Smart electricity

Project report: IT2901

Spring 2014

Beate Baier Biribakken
Tor-Håkon Bonsaksen
Lars Erik Græsdal-Knutrud
Per Øyvind Kanestrøm
Håvard Holmboe Lian
Pia Karlsen Lindkjølen

Contents

1 Introduction

the problem we're going to solve, as-is and to-be

1.1 About the assignment

Specification

1.2 About the customer

2 Project schedule

2.1 Team organization

In order to exploit the team's resources in the best way possible, all the team members has summed up their background knowledge and which role they wish to have in the project. The team has organized its structure the based on this information.

The team members

The team consists of six persons, all in their final year of a bachelor in Informatics. All the members have previous experience on working in teams on educational projects.

Beate Baier Biribakken

Beate have previously worked at Student-Media AS and Sportradar AS as web developer. From these experiences, she gained knowledge about Linux and web development, such as PHP, JavaScript, HTML and CSS. She also has some experience with Scrum, Java and Android.

Tor-Håkon Bonsaksen

Tor-Håkon has a trade certificate in data electronics and broad experience with web development through extracurricular activities within the groups dotKom and Casual gaming. This includes knowledge about Python, Django, HTML, CSS and JavaScript. In addition, he works with Android development in his spare time. He also has some experience with Java through school.

Lars Erik Græsdal-Knutrud

Lars Erik has experience with Java, C#, C++ and SQL through his ongoing education. As the internal systems developer for Orakeltjenesten Dragvoll he also has some experience with PHP and server environments.

Per Øyvind Kanestrøm

Per Øyvind is an GNU/ Linux user. He is currently working on a web project in PHP/ Symphony2 and an app project on the Android platform. From school he has experience in Java, Python, Scrum and MYSQL.

Håvard Holmboe Lian

Håvard has experience with Java, Python, SQL, C, C#, and VHDL from school. He has also worked with writing C code for embedded systems with and without an OS (Linux).

Pia Karlsen Lindkjølen

Pia has some experience with Java, Python, MYSQL and scrum from school projects. She also have some experience with project management.

2.2 Scrum

The assignment at hand will demand innovation and frequent changes underways. What the group needs is an agile development framework to follow. First and foremost it must also be a development framework that is know to the group so no time will be spent trying to learn a new process. The standard choice for this is to follow the Scrum model. All the members of the group have previous experience with this from the course IT1901.

The Scrum model is made as an iterative and incremental agile software development framework. The main principle is that a small group is focusing on reaching a common goal.

The scrum process consists of intervals of sprints that are 1 to 4 weeks long. Each sprint has three important parts. The first part is the sprint planning meeting, then begins the daily scrum meetings and the process is ended with a so called end meeting.

Sprint planning

Then the sprint planning begins. The objective of the sprint planning is to find out what work to be done and prepare a sprint backlog that consists of the tasks to be done and how much time the team thinks it will take to complete it.

Daily meetings

End meeting

that consists of a sprint review and retrospective meeting for the last sprint. Thus progress is reviewed and accumulated lessons from the sprint can be taken in account for the next sprint.

Our process

We have chosen to have sprint that are two weeks long. Having daily meetings is not possible with our schedule, since this course demands 20 hours each week and the team members is taking different courses besides this one. Thus we have ended up with having two meetings each week that begins with a 'daily meeting'.

We have also decided to use a Scrum tool to help us in the process. It will give us automatic sprint backlogs and time measurements for each group, and graphs. Lots of graphs.

Out team has previously had experience with scrum tools like icescrum. This experience was not a jolly one. Thus a search for a better tool was done. The tool of choice ended up being Yodiz.

2.3 Main responsibilities

Role	Description
Project leader: Pia	Keeping team updated and monitor the project's status
Deputy project leader: Lars Erik	Fill in whenever the project leader is incapable to perform all duties
Scrum-master: Per Øyvind	Make sure that Scrum-process goes as smooth as possible, that the team provides necessary documentation and keep track of the project process
Customer relations: Pia	All customer communication mainly goes through this person.
Development: Tor-Håkon	Keep track of the technological development progress, make sure it is going by schedule and take necessary action
Report: Beate	Monitor the report's progress, spell check and review content.
Testing: Håvard	Make sure that the code is properly tested during the development process to detect possible errors, deficiencies and bugs and take the necessary action
Secretary: circulates	Take note of important information during meetings within the team and with the customer.

2.4 Planning

Time schedule

The assignment and team members were assigned January 20. The initial meeting with the customer took place January 24, where the assignment was presented and discussed. Oral presentation of the project will be performed March 19 and the final deadline for submission of the project is set to May 30. The deadline for new specification that would result in major changes to the software is set to March 21., due to time restrictions.

Available hours

Each sprint has a duration of two weeks, excluding the period from April 4. to April 21., when the entire group leaves for a school field trip to China.

Table 2.1 lists the project's available hours and is based on that all team members spends twenty hours on the project every week.

1darkgraylightgray

Sprint #	**Dates**	**Days**	**Hours**
Sprint 1	January 27. - February 7.	10	240
Sprint 2	February 10. - March 21.	10	240
Sprint 3	March 24. - April 4.	10	240
Sprint 4	April 21. - May 2.	10	240
Sprint 5	May 5. - May 16.	10	240
Sprint 6	May 19. - May 30.	10	240
Total		**60**	**1440**

Table 2.1: Available hours

Milestones

1darkgraylightgray

Milestone #	**Description**	**Deadlines**
Milestone 1	Project report - preliminary version	February 9.
Milestone 2	Project report - mid-semester version	March 16.
Milestone 3	Peer evaluation	March 23.
Milestone 4	Project report - final version	May 30.

Table 2.2: Milestones

2.5 Development method

Will you work iteratively and/or incrementally, will you make a mock-up or prototype; phases and iteration

Risk analysis

As part of our project planning, we have outlined potential risks to the progress of our work. A risk is defined as an unwanted event that affects the process. We acknowledge the possibility of challenges and problems arising during the work, such as technical problems, human error or personal problems. These are risks for both individuals and the group as a whole. Effects of these problems include delays, conflict and anything that will slow the progress, and ultimately lead to failure to meet deadlines. One of the most prevalent and dangerous risks is loss of motivation from one or more members of the group. Our assessment of the risks to this project is included in attachment E. The risk elements are sorted by their importance. Importance is calculated with two factors in mind; the calculated probability of the event actually occurring and the effect the event will have on the project. In the table given below, we analyze the elements we consider risks to our project. Likelihood (L) and effect (E) is measured on a scale from 1 to 9. For likelihood, a 9 is very likely and a 1 is very unlikely. For effect, a 9 is devastating and a 1 is not much effect. The table is sorted from high to low importance (I). Importance is the product of probability and effect.

Description	L	E	I	Preventive action	Remedial action
Customer requirements exceeds predicted amount of time				Continiously revise the workload and prioritize	
Underestimation of workload				Continiously revise workload and how much time is left, and prioritize	Reestimate continiously
Data loss				Use version control system	Restore data from previous versions
Conflict in group				Have in mind that people often misunderstand each other and have an open dialog to solve problem.	Contact supervisor for help.
Communication failure				Make e-mail-list and exchange contact information. Be explicit.	Check e-mail and phones multiple times a day.
Unbalanced workload				Coordinate with entire team and log hours	Reallocate tasks
Issues with software or tools				Choose software the team members are familiar with	Hold workshops and view tutorials
Eclipse and IntelliJ is imcompatible					
External services unavailable					
Team member unavailable				Good infrastructure for communication and progress reports	Keep in touch with unavailable team member or redistribute tasks
Team member drops out				Keep track of the effort and the hour's log public	Contact supervisor and redistribute tasks
Illness				Multiple team members work on the same task	Allocate sick member's task to remaining members
Supervisor unavailable				Keep regular contact with supervisor	Discuss problem internally and contact department
Customer unavailable				Keep regular contact with customer	Discuss problem within team and contact supervisor

Architecture

3 Research

3.1 Project management

4 Development environment

- tools for programming, testing, versioning,

4.1 Development tools and libraries

The team members have decided to use the integrated development environments Eclipse and IntelliJ, which is well integrated with Android.

All the team members already had experience with Eclipse, but some of us also wanted to try out IntelliJ, as it appeared to have more features, such as a faster compiler, better search function and autocompletion of code, than Eclipse. However, the remaining part of the team considered the learning curve to switch to IntelliJ to be greater than the advantages and therefore chose to stick to Eclipse.

4.2 Project management and versioning

GitHub

Code convension

Java and Android's code convensions.

4.3 Documentation

Google Drive, LaTeX, JavaDoc

4.4 Communication

Google groups e-mail

4.5 Testing

4.6 Documentation

documenting, archiving,

4.7 Communication

Within the team

With the customer

4.8 Requirements

main requirements. Must be sorted out with customer.

4.9 Alternative solutions

check if there already exist solutions for our problem and make evaluation of them. Evaluation criterias should be discussed with customer.