Distributed Software Engineering Laboratory

Project Coffee

Date	15 December 2014
Authors	Group 2

1 Introduction

In this report, we will explain the process of development and implementation of requirements for the Coffee software system. Firstly, we accepted the software requirements specification written by the Kazan3 team, then we agreed with the front-end team Rio Cuarto2 about the development methodologies and techniques which we will follow in order to deliver the system.

2 Development

2.1 Development techniques and used tools

We used Eiffel programming language, Eiffel Studio 14.05 as the environment, SQLite as database support, Tortoise SVN [1] as version control, batch scripting as test mechanism, Google Chrome/Mozilla Firefox as test browsers.

3 Implemented requirements

During the allocated time we managed to implement the following functional requirements:

- FR-01 – FR-10
- FR-12 – FR-14

We focused specifically on these requirements because of the risks and priorities which was stipulated in the software requirements specification. By implementing the most critical requirements we wanted to ensure the core functionality of the system.

4 Non-implemented requirements

We couldn’t implement the following requirements during the time we had on our disposal:

- FR-11

Non-implemented requirement is not critical for the Coffee software system, but on the other hand the system itself would be more useful and intuitive if we had implemented them. Major reason for not implementing this requirement is lack of time. In the beginning of development process we spent more time learning how to work in distributed manner, which eventually led to the lack of time for optional requirement like this.

5 New requirements

During the work on project, we noticed we could add features and enhance user experience and system itself. These features are:

- Sprint statistics – For each previous sprint system computes number of hours estimated, number of spent hours, total number of tasks and number of completed tasks
- Burn-down chart

[1] http://tortoisesvn.net/

Even though these requirements are not implemented, they can easily be added to the system without many changes. Database is designed to support these features, so upgrading a system can be done only by adding necessary methods and corresponding views.

6 Conclusion

We had whole software development cycle to follow and participate in the new distributed environment. It was a real experience from which we learned a lot, we faced the whole new level of problems both from programmer’s and manager’s points of view. There were situations which we can better address now. Problems in distributed computing are specific in their own way, there is a lot of room for improvement but this project taught us the very core and principles that we should follow.