Progetto di Ingegneria del Software 2

SWIMv2

Planning Document

A.A. 2012/2013

Autori:
Emanuele Uliana (799256), Gabriele Rufolo (743695), Walter Rubino (742519)

Docente:
Prof.ssa Raffaela Mirandola

Versione 1.2 del 03/12/2012

Indice

1 Project description and deadlines

The goal of the project is to build up a social network where people can, after a registration, search for help and support from users who have a particular ability that can be useful to solve determinate problems.

This system is called SWIM and it consists in a web application reachable by a common browser; basically it works like this: there are three type of users with different capabilities: the guests, which can only search for help among all the registered users, or do the registration process; the registered users which are able to declare a set of abilities, to search for other users, to send/accept friendship request, to search for help from both friends and anyone other (as long as he/she is registered), to give a feedback about the support he/she received, and finally to send to an administrator a request for upgrading his/her set of abilities.

The last type of users is obviously the administrator who defines the entire set of abilities and manages the requests from the common registered users. The implementation will be discussed in part later in this document and mainly in the next one(s): the RASD and the DD.

1.1 Deliveries

Deadline	Delivery
27 October 2012	Planning Document
24 November 2012	RASD
22 December 2012	Design Document
26 January 2013	Implementation
2 February 2013	Testing

Note that we don't indicate the dates with the American format in which the month precedes the day, because we don't like it.

- **Planning Document**: 27 October 2012 is the last day to confirm the group, formally by compiling a form with the anagraphics of the members, and to produce the so called "Project Planning Document" (this one).
- **RASD**: 24 November 2012 is the last day to produce the "Requirements analysis and specification document", also known as "RASD": it contains the use scenarios and the use cases, the description of all the requirements in natural language, their specification possibly through a mathemathical model, some UML diagrams and some Alloy examples. (Please note that in this version 1.0 we actually don't know how to use UML and Alloy in the RASD, so now we can't be more specific; in the next version this issue will be fixed).
- **Design Document**: 22 December 2012 is the last day to produce the Design Document: basically the UML schemes/diagrams that (possibly) fully describe the modules, the classes and everything else about the system.
- **Implementation**: 26 January 2012 is the big day/the X hour: this is the deadline for the implementation of what is described in the RASD, following and compatibly with the DD.
- **Testing**: in the last week the goal is to find odd behaviours in another group project; even if we were toldto not overworking on this part, since we are very interested in computer security, we will. Moreover we're going to search for vulnerabilities by exploiting SQL Injection and XSS: we think that testing the security of a database is very important.

2 Who is this project for? Who are the stakeholders?

The system is thought for anyone who needs help and wants to search online for a solution, rather than calling an expert and so having to pay him. The interface is supposed to be user-friendly to allow any people, even person who lacks in computer science knowledge, to use it without problems.

3 About the team

The team is made by three students of Computer Science Engineering at Politecnico di Milano: Emanuele Uliana (mat. 799256), Gabriele Rufolo (mat. 743695) and Walter Rubino (mat. 742519); the referring teacher is prof. Raffaela Mirandola, who is the holder of the course "Ingegneria del Software 2" and a member of the DEI, in particular of the DEEPSE.

4 Development

4.1 Developing plan

We are not going to choose the "Waterfall lifecycle" for two reasons: the first is that we lack in experience, so it's irrealistic to think that we are able to go through the various phases without having to change something in the previous ones; the second is that we do not like it at all: we think it's not the best way to proceed, expecially because it's likely that, due to unexpected issues, at least a part of the project needs to be rethought from the beginning or from a previous phase.

Rather we are going to modify the planning, the requirements or the design whenever we understand that in the implementation there is something unfeasable or too difficult to make being coherent with the previous phases; however we are confident to limitate these cases by overworking on the phases that precede the code writing. Basically the subdivision of the activities works like this:

- Project Planning
- Requirements Analysis
- Design
- Implementation
- Testing where every phase corresponds more or less to the deliveries we have already spoken about.

4.2 How do we work?

Whenever is possible (I mean when we are at Polimi and we don't have lessons at least for the next two hours) we work together: it may be that one of us is writing in Java while the others give an opinion about the complexity of the methods or the legibility of the code; it may be that someone is programming and someone else is searching online for something the writer doesn't know...Anyway we take the critical decision together for sure: for example the technology (I mean MYSQL for the database, etc.) or the general appearence of the web page.

However the main part of the work is done individually at home: we decide everyday or so what everyone of us is going to do and when we meet the day after we take the stock of the situation.

4.3 About the technologies we use

Here we list the technologies used in the project (by technologies we don't mean the software: this will be in the next section): HTML 5.0, CSS and Javascript for the web page; Java Enterprise Edition for the main application and MYSQL for the database. We are forced to use JEE by contract and we choosed together the other ones.

4.4 About the machines and the tools

Emanuele has an Asus laptop with GNU / Linux OS, Gabriele has a laptop with Windows and Walter owns a MacBook Air. This is a positive thing because we can check the actual portability of Java (maybe so much enthusiam is not justified, because is wery well known that many glitches appear in these cases).

As for the software used: first of all Eclipse for Java EE, the main tool for developing the project; we installed some plugins: m2e (maven) and svn (subversion); then Google Code as a Subversion Repo, Kile and MacTeX for writing documents in LATEX(every single document of the project is written in LATEX, then converted into a PDF file), various text editors (Libre Office Writer, Microsoft Word for Mac, Gedit) and different HTML/CSS tools, phpmyadmin for the database part.

4.5 Time estimation

The graphic below represents an estimation of the hours of work. Our convention is: when we work together we count the hours only one time; when we work independently from the others at home, everyone counts the hours he worked on the project. So if a day Emanuele works from 15.00 to 19.00, Gabriele from 15.00 to 19.00 and Walter from 15.00 to 19.00 too, and everybody is at home, the total hours counted are 12.

In this 1.2 version we have to say that we have almost no elements to estimate the final number of hours of work, so we rely on Statistics: we have a graphic with the number of hours of various projects made by other students last year; we eliminate from the sample the groups which ended in less than 150 hours, because we think they meant by that value the number of man hours (basically the real number of hours divided by two or three), and the groups which ended in more than 800 hours, because we think that, even working 8 hoursperday they would have needed more than three months to reach that value and if we consider that the total time for the project is slightly less than four months and almost every morning (and a good part of the afternoon) is not available because of the lessons of the other courses, either they worked during the night, or the estimation they made is not good.

Anyway, after this we have our sample; the situation is that we don't know a parameter which is the expected value and the statistics says that the best estimator for it is the sample mean. So the only reasonable way for us to estimate the total is to take the sample mean of the data we have.

We understand that the estimator is very likely to be biased, expecially because the project of this year is different, the approach will necessarily be different (due to the fact that people are different) and the risk factors (they are not obviously equal to the last year ones) probably will play an important role, but I repeat, we have no other choice.

In the last version we will include the real number of hours we used for the whole project; for now, along with this document, in the google code repo can be found a Gantt diagram not so precise (basically it says everyone is going to do everything).

Task	Hours
Requirements	60
Design	70
Implementation	240
Testing	30
Total	400

4.6 Useful knowledges we have and we use in the project

- Propositional and first order logic
- Java SE/EE 6/7
- LATEX
- HTML/CSS/Javascript
- SQL/PHP
- SQL Injection/XSS

5 Risks

Of course no project is risk-free and ours is no exception: many unwanted things can happens and delay some phases of the work; we divided these risks in two categories: human risks and software risks.

The first group contains all the problems that might rise up among the team: misunderstandigs between the components, a sudden illness of one of them, a low capability to cooperate, a low productivity when working alone, ecc.

The second group contains all the issues linked to the hardware/software: a computer might break down, there may be problems while installing/using all the software needed (and we were told form prof. Mirandola that in the past this happened several times, so, even if we can't say it for sure, we can expect we will be facing these problems) and there may also be portability problems, mainly connected to Java and to the web browsers.

Type of risk	Description
Human	• Misunderstandings • Indivudual work • Low capability of working together • Illness of someone • Group size
Computer	• Phisical vulnerabilities • Problems with SVN/software in general • Portability • Bugs

6 Various and personal thoughts and considerations about the project

This section is going to be filled through and after the project.