Implementation Report

Edoardo Scibona
Mat. 836884

January 27, 2015

Contents

1 Introduction

In this document we report the changes, with respect to the RASD and the DD, that we made during the implementation phase and the motivations behind those changes.
At the end there is a brief summary of the implementation phase and some informations.

2 Changes pertaining the RASD

- **User participating to an outdoor event are notified as soon as possible**
  Contrary to what is stated in section *3.1 Functional requirements* the system does not notify the users a day in advance but as soon as a new weather forecast for the event is available. This is because the weather manager checks periodically for weather forecast and the weather forecast associated to an event is updated as soon as possible. Notifications are generated at update time.

- **Past events can be modified**
  Contrary to what is stated in section *2.5.2 Assumptions* we chose to let users to be able to modify all of their events. The are two main reasons: the first one is that they can only modify the event's name and description and that operation is not particularly heavy, the second reason is that they may want to write down something that happened or something to remember about the event or something that they want to share with who attended the event.

- **User names are case insensitive** this was not explicitly stated in the RASD but we chose this option as the database system did not take into account case sensitivity and it would refuse to insert duplicate usernames.

- **Class Diagram** while the provided class diagram was only a rough draft for the system to be, the final implementation benefited from its skeleton but also deviated from its appearance in part thanks to the technologies used and in part thanks to the choices made during development.

3 Changes pertaining the DD

- **Logical data model** the logical data model provided in section *3.2* underwent the following changes in order to better adapt to the JEE database interface or to answer to development choices.

  User(ID, Username, Password, Groupname) *Groupname* was introduced for the Realm based authentication.

  Event(ID, Type, Location, Date, StartingTime, EndingTime, Description, Weather, Owner) *EndingTime* substituted *Duration* to provide more consistency with *StartingTime*; *Owner*, a foreign key to User, replaced the bridge table *EventOwners* in order to facilitate relationships maintenance.

  Notification(ID, Type, Status, Recipient) *Recipient*, a foreign key to User, replaced the bridge table *PendingNotifications* in order to facilitate relationships maintenance.

- **User Experience** Event pages such as *Details*, *Update* and *Delete* were implemented as a single window, always shown in the personal page, that retained the flow describe by the UX diagrams.

4 Informations

4.1 Effort

As of January 27, 2015 approximately **75 hours** have been spent during the implementation phase.

4.2 Tools used

Texmaker 4.1 to write this document and the three manuals.
`http://www.xm1math.net/texmaker/`

NetBeans IDE 8.0.2 to develop the MeteoCal application.
`https://netbeans.org/`

Glassfish Server 4.1 to deploy the MeteoCal application.
`https://glassfish.java.net/`

Google Code to host the git repository of the project
`https://code.google.com/`

Git 2.2.1 as the version control system for the project
`http://git-scm.com/`