upprätthåller vårt sociala kontrakt. Det är bra att vi följer våra storys på Trello. Vi har arbetat tillsammans på samma plats vilket vi tycker är bra, då vi kan hjälpa varandra, framförallt nu i början då vi måste komma in i systemet.

### VAD HAR VARIT DÅLIGT:

Magnus tycker att vi använder Facebook chatten för mycket, för han har inte Facebook på datorn. Det är löst nu. Vi har inte använt Slack mycket för kommunikation, vi ska försöka använda den mer. Framförallt har de tekniska problemen varit stora. Vi hade estimerat fel hur mycket problem vi skulle få med att få igång API:n samt sätta igång arbetet i C#.

### VAD VILL VI FÖRÄNDRA:

Vi borde dela upp stories mer så att flera kan jobba på det. Viktig information och länkar vore bra att pinna på slacken. Vecka för vecka bestämmer vi vad vi ska jobba med, utifrån vilka storys vi har valt. Fråga andra grupper när vi behöver hjälp (på Slack, eller annat).

---

### VAD BEHÖVER VI HJÄLP MED:

Just nu behöver vi hjälp med att få igång API:n (VirtualBox etc)  
Hur skickar man meddelanden till API:n, prenumererar på meddelanden  
Inga fler frågor gällande vad agenten gör? Funderar vi på.

### INFÖR NÄSTA:

Forsätta med in-progress stories.

# Retrospective 3/5

## VAD HAR VI GJORT:

Vi har valt huvud design för vår applikation.

Vi kan skicka ett meddelande i JSON eller XML format till API:t

Bryta ner meddelanden till XML

Förra veckans story att skicka enkelt meddelande till API:t är således avslutad.

HTML för meny och annat som syns på alla sidor, samt för sidan som skickar in ett meddelande

HTML för dashboard-sidan

## VAD HAR GÅTT BRA:

Vi samarbetar bra, inget bråk.

Olle har använt Slacken och fått hjälp, vilket var ett av våra mål från förra sprinten att göra mer.

Vi löste en del tekniska problem. Vi arbetar oftast tillsammans vilket vi tycker är bra.

## VAD HAR VARIT DÅLIGT:

Nya tekniska problem.

Vi har haft lite stora uppgifter som varit svåra att dela upp, vilket har lett till att somliga inte vetat vad de ska göra.

Vi hann inte klart med allt som vi tagit åt oss.

## VAD VILL VI FÖRÄNDRA:

Försöka arbeta i 2-manna team, en som löser back-end och en som löser front-end så att vi får en hel tårtbit för varje story.

Ha mindre stories som är enkla att dela upp så att alla får något att göra.

Ha mål för varje möte/dag så man vet vad man gör

Ha stand-up meetings varje gång vi ses.

---

## VAD BEHÖVER VI HJÄLP MED:

Hämtar man ut PortCallMessage via MessageBroker i JSON format så blir datumet fel

## INFÖR NÄSTA:

Implementera en databas som kan spara information om skepp etc

Göra klart fler sidor designmässigt

Skapa ett meddelande att skicka till PortCDM

# Retrospective 10/5

## VAD HAR VI GJORT:

Vi kan skicka in meddelanden till PortCDM med nästan alla Service States.

Vi har skapat en databas för att hantera information om Portcalls

Vi har gjort en ny timeline sida, och responsiv meny

Vi kan prenumerera på portcalls

## VAD HAR GÅTT BRA:

Vi har fått mycket gjort. Det har varit bra uppdelade stories så att alla har kunnat ha något att göra. Samarbetet har fungerat bra, vi har löst diverse problem som dykt upp tillsammans. Vi har fortsatt arbetat mycket tillsammans, men uppgifterna har varit mer uppdelade.

## VAD HAR VARIT DÅLIGT:

Några tekniska problem.

Vi har inte haft några stand-up möten, och vi har inte hunnit med alla storypoints.

Vi har underskattat hur mycket som krävs för vissa stories, och fyllt upp för att fylla hela sprinten.

## VAD VILL VI FÖRÄNDRA:

Vi SKA ha stand-up möten den här sprinten.

Estimera storypoints på ett bättre sätt; testa pokerplanning.

-----

## VAD BEHÖVER VI HJÄLP MED:

## INFÖR NÄSTA:

Utveckla vår egna databas, fortsätta bygga på det system vi gjort, göra tester på Databasen etc

# Retrospective 17/5

## VAD HAR VI GJORT?

Lagt till funktionalitet för att lägga till, ta bort skepp. Lägga till och ändra kommentaren på ett skepp. Databasehandler och html för detta.

På timeline kan man nu hämta portcallid från databasen och med det portcallid:t hämta alla meddelandet från portcdm.

Utökad design och funktionalitet för att skicka meddelanden. Vi kan nu skicka alla meddelanden som vi bör kunna skicka.

Fixat till småsaker med css. (Bakgrunden t.ex)

## VAD HAR GÅTT BRA?

Inte lika mycket tekniska problem. Bra med standups, tre stycken. Det gav bra översikt, lättare att samarbeta. Uppdelning av arbete har gått bra. Bättre uppskattning av tiden.

## VAD HAR GÅTT DÅLIGT?

Lite tekniska när problem när vi bytte version till sandbox. Tekniska problem med olika plattformar. Många fler issues enligt code climate.

## VAD VILL VI FÖRÄNDRA?

Inget. Vi vill fortsätta med standups.

# Retrospective 24/5

## VAD HAR VI GJORT:

Färdig prototyp av programvaran.

Vi har gjort Dashboard som hämtar kommentarer och nästkommande skepp.

Vi har kopplat ihop så att man kan gå från skeppssidan till meddelande eller tidslinje för ett skepp. Responsiv design och prototyp.

Gruppera portCallmessages i Timeline.

Kan skapa ett enkelt Departure Message.

Finjusterat Send-message funktioner.

Lagt till UTC klocka.

Mer info på shipssidan

Kommentarer i timeline, och logotyp.

## VAD HAR GÅTT BRA:

Vi har uppskattat uppgifterna bättre och därmed nått vårt mål den här sprinten.

Vi har haft stand-ups, vilket va bra.

Samarbetet har fungerat ypperligt och vi har fått gjort mycket. Många har hjälp till med många olika ställen.

## VAD HAR VARIT DÅLIGT:

Vissa saker fungerar fortfarande inte på Mac vilket har varit ett problem. MEN vi har då delat upp arbetet så att alla har haft någonting att göra.

## VAD VILL VI FÖRÄNDRA:

Vi har inget mer o göra nu.



---

# Appendix C

## Scrum Strategies

Aleksandar Babunovic, Ken Bäcklund, Magnus Rönnberg, Johanna Torbjörnsson, Robin Bengtsson, Olle Persson

### Take priorities into account

In order to meet the clients demands one needs to take the priorities of the user stories in account while choosing assignments. For example, at the Lego exercise the project owner prioritized roads for the city above all else. Despite that, no one took it upon themselves to construct roads to the city. No one even consulted the project owner.

During yesterday's Lego exercise it became apparent that if you use scrum for a project you must always communicate with the project owner to find out the thoughts and needs of the customer. To make a customer happy you not only have to finish the tasks, the customer might also want them in a specific order.

### Find the purpose

If the client has a very specific request, it's important to find the purpose of that request. The client may think they need a very specific thing, but in reality their wishes may come from an underlying need that can be solved in a better way.

During the lego exercise, we got to build a park that had the specification that it should include 28 trees. We had a very hard time building that many trees and fitting them into the park. If we would have asked Håkan why he wanted 28 trees he might have answered that he just wanted the park to look nice, and we could have proposed that 10 trees would be enough to achieve that. That would have saved us a lot of time and stress.

During the project we will implement this technique by communicate with the client and try to find all their underlying needs, and if needed come up with other solutions than what the client came up with themselves.

### Clear Roles

To make our work-hours effective; we will divide (as far as possible) the workload amongst the people in our group. It is important that every person *clearly* knows his/her assignment, and how this assignment is to be joined to the project as a whole. It is our assumption that this will help us work effectively and avoid situations where people are working on the same thing, or the well known "Too many cooks spoil the broth" saying.

During the Lego exercise we divided the workload so that one or two people built the project and the rest gathered the necessary pieces; this made our work very effective.

---

## Appendix D

### Key Performance Indices

#### Burn up

What: A burn up chart is going to be used to measure work progress throughout the project.

Why: We decided to use a burn up chart since the size of the project may change. This makes it difficult to use a burn down chart. A burn up chart is easy to interpret and update. This gives us a quick way to update and measure our work progress. Measuring our progress every sprint gives us a good idea of if something needs to change or if the workflow we put into place is working well.

How: After every sprint we will save our completed story points. This will enable us to create a chart to visualize our progress.

#### Velocity

What: This will measure our velocity between sprints, and our estimated story points each sprint.

Why: We have come to a conclusion that this KPI will be more suitable to measure our estimated story points to the actual points until completion. This also allows us to compare time spent between each sprint.

How: The KPI has two bars at each sprint. The x-axis represents each sprint while the y-axis represents story points. Two bars at each sprint shows the estimated and completed story points.

#### Quality check










What: Each sprint, we will use an appropriate tool to measure the quality and the amount of bugs in the code. We will try to keep the quality on the same level through the entire project.


Why: This kpi is a good compliment to the other two, since it focuses on the code. This kpi will encourage us to value quality over quantity.

How: One option is Code Climate, a tool that measures quality, correctness, security, and style issues among other things. Another option is Findbugs. Code climate is a better option since it offers more features, and hopefully we will get a student account for free.


# Appendix E

## The Business Model Canvas

The Business Model Canvas		Designed for: Grupp Runda Bordet	Designed by: Grupp Runda Bordet	Date: 20170406	Version: 1
<b>Key Partners</b>  <b>PortCDM</b> <b>Andra aktörer i hamnen</b>	<b>Key Activities</b>  <b>Mjukvaruutveckling</b> <b>Support</b> <b>Marknadsundersökning</b>	<b>Value Propositions</b>  <b>Underlätta agentens arbete och kommunikation med andra aktörer i hamnen</b>	<b>Customer Relationships</b>  <b>Plattformssupport</b>	<b>Customer Segments</b>  <b>Skeppsagenter i hamnar som använder portCDM</b>	
	<b>Key Resources</b>  <b>PortCDM</b> <b>Utvecklingsplattform</b>		<b>Channels</b>  <b>Applikation som möter kundens behov</b>		
<b>Cost Structure</b>  <b>Lön till utvecklare</b> <b>Lön till support</b> <b>Serverkostnad</b>		<b>Revenue Streams</b>  <b>Inköpskostnad</b> <b>Månadskostnad för support</b>			



DESIGNED BY: Strategyzer AG  
The masters of Business Model Generation and Strategyzer



strategyzer.com

---

## Appendix F

### Report from Git Inspector

Statistical information for the repository 'SoftwareEngineeringProject' was gathered on 2017/05/30.  
The following historical commit information, by author, was found in the repository:

Author	Commits	Insertions	Deletions	% of changes
Aleksandar Babunovic	10	543	329	5.05
Ello	36	1996	1373	19.50
ElloP	15	3155	1772	28.52
Olle Persson	2	373	766	6.59
Robin	1	230	0	1.33
Robin Bengtsson	13	879	208	6.29
gusbacke	11	194	27	1.28
gusronmae	25	2257	509	16.01
jtorbjornsson	39	2087	578	15.43

Below are the number of rows from each author that have survived and are still intact in the current revision:

Author	Rows	Stability	Age	% in comments
Aleksandar Babunovic	162	29.8	0.9	7.41
Ello	2490	124.7	0.5	12.61
Robin Bengtsson	561	63.8	0.5	4.10
gusbacke	53	27.3	0.3	1.89
gusronmae	930	41.2	0.6	1.29
jtorbjornsson	1294	62.0	0.5	2.24

Aleksandar Babunovic:	Aleksandar Babunovic
Olle Persson:	Ello, ElloP, Olle Persson
Ken Bäcklund:	gusbacke
Magnus Rönnberg:	gusronmae
Robin Bengtsson:	Robin, Robin Bengtsson
Johanna Torbjönsson:	jtorbjornsson