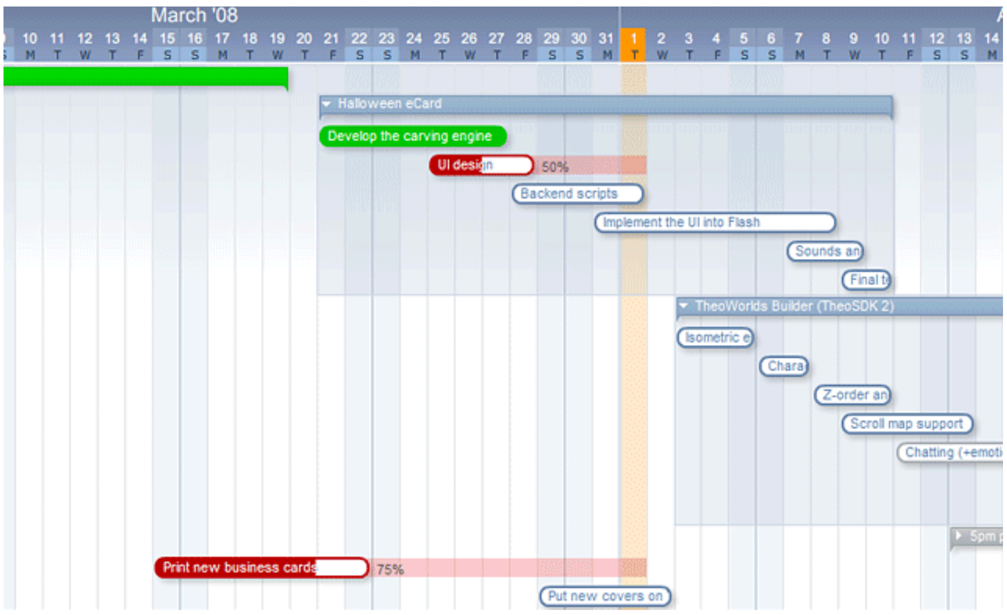
Linnéuniversitetet
Kalmar Växjö

Project Course in Computer Science, 1DV508

Project Specification

The goal of the project is to develop a Timeline Manager application in Java. The application shall be a stand-alone program that is started from an executable JAR-file. The project group is required to use GitHub for source code management. An example of a timeline manager can be seen in the figure below.



Requirements

The application shall have two major modes: Create mode and Display Mode. The requirements for each mode as well as some general requirements are listed below.

Create mode requirements:

- The user shall be able to add multiple (more than one) timelines with separate start- and end time.

Dr. Johan Hagelbäck
johan.hagelback@lnu.se
Dept. of Computer Science

- The user shall be able to add any number of events to a timeline.
- There are two types of events: events with duration and events without duration.
- When adding an event with duration the user shall specify a title, a descriptive text, and start- and end time.
- When adding an event without duration the user shall specify a title, a descriptive text, and a time.
- The user shall be able to delete a complete timeline or an event on a timeline.
- The user shall be able to modify the title, descriptive text or time(s) for an event.

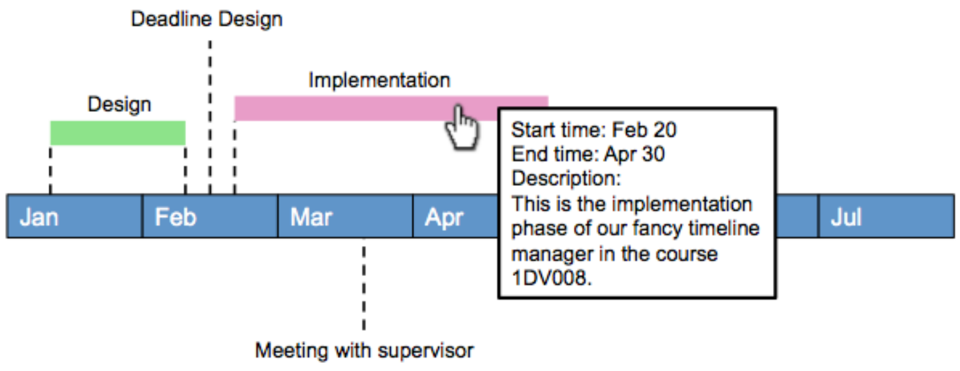
Display mode requirements:

- In display mode all timelines and all events on each timeline shall be shown.
- Events shall be placed at the correct time in the timeline.
- The title shall be shown for all events.
- For events with duration, the time span shall be shown with a bar.
- The user shall be able to click on an event to open a popup window showing title, descriptive text and time(s) for the event.
- The user shall be able to scroll horizontally and/or vertically if all timelines cannot fit in the display window.

General requirements:

- The user shall be able to save and load timelines.
- The data shall be stored in a text file or SQL database (optional).

The figure below shows an example of how the display mode can look like.

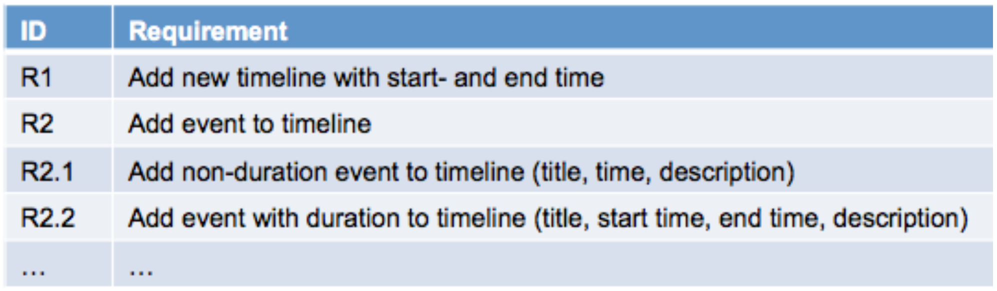


Dr. Johan Hagelbäck
johan.hagelback@lnu.se
Dept. of Computer Science

Documentation

The following documents shall be produced during the project:

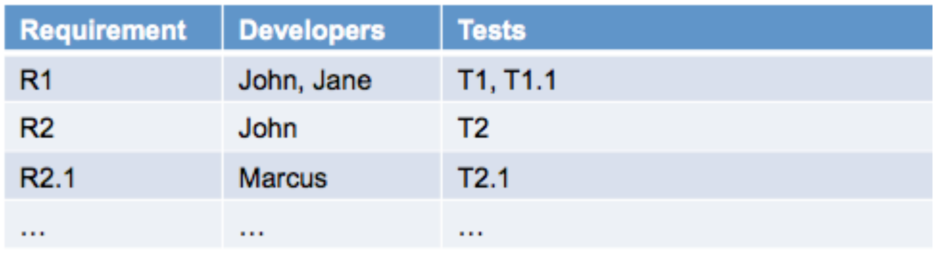
1. Analysis document with the requirements for the application. Each requirement shall have a description and an assigned unique ID:



ID	Requirement
R1	Add new timeline with start- and end time
R2	Add event to timeline
R2.1	Add non-duration event to timeline (title, time, description)
R2.2	Add event with duration to timeline (title, start time, end time, description)
…	…

2. A high-level design document with modules and classes, and how they are connected. The design document shall be updated during the development phase of the project to reflect design changes.

3. An implementation plan for each implementation delivery. The plan shall contain which requirements to implement in this delivery, who shall implement them and which tests to run:

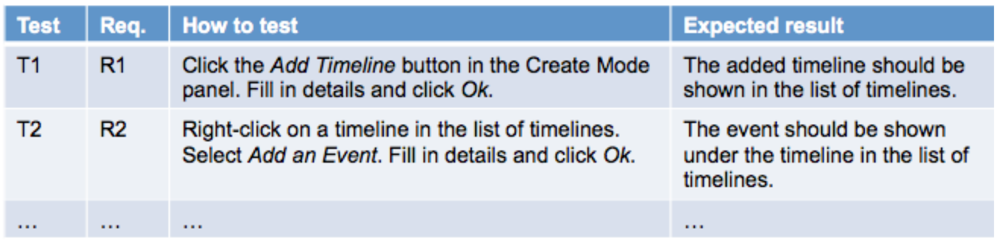


Requirement	Developers	Tests
R1	John, Jane	T1, T1.1
R2	John	T2
R2.1	Marcus	T2.1
…	…	…

4. A detailed design document with UML class diagrams for each implementation delivery.

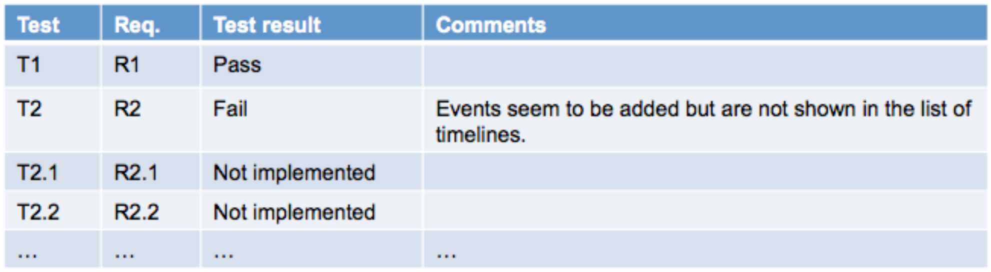
Dr. Johan Hagelbäck
johan.hagelback@lnu.se
Dept. of Computer Science

5. A test case document with test cases for each implementation delivery. Each test shall have an ID number, which requirement is tested, how to test it and expected result:



Test	Req.	How to test	Expected result
T1	R1	Click the *Add Timeline* button in the Create Mode panel. Fill in details and click *Ok*.	The added timeline should be shown in the list of timelines.
T2	R2	Right-click on a timeline in the list of timelines. Select *Add an Event*. Fill in details and click *Ok*.	The event should be shown under the timeline in the list of timelines.
…	…	…	…

6. A test result document for each implementation delivery where the test results are documented. An example is shown below.



Test	Req.	Test result	Comments
T1	R1	Pass	
T2	R2	Fail	Events seem to be added but are not shown in the list of timelines.
T2.1	R2.1	Not implemented	
T2.2	R2.2	Not implemented	
…	…	…	…

7. A delivery test case document including tests that cover all requirements for the whole system.

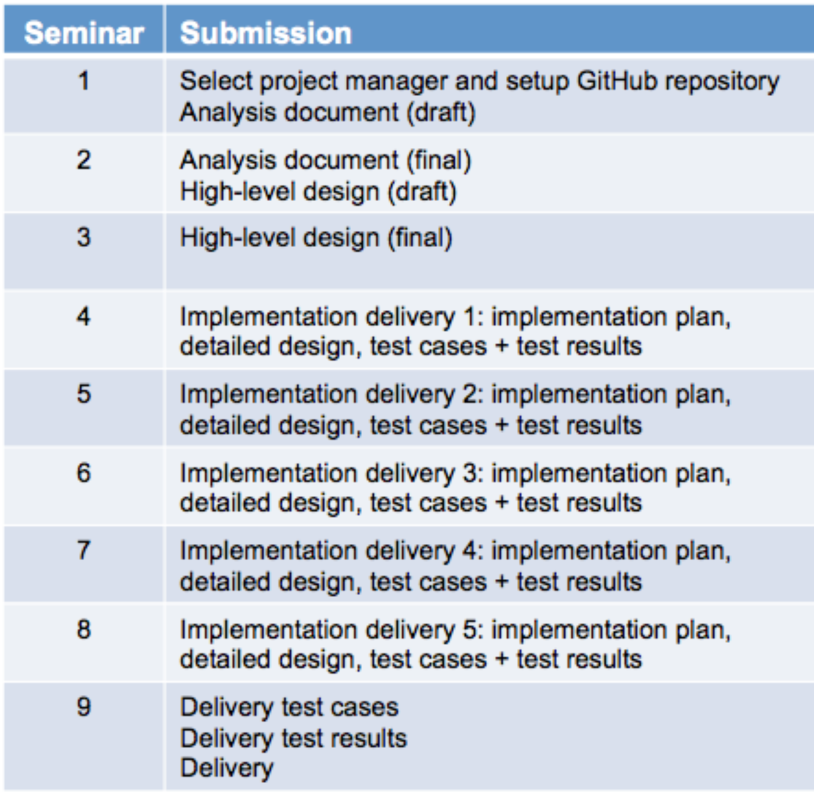
8. A delivery test result document.

9. A written manual explaining how to use the application for a potential user.

Dr. Johan Hagelbäck
johan.hagelback@lnu.se
Dept. of Computer Science

Submissions

Each course week the project group shall attend a supervision seminar (in total 9 seminars). The day before each seminar the group shall hand in a weekly submission at the course page on Mymoodle. What to submit each week is specified in the table below:



Seminar	Submission
1	Select project manager and setup GitHub repository Analysis document (draft)
2	Analysis document (final) High-level design (draft)
3	High-level design (final)
4	Implementation delivery 1: implementation plan, detailed design, test cases + test results
5	Implementation delivery 2: implementation plan, detailed design, test cases + test results
6	Implementation delivery 3: implementation plan, detailed design, test cases + test results
7	Implementation delivery 4: implementation plan, detailed design, test cases + test results
8	Implementation delivery 5: implementation plan, detailed design, test cases + test results
9	Delivery test cases Delivery test results Delivery

Final submission

The final submission shall consist of:

- Link to GitHub repository.
- Executable JAR-file.
- Manual for the application.
- All documents; analysis, high-level design, detailed design, test cases and rest results.

Each student shall also submit an individual report of 2-4 pages about the student's experience and lessons learned from working in a group project in computer science.

Dr. Johan Hagelbäck
johan.hagelback@lnu.se
Dept. of Computer Science