Software Engineering 2—Project description

READ THIS VERY CAREFULLY—NO EXCUSE FOR IGNORING WHAT WE WRITE HERE

MeteoCal

1 Problem description

The X software house wants to offer a new weather based online calendar for helping people scheduling their personal events avoiding bad weather conditions in case of outdoor activities. Users, once registered, should be able to create, delete and update events. An event should contain information about when and where the event will take place, whether the event will be indoor or outdoor. During event creation, any number of registered users can be invited. Only the organizer will be able to update or delete the event. Invited users can only accept or decline the invitation.

Whenever an event is saved, the system should enrich the event with weather forecast information (if available). Moreover, it should notify all event participants one day before the event in case of bad weather conditions for outdoor events. Notifications are received by the users when they log into the system.

2 Your task

You must develop the system using the Java EE platform. In particular, you will use EJBs to develop the business logic. The user interface can either be a web application or a normal Java application. In both cases the user interface has to interact with the business logic. During the development you should fulfill the deadlines indicated in Section 3 and should abide by the rules described in Section 4.

3 Steps and deadlines

The steps you must proceed through and the milestones for your development process are the following:

Deadline 2 November 2014, 23.59: *Group registration*. You should form your group and register it by going through the following steps:

1. Create a repository for your project on GoogleCode (https://code.google.com). Make sure that you give a meaningful name and description for your project. Use the version control system you know most (our preferred ones would be subversion or git). Make sure that all group members have a google code account and have access to the repository.
2. Register your group by filling in the following form http://goo.gl/forms/xFQ3U5F4Yx. Both GoogleCode and the registration form will be presented during the project lab of October 30th.

From that point on **you will start keeping track of the number of hours each group member works toward the fulfillment of each deadline**.

Deadline 16 November 2014, 23.59: *Requirements analysis and specification document (RASD)*. The RASD contains the description of the scenarios, the use cases that describe them, and the models describing requirements and specification. You are to use a suitable mix of natural language, UML and Alloy. UML and Alloy MUST be part of the documentation. You must also show that you used the Alloy tool for analysis, by reporting the models you obtained by using it. Of course, the initial written problem statement provided above suffers from the typical drawbacks of natural language descriptions:

it is informal, incomplete, uses different terms for the same concepts, etc. You may choose to solve the incompleteness and ambiguity as you wish, but be careful to clearly document the choices you make. You will also include in the document information on the number of hours each group member has worked towards the fulfillment of this deadline.

Deadline 7 December 2014, 23.59: *Design document (DD)*. DD must contain a functional description of the system, and any other view you find useful to provide. You should use all the UML diagrams you need to provide a full description of the system. Alloy may also be useful. You will also include in the document information on the number of hours each group member has worked towards the fulfillment of this deadline.

Deadline 25 January 2015, 23.59. *Implementation*. You should provide an implementation for the requirements you specified in the RASD, following the design you specified in the DD. You are requested to release source code and executable, installation and use manuals, system test cases. You will also include in the release a document containing information on the number of hours each group member has worked towards the fulfillment of this deadline.

Deadline 8 February 2015, 23.59. *Acceptance testing*. You will define acceptance test cases and report on the execution of tests for the system developed by a different group. We will give you the assignments. In this case you will act as a quality assurance group. You will also include in the document information on the number of hours each group member has worked towards the fulfillment of this deadline.

Deadline 10 February 2015, 23.59. *Project reporting*. You will apply the Function Point approach to your project and check if the results you get are similar to the actual size of your project. Moreover, you will use the project's actual size to apply the formulas from COCOMO and compare the resulting effort with the one you have actually spent. The project reporting document will contain the results of this analysis.

Date to be defined. *Final presentation*. You will have to present your project providing an overview on your requirement and design decisions, a demo of the system, describe your code and test cases and present the results of the acceptance testing you have performed. Overall, groups of two students will have 30 minutes, groups of three students will have 40 minutes, while single students will have 20 minutes.

4 Rules

The project should be developed by groups of 1, 2 or 3 persons (two persons is the suggested size).

- **Groups composed of one person** will develop the entire set of functions described above.
- **Groups composed of two persons** will extend the system as follows:
  - A user A should be allowed to make his/her calendar public, that is, visible to all other registered users. These last ones will see all the time slots in which A is busy but without seeing the details of the corresponding events, unless they have been defined as public. Events can be defined as public or private by their owners, upon creation. If an event is public, all the registered users can see its details, including the corresponding participants.
  - In case of bad weather conditions for outdoor events, three days before the event, the system should propose to its creator the closest (in time) sunny day (if any).
- **Groups composed of three persons** will extend the system by adding all requirements for groups composed of two persons, and the following additional requirements:

    - Implement email notifications (both for invitation and cloudy outdoor events alerts)
    - Allow users to import and export their calendar
    - Manage time consistency when creating an event by avoiding conflicts with existing events.
    - Update weather information associated to events periodically, e.g., every 12 hours, and, of course, notify outdoor event participants in case the forecast has changed.
- **Telecom students**: Students enrolled in "Telecommunication Engineering/ Ingegneria delle Telecomunicazioni" and following the Internet studies will obey to the rules valid for the students enrolled in "Computer Science and Engineering/Ingegneria Informatica". The others following the traditional telecommunication studies will either develop a more extensive testing activity instead of implementing the solution using JEE or they will join computer science students in their groups. A group featuring a telecom non Internet student should do the implementation part as if it was composed of a number of students not comprising the telecom student. For instance a group of three people featuring a telecom non Internet student should do the implementation part mandated for a group of two.
- **Each group MUST provide the requested artifacts within the stated deadlines. A delay of a few days will be tolerated but it will result in a penalty in the final score. It is mandatory to provide these artifacts and to present them to the reference professor in a final meeting that will be scheduled on the web.**
- **The material presented in one artifact is not fixed in stone. You can provide updates as part of the following deliverables. For instance, if you realize that you need to modify the RASD document and you are in the implementation phase, you can include in your implementation deliverable a short explanation of why you need to change the RASD, and update this last document accordingly by including, at the end of the new version, a short summary of changes.**
- For any question related to the project that could be interesting also for the other groups please use the forum available on the course website. We will answer as promptly as possible.