Politecnico di Milano

Master of science in computer science and engineering

MeteoCal

A WEATHER BASED ONLINE CALENDAR

Previous Document rectification

Teachers

Raffaela Mirandola
Marco Miglierina

Students

Alessandro Negrini 836806
Andrea Gulino 836681
Paolo Guglielmino 837055

Academic Year 2014/2015

Contents

1 Introduction

Readers can find RASD modification in RASD Document V2 file.

Due to time constraints we couldn't develop all the functionalities that we stated in our previous documents.
As a matter of this fact, the purpose of this document is to explain why and what we miss to develop.

But first, we want to emphasize the fact that the functionalities that are missing are not mandatory, but they are just extra functionalities that we add during planning phase in order to improve the application, and improve the user experience.

Anyway, we believe that all mandatory requirements are met according to teacher request.

2 RASD Rectification

The first rectification we must do is related to events. In fact, we expected to develop a module that would allowed users to leave comments to the events they would have been invited to. In the final version of our application there isn't this opportunity.

Another modification we did is regarding to weather notification policy. In our old RASD we predicted that the system updates periodically every 12 hours and in case of bad weather is expected is sends users invited the notification. Actually, in the application we chose to adopt this policy:

1. The system updates every 1, 5 or 12 hours (according to current administrator settings) the weather and in case the weather has changed with respect the previous forecast it sends a notification to all users invited.

2. If three days before the weather is worse than the one selected in creation event phase, the system sends to the administrator the email that invites him to choose a new date (the system provides the closest sunny day) or invites him to cancel event. Anyway, the user can also ignore the mail.

3. If the administrator decides to ignore the notification, if one day before the event is still bad weather the system sends to every invited user a notification.

Another thing that wasn't took into account in developing phase was the management of time zone.

Another point that wasn't developed, is the one that in case of bad weather the system provides the most suitable place.

As for the interface, our basic structure is maintained almost equal to the one presented in RASD, except for the fact that the user can invite other users only after having created the event. In our mockup we expected to have a "Add friend" button in event creation page. Instead, the user can invite someone just after having created the event.

Finally, when a user creates a new account, we don't send him/her an email in order to confirm the operation, but we directly activate his/her account without sending email.

3 Design Document Rectification

According to Class Diagram, Entity Relationship diagrams, we didn't use the attribute *lastUpdate* in entity *W-FORECAST* because we think that it is an useless attribute. The system updates periodically (every 1, 5 or 12 hours) without checking at it.

3.1 UX diagrams

UX diagrams have been respected quite completely. Here are listed some little modifications, that won' t affect the diagrams in the old DD document at all . Anyway in the second version od DD you'll find the correct diagrams.

- Unlike the first diagram, we don't have only one screen compartment dealing with lastRecentEvents, but we have three compartments about accepted events, declined event, and pending events.
- The import/Export Calendar is not a part of User Profile screen, but a part of User Home page.
- Since we haven't implemented comments, we have to remove the screen compartment "Comments" from the Event Management UX diagram.

  All these updates are fixed in the newer version of DD Document.

3.2 Entity Beans Model

Compared to what we had expected, we made a couple of changes during the implementation phase:

- Removed the entity Comment, due to time issues we didn't implement this functionality.
- Changed some attribute in Location, for example Coordinates because we needed them to represent the Google Map.

3.3 Business Logic EJB

In principle, the structure follows the one we have made in the first version of DD. However there are some differences we want to point out:

- Some controller names are different, and contains the word "Facade" instead of "Manager", we also created a Facade that contains all common methods used by the other Facades.

- The controllers with named as "Loader" were incorporated into another controller, for example UserDataLoader doesn't exist any more, its functions are performed by UserManager.
- SearchUserManager doesn't exist any more, it is replaced by SearchUser (Servlet).
- Social functions, like Follow/Unfollow, are implemented in UserManager.
- We split some boundaries in order to avoid too much complex classes.

4 Appendix A: References

1. Past example of Test Cases document
2. Lecture slides
3. http://users.csc.calpoly.edu/ jdalbey/206/Assign/HowToTestCase.html