Project Final Report - PFR

October 24, 2018

Contents

Document history

Version	Date	Resp	Description
0.1	2018-10-12	PG	First draft

1 Referenced documents

1. SDP - Software Developement Plan, v. 1.0

2 Historical overview of the project

2.1 Phases

2.1.1 Phase 1

During this phase the documents SDP, SRS and SVVS were produced. This phase ended up requiring two formal meetings instead of one, resulting in the suggested deadline from the SDP being pushed. The suggested deadline was 16/09 and the documents were put in baseline at 02/10.

2.1.2 SDP

The suggested time estimation for the SDP was 10 hours, in table 1 the actual time resulted in 23.75 hours(1425 minutes).

Week	**Documentation**	**Informal Review**	**Formal Review**	**Rework**	**Total time**
1	135	0	0	0	135
2	240	0	0	0	240
3	120	45	60	360	585
4	0	135	60	270	465
Sum	495	180	120	630	1425

Table 1: Weekly overview of SDP. Numbers in minutes

2.1.3 SRS

The SRS was suggested to take 8 hours according to the SDP since it had to be finished before the SVVS could finish. The actual time resulted in 50 hours(3005 minutes) according to table 2.

Week	Documentation	Informal Review	Formal Review	Rework	Total time
1	120	90	0	0	210
2	940	215	0	315	1470
3	240	175	60	180	655
4	120	0	60	360	540
5	0	0	0	60	60
6	0	0	0	15	15
7	0	0	0	55	55
Sum	1420	480	120	985	3005

Table 2: Weekly overview of SRS. Numbers in minutes

2.1.4 SVVS

The SVVS took 28.5 hours(1710 minutes) according to table 3. According to the SPD this document was suggested to take 10 hours.

Week	Documentation	Informal Review	Formal Review	Rework	Total time
1	180	0	0	0	180
2	390	0	0	150	540
3	120	90	60	210	480
4	0	0	60	450	510
Sum	690	90	120	810	1710

Table 3: Weekly overview of SVVS. Numbers in minutes

2.2 Phase 2

In this phase the STLDD and SVVI were to be produced. The phase was set to start 17/09 according to the SPD which it also did. The deadline of the phase was set to 27/09, and the documents was approved for baseline at 9/10.

2.2.1 STLDD

The STLDD was set to take 8 hours in the SDP. In table 4 the actual time resulted in 82.667 hours(4960 minutes).

Week	Documentation	Informal Review	Formal Review	Rework	Total time
3	120	0	0	240	360
4	990	0	0	180	1170
5	965	215	60	270	1510
6	1740	0	60	240	2040
Sum	3815	215	0	930	4960

Table 4: Weekly overview of STLDD. Numbers in minutes

2.2.2 SVVI

The estimated time for the production of the SVVI was 10 hours. From time reports the actual time resulted to 22.5 hours.

Week	Documentation	Informal Review	Formal Review	Rework	Total time
2	60	0	0	0	60
3	180	0	0	0	180
4	300	0	0	0	300
5	300	30	60	360	750
7	0	0	0	120	120
Sum	840	30	0	480	1350

Table 5: Weekly overview of SVVI. Numbers in minutes

2.3 Phase 3

Phase 3 was estimated to be ready to begin at 01/10 which it also was. During this phase the code for the application was developed.

2.3.1 SDDD

In the SDP it is suggested that 14 hours is going to the development of the SDDD. From table 6 the actual time resulted to 148 hours (8930 minutes)

Week	Documentation	Informal Review	Formal Review	Rework	Total time
4	1200	0	0	0	1200
5	3510	0	0	0	3510
6	3740	0	0	0	3740
7	480	0	0	0	480
Sum	8930	0	0	0	8930

Table 6: Weekly overview of SDDD. Numbers in minutes

2.4 Phase 4

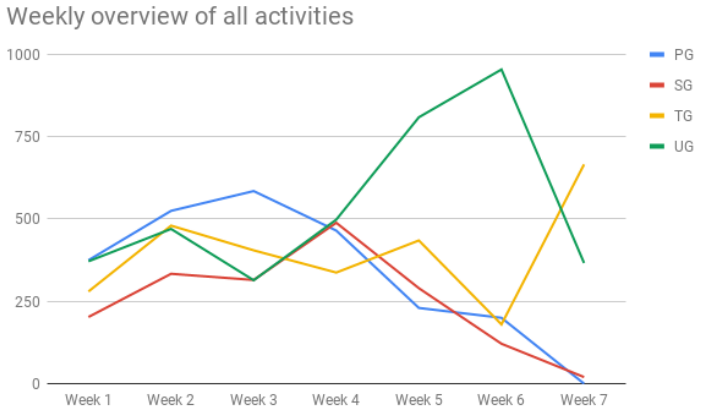
Since phase 4 was initiated week 7 and no work was done during this week on SVVR and SSD resulting in no time reports for these documents.

2.4.1 PFR

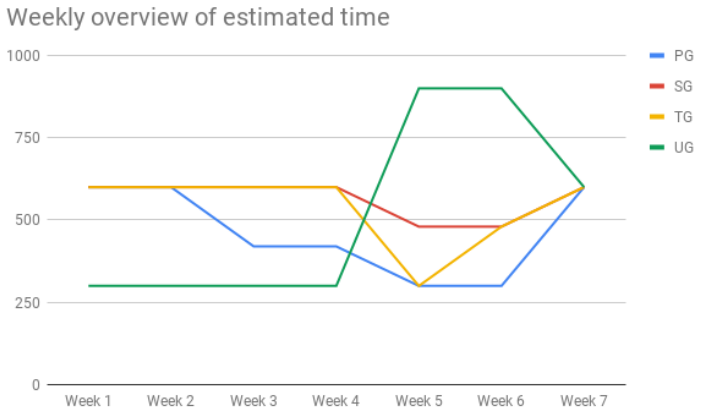
Week	Documentation	Informal Review	Formal Review	Rework	Total time
6	60	0	0	0	60
7	390	0	0	0	390
Sum	450	0	0	0	450

Table 7: Weekly overview of PFR. Numbers in minutes

2.5 Weekly Summaries



(a) Overview of actual time spent on all activities



(b) Overview of estimated time spent on all activities from table 3 in the SDP

Figure 1: Weekly overview of all activities for each group, compared to weekly estimated time for each group. Y-axis is given in minutes.

3 Evaluation of the project

3.1 Evalutation of the data

In all the tables of section 1 the estimated time from the SDP differs much in the way that they are way lower, than the actual time time spent working on the documents. The reason why there is such a huge gap depends on many

factors.

One of these factors is that everyone's work in one document is summed up in the time reports. Meaning that if two people work for 2 hours at the same time in one document, their work will summed and reported as 4 hours. This was something the PG didn't think of when they were estimating the amount of time required for each document. Instead the project leaders estimation is based on how many points this course is worth and some guessing according to the SDP. Table 1 in the SDP suggests an estimation in hours for each document and not the groups summed effort, the SDP rather suggests the effective effort put in on that document. The estimations in table 1 of the SDP can therefore be seen as an average effort of each group member responsible for that document is thought to put in. By multiplying the amount of group members with the estimated time for each document in the SPD, table 8 gives gives a better comparison between the SDPs estimations and the actual time spent.

	SDP	**SRS**	**SVVS**	**STLDD**	**SVVI**	**SDDD**
Total actual time	24	50	29	83	23	148
New estimated time	20	32	40	32	40	84

Table 8: Table comparison of new estimated times. The new estimated times are given by multiplying the amount of group members responsible for that document with the estimated time in table 1 of the SDP. Numbers are given in hours.

The new estimated times still differs much from the actual time spent. Another factor is that informal reviews and formal review meetings are not counted in with the suggested time estimations from the SDP, as it is with the actual work time. In the SDP it was suggested that 1-2 hours each week goes to informal reviews depending on the document being reviewed. However, it does not state how many or who is supposed to do the informal review for the document. During the course meetings it was decided that informal reviews would be done digitally and anyone could do them. Therefor it would not make any sense to add the the times for informal reviews to the estimated times. But by subtracting the time taken for informal and formal reviews of the actual time the following table 9 is created.

	SDP	SRS	SVVS	STLDD	SVVI	SDDD
Total actual time	24	50	29	83	23	148
Informal Review	3	8	1.5	3.5	0.5	0
Formal Review	2	2	2	1	1	0
Actual time without reviews	19	40	25.5	78.5	21.5	148
New Estimated time	20	32	40	32	40	84

Table 9: Table comparison of actual time spent without informal reviews and formal review meetings on documents, and new estimated times. Numbers are given in hours.

There are still differences between the estimated time and actual time spent. Another reason why the numbers differ is the lack of time reports. This reason is especially true for TG as seen in table 10.

	Week 1	Week 2	Week 3	Week 4	Week5	Week 6	Week 7
PG	2/2	2/2	2/2	2/2	2/2	1/2	1/2
SG	4/4	4/4	4/4	4/4	4/4	4/4	4/4
TG	3/4	3/4	3/4	3/4	2/4	1/4	1/4
UG	6/6	6/6	6/6	6/6	6/6	4/6	3/6
Total	15/16	15/16	15/16	15/16	14/16	11/16	9/16

Table 10: Table of time reports per group in each week

In the following sections each group will cover why the data differs for their respective document and more in-depth analysis of what happened, why it took longer than expected or faster.

3.2 PG

3.2.1 Evaluation of time plan

As a part of the Software Development Plan, PG estimated the time each of the project's documents would take to produce. These estimations turned out to be underestimated by a large margin. PG has tried to analyse the reason for this error and several plausible reasons were discovered.

Firstly, the estimations were very rough and no advanced techniques were used. The reason for this was that PG had no prior experience of similar projects and there was only a basic understanding of each document. Additionally, the estimations were mainly based on the time it would take to write the initial version, not including reviews or corrections. There was also a misunderstanding regarding how time would be reported. All of these reasons contributed to the initial estimates being rather poor.

Also, several documents had to be formally reviewed two times. As a result, the time needed for these documents increased greatly. A contributing factor to

why these documents failed their first formal review was poor informal reviews. As a result, a lot of time was spent on these documents.

Another factor which likely affected the actual numbers was the time reporting itself. This was a risk which was identified in the SDP but it is difficult to determine its effect. It seemed that many time reports were estimations done retrospectively, some done much later that preferred. Additionally, some reports were even missing or incomplete. As a result, the numbers presented might not accurately reflect the amount of work that was actually done.

Lastly, the estimates were based on a very unclear view of what the final product would actually be. Since the estimates were done before any requirements or technical details were determined, PG had difficulties in estimating times for phases later in the projects. For example, the time needed for STLDD and SDDD depend heavily on the complexity and amount of requirements

To summarise, the time estimations were generally very inaccurate. Although there were many possible contributing factors, PG believes many of them could be avoided or included in future estimates. As such, time estimates of future projects of similar nature will hopefully be more accurate.

3.2.2 Evaluation of communication

Communication is crucial for the vast majority of projects and this project was no exception. In order to enable good communication, PG set up Slack to be used and weekly meetings to be held. The goal was for team members to easily have access to relevant information and be informed when needed. Although this was a good start, there were some flaws and shortcomings that made communication sub-par.

One problem was that the communications channels themselves were perhaps not ideal or not utilised effectively. Many experienced Slack to be good for sending out information but not a great place to hold discussions or conversations. Some also had problems with receiving notifications from Slack. As such, communication via Slack proved to be troublesome and many switched over to using other alternatives.

The weekly meetings could also have been better. This might have been a result of the meetings not having a clear goal. This was most evident during the first few meetings, since the different groups had not received instructions on what to communicate during the meeting. PG then informed the rest of the group to prepare a short update for each meeting. As a result, the meetings improved slightly. However, the meetings probably had more potential to be a better platform for communication and discussion. Another problem with the weekly meetings was attendance. Due to conflicting schedules and other reasons, there was a problem with trying to find a time where all members could participate.

Another factor that affected communication was the group structure, which was mainly hierarchical. There was a problem with information flowing between the different groups and many experienced that they were missing out on information. Although the weekly meetings were meant to improve communication

within the group, they were perhaps not frequent enough or the right platform to handle some matters. Additionally, there was no clear communication plan so communication was often unorganised or ad-lib.

This project has shown that software engineering is a highly social activity and good communication is key. There are many tools to use to organise and coordinate team members, but in-person meetings are imperative. As the amount of people involved increase, it is also important to create a clear communication plan to make sure information flows correctly.

3.2.3 General Evaluation

The beginning of the project was especially chaotic because it started so abruptly. There were a lot of new concepts the group had to learn and the project had to be organised very quickly. PG was initially keen on looking for alternatives for ePuss but decided to go with ePuss due to time constraints. The project management tool "Monday" was supposed to be used but due to costs it was disregarded. PG felt that more time before the actual project started would result in less initial confusion and a more fleshed out project organisation.

One problem the group experienced was a low amount of informal reviews. PG believes there are two causes for this. The first cause is that there was no responsibility or obligation to do them. The second cause is that the time period for informal reviews was rather short.

Despite all of this, the group was able to produce quality work. The group put in a lot of effort and were able to learn the formal process very quickly. The groups were also very ambitious and highly autonomous.

3.3 SG

3.3.1 Evaluation of SRS

The biggest challenge was to make everyone be a part of making the requirements and scenarios. We had a very slim time-frame and the group members all had different schedules. This meant that we could not have a workshop or bigger discussions like we wanted. Instead a lot of time had to be put into working it out ourselves.

Another reason for it taking so long was that the time required was largely underrated. Making consistent requirements that cover everything took multiple editions, just because of how exhaustive they had to be.

Sometimes it was unclear what was meant with different requirements that the different groups sent in. A lot of overlap was found as well. This together with the last paragraph was mainly because of a lack of communication.

3.3.2 Evaluation of STLDD

Again we had problems with our prediction of the amount of time required. This is because the prediction was made before the SRS was made, and therefore not

a clear picture of all the functionality was there. This meant that there was not enough time.

Because of delays already from the start with the SRS, our plan for how to make the STLDD did not plan out. This proved to be another factor to why it took so much extra time. However as with the SRS the amount of time needed was underrated.

Mainly problems with our time frame led to all other problems arose. We did not have enough time to communicate and have discussions with the different groups. We could not either have meetings in our group and discuss what we had done and improve or correct. There was not a open dialogue with the UG, which meant that we had to make decisions at their behalf.

3.3.3 General evaluation

Lack of time and communication in the group, not enough meetings and such meant that not everything was done in time always. This also resulted in that a lot of work had to be done in the nick of time. There was not always enough time for evaluation either.

The lack of time also contributed to lack of communication with other groups. A lot of things we planned, like workshops and meetings could not be held due to time limits.

3.4 TG

3.4.1 Evaluation of SVVS

Being the first document we created this was probably the hardest. However the combination of the project instruction and example SVVS available there was a pretty clear picture of what was needed to be done. The biggest hurdle TG faced during the creation of this document was simply the deadline. Not that it was too soon but the exact date was too unclear. TG thought the date was set for the first Friday and planned the work accordingly, however TG were told that it wasn't actually until the next week so the work went cold. This was then closely followed up by being Friday evening that the actual deadline was Sunday night which caused some panic and a perhaps rushed document. Once TG started working it went without any big issues. Fixing the unclear deadlines and coordinating with PG would've made this first step a million times easier.

Formal reviews were the biggest help in this step, as it was clear what we had missed and easily fixable.

3.4.2 Evaluation of SVVI

This document went extremely smooth given the circumstances. Once again the biggest issue was the deadline. The test lead got information about a hard deadline very close to the SVVS deadline so work started immediately with the not reviewed version of SVVS and SRS as the references. Once some of the work was completed TG were told by PG there wasn't any deadline at all and

we had to wait for the SRS to be fully finished. During the wait we did not get or seek any information how the SG work was coming along. This is where communication could have helped, because we were once again faced with a hard deadline without much notice.

In the end however the group are very happy with the result. The main unclear thing is the actual amount of time it takes to write this document, which is a lot.

3.4.3 Evaluation of SVVR

Another document which went very smooth. Unclear deadlines were only a slight issue. The largest hurdle this time was the lack of an example document to cross reference, but the group believes they have the correct information added. Testing for this document went easy and work was done fast.

3.4.4 General Evaluation

TG are in general very happy with their work, given the many unclear circumstances. It was not clear in the beginning when and what work was to be made, which wasn't helped with a big lack of meetings with both PG and SG but mainly within TG. Once testing actually began and documents were to be written work got done effectively but detailed.

The main issue to take away is to begin sooner. For example: a Jenkins solution was created to handle regression testing and to aid the developers, but this solution was never implemented. This was mainly due to time constraints at the end of the project. The solution wasn't set-up until the second to last week and TG wasn't ready to learn groovy and Jenkins pipelines in such a short time, especially with the need to create all the necessary scripts to run a pipeline for each developer group.

3.5 UG

The work that the Software Developers (here called UG) were responsible for worked well overall, however each group will present challenges and solutions discovered in the sections below.

3.5.1 Front-end

The responsibility for the front-end group was to develop the part of the system that the user interacts with. That is, the website that is displayed when connecting to localhost after starting the server. During the meeting before STLDD was to be written, several design ideas were discussed within the front-end group and with the front-end contact from the system architects. However, these design ideas were only spoken at that meeting, there were no traces of it in the STLDD and nothing was changed in the SRS even though we discussed what would have to be changed. This led to great confusion during the development due to us not knowing if we should follow the design specified at the

meeting, if we should follow the SRS or if we should do something in between and submit problem reports for everything that was different.

The total time spent for the front-end group far exceeded all expectations. One of the reasons for this is that half of the group had no experience with front-end development before which led to that person not being able to contribute very much. Furthermore, there were way too many of the requirements in the SRS that were front-end centered due to there being so many error checks in all scenarios leading to us spending a tremendous amount of time during the development.

Moreover, back-end had way too much to do, due to one of the members there having no experience with SQL, leading to us having to help them. Lastly, it also caused problems that the resource-methods provided by back-end and the algorithm-group that were supposed to just be called in front-end did not work properly, leading to front-end having to solve the issue in most cases due to the fact that we were the ones that were using the methods and therefore were most familiar with what was going on even thought it was not related to front-end.

3.5.2 Back-end

The back-end developers' main responsibility was the database and the functionality related to data management. Since the database is written in SQL, most of the work have focused on implementing SQL-functionality. Since both the front-end and the algorithm solutions depend on the database, a lot of work have been done to tie the different parts together with various data access classes. The line between back-end and algorithm is quite vague, therefore it was decided that the two groups should merge to one group, but both groups still kept their main responsibility. This gave a streamlining effect on the development because before this decision was made there was a lot of discussion about which group should do which classes. When there was only "one group", everybody was responsible for all the classes and there was no question marks about what classes either group could implement.

The total amount of hours put in to the SDDD exceeded expectation. The main reason for this was, just like in the front-end case, just one of the developers had previous experience in SQL. This meant there was basically only one developer that could effectively implement while the other had to simultaneously learn SQL while implementing it. This slowed down the process a lot. This would have been hard to avoid without having two developers in the back-end group comfortable with SQL, but on the other hand now a developer got to learn SQL from a more experienced programmer. Another thing that not necessarily slowed down the work process, but drove up the hours spent on the project was the fact that there were some classes that were very central in the program and had to be implemented first. This led to some bottlenecks in the developing process. So even if the entire UG was gathered and ready for implementing they had to wait for one developer finish and pushing the code before the they could continue their work.

The communication experience for the back-end is really well described by the algorithm group in the next section.

3.5.3 Algorithm

The main responsibility of the Algorithm Group (hereafter called AG), consisting of two people, was to develop the algorithm for matching a User searching for a Ride with different Rides. This algorithm had to comply to the requirements specified in the SRS.

The total amount of hours that AG put on actually developing the algorithm for matching a User with Rides was less than the expected time. This, however, does not show in the diagram in figure 1a since only the total time of the whole Software Development group shows. The main reason for this is that AG had a meeting early on with one of the System Architects (SG), where the task was specified, put in relation to the rest of the back-end code as well as examples of rather specific solutions to the problem - such as the use of merge-sort in the sorting algorithm. Alternative solutions were discarded early on, which made for a clear, straight forward path for AG to follow. This also made it possible for AG to start sketching on this code before the development phase, resulting in less time spent on developing the algorithms during that phase.

When the actual development phase was initialised, one of the things that aided the coding was the UML diagram that the System Architects had made. This provided all the developers with a structure to build on, even though some things were altered, and made discussing solutions between the development groups easier. To simplify the communication between the developers when it came to what code had actually been written, an informal use of Trello was initialised. Trello is an online dashboard used in Scrum-like projects, with one backlog (tasks to be done), one doing-log and one done-log. This also enabled developers to choose what code they wanted to do, which was to much aid when AG helped the back-end developers with their tasks.

During the times most of the code was developed, all developers sat together in a room and coded. This helped discussions and problem solving, as everyone involved was present. Since it also seemed like the majority of the developers had the same level of ambition regarding the project and entire course, it was not difficult to find time to develop nor communicate with the other developers when they were not physically present. This work ethic and easy communication was what was crucial when addressing the fact that the time for development was changed from three to (practically) less than two weeks, but the developers still made the original deadline.

4 Suggestions for improvement and procedures

During the project each group has encountered different problems and difficulties as well as things that have worked out well. The groups have their own suggestions of how to improve on the negative aspects and how to further

strengthen the parts of the project that went well.

4.1 PG

One of the responsibilities the PG had was to divide the other group members into roles. This was done by mailing out a form to all members in the project group(except PG). In the form the members rated the different roles on a scale of 1-5 with 5 being that they are very interested with working in that role. They could also rate how much they wanted a leader responsibility. PG did the best they could to divide everyone into a role that they were interested in. However, even if they got a role they were interested in, they may have lacked any previous experience working with it and this led to some complications within UG. A suggestion of improvement for this is to include a field within that form, that requires the user to rate how much previous experience this person has within that role. This can ensure that the groups has more members with experience of that role.

Based on the difficulties mentioned in the evaluation, there are several improvements that could have been implemented.

- Designate more time designated for informal reviews and have some people individually responsible for a review. This would encourage more informal reviews of higher quality.
- Define a clear communication plan that includes weekly updates and plans from each group. Also include more structured meetings with the whole group and between specific groups.
- Utilising the structure in a better way in regards with communication and responsibility.

Having two members in PG helped greatly in most regards. To further strengthen this, PG could perhaps define what their individual responsibilities were. There was a lot of parts of the project so having distinct work areas would mean that PG could specialise in some areas.

4.2 SG

Make sure to communicate a lot within the project group. However mainly make sure that you plan for enough time so that you can have meetings and discuss the documents regularly. Sometimes it felt like the other groups waited for information we did not know they wanted.

Make sure to get a grip on how experienced the developers are with e.g. JavaScript and SQL. This to not have to change or remove requirements because they were too complicated to implement. This is connected to communication as well.

More clear planning, and meeting in person more often then using messaging as communication. Make sure to set deadlines to plan for evaluation and

correction meetings as well and add consequences if someone does not do their part.

In the time frame we had and with everyone's different schedules this is not as trivial as it sounds. However more meetings with the entire group is another suggestion. And to have a set goal with the meetings, e.g. discuss requirements or set a time frame.

4.3 TG

Have as many meeting as in necessary so the whole group knows each other and the work you should both do together and individually as soon as possible. Also make sure during these meetings everyone knows how the workload is set up with intense periods versus slow periods and make sure to plan ahead.

Start early with big projects such as Jenkins as there are many many parts to get that moving and being a proactive part of the development process. Also the SVVI should be started early as there are a ton of small parts which all need to be completed.

Communicate as much as possible, not only a few days before deadlines. Make sure everyone is on the same page at all times. Try to talk with SG and UG as much as possible to get a grasp of their needs instead of just looking at documents.

4.4 UG

4.4.1 Front-end

After a project such as this one, many suggestions for improvement come to mind for all of the project groups. For the front-end one main improvement would be communicating more with the system architects. During the time of the project there was really only one meeting between the front-end developers and system architects. Since very little of the ideas that the meeting brought were actually implemented in documents such as the SRS or the STLDD, the feeling afterwards was that the meeting was basically meaningless. Several more meetings with the system architects and a dialogue during the process of changing the SRS, and other documents, would have been of great help.

The fact that the total time spent was so great could have been avoided by considering a couple of things. First of all, the number of requirements could have been cut down and the difficulty of those requirements could have been chosen more suitable to the level of experience in the front-end group. This could have been achieved, again, with better communication and understanding between the people creating the requirements and the front-end group. Secondly, the fact that the deadlines for the SRS was pushed back so many times it shortened the deadline for the developers. The solution to this is improved planning of deadlines and actually sticking to the deadlines. Some parts of the SRS could have been prioritised in order to allow the developers to start coding earlier instead of having to do all the work during a very short period of time.

4.4.2 Back-end

The biggest problem with the entire process was without doubt the communication. This was also bound to be the problem given the size of the group. The information was distributed through Slack, which is a messaging service on a website. The main problem was that almost all information was distributed to all the different groups, which led to the back-end group not having many responsibilities before phase 3. This eventually led back-end no longer reading the messages. When it finally was time for the back-end group to be active, a lot of the messages were missed since the information load was so high. There was a solution for this by having different Slack messaging groups, but it wasn't consistently used.

It was also planned that the back-end group would have some kind of system architect to guide us or explain how the program should work, but there was only one very short meeting that didn't result in anything productive. The back-end group had to try to figure out how the back-end part of the system should be implemented by reading the SRS and STLDD/UML documents. Some guiding in the beginning of the process would probably have made the working process easier and a more lot fluent.

Another improvement that could have been done, is giving specific people more responsibilities. For example when the internal reviews in different documentations were done, the task was described in a way that "somebody" should do it. Since there were 16 people in our group, everybody thought that someone else would do it, which resulted in just a few people actually doing it. If everybody was responsible for the internal reviews one week for example, it would have probably worked better.

One other thing that occurred during the process is that there was a lack of responsibility. Since the back-end group wasn't heavily involved in the work the first 5 weeks of the project and then suddenly had to work a lot, the feeling that we cared about having a good final product never really existed. This might just have been a lack of character in the back-end group. Anyway, a solution to this problem could have been solved by involving the back-end group more in the earlier stage of the project. This might also be one of the reasons that companies tend to change from this type of waterfall method programming to a more agile type of programming where everybody is more active in all stages of development.

4.4.3 Algorithm

Despite all the positive comments on the development in section 3.5.3, there were some things that AG felt could be improved. One of these things were that a plan for redistributing resources (i.e. developers) could have been made before the development phase begun, by the System Architects. For example, AG were done with their code quite quickly, however they felt they were not enough prepared for helping for example back-end in an efficient way. Had AG formally been instructed to prepare for helping front-end and/or back-end

(maybe one developer per other group), the developing could have worked even better.

Another thing that should be noted for improvement was the communication between the Software Developers (UG), System Architects (SG) and Project Leaders (PG). During the entirety of the project, AG had to ask for information to get it, rather than be informed when decisions were made. This lead to a lot of confusion the first weeks, since AG did not know in which direction the project was headed, nor the status of the other groups and their tasks. This lead to some frustration within AG. After a couple of weeks the status of the other groups were easier to grasp, since the PG started each group meeting with a status update with each group. However, out of all group meetings, the entire group was only gathered once. In general, the attendance of meetings was low, which gave an overall unprofessional impression of those not attending.

The group meetings contained mostly of updating status between the groups (once that had been initialised), and a lot of decision making happened on Slack after each meeting. This lead to an unclear and poor communication, which then showed since several deadlines had to be pushed. The meetings and the communication in the group could have been easily improved had decisions been made before group meetings, communicated during meetings and reinforced through Slack afterwards, instead of group meetings only consisting of information that decisions were to be made after the meeting and communicated in Slack. Because of this, basically all communication was made through Slack, which did not even function properly for all group members since no notifications were sent when someone posted information. It also led to each group meeting lasting a maximum of 15 minutes, once each week, where the lecturer in the course clearly stated that two meetings or more per week would be recommended.

As earlier mentioned, AG had a meeting early on with one of the System Architects, however it was held on the initiative of AG. It seems like the other developer groups did not have a meeting with their respective System Architects, which resulted in some conflicting ideas on program design when the whole group of developers first met. This was easily solved, however it would have been a shorter starting distance if all developers started on the same page.