Rapport 1

Gruppenummer: 35

Kursus 02161
Software Engineering

Lukas Villumsen (s144451)
Sebastian Nyholm (s144434)

13. marts 2015

Indhold

1 Introduction

For a company to be properly able to use its resources it needs to have a plan. By letting the leaders know who is available at which time frames, we make sure that there can be no mistakes between the leaders and the other employees. Each of the employees have their own schedule to keep, and they can always look it up. Leaders will be able to modify the employees schedule by adding new tasks. Employees should also have the liberty to provide the system with the amount of time spent working.
We accomplished this by making one big schedule, containing all events in terms of projects and tasks. The schedule is modified by all project leaders who individually add single projects, and various tasks within the project. Thus, the schedule is updated whenever a new project or task is created, deleted or modified.
All sorts of *absence* is considered the same. Absence includes vacation, illnesses, courses, ect. Absence is too considered a project to which employees are assigned whenever appropriate. When an employee is assigned this project, all other activities are suspended, and resumed once the employee returns.

2 Glossary

Employee: A person working in the company Softwarehuset A/S. An employee can be added to an existing project, scheduled to perform sub-task.

Project: Main scheduled objects, divided into sub-tasks which are to be carried out and solved by the employees connected to them.

Project leader: An employee that has been assigned leader of the project. The project leader is in charge of managing employees and their time, working on certain tasks.

Schedule: Each of the employees has an individual schedule to keep, where he/she is appointed to specific tasks under a project.

Add tasks to a project: The project leader can add tasks to a project.

Remove tasks from a project: The project leader can remove tasks from a project.

Active project: A project which end time has not been exceeded.

Task: A part of a project. Contains expected time and employees working on it.

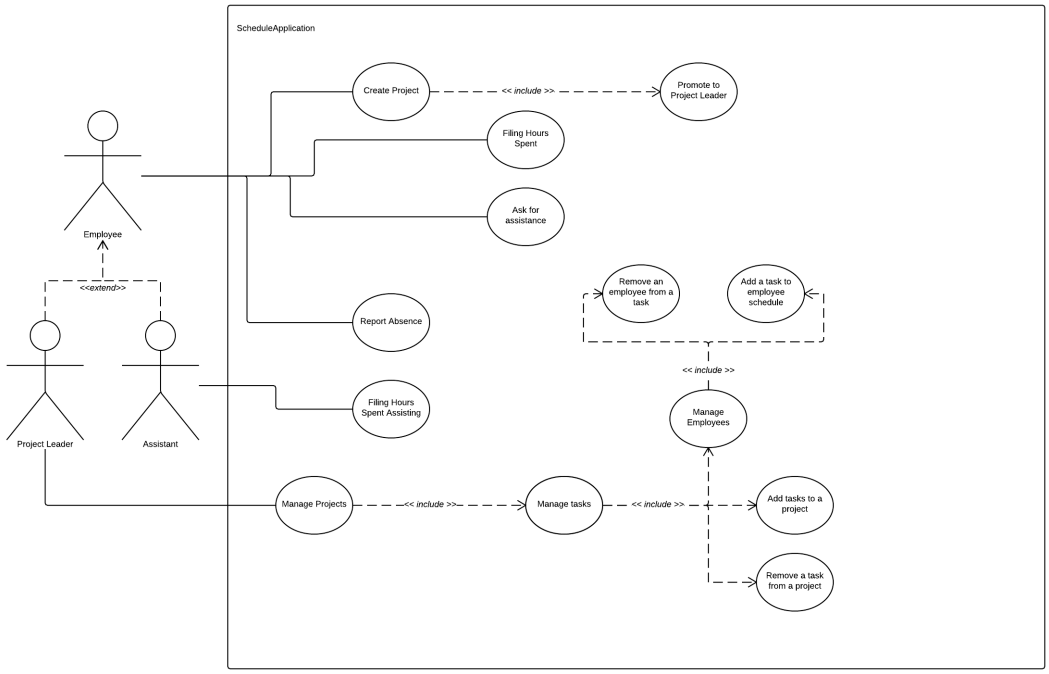
Add an employee to work on a task: The project leader can add employees to a task with a given time.

Remove an employee from a task: The project leader can remove employees from a task.

Add time spent on a task: Employees connected to a specific task can add the time they spent on it.

Absence: Absence is a unique *project* which employees are assigned whenever not present during working hours.

3 Use Case Diagram



Figur 1: Use case diagram

4 Detailed Use Case

4.1 Create project

Name: Create a new project
Description: An employee creates a project and is assigned leader of the project.
Actor: Employee
Main Scenario:

1. A chosen employee creates a new project.
2. All needed information in the project form gets filled in.
3. The project is created.
4. The chosen employee automatically becomes leader of the project.

Alternative Scenario:

a1 A chosen employee creates a new project.

a2 All needed information in form of the project is filled in.

a3 There is already an active project with that name.

a4 The project is not created and no leader is being assigned.

4.2 Add Task

Name: Add a task to a project
Description: A project leader adds a task for a project.
Actor: Employee
Precondition:

§ The project leader has found the project.

§ The employee must be assigned leader of the project in order to add tasks to that project.

§ The project must be active.

Main Scenario:

1. The project leader adds a task to an assigned project by filling in the needed information.
2. The leader assigns employees to the task.
3. The task can now be seen in the assigned employees' schedule.

Alternative Scenario:

a1 The project leader fails to provide with the information required to create a new task.

a2 The task is not created and notifies the leader of this.

4.3 Set time

Name: Time spent working.
Description: An employee files the amount of time spent working on a given task.
Actor: Employee
Precondition:

§ The employee is assigned to task.

Main Scenario:

1. The employee files the time spent working on the given task.
2. The task adds the time filed to the specific employee.
3. The task adds the time filed to its total time.

Alternative Scenario:

a1 The employee files more time than the estimated time.

a2 The task notifies the user of this and asks for confirmation

a3 The employee corrects it.

a4 The task updates.

4.4 Absence

Name: An employee is absent
Description: An employee asks for vacation.
Actor: Employee
Precondition:

§ The request for vacation has been approved.

Main Scenario:

1. The project leader of the 'absence' project has been notified of the coming absence.
2. The project leader adds the employee to the project.
3. Suspending the employee from all other activities.

Alternative Scenario:

a1 The project leader of the 'absence' project has been notified of the coming absence.

a2 The project leader adds the employee to the project.

a3 Suspending the employee from all other activities.

a4 The employee cancels his vacation.

a5 The employee is removed from the absence project.

a6 Resuming all of the employee's activities.

4.5 Get Schedule

Name: Get an employees schedule
Description: A project leader retrieves the schedule of an employee to determine his/her availability for a project
Actor: Project leader
Precondition:

§ The project in focus is active.

Main Scenario:

1. The project leader needs to add employees to a project.
2. The project leader checks the schedule of a specific employee.
3. The employee is not working on a another project in the specified period.
4. The project leader adds the employee to his/her project for the period

Alternative Scenario:

a1 The project leader needs to add employees to a project.

a2 The project leader checks the schedule of a specific employee.

a3 The employee is busy working on a another project in the specified period.

a4 The project leader cannot add the employee to his/her project for this period.

5 User Stories

The above use cases converted to user stories.

5.1 LogIn

"As an employee I would like to log into the system, when I arrive at work."

5.2 CreateProject

"As an employee I would like to create a new project and be assigned as leader of it."

5.3 DeleteProject

"As a project leader I would like to delete a project, that I accidentally created."

5.4 AddTask

"As a project leader I would like to add a task to a project, if it exists."

5.5 CheckSchedule

"As an employee, I would like to check my schedule, to see what I should work on in the coming week."

5.6 CheckTotalSchedule

"As an project leader, I would like to check the schedule for all employees, so I can find employees for a task in a specific week."

5.7 AddEmployeeToTask

"As a project leader I would like to add multiple employees, that is free to work, to a task."

5.8 SetTime

"As an employee working on a task, I would like to update the time that I have worked on the task."

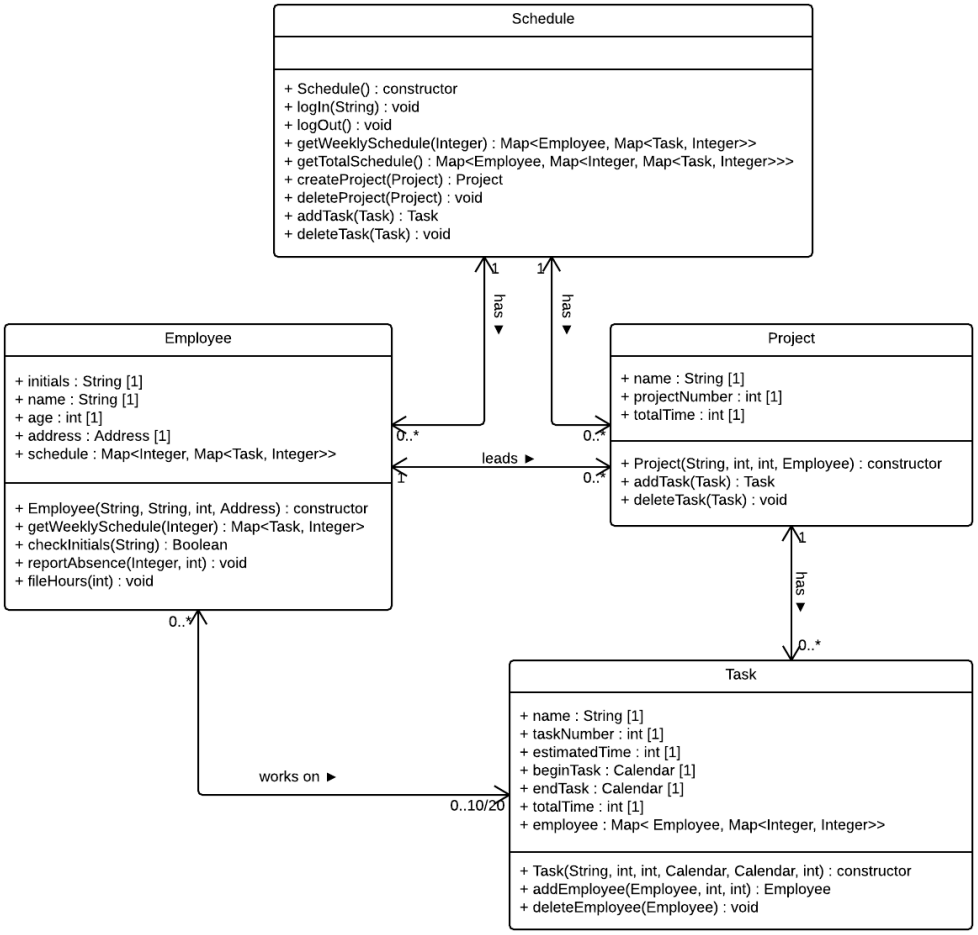
5.9 ReportAbsence

"I, an employee, feel sick this morning and would like to let my project leaders know."

5.10 LogOut

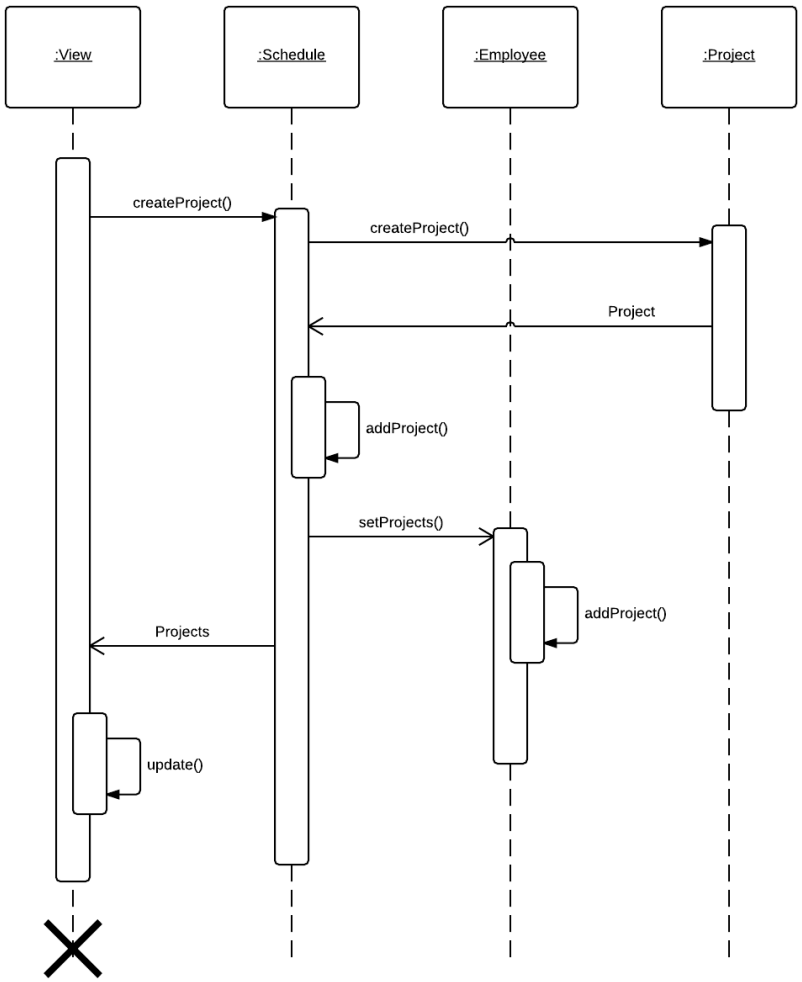
"As an employee I would like to log out when I am done for the day."

6 Class Diagram



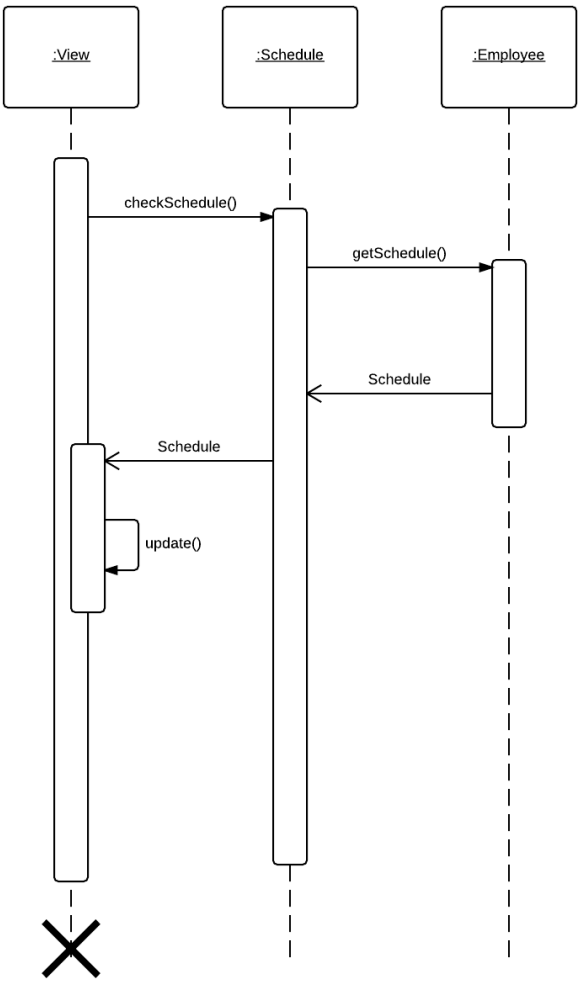
Figur 2: Class Diagram

7 Sequence Diagram - Create Project



Figur 3: Sequence Diagram - Create Project

8 Sequence Diagram - Check Schedule



Figur 4: Sequence Diagram - Check Schedule

9 Discussion

Our plans with this project, is to meet all the customers requirements, but since the time is limited, we will take the most important at first and then add the less important after. This method is also called agile software development.
The way we have designed our software so far, is by an architecture where the class Schedule is our main model, which can be seen on the *class diagram*. We have to go through this class to interact and/or manipulate any underlying objects. We chose this architecture because it gives a structured data setup, where many of the less important classes are not connected to the core elements of the system. This eases the maintenance and future implementations.