Project Presentation

POLITECNICO DI MILANO





Andrea Celli, Stefano Cereda

Politecnico di Milano

February 20, 2015

Overview

- Requirements and goals
- Design
- Code
- Project demo
- Project reporting
- Acceptance
- Questions

Main functionalities

- An on-line calendar
- A system to manage activities according to weather forecasts

More specifically: 3 main "families" of functionalities. For each of them we specified requirements

Managing users

Functional requirements:

- Register to system
- Login/Logout
- Modify password
- Update personal info

Non-functional-requirements:

- Password must be stored securely
- System must support high numbers of users

Managing calendars

Functional requirements:

- Add new events
- Modify/delete an existing event
- View personal schedule
- View event details
- Send invitations
- Reply to invitations
- See other users' schedule (if possible)
- See other users' public events details
- Receive "event changed" notification

Non-functional requirements

- None

Managing forecasts

Functional requirements:

- Send a notification the day before an event in case of bad weather to all the event's participants
- Propose an alternative schedule three days before an event in case of bad weather to the event creator
- Show the weather forecasts for the scheduled events

Non-functional requirements:

- The displayed forecasts should be updated every 24 hours
- The system has to interface with a meteo service to collect forecasts

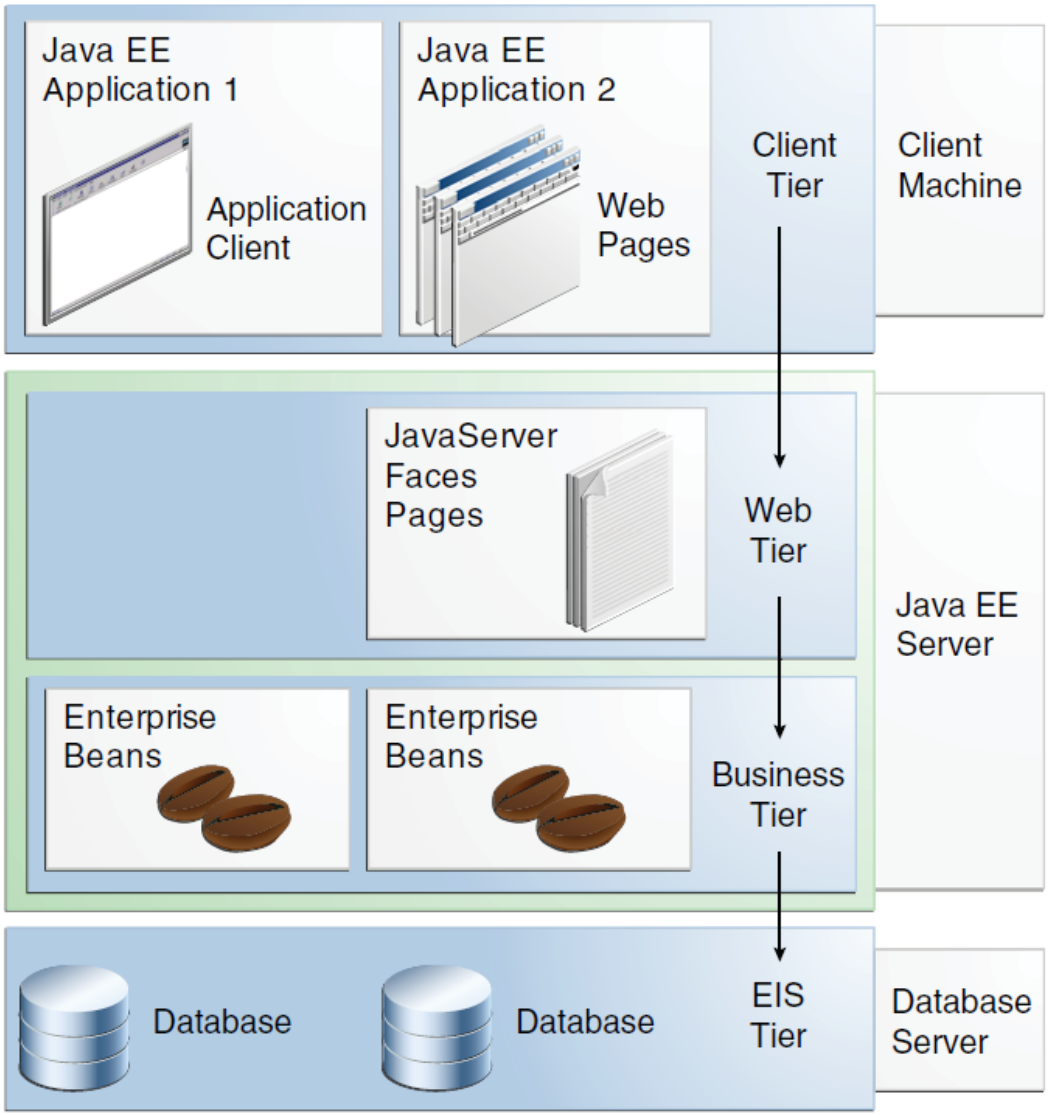
Goals

- Allow the registration of new users
- Allow users to view, create, update and delete events
- Allow users to invite other users to their events
- Allow invited users to accept or decline invitations
- Allow users to see other users' public calendar
- Allow users to see other users public events details
- Send a notification to all the participants one day in advance in case of bad weather
- Propose an alternative schedule to the event creator three day in advance in case of bad weather
- Notify all the event's participants if the creator changed its details
- Allow users to modify their data

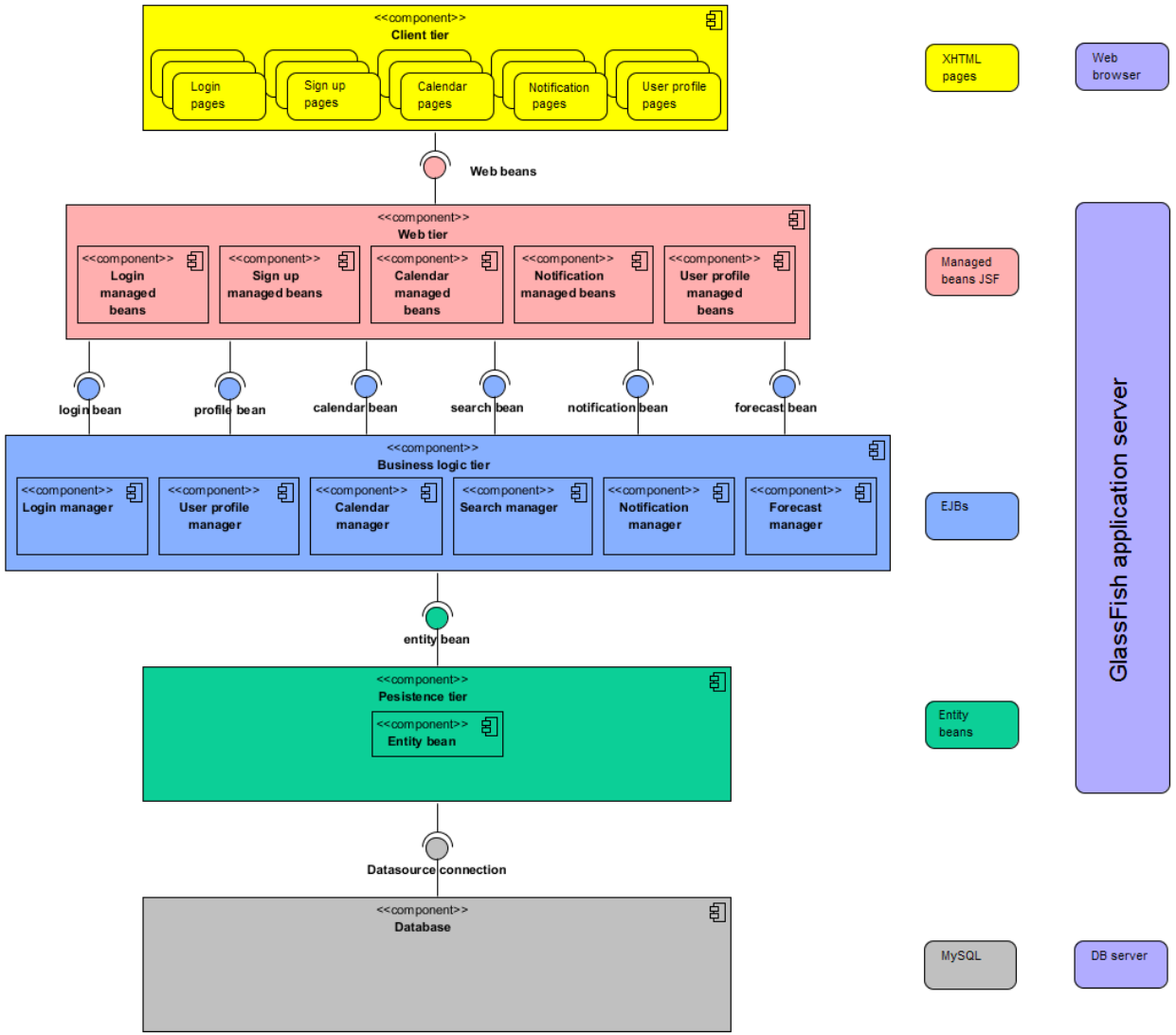
Design overview

We followed the usual 4 tier JEE architecture

Design overview



Design overview



Subsystems

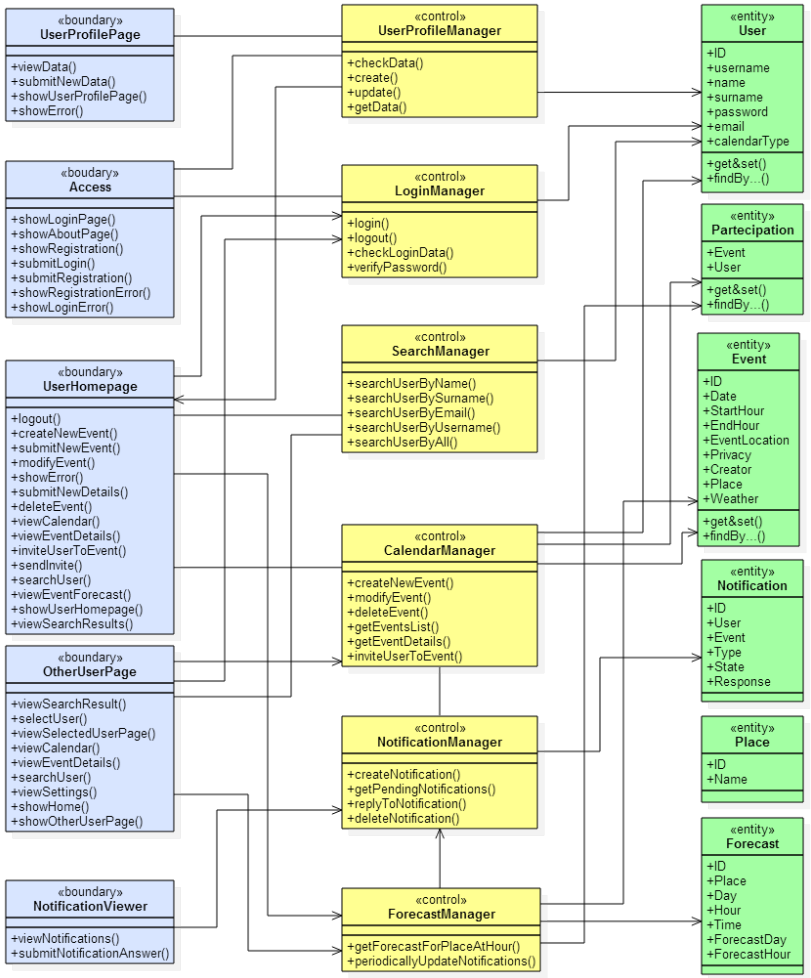
Our system can be easily divided into smaller subsystems with a high cohesion:

- Event managing
- Forecast managing
- Notifications managing
- User managing

The subsystems access each other through a little number of public methods

The system can be divided in three areas according to an MVC pattern

BCE


«boundary»
UserProfilePage
+viewData()
+submitNewData()
+showUserProfilePage()
+showError()
«control»
UserProfileManager
+checkData()
+create()
+update()
+getData()
«entity»
User
+ID
+username
+name
+surname
+password
+email
+calendarType
+get&set()
+findBy...()
«boudary»
Access
+showLoginPage()
+showAboutPage()
+showRegistration()
+submitLogin()
+submitRegistration()
+showRegistrationError()
+showLoginError()
«control»
LoginManager
+login()
+logout()
+checkLoginData()
+verifyPassword()
«entity»
Partecipation
+Event
+User
+get&set()
+findBy...()
«control»
SearchManager
+searchUserByName()
+searchUserBySurname()
+searchUserByEmail()
+searchUserByUsername()
+searchUserByAll()
«boundary»
UserHomepage
+logout()
+createNewEvent()
+submitNewEvent()
+modifyEvent()
+showError()
+submitNewDetails()
+deleteEvent()
+viewCalendar()
+viewEventDetails()
+inviteUserToEvent()
+sendInvite()
+searchUser()
+viewEventForecast()
+showUserHomepage()
+viewSearchResults()
«entity»
Event
+ID
+Date
+StartHour
+EndHour
+EventLocation
+Privacy
+Creator
+Place
+Weather
+get&set()
+findBy...()
«control»
CalendarManager
+createNewEvent()
+modifyEvent()
+deleteEvent()
+getEventsList()
+getEventDetails()
+inviteUserToEvent()
«entity»
Notification
+ID
+User
+Event
+Type
+State
+Response
«boundary»
OtherUserPage
+viewSearchResult()
+selectUser()
+viewSelectedUserPage()
+viewCalendar()
+viewEventDetails()
+searchUser()
+viewSettings()
+showHome()
+showOtherUserPage()
«control»
NotificationManager
+createNotification()
+getPendingNotifications()
+replyToNotification()
+deleteNotification()
«entity»
Place
+ID
+Name
«boundary»
NotificationViewer
+viewNotifications()
+submitNotificationAnswer()
«control»
ForecastManager
+getForecastForPlaceAtHour()
+periodicallyUpdateNotifications()
«entity»
Forecast
+ID
+Place
+Day
+Hour
+Time
+ForecastDay
+ForecastHour

Design principles

- Good days searched in daily forecast and returned as a list (design for flexibility)

Design principles

- Good days searched in daily forecast and returned as a list (design for flexibility)
- One parser for both 3-hours/daily (design for re-usability)
- JSON objects downloaded accordingly (design for re-usability)

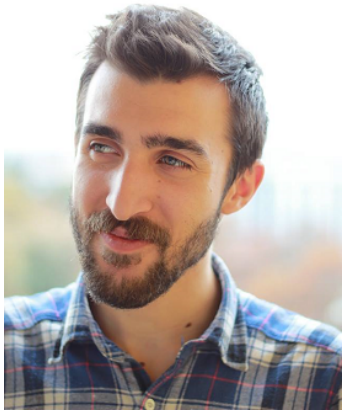
Design principles

- Good days searched in daily forecast and returned as a list (design for flexibility)
- One parser for both 3-hours/daily (design for re-usability)
- JSON objects downloaded accordingly (design for re-usability)
- The passed events are searched in the db (design defensively)

MeteoCal demo

Let's try it!

Overall project time



- RASD: 30h
- DD: 30h
- DEV: 150h (front-end)

- RASD: 30h
- DD: 35h
- DEV: 150h (back-end)

$$\Downarrow$$

$$\text{TOT} = 425\text{h}$$

Function point

- $UFP = 90 FPs$
- $LOC = AVG * UFP = 46 * 90 = 4140$

Function point

- $UFP = 90 FPs$
- $LOC = AVG * UFP = 46 * 90 = 4140$
- Actual size = 4372 LOC
- Difference of 5,5%
- Pretty good estimation

COCOMO II

- Effort estimation
  - PM = 8.5 person-months
- Schedule estimation
  - TDEV = 7.5 months
  - Number of people = 2 (approximately)

Acceptance

Acceptance testing of
`https://code.google.com/p/meteocal-iodicefinardi/`

Requirements

- Dublin IE or Dublin US
- Good weather vs Desired weather
- Bad weather alert: is the creator a participant?

Design

- Date of birth and city: useless and wrong
- Closest day with good weather: embed in the notification or not?
- Forecast: daily vs 3-hours

Failing tests

The two tests EventManagerTest.newEventShouldBeSavedOnce and CalendarManagerIT.eventsGetAddedToCalendarOfCreator fail when executed after 12.00 a.m.

Failing tests

The two tests EventManagerTest.newEventShouldBeSavedOnce and CalendarManagerIT.eventsGetAddedToCalendarOfCreator fail when executed after 12.00 a.m.

```
//Creating event
Calendar calendar = Calendar.getInstance();
Date date = calendar.getTime();
Date startingTime = new Date();
startingTime.setDate(8);
startingTime.setMinutes(00);
Date finishingTime = new Date();
finishingTime.setHours(12);
finishingTime.setMinutes(00);
```

Credentials

The browser proposes to save the date of birth as username

Wrong forecast

During the periodical update the system downloads the forecast for the following day even for three days events

Wrong forecast

During the periodical update the system downloads the forecast for the following day even for three days events

```
String[] forecastDownloaded;
int daysFromCurrentDateOfEvent;
if(recipient=="creator") {
    daysFromCurrentDateOfEvent=3;
} else {
    daysFromCurrentDateOfEvent=1;
}
```

Questions

Questions time!