

Czech University of Live Sciences Prague

Faculty of Economics and Management

Department of Information Engineering



Diploma Thesis

VIRTUAL TEAM BUILDER HELPER

Author: Bc. Mario Alberto ROW MORA

Supervisor: Ing. Josef Pavlíček, Ph.D.

© 2013 CULS in Prague

Acknowledgements

I would like to thank all my family and my friends for all the support and encouragement during my master. Specially, I want to thank my mom and my dad. I love you mom and dad!)

To my colleagues for support me, while writing this thesis and also my teachers in this university for their time and engagement.

In Prague March, 2013

Bc. Mario Alberto Row Mora

Declaration

I declare that I have completed this thesis independently and I have listed all the literature and publications used.

I have no objection to usage of this work in compliance with the act 60 Zákon č. 121/2000Sb. (copyright law), and with the rights connected with the copyright act including the changes in the act.

In Prague, March 2013

.....

Bc. Mario Alberto Row Mora

POMOCNÍK PRO VIRTUÁLNÍ TEAMBUILDING

VIRTUAL TEAM BUILDER HELPER

Souhrn

Každodenní práce v mnohých, ne-li všech, dnešních společnostech je založena na týmové práci. Týmy v takovýchto společnostech jsou navíc velmi často založeny na mezinárodní úrovni. Takovýto tým, kdy jsou členové týmu odděleni geograficky (např. Madrid, New York, Praha) se označuje termínem „distribuovaný tým“. Naskýtá se tak otázka: Pokud jsou členové jednoho týmu odděleni značnou vzdáleností, jakým způsobem budovat a podporovat vztahy mezi členy takového týmu?

Popularita distribuovaných týmů se vzhledem k ekonomickým, synergickým, ale i jiným důvodům zvětšuje a zvětšují se i počty členů takovýchto týmů. Cena za budování důvěry v distribuovaných týmech prostřednictvím osobních setkání je cenově vysoká (Games for virtual team building, 2008). Virtuální nástroje a sociální sítě mohou být cenově dostupnou alternativou umožňující bližší kontakt pro členy týmu a tak zpříjemnit jejich každodenní práci.

Tato práce je zaměřena na problematiku budování vztahů v distribuovaných týmech prostřednictvím sociálních sítí. Výstupem této práce jsou: prototyp nástroje umožňujícího a podporujícího tým a týmovou práci společně se specifikací požadavků, návrhem architektury a testů pro takovýto nástroj.

Klíčová slova

Virtuální, budování týmu, Java, Java EE, sociální síť, streaming, architektura, UML, xhtml, vývoj webových aplikací, css.

Abstract

The daily work in any company is usually based on teamwork. Team members can be spread among different geographical regions (e.g. Madrid, New York, Prague). The teams based in this manner are called "distributed teams". It raises the question: If team members are separated by great distances, how to build and foster relationships between the members of such a team?

The distributed teams are commonly increasing day by day in today's workplace. For these teams, face-to-face meetings and teambuilding activities where members can most easily build trust are rare and often cost-prohibitive (Games for virtual team building, 2008). Virtual tools and social networks may offer an alternate means for those who cannot have a close contact. It helps team members to get know each other better and it can also make the daily work more pleasant.

The aim of this thesis is to investigate the team buildings and social network approaches; and the results suggest a tool that can help organizations with virtual team buildings. The thesis includes an example of the software development process, requirement specification, architecture design, implementation and testing of this kind of tool.

Keywords

Virtual, teambuilding, helper, tool, Java, Java EE, streaming, social network, architecture, UML, xhtml, web development, css.

Table of Contents

Acknowledgements.....	III
Declaration.....	V
Souhrn.....	VIII
Klíčová slova	VIII
Abstract.....	IX
Keywords	IX
Table of Contents.....	XI
List of Figures.....	XIV
List of Acronyms	XV
1 Introduction.....	17
1.1 Motivation	17
2 Objectives and Methodology	19
2.1 General Objective.....	19
2.2 Specific objectives.....	19
2.3 Scope of the project.....	19
2.4 Methodology	19
2.4.1 UML.....	20
3 System requirement specification.....	22
3.1 Functional requirements.....	22
3.1.1 Basic features	22
3.1.2 User interface	22
3.1.3 Non-functional requirements	22

3.1.4	Performance issues.....	23
4	Literature Overview	25
4.1	Teambuilding and virtualization	25
4.1.1	Team building.....	25
4.1.2	Social Networks.....	26
4.1.3	Virtual organization	28
4.2	Web Application concepts	29
4.2.1	The Web.....	29
4.2.2	HTTP and its starts.....	29
4.3	Technology overview.....	30
4.3.1	HTML & XHTML.....	30
4.3.2	CSS	31
4.3.3	Essentials of JEE Application.....	32
4.3.4	Java EE Clients	33
4.3.5	Application server.....	33
4.3.6	Servlet	35
4.3.7	JavaServer Pages.....	36
4.3.8	Enterprise JavaBeans	36
4.4	JEE tools and frameworks.....	38
4.4.1	GlassFish Server	38
4.4.2	Ajax.....	39
4.4.3	MySQL Workbench.....	39
4.4.4	NetBeans	39
4.4.5	GIT	40
5	System analysis and implementation	44

5.1	Use Cases diagram	44
5.1.1	Actors	44
5.2	Classes diagram.....	46
5.3	Data Model.....	47
5.4	Deployment diagram	48
5.5	Interface design	49
5.6	Database design.....	50
6	Testing.....	52
6.1	Testing essentials.....	52
6.1.1	V&V	52
6.1.2	Criteria	53
6.1.3	Black-box vs. White-box	54
6.2	Tools for testing	56
6.2.1	TestNG	57
6.2.2	Unit test.....	57
7	Conclusion	59
	Bibliography	61
	Appendices.....	64
	Appendix A - Installation Guide.....	64
	Appendix B - SQL Commands constraints.....	65
	Appendix C – Some source code	68

List of Figures

Figure 1 Sociogram.....	27
Figure 2 Example a rule.....	31
Figure 3. Multitiered Applications.....	34
Figure 4. Server Communication.....	35
Figure 5 Representation of the enterprise beans functionality.....	37
Figure 6 Local version control diagram.....	40
Figure 7 Git Command-line interface.....	41
Figure 8 Git repository browser.....	42
Figure 9 User Roles.....	45
Figure 10 Uses case diagram.....	45
Figure 11 Class diagram.....	46
Figure 12 The Model diagram.....	47
Figure 13 Deployment diagram.....	48
Figure 14 Interface.....	49
Figure 15 Database design.....	50
Figure 16 Early Software V&V.....	53
Figure 17 PoC and PoO at black box.....	55
Figure 18 PoC and PoO at white box.....	56
Figure 19 Basic TestNG test.....	57

List of Acronyms

AJAX	
Asynchronous JavaScript And XML ...	22, 39
AS	
Application Server	64
CERN	
European Organization for Nuclear Research	29
DBA	
Database Administrator	39
e.g.	
for example (exempli gratia).....	26
IETF	
Internet Engineering Task Force.....	30
JMS	
Java Message Service	38
PoC	
Point of Control.....	54
PoO	
Point of Observation	54
RIA	
Rich Internet Application.....	39
SGML	
Standard Generalized Markup Language	30
UML	
Unified Modeling Language	19, 20, 44
V&V	
Verification and Validation.....	52
XHTML	
(Extensible HyperText Markup Language	22

1

Introduction

The basic concept of team building is use for companies to make the employees closer each other's.

The aim of this work is make a research of team-building concepts and its characteristics; as well propose such a tool (system) and its implementation that could help companies with this task – encourage a team building activities, especially in that kind of teams where the distance is a huge negative factor.

In addition and as part of the work, the proposed system will be implemented using innovative technology that Java offers.

1.1 Motivation

The concept of teambuilding is extremely important in every organization. Nowadays could be this concept understanding, it means as a regular meetings from members of one organization, normally in not working hours and in not workplace. But the concept is changing; it is because the technology is shortening the distances, but the taste of face-to-face conversations or simple kind of affection for good action sometimes is not possible as often as someone could wish.

2

Objectives and Methodology

2.1 General Objective

Conduct an analytical study of the teambuilding and social networks approaches as important part in the environment of any organization, taking in consideration the new concept of virtual distributed teams and propose a tool that could help those organizations to make their members relationships stronger.

2.2 Specific objectives

- Retrieve information about teambuilding and social networks
- Analyze the information and formulate the software requirements.
- Select the development tools and technology.
- Design the architecture.
- Implement the software.
- Make software testing

2.3 Scope of the project

- Research about the teambuilding concept
- Propose a tool that will help to incentive the teambuilding activities in organizations.

2.4 Methodology

Java Platform, Enterprise Edition as technology for development and running

To use Java EE as technology development and UML as tool that help in all the documentation of the project.

2.4.1 UML

The Unified Modeling Language¹ UML is a language that helps to specify, visualize and document models of software systems, including their structure and design, in a way that meets all of these requirements; in addition can be used for business modeling and modeling of other non-software systems (Object Management Group, Inc., 2005)

It is just a notation but its goal is to model systems using object-oriented concepts; also to visualize the software structure and to create modeling language useable by both humans and machines.

¹ UML was approved by the Object Management Group™ (OMG™) as a standard in 1997. Last release in August 2011 with the version 2.4.1

3

System requirement specification

3.1 Functional requirements

3.1.1 Basic features

1. System should allow the creation de new accounts
2. System should allow users to login/logout to his account
3. System should allow user to edit his or her data profile
4. System should allow send and received message to/from another user
5. System should allow send receive and display multimedia

3.1.2 User interface

In the develop of the interface the technology apply will be

- XHTML 1.0 transitional and AJAX to provide using RichFaces component library as a rich user interface.
- User interface will contain context help for individual fields and related validation with AJAX behavior.

3.1.3 Non-functional requirements

- The system will be develop in Java EE 6 and will be deploy on GlassFish Server version 3.1.2
- There are more technology involved as JSF and facelets, RichFaces, EJB.
- Client part of the application must be able to run in browsers with enabled JavaScript and support for cookies (Internet Explorer 6.0 or higher, Mozilla Firefox 2.0 or higher, and Chrome 22.0 or higher).

3.1.4 Performance issues

On an application server with a standard 2 GHz processor type IA32 2, 2 GB RAM and 7200rpm drive in RAID 3, the application shall manage up to 100 users connected simultaneously and operate an average rate of 1 operation in 10 seconds. Waiting time per operation must not exceed 5 seconds.

4

Literature Overview

4.1 Teambuilding and virtualization

The first concept that I would like to specify is a team, there are many different kinds of teams. Therefore, the boundary of this document will be specific a terms in working environment. Below is a list of a few common types and their descriptions.

- a) **Project team:** there are specific teams created to complete a project. They are usually temporary, only exist while the project is being completed, and are disbanded after the project is completed. Even though there teams are temporary, they still can benefit from teambuilding activities.
- b) **Working teams:** There are teams that work together every day with a purpose. The group is more permanent than the project team, although team members may come and go.
- c) **Distributed teams or virtual teams:** In the world of technology, there is a new breed of team – the virtual team. People now work together over great distances without actually meeting in person. There virtual teams can encounter a number of communication problems. The internet and e-mail are not always the best medium to communicate clearly. Taking on the phone is a little better, but you still cannot see a person's body language or other nonverbal communications.

(Peragine, 1970)

4.1.1 Team building

Team Building is the cooperative process that a group of individuals uses to solve both physical and mental challenges

4.1.2 Social Networks

Social networks are defined as a group of actors (individuals, groups, organizations, communities, societies, etc.) linked to each other through a relationship or a set of social relations. Social networks are supported by the Social network analysis (SNA) which focuses on the relationships between actors and their social behavior. The notion of a *social network* has attracted considerable interest and curiosity from the social and behavioral science community in recent decades. Much of the interest can be attributed to the appealing focus of social network analysis on relationships among social entities, and on the patterns and implications of these relationships. (Wasserman, et al., 1994).

4.1.2.1 Origin of Social Networks

A *social network* is a social structure that is made up of groups of individuals, which are called nodes and which are connected by some kind of relationship (e.g., friendship, business, kinship, etc.). Such structures are often used to model a social situation. In 1930 appears the sociometry as a way of formalizing the social sciences, where it was used the statistic to study populations and the graph theory helps to model the relationship between people (Scott, 2000)

In recent years, the theory about the structure of social networks was taken up; it is because to the high number of system in the Web that could make virtual communities that can be represented as a social network structure.

4.1.2.2 Mathematical Representation

It is crucial the background of the representation and analysis of social network. The principal types of data are 'attribute data' and 'relational data', where attribute data relate to the attitudes, opinions and behavior of agents, in so far as there are regarded as the properties, qualities or characteristics that belong to them as individuals or groups (i.e. income, occupation, education etc.).

The relation data, on the other hand, are the contacts, ties and connections, the group attachments and meetings, which relate one agent to another. (Scott, 2000)

Associated to those terms we can indicate that the analysis can be represented using the graph theory, applying participants and their relationships. In more practical terms the concept of sociometry, it is the study of relationships within a group of people.

In the mathematical computers sciences point of view, it is important consider the tem **relations** and **graphs**. Where Relation “R” can explain, first the description (records) of objects that in this case are: the employees in the company and those employees as part of a team members. And then describe the attributes of a given set and the relations between two or more objects; we can extract information as how strong can be that relation depending of different parameters, like comments, message to one member to another and so on. (Vaníček, et al., 2008)

So we can user transitive relation graph so show the members and his internal relations between them.

With those terms, we can start using the concept of sociometry

In Figure 1 Sociogram, for example, person A is the recipient of friendship choices from all the other members of a group, yet A gives reciprocal friendship choices only to persons B and C is, therefore, the star of attraction within the group. And then we can talk about the social forces in which the group was located. (Scott, 2000)

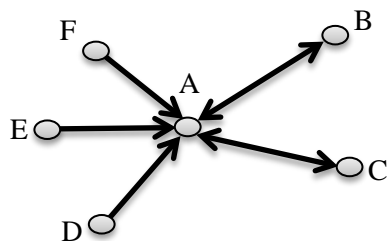


Figure 1 Sociogram

4.1.3 Virtual organization

In now days the complexity of interaction between members of teams is increasing, let's take in consideration if you do have a place to work, the people in it come and go at all hours, in various states of dress ranging from the traditional to the bafflingly outré, there is option work out, from your home, at least some of the time.

Moreover, people come and go from employment in your company with astonish rapidity. Some work for other firms that are temporarily connected with you in joint alliances. Some work for subcontractors that have long-term relationship with your company, and some simple work for themselves, having given up on corporate employment altogether. (Harvard Business Publishing, 2001)

Work with teams of people you never see and may have never met except in the virtual sense, is difficult to work on it. The complexity of virtual organization adds up to one certain issue: good communication has become more difficult than ever.

Virtual communication puts stress on three competencies that have always been important: accountability, trust, and adaptability.

The virtual workspace is designed to handle the communication needs; there are a variety of Internet-based products that might broadly be described as **virtual workspaces**, most of them free or low cost, have spring up to address the challenges involved in keeping projects teams in touch. Virtual workspaces typically provide a password-protected Web site with services ranging from e-mail to information storage to chat rooms. The sites usually offer

4.2 Web Application concepts

From the beginning of the World Wide Web (1979–1991) developed at CERN allowed Internet penetration in the most unexpected places and exploded in popularity on a world wide scale. (Ben, 1995)

The original idea was very basic with server – client, sharing information in the Internet. But in nowadays the complexity of systems is not as simple as it was think, there are a lot of factors that have strong influence in development decisions, we can mention some of them: dramatic quantity of users, quantity of multiple uses access in the same time, real-time systems, scalability, security, etc.

4.2.1 The Web

Let's see some important concepts that will show up the background of the thesis work.

4.2.2 HTTP and its starts

HTTP is the protocol behind the World Wide Web. With every web transaction, HTTP is invoked. HTTP is behind every request for a web document or graphic, every click or a hypertext link, and every submission of a form. It is useful because it provides a standardized way for computers to communicate with each other. Also specifies how clients request data and how servers respond to the request (Wong, 2000).

Another and maybe more technical definition is HTTP is an application-level protocol for distributed, collaborative, hypermedia information systems. It is a generic, stateless, protocol which can be used for many tasks beyond its use for hypertext, such as name servers and distributed object management systems, through extension of its request methods, error codes and headers. A feature of HTTP is the typing and negotiation of data representation, allowing systems to be built independently of the data being transferred. (Fielding, et al., 1999).

4.3 Technology overview

In this chapter I will discuss the technology used during the development of the project.

4.3.1 HTML & XHTML

There have been several versions of HTML since the Web began, and the development of the language is overseen by an organization called the W3C (World Wide Web Consortium).

The last major version of HTML was HTML 4.01 in December 1999. In January 2000, some stricter rules were added to HTML 4.01, creating what is known as XHTML (Extendible Hypertext Markup Language) (Duckett, 2010).

Let's review both concepts:

HTML is Acronym for **H**ypertext **M**arkup **L**anguage; the markup language used for documents on the World Wide Web. A tag-based notation language used to format documents that can then be interpreted and rendered by an Internet browser. HTML is an application of SGML (Standard Generalized Markup Language) that uses tags to mark elements, such as text and graphics, in a document to indicate how Web browsers should display these elements to the user and should respond to user actions such as activation of a link by means of a key press or mouse click.

Defined by the Internet Engineering Task Force (IETF), included features of HTML common to all Web browsers as of 1994 and was the first version of HTML widely used on the World Wide Web. HTML was proposed for extending HTML 2 in 1994, but it was never implemented. HTML 3, which also was never standardized or fully implemented by a major browser developer, introduced tables. HTML 3.2 incorporated features widely implemented as of early 1996, including tables, applets, and the ability to flow text around images. HTML 4, the latest specification, supports style sheets and scripting languages and includes internationalization and accessibility features. Future HTML development will be carried out by the World Wide Web Consortium (W3C). (Microsoft Corporation, 2002).

XHTML is a short for **E**xtensible **H**ypertext **M**arkup **L**anguage; it is a markup language incorporating elements of HTML and XML.

Web sites designed using XHTML can be more readily displayed on handheld computers and digital phones equipped with microbrowsers. XHTML was released for comments by the World Wide Web Consortium (W3C) in September 1999. See also HTML, microbrowser, XML (Microsoft Corporation, 2002).

4.3.2 CSS

CSS works by allowing you to associate rules with the elements that appear in the document. These rules govern how the content of those elements should be rendered. Figure 2 shows you an example of a CSS rule, which as you can see is made up of two parts:

- The *selector*, which indicates which element or elements the declaration applies to.
- The *declaration*, which sets out how the elements should be styled.

$$\begin{array}{c} \textit{selector} \qquad \qquad \qquad \textit{declaration} \\ \widehat{h1} \quad \underbrace{\{ \textit{font - family} : \textit{arial}; \}} \\ \qquad \qquad \qquad \textit{property} \qquad \qquad \qquad \textit{value} \end{array}$$

Figure 2 Example a rule

This is very similar to the way that HTML/XHTML elements can carry attributes and how the attributes controls a property of the element. (Duckett, 2010)

4.3.3 Essentials of JEE Application

In this section is important to explain how a Java EE Application works, showing up all the factors that are involved in the JEE environment.

The Java EE application model begins with the Java programming language and the Java virtual machine. The proven portability, security, and development productivity and provide forms on the basis of the application model.

The Java EE application model defines an architecture for implementing services as multitier applications that deliver the scalability, accessibility, and manageability needed by enterprise-level applications. This model partitions the work needed to implement a multitier service into the following parts:

- The business and presentation logic to be implemented by the developer
- The standard system services provided by the Java EE platform

Java EE Components

Java EE applications are made up of components. A Java EE component is a self-contained functional software unit that is assembled into a Java EE application with its related classes and files and that communicates with other components. (Eric, et al., 2013)

The Java EE specification defines the following Java EE components:

- Application clients and applets are components that run on the client.
- Java Servlet, JavaServer Faces, and JavaServer Pages (JSP) technology components are web components that run on the server.
- Enterprise JavaBeans (EJB) components (enterprise beans) are business components that run on the server.

4.3.4 Java EE Clients

The client in Java EE is usually either a web client or an application client.

In this project the focus is in web clients, and in web clients consist in tow part:

- Dynamic web pages containing various types of markup language (HTML, XML, and soon), which are generated by web components running in the web tier
- A web browsers, which renders the pages received from the server.

A web client is sometimes called a *thin client*, because usually do not query databases, execute complex business rules, or connect to legacy applications.

4.3.5 Application server

Application server is a server program on a computer in a distributed network that handles the business logic between users and backend business applications or databases. Application servers also can provide transaction management, failover, and load balancing. An application server is often viewed as part of a three-tier application consisting of a front-end GUI server such as an HTTP server (first tier), an application server (middle tier), and a backend database and transaction server (third tier). (Microsoft Corporation, 2002). The server can be program provides its services to the client program that resides either in the same computer or on another computer connected through the network.

The Java EE platform uses a distributed multitiered application model for enterprise applications. Application logic is divided into components according to function, and the application components that make up a Java EE application are installed on various machines, depending on the tier in the multitiered Java EE environment to which the application component belongs. (Eric, et al., 2013)

Figure 3 shows two multitiered Java EE applications divided into the tiers described in the following list.

- Client-tier components run on the client machine.
- Web-tier components run on the Java EE server.
- Business-tier components run on the Java EE server.
- Enterprise information system (EIS)-tier software runs on the EIS server.

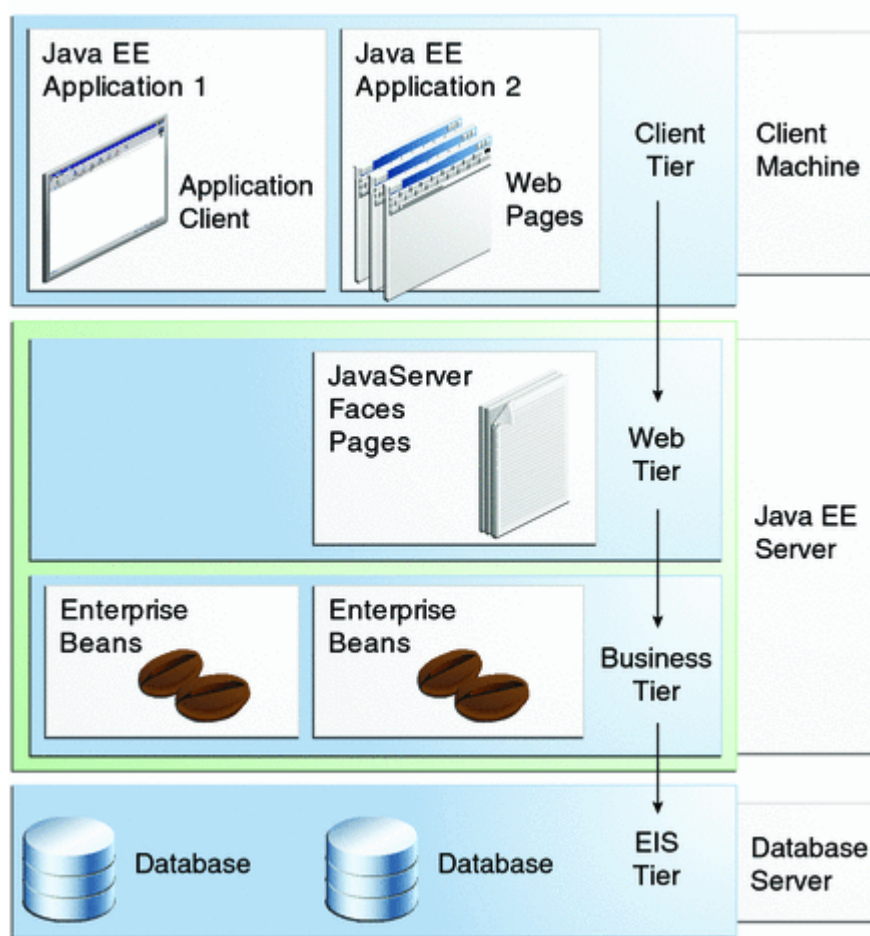


Figure 3. Multitiered Applications

(Eric, et al., 2013)

Java EE Server Communications

Figure 4 shows the various elements that can make up the client tier. The client communicates with the business tier running on the Java EE server either directly or, as in the case of a client running in a browser, by going through web pages or servlets running in the web tier.

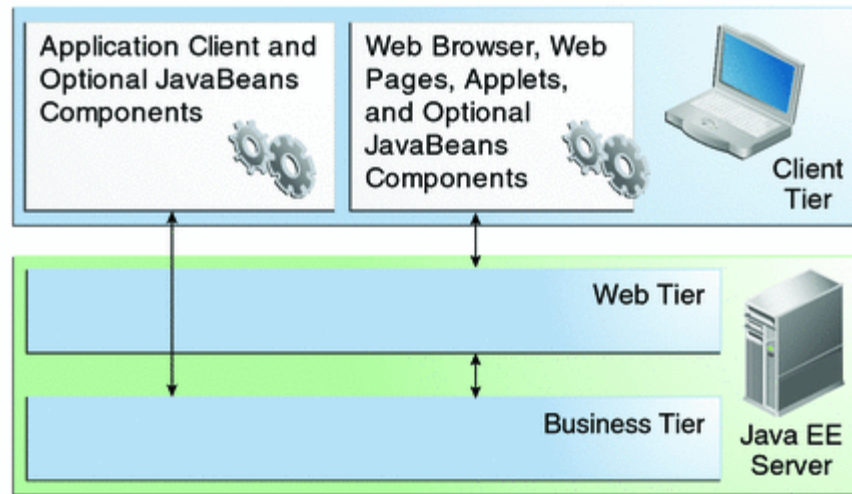


Figure 4. Server Communication

4.3.6 Servlet

Servlet is a Java class that is used to extend the capabilities of servers that host applications. Servlets can respond to requests and generate responses. The base class for all servlets is *javax.servlet.GenericServlet*, this class defines a generic, protocol-independent servlet. (Heffelfinger, 2007)

The most common type of servlet is an HTTP servlet; it must implement one or more methods to respond to specific HTTP requests. For example of HTTP Request we can see as GET, POST, PUT and DELETE.

4.3.7 JavaServer Pages

JavaServer Pages (JSP) technology lets you put snippets of servlet code directly into a text-based document. A JSP page is a text-based document that contains two types of text:

- Static data, which can be expressed in any text-based format such as HTML or XML
- JSP elements, which determine how the page constructs dynamic content

(Eric, et al., 2013)

The main features of JSP technology are as follows:

- A language for developing JSP pages, which are text-based documents that describe how to process a request and construct a response
- An expression language for accessing server-side objects
- Mechanisms for defining extensions to the JSP language

The JSP elements in a JSP page can be expressed in two syntaxes, standard and XML, though any given file can use only one syntax. A JSP page in XML syntax is an XML document and can be manipulated by tools and APIs for XML documents. (Jendrock, et al., 2007)

4.3.8 Enterprise JavaBeans

The word “enterprise” has magical powers in computer programming circles. It can increase the price of a product by an order of magnitude, and double the potential salary of an experienced consultant (Weaver, et al., 2004)

J2EE provides a collection of standardized components that facilitate software deployment, standard interfaces that define how the various software modules interconnect, and standard services that define how the different software modules communicate; therefore the multi-tier architecture is active part in Enterprise JavaBeans.

Services provided by the EJB container

The function of EJB components is mainly based on the work of the EJB container. The EJB container is a Java program that runs on the server and that contains all the classes and objects needed for the proper functionality of the enterprise beans.

The Figure 5 shows the representation of the enterprise beans functionality; there is clearly 3-tiers architecture: first, you can see that the client makes a request to the bean and the server that contains the bean are running on different Java virtual machines. They may even be on different hosts. Finally, the bean processes the request making the corresponding access to the database.

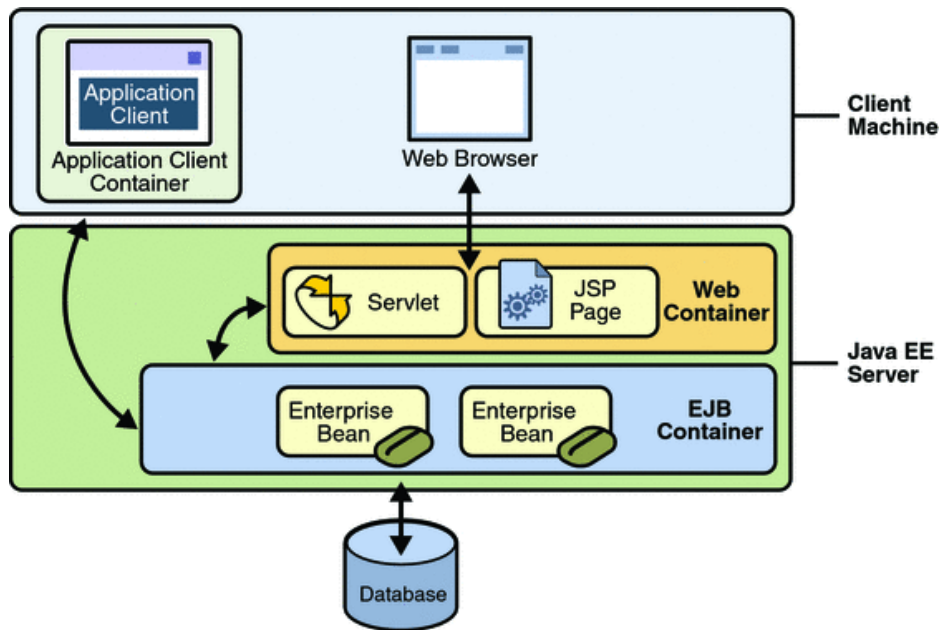


Figure 5 Representation of the enterprise beans functionality

(Oracle and/or its affiliates, 2010)

Enterprise beans are Java EE components that implement Enterprise JavaBeans (EJB) technology. Enterprise beans run in the EJB container, a runtime environment within the GlassFish Server. Although transparent to the application developer, the EJB container provides system-level services, such as transactions and security, to its enterprise beans. These services enable you to quickly build and deploy enterprise beans, which form the core of transactional Java EE applications.

In programming terms an enterprise bean is a server-side component that encapsulates the business logic of an application.

Enterprise beans are either session beans or message-driven beans.

- A session bean represents a transient conversation with a client. When the client finishes executing, the session bean and its data are gone.
- A message-driven bean combines features of a session bean and a message listener, allowing a business component to receive messages asynchronously. Commonly, these are Java Message Service (JMS) messages. (Eric, et al., 2013)

4.4 JEE tools and frameworks

4.4.1 GlassFish Server

The server selected for the development is a GlassFish Server Open Source Edition 3.1.2.

GlassFish Server provides a lightweight, modular server for the development of Java Platform Enterprise Edition (Java EE) 6 applications and Java Web Services. It delivers enterprise performance, scalability, and reliability. (Oracle and/or its affiliates, 2012)

It is a project, started by Sun Microsystems for the Java EE platform and now sponsored by Oracle Corporation. There are two modalities, the supported version is called Oracle GlassFish Server, but in my case The GlassFish that is free software, dual-licensed under two free software licenses: The Common Development and Distribution License and the GNU General Public License.

4.4.2 Ajax

AJAX is a group of interrelated web development techniques used for creation an interactive dynamic web pages or known as RIA. This kind of application is executed in the client side in asynchronous form. With Ajax, web applications can send data to, and retrieve data from, a server asynchronously (in the background) without interfering with the display and behavior of the existing page.

4.4.3 MySQL Workbench

MySQL Workbench is a unified visual tool for database architects, developers, and DBAs. MySQL Workbench provides data modeling, SQL development, and comprehensive administration tools for server configuration, user administration, and much more. MySQL Workbench is available on Windows, Linux and Mac OS. (Oracle Corporation and/or its affiliates, 2010)

4.4.4 NetBeans

NetBeans is an IDE, an open-source tool for developing with Java, JavaScript, PHP, Groovy, Apache, C/C++/Fortran (Oracle Corporation, 2011). The version used 7.3 Beta 2.

Net beans began in 1996 as Xelfi, a Java IDE student project under the guidance of the Faculty of Mathematics and Physics at Charles University in Prague. In 1997 Roman Saněk formed a company around the project and produces a commercial version of the NetBeans IDE until it was bought by Sun Microsystems in 1999. Sun open-sourced the NetBeans IDE in June of the following year. In 2010, Sun was acquired by Oracle. (Tulack, 2006)

The platform offers reusable services common to desktop applications, allowing developers to focus on the logic specific to their application. Among the features of the platform are:

- User interface management
- User settings management
- Storage management
- Window management
- Wizard framework
- NetBeans Visual Library
- Integrated development tools

NetBeans IDE is a free, open-source, cross-platform IDE with built-in-support for Java Programming Language. (Oracle and/or its affiliates, 2013)

4.4.5 GIT

Git is a Version Control System that records changes to a file or set of files over time so that you can recall specific versions later. (Chacon, 2009).

In the practical project of this thesis, GIT was a key during the development. In web design makes a single change in layout, image, url link or attribute in css file can put yourself in troubles, so keeping versions as small as possible help you in case of rolling back to previews version. I used this tool as local version control system see Figure 6 (Chacon, 2009)., but it has more powerful use like centralized version control system or distributed version control system.

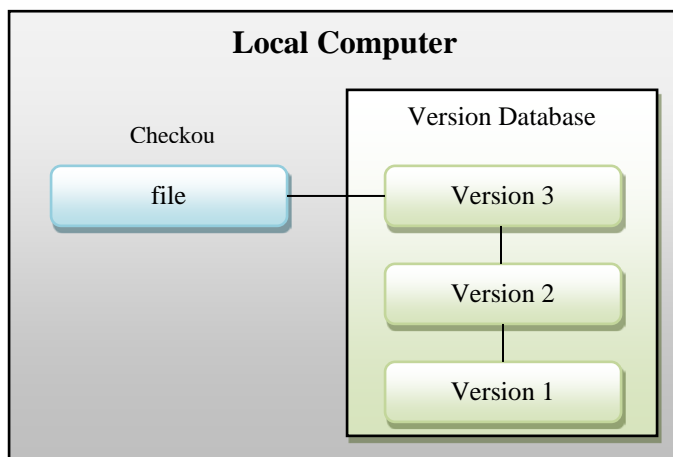
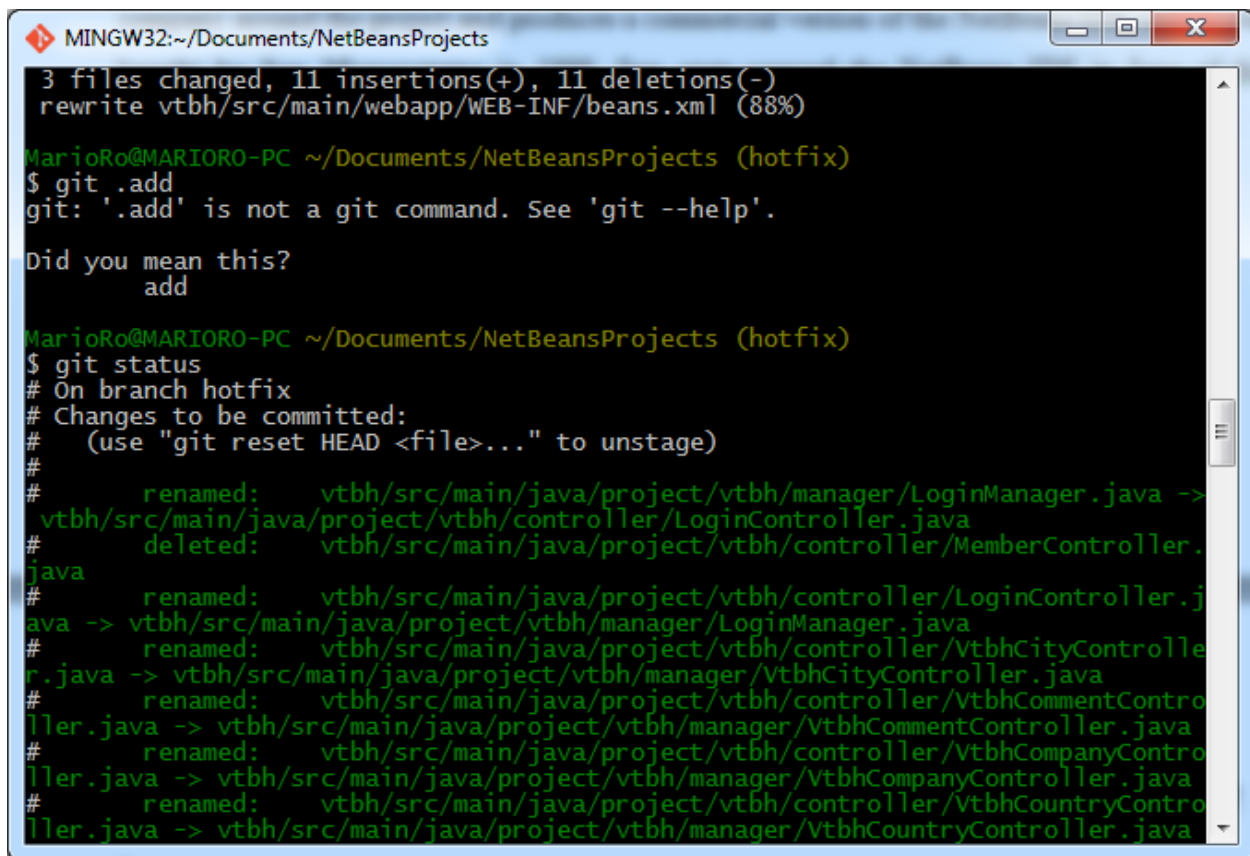


Figure 6 Local version control diagram

Git is an easy-use tool, notwithstanding when you have to use a command-line interface; see Figure 7.



```
MINGW32:~/Documents/NetBeansProjects
3 files changed, 11 insertions(+), 11 deletions(-)
rewrite vtbh/src/main/webapp/WEB-INF/beans.xml (88%)

MarioRo@MARIORO-PC ~/Documents/NetBeansProjects (hotfix)
$ git .add
git: '.add' is not a git command. See 'git --help'.

Did you mean this?
  add

MarioRo@MARIORO-PC ~/Documents/NetBeansProjects (hotfix)
$ git status
# On branch hotfix
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       renamed:    vtbh/src/main/java/project/vtbh/manager/LoginManager.java ->
vtbh/src/main/java/project/vtbh/controller/LoginController.java
#       deleted:    vtbh/src/main/java/project/vtbh/controller/MemberController.
java
#       renamed:    vtbh/src/main/java/project/vtbh/controller/LoginController.j
ava -> vtbh/src/main/java/project/vtbh/manager/LoginManager.java
#       renamed:    vtbh/src/main/java/project/vtbh/controller/VtbhCityControlle
r.java -> vtbh/src/main/java/project/vtbh/manager/VtbhCityController.java
#       renamed:    vtbh/src/main/java/project/vtbh/controller/VtbhCommentContro
ller.java -> vtbh/src/main/java/project/vtbh/manager/VtbhCommentController.java
#       renamed:    vtbh/src/main/java/project/vtbh/controller/VtbhCompanyContro
ller.java -> vtbh/src/main/java/project/vtbh/manager/VtbhCompanyController.java
#       renamed:    vtbh/src/main/java/project/vtbh/controller/VtbhCountryContro
ller.java -> vtbh/src/main/java/project/vtbh/manager/VtbhCountryController.java
```

Figure 7 Git Command-line interface

Some commands used are:

- Gitk: Start the git repository browser, please see Figure 8
Calling the command: \$ gitk + enter
- Git add: Add file contents to the index
e.g. \$ git add .
- Git commit. Record changes to the repository
e.g. \$ git commit -a -m 'Message adding added'
- Git checkout. Checkout a branch or paths to the working tree
e.g. \$ git checkout 5cce911dc3964d30914ee929d76617896b51dbf0

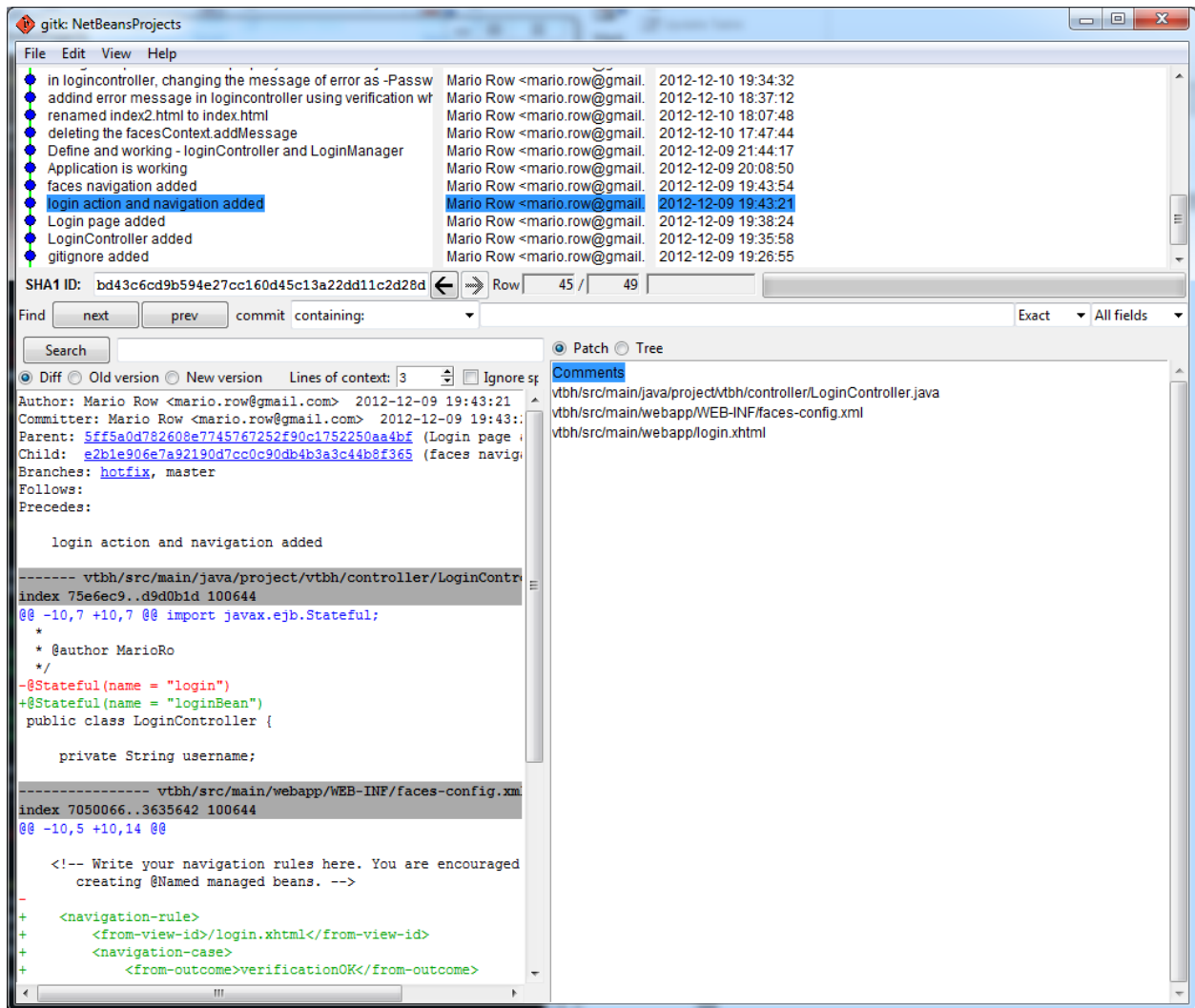


Figure 8 Git repository browser

Git is a member project of Software Freedom Conservancy. Conservancy is a not-for-profit organization that provides financial and administrative assistance to open source projects.

5

System analysis and implementation

This chapter will be focused to analyze the system in base of the list of requirements. The aim of this process is to obtain a detailed specification of the system. All the diagrams in the chapter are designed using the methodology UML, and the used tool was Visual Paradigm for UML 10.0 Community Edition.

5.1 Use Cases diagram

In this interaction model we can start understanding the relationship between the actors and the system. The use case diagram formally depict a structure of system functionality with respect to its communication with a surrounding (actors) (Vrana, 2009)

For the user cases diagram please see below the *Figure 10* and it was modeling in base of the requirement that you can see in the section 3.1.1

5.1.1 Actors

The first step in the analysis is the identification of the Actors who will interact with the system. There are two actors:

- **User:** The user will have the more active part in the system (There are the employees), because the virtual teambuilding helper will be focused in theirs activities as important key in the company organization.
- **Company:** or call administrator, the person or group of them who will be in charge of the administration part (Creation/Modification/Deleting accounts) and as well the stimulation of teambuilding activities.

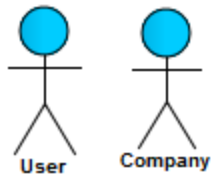


Figure 9 User Roles

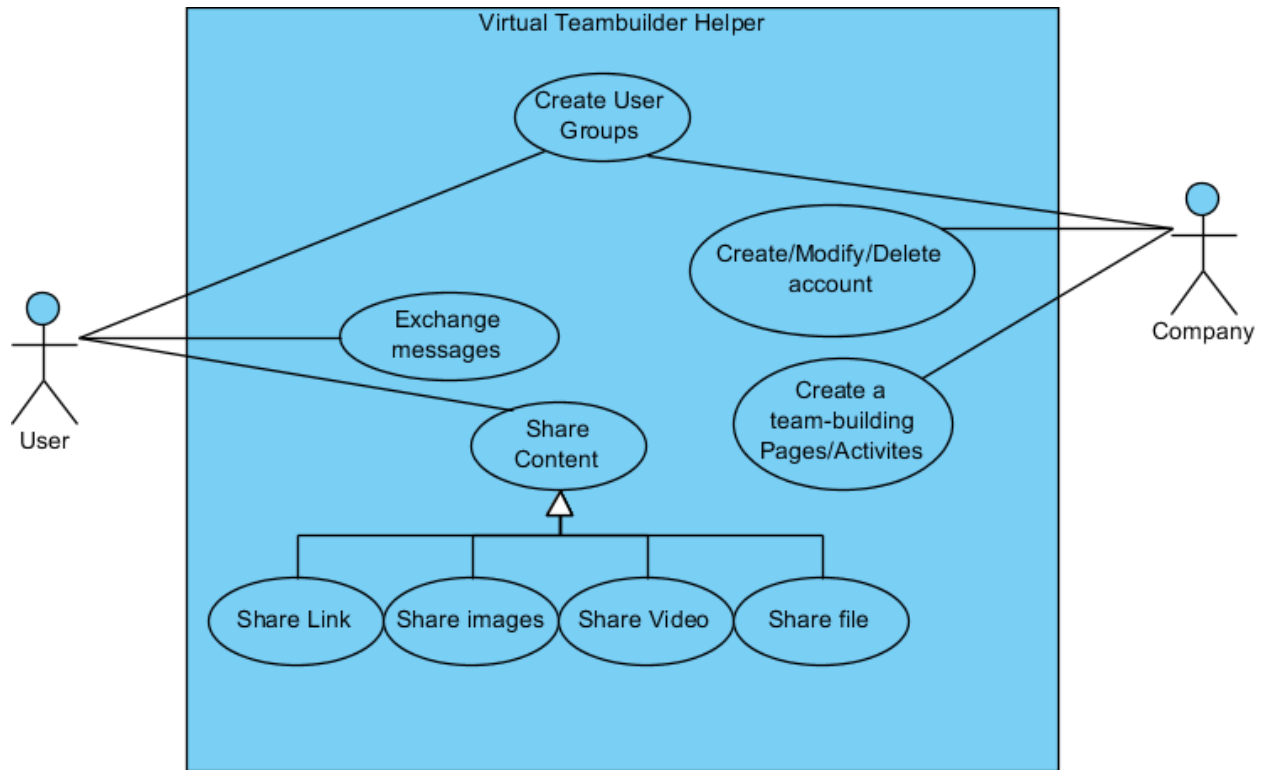


Figure 10 Uses case diagram

5.2 Classes diagram

In this part I describe all the classes that have emerged in the analysis of use cases.

In this diagram you can identify the roles of the actors in the system, their attributes, relations between them and other components that are required for its design.

Class diagram is defined as a tool for accurate and easy expression of an object model. (Vrana, 2009)

Please see below Figure 11 that shows the classes defined in Classes diagram

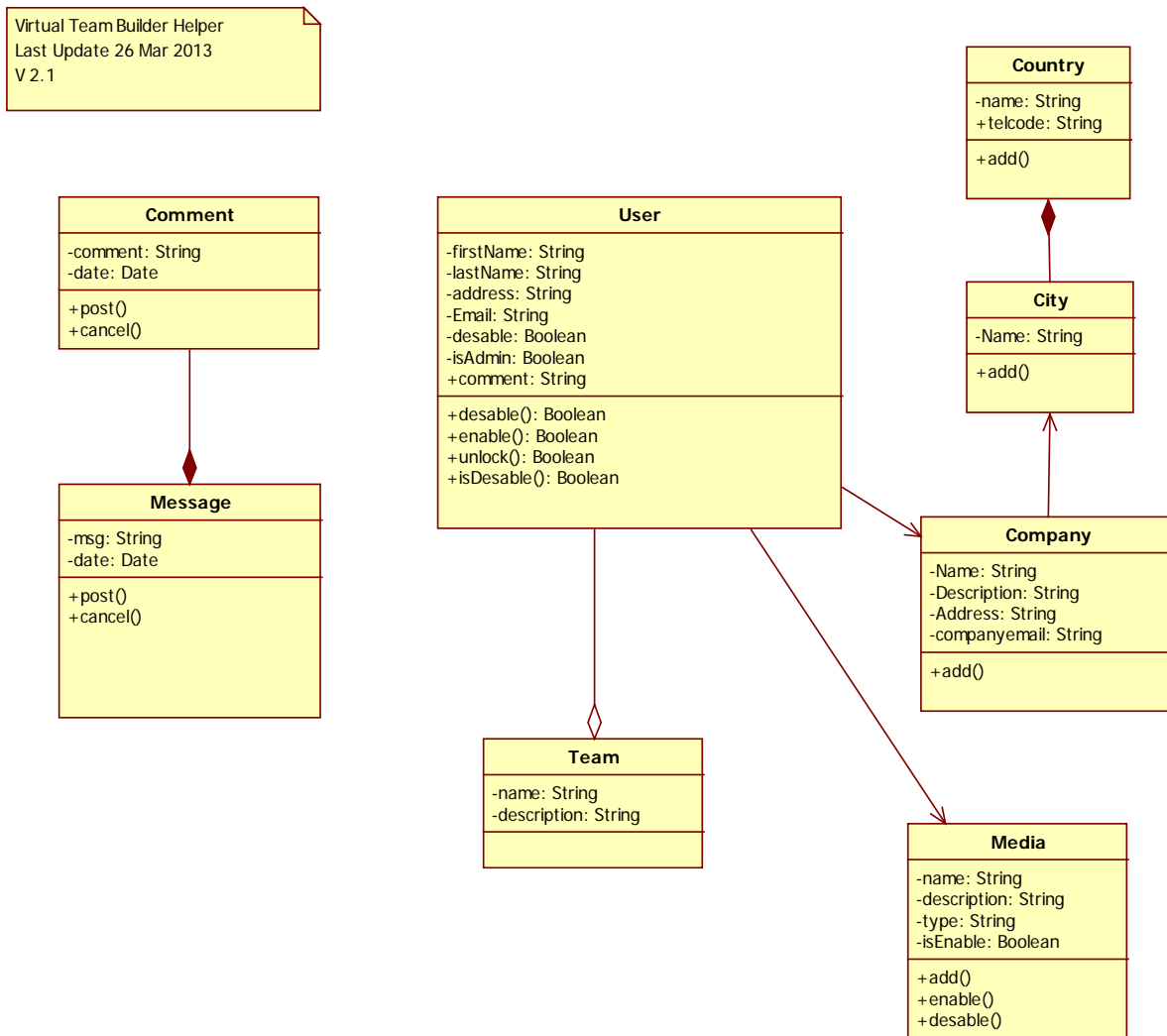


Figure 11 Class diagram

5.3 Data Model

The conceptual model is made using the tool StartUML and represents the extraction concepts and the relationships between them

Please see below Figure 12 that shows the Model diagram

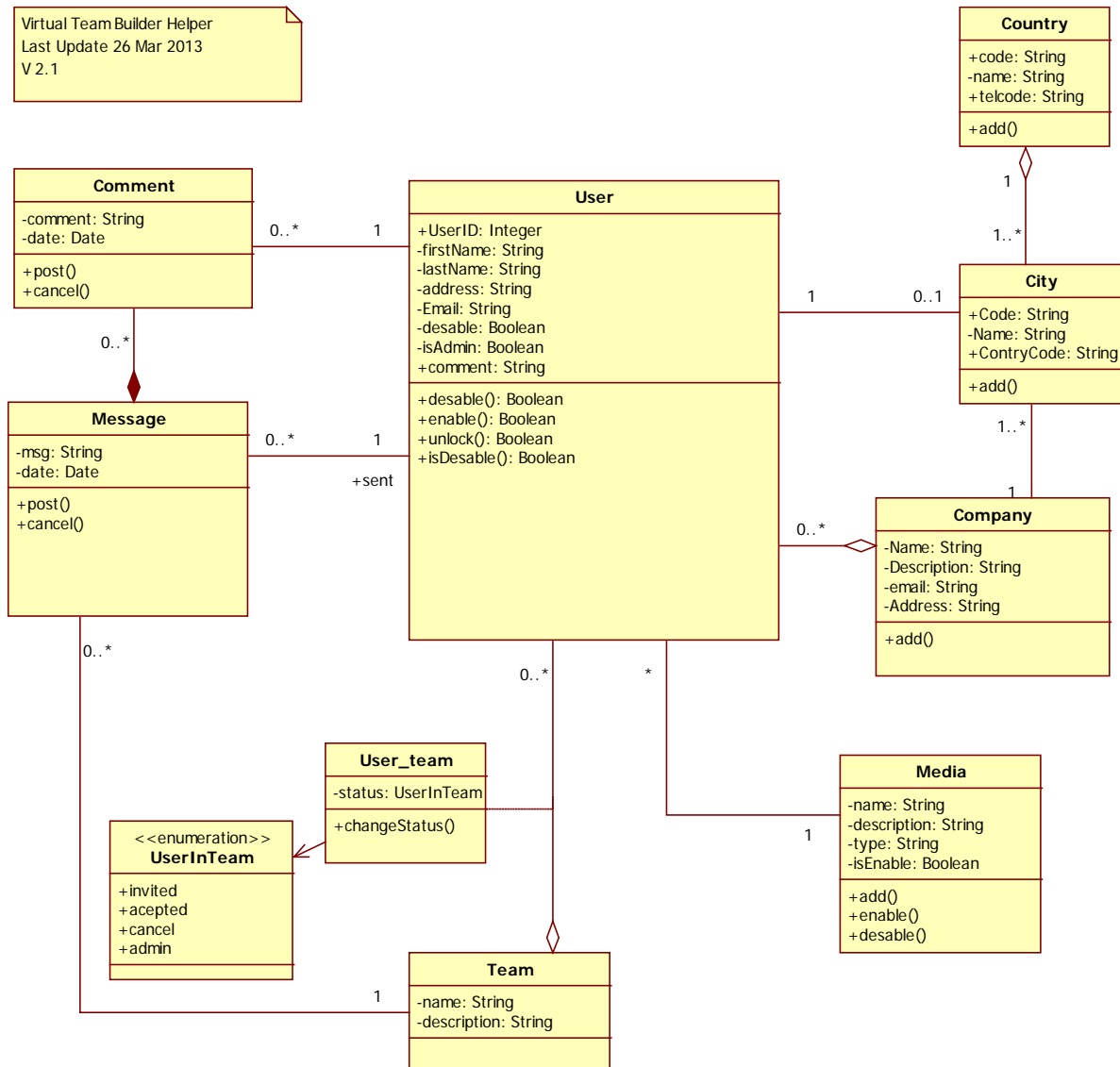


Figure 12 The Model diagram

5.4 Deployment diagram

Deployment diagrams are used to visualize the topology of the physical components of a system where the software components are deployed.

Therefore, deployment diagrams are used to describe the static deployment view of a system. It consists of nodes and their relationships.

The purpose of deployment diagrams are, visualize hardware topology of a system, describe the hardware components used to deploy software components and describe runtime processing nodes.

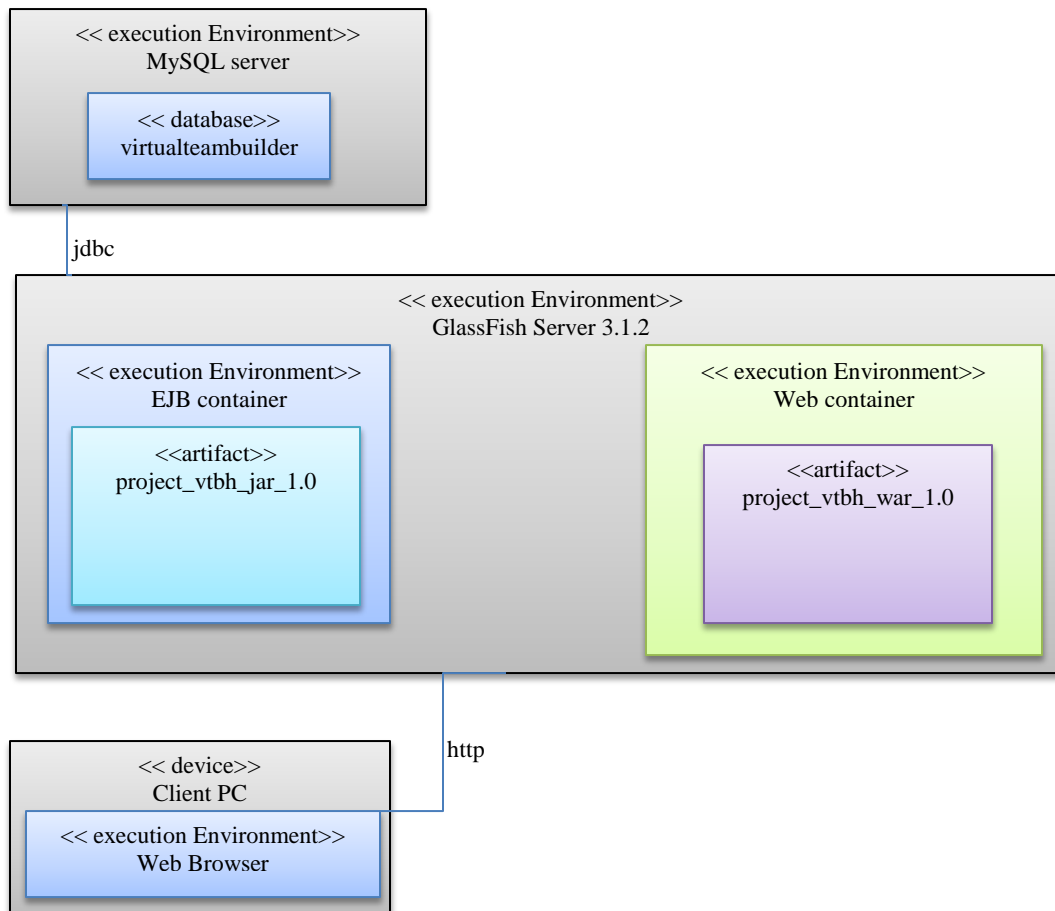


Figure 13 Deployment diagram

5.5 Interface design

The Figure 14 shows the first web page where the user has to introduce his user id and password.

The screenshot displays a web interface for 'Virtual TeamBuilder Helper'. At the top right, there is a navigation bar with links for 'Home | search | Login'. The main heading reads 'Welcome to Virtual TeamBuilder Helper!'. Below this, a sub-heading states 'This application is a practical work of my MSc Dissertation'. A section titled 'User Authentication' contains a form with the instruction 'Please, introduce your username and password'. The form includes two input fields: 'Username:' and 'Password:', each followed by a text box. Below these fields is an 'Accept' button. To the right of the login form, a grey box titled 'Teambuilding and virtualization' contains text explaining that team building is a cooperative process and lists three types of teams: Project team, Working teams, and Distributed teams or virtual teams. At the bottom center, there is a footer with the following text: 'Dissertation Project', 'Msc in System Engineering and Informatics', 'Author: Mario Alberto Row Mora', 'Supersitor: Pavel Pavlíček', and 'Czech University of live Sciences Prague ©2013'.

Figure 14 Interface

6

Testing

Let's see the definition of test: to check program correctness by trying out various sequences and input values. (Microsoft Corporation, 2002)

This chapter is focus in two main parts; the first is a quite introduction about the issue of software testing and the second is about JEE testing tools.

6.1 Testing essentials

6.1.1 V&V

In verification and validation (V&V) of software requirements and design specifications are to identify and resolve software problems and high-risk issues early in the software life-cycle. The main reason for doing this is indicated in Figure 16, (Guidelines for Verifying and Validating Software Requirements and Design Specifications, 1979). It shows that savings of up to 100:1 are possible by finding and fixing problems early rather than late in the life-cycle.

The following concepts have to be clarify,

Validation – to establish the truth of the correspondence between a software product and its specification. Corresponding with the question “Am I building the product right?”

Verification – to establish the fitness or worth of a software product for its operational mission. Corresponding with the question “Am I building the right product?” (Guidelines for Verifying and Validating Software Requirements and Design Specifications, 1979)

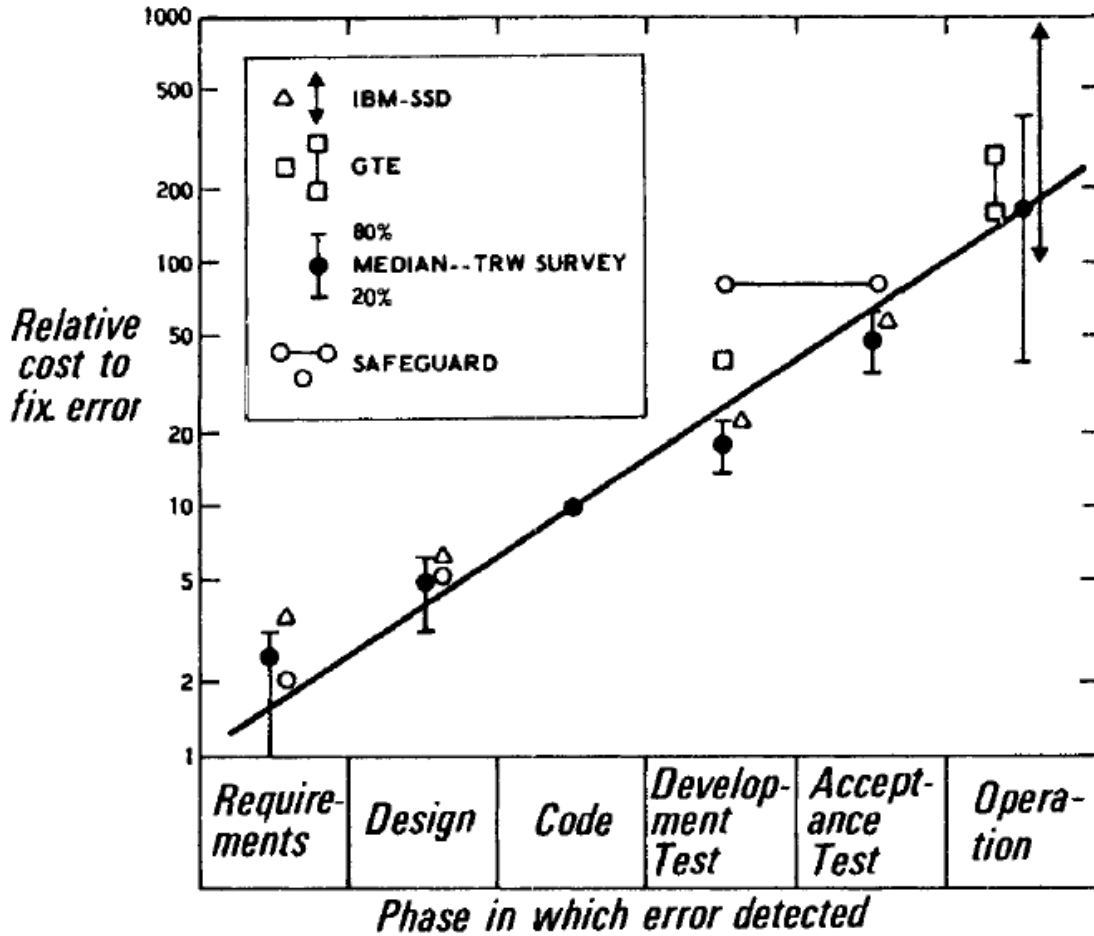


Figure 16 Early Software V&V

6.1.2 Criteria

It is important determine the measures for requirements and design specifications, as exposed by Barry W. Boehm there are four criteria and those are completeness, consistency, feasibility and testability.

Completeness, a specification is complete to the extent that all of its parts are present and each part is fully developed.

Consistency, a specification is consistent to the extent that its provisions do not conflict with each other or with governing specifications and objectives.

Feasibility, a specification is feasible to the extent that the life-cycle benefits of the system specified will exceed the life-cycle costs. Thus, feasibility involves more than verifying that a system can be developed which satisfies the functional and performance requirements. It also implies validating that the specified system will be sufficiently maintainable, reliable, and well human-engineered to keep the system's life-cycle balance sheet positive.

Further, and most importantly, it implies the identification and resolution of any high-risk issues involved, before the commitment of large numbers of people to detailed development.

And testability, a specification is testable to the extent that one can identify an economically feasible technique for determining whether or not the developed software will satisfy the specification. In order to be testable, specifications must be specific, unambiguous, and quantitative wherever possible.

6.1.3 Black-box vs. White-box

This section describes techniques for testing of software by executing the test-objects on a computer. Several different approaches are available for testing the test object. They can be categorized into two groups: black and white box testing.

6.1.3.1 Black-Box

Using black box testing, the test object is seen as a black box. Test cases are derived from the specification of the test object. The behavior of the test object is watched from the outside (PoO – Point of Observation is outside the test object). The operation sequence of the test object can only be influenced by choosing appropriate input test data or by setting appropriate preconditions. The →PoC (→ Point of Control) is also located outside of the test object. Test cases are designed by using the specification or the requirements of the test object.

Black box testing is predominantly used for higher levels of testing even though it is reasonable in component tests. Any test design before the code is written (test-first programming, test driven development) is essentially black box driven. (Spillner, et al., 2011)

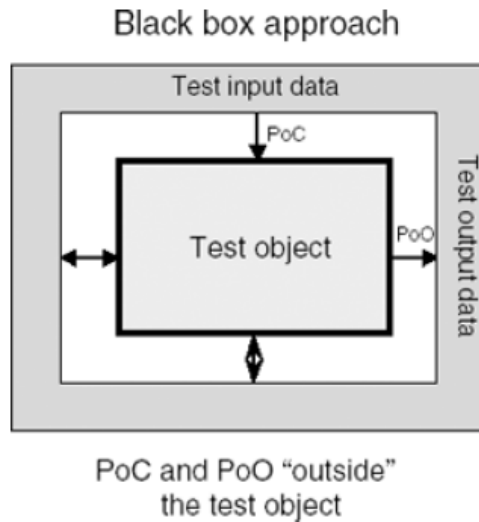


Figure 17 PoC and PoO at black box

6.1.3.2 White Box testing

In white box testing, the source code is known and used for test design. While executing the test cases, the internal processing of the test object, as well as the output, is analyzed (the Point of Observation is inside of the test object). Direct intervention in the process of the test object is possible, but should be used only in special situations, e.g. to execute negative testing when the component's interface is not capable of initiating the provoked failure (the Point of Control can be located inside the test object). Test cases are designed to cover the program structure of the test object.

White box testing is also called structural testing, because the test designer considers the structure (component hierarchy, flow control, data flow) of the test object. White box testing can be applied at the lower levels of the testing, i.e., component and integration test. (Spillner, et al., 2011)

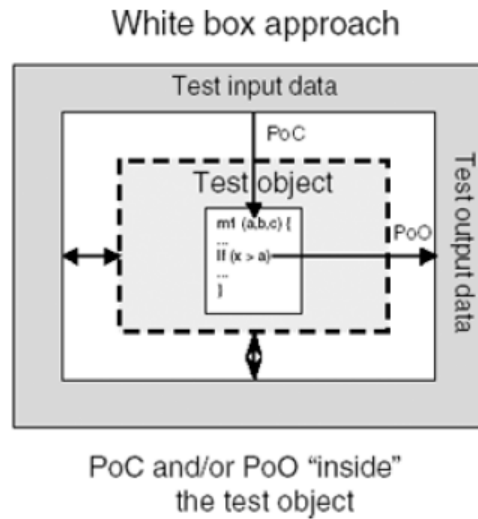


Figure 18 PoC and PoO at white box

6.2 Tools for testing

Java Enterprise Edition is a comprehensive platform for developing complex information systems. Nevertheless, there are several differences between testing web applications and JEE platform applications. This platform requires run in special environment (application server), than can be one server or distributed servers. Moreover, the Information systems are using relation databases or data warehouses of data persistence; as well with several distributed departments and heterogeneous legacy business systems. In cases we are dealing with such complicated system, it is useful to know some advanced techniques of testing, such as Mocking objects – that is a creating an artificial environment and interface, which emulates the services of an application or database server.

6.2.1 TestNG

TestNG is a testing framework for testing and working with Java and is based on JUnit. TestNG is designed to cover all categories of tests: unit, functional, end-to-end, integration, etc.

Annotations were formally added to the Java language, and TestNG made the choice to use annotations to annotate test classes. Figure 19 ((Beust, et al., 2007)) shows a basic example.

```
1. public class FirstTest {
2.     @BeforeMethod
3.     public void init() {
4.     }
5.     @Test(groups = { "unit", "functional" })
6.     public void aTest() {
7.     }
8. }
```

Figure 19 Basic TestNG test

6.2.2 Unit test

A unit test is a piece of code written by a developer that exercises a very small, specific area of functionality of the code being tested. Usually a unit test exercises some particular method in a particular context. For example, you might add a large value to a sorted list, and then confirm that this value appears at the end of the list. Or you might delete a pattern of characters from a string and then confirm that they are gone. (Hunt, et al., 2003)

Unit tests are performed to prove that a piece of code does what the developer thinks it should do.

7

Conclusion

Teambuilding is very popular way how to build good relations between members of distributed teams. The degree of stress, pressure and sometimes over-work, made the teambuilding as an open-door escape, where the members of the team could calm down emotions and make stronger the relationship between them. They are also costly and social networks offer less expensive alternative.

There are many technologies that social networks could be based on. I have used the combination of Java Enterprise Edition with Java Beans, JSF, Facelets, Richfaces, MySQL because they offer a lot of tools and functionalities. On the other side, these types of technologies could be very complex for some applications. The cost of resources, maintenance and support for applications based on this combination of technologies could be in some cases too high.

Using software for version control system helps in the development stage of every project. *Git* is a simple but powerful tool that helped me to track changes in my project and keep saving every single change.

In this master thesis my project Virtual Team Builder Helper helped me understand and gave me a huge experience about the software development and Java Enterprise Edition technology.

As a final point, further work could be to include more games and interactive real-time tools that could increase the sense of strong relationship between the members of the team. Also another point that could be improved is in terms of data security.

Bibliography

Ben, Segal . 1995. A Short History of Internet Protocols at CERN. *Ben Segal's Home Page*. [Online] April 1995. <http://ben.home.cern.ch/ben/TCPHIST.html>.

Beust, Cédric and Suleiman, Hani. 2007. *Next Generation Java Testing: TestNG and Advanced Concepts*. Boston : Addison-Wesley, 2007. ISBN 0-321-50310-4.

Chacon, Scott. 2009. *Pro Git*. New York : Appress, 2009. ISBN 978-1-4302-1834-0.

Duckett, Jon. 2010. *Beginning HTML, XHTML, CSS, and JavaScript*. Indianapolis : Wiley Publishing, Inc., 2010. ISBN: 978-1-4571-0728-3.

Eric, Jendrock, et al. 2013. *The Java EE 6 Tutorial*. [Document] Redwood : Oracle and/or its affiliates, 2013.

Fielding, R and UC, Irvine. 1999. *Hypertext Transfer Protocol -- HTTP/1.1*. 1999. p. 1. RFC2616.

Games for virtual team building. **Ellis, Jason B, Luther, Kurt and Bessiere, Katherine. 2008.** New York : s.n., 2008. The 7th ACM conference on Designing interactive systems. pp. 295-304. ISBN 978-1-60558-002-9.

Guidelines for Verifying and Validating Software Requirements and Design Specifications. **Boehm, Barry W. 1979.** [ed.] P.A. Samet. Redondo Beach : North Holland Pub. Co, 1979. Proceedings of the European Conference on Applied Information Technology of the International Federation for Information Processing. pp. 711-719. ISBN 0444853707.

Harvard Business Publishing. 2001. Communicating with Virtual Project Teams/Creating Successful Virtual Organizations - Virtual Communication. *Harvard Business School Working Knowledge Archive*. [Online] 3 26, 2001. [Cited: 3 112, 2013.] <http://hbswk.hbs.edu/archive/2122.html>.

Heffelfinger, David R. 2007. *Java EE 5 Development using GlassFish Application Server*. Birmingham : Packt Publishing Ltd., 2007. ISBN 978-1-847192-60-8.

Hunt, Andrew and Thomas, David. 2003. *Pragmatic Unit Testing: In java with JUnit.* Raleigh : The Pragmatic Programmers, LLC., 2003. ISBN 0-9745140-1-2.

Jendrock, Eric, et al. 2007. *The Java EE 5 Tutorial. For Sun Java System Application Server 9.1.* Redwood : Oracle and/or its affiliates, 2007.

Microsoft Corporation. 2002. *Microsoft Computer Dictionary.* Washington : Microsoft Press, 2002. Vol. V. ISBN 0-7356-1495-4.

Object Management Group, Inc. 2005. Introduction To OMG's Unified Modeling Language™ (UML®). *Object Management Group.* [Online] July 2005. [Cited: January 10, 2012.] http://www.omg.org/gettingstarted/what_is_uml.htm.

Oracle and/or its affiliates. 2012. *GlashFish Server Open Source Edition. Release Notes. Release 3.1.2 and 3.12.2.* July 2012.

—. **2013.** Java EE and Web Application Development. *NetBeans.* [Online] 2013. [Cited: 3 13, 2013.] <http://netbeans.org/features/java-on-server/>.

—. **2010.** Securing Java EE Applications. [book auth.] Eric Jendrock. *The Java EE 5 Tutorial. For Sun Java System Application Server 9.1.* s.l. : Oracle and/or its affiliates, 2010, p. 796.

Oracle Corporation and/or its affiliates. 2010. MySQL Workbench 5.2. *MySQL.com.* [Online] March 26, 2010. [Cited: October 23, 2012.] http://wb.mysql.com/?page_id=49.

Oracle Corporation. 2011. NetBeans IDE 7.0.1 Release Notes. *NetBeans.* [Online] October 17, 2011. [Cited: January 23, 2012.] <http://netbeans.org/community/releases/70/relnotes.html>.

Peragine, John N. 1970. *365 Low or no cost workplace teambuilding activities.* Florida : Atlantic Publishing Group, Inc., 1970. 978-1-60138-043-2.

Scott, John. 2000. *Social Network Analysis a handbook.* 2nd. London : SAGE Publications Ltd, 2000. p. 8. ISBN 978-0-7619-6339-4.

Spillner, Andreas, Linz, Tilo and Schaefer, Hans. 2011. *Software Testing Foundations.* [ed.] Michael Barabas. 3rd. Santa Barbara : Rocky Nook Inc., 2011. ISBN: 978-1-933952-78-9.

Tulack, Jaroslav. 2006. Happy Birthday NetBeans -- interview with Jaroslav "Yarda" Tulach. [interv.] Roman Strobl. *NetBeans Interviews - Yarda Tulach.* 10 11, 2006. <http://netbeans.org/community/articles/interviews/yarda-tulach.html>.

Vaniček, Jiří, et al. 2008. *Mathematical foundations of computer sciences.* Prague : Kernberg Publishing, s.r.o.& Alfa Publishing, s.r.o., 2008. ISBN 978-80-87168-06-6.

Vrana, Ivan. 2009. *Projecting of information systems with UML.* Prague : Česká zemědělská univerzita v Praze - Provozně ekonomická fakulta, 2009. p. 59. ISBN 978-80-213-1976-9.

Wasserman, Stanley, Faust, Katherine and Iacobucci, Dawn. 1994. *Social Network Analysis: Methods and Applications (Structural Analysis in the Social Sciences).* Cambridge : Cambridge University Press, 1994. p. 3. ISBN 0 521 3870 8.

Weaver, James L., Mukhar, Kevin and Crume, Jim. 2004. *Beginning J2EE1.4: From Novice to Professional.* New York : Springer-Verlag, 2004.

Wong, Clinton. 2000. *HTTP Pocket Reference.* Sebastopol : O'Reilly & Associates, 2000. ISBN 1-5659-862-8.

Appendices

Appendix A - Installation Guide

1. Download *GlassFish AS 3.1.2 Final* from the side:

<http://glassfish.java.net/downloads/3.1.2-final.html>

2. Unzip the downloaded package
3. Start the *GlashFish Server* from the command line use:

```
asadmin start-domain -verbose
```

*With no arguments, the start-domain command initiates the default domain, which is domain1.

4. Download MySQL 5.5.28 from the side:

<http://www.mysql.com/downloads/>

5. Configuration of use and password to access the database

```
$ mysql --user=root --password  
(Will ask the password as administrator, by default it is empty)  
mysql> GRANT ALL PRIVILEGES ON *.* TO 'scs'@'localhost' IDENTIFIED BY 'scs' WITH  
GRANT OPTION;  
mysql> GRANT ALL PRIVILEGES ON *.* TO 'scs'@'%' IDENTIFIED BY 'scs' WITH GRANT  
OPTION;  
mysql> exit
```

6. Create the database from “SQLscript\VTBHDatabase.sql”

```
$ mysql --user=scs --password=scs < ejemplo-JEE.sql
```

7. Deploy the program

```
as-install/bin/asadmin deploy vtbh.war
```

8. Access the *Virtual TeamBuilder Helper* application by typing the following URL in your browser:

<http://localhost:8080/vtbh>

9. Stop the application server GlassFish, from the command line user:

```
asadmin stop-domain domain1
```


Appendix B - SQL Commands constraints

The code of the implementation in SQL is using MySQL Community Server:

```
1. -- vtbh_userUserID-- Database: `virtualteambuilder`
2. USE virtualteambuilder;
3.
4. DROP TABLE IF EXISTS `Vtbh_Team_Users`;
5. DROP TABLE IF EXISTS `Vtbh_Boundary_Access`;
6. DROP TABLE IF EXISTS `Vtbh_Media`;
7. DROP TABLE IF EXISTS `Vtbh_Comment`;
8. DROP TABLE IF EXISTS `Vtbh_Message`;
9. DROP TABLE IF EXISTS `Vtbh_Team`;
10. DROP TABLE IF EXISTS `Vtbh_User`;
11. DROP TABLE IF EXISTS `Vtbh_Company`;
12. DROP TABLE IF EXISTS `Vtbh_City`;
13. DROP TABLE IF EXISTS `Vtbh_Country`;
14.
15.
16. -----
17. -- Table structure for table `vtbh_Country`
18. CREATE TABLE `Vtbh_Country` (
19.   `code` CHAR(3) NOT NULL,
20.   `name` CHAR(80) NOT NULL DEFAULT '',
21.   `telCode` CHAR(8) DEFAULT NULL,
22.   PRIMARY KEY (`Code`)
23. );
24.
25. -----
26. -- Table structure for table `vtbh_City`
27. CREATE TABLE `Vtbh_City` (
28.   `code` INT(11) NOT NULL AUTO_INCREMENT,
29.   `name` CHAR(100) NOT NULL DEFAULT '',
30.   `Countrycode` CHAR(3) NOT NULL DEFAULT '',
31.   PRIMARY KEY (code),
32.   CONSTRAINT FK_City_Country FOREIGN KEY (Countrycode) REFERENCES vtbh_Country (`code`)
33. );
34.
35. -- Table structure for table `vtbh_Company`
36. CREATE TABLE `vtbh_Company` (
37.   `name` VARCHAR(100) NOT NULL DEFAULT '',
38.   `description` VARCHAR(500) NOT NULL DEFAULT '',
39.   `address` VARCHAR(50) NOT NULL DEFAULT '',
40.   `email` VARCHAR(75) NOT NULL DEFAULT '',
41.   `CompanyID` INTEGER NOT NULL,
42.   PRIMARY KEY (CompanyID),
43.   `Citycode` INT(11),
44.   CONSTRAINT FK_Company_City FOREIGN KEY (Citycode) REFERENCES vtbh_City (`code`)
45. );
```

```

46. -- -----
47. -- Table structure for table `vtbh_User`
48. CREATE TABLE `vtbh_User` (
49.   `UserID` INTEGER NOT NULL,
50.   `Username` VARCHAR(35) NOT NULL DEFAULT 0,
51.   `firstName` VARCHAR(70) NOT NULL,
52.   `lastName` VARCHAR(70) NOT NULL,
53.   `address` VARCHAR(100) NOT NULL DEFAULT '',
54.   `email` VARCHAR(75) NOT NULL,
55.   `pass` VARCHAR(256) NOT NULL,
56.   `isLocked` BOOLEAN NOT NULL DEFAULT 0,
57.   `desable` BOOLEAN NOT NULL DEFAULT 0,
58.   `hasChangePass` BOOLEAN NOT NULL DEFAULT 1,
59.   `lastLoginDate` DATETIME NOT NULL,
60.   `lastPassChange` DATETIME NOT NULL,
61.   `isAdmin` BOOLEAN NOT NULL DEFAULT 0,
62.   `failedPassAttemptCount` INT(11) NOT NULL,
63.   `comment` VARCHAR(100) NOT NULL DEFAULT '',
64.   `version` INT(11) DEFAULT NULL,
65.   PRIMARY KEY (`UserID`),
66.   `idCompany` INT(11),
67.   `Citycode` INT(11),
68.   CONSTRAINT `FK_User_Company` FOREIGN KEY (`idCompany`) REFERENCES `vtbh_Company` (
`CompanyID`),
69.   CONSTRAINT `FK_User_City` FOREIGN KEY (`Citycode`) REFERENCES `vtbh_City` (`code`)
70. );
71. -- -----
72. -- Table structure for table `vtbh_Media`
73. CREATE TABLE `vtbh_Media` (
74.   `mediaID` INTEGER NOT NULL AUTO_INCREMENT,
75.   `name` VARCHAR(100) NOT NULL,
76.   `description` VARCHAR(100) NOT NULL DEFAULT '',
77.   `type` VARCHAR(3) NOT NULL,
78.   `url` VARCHAR(3) NOT NULL,
79.   `isEnabled` BOOLEAN NOT NULL DEFAULT 1,
80.   `data` LONGBLOB NOT NULL,
81.   PRIMARY KEY (`MediaID`),
82.   `UserID` INTEGER,
83.   CONSTRAINT `FK_Media_User` FOREIGN KEY (`UserID`) REFERENCES `vtbh_User` (`UserID`
)
84. );
85. -- -----
86. -- Table structure for table `vtbh_Team`
87. CREATE TABLE `vtbh_Team` (
88.   `teamID` INTEGER NOT NULL AUTO_INCREMENT,
89.   `name` VARCHAR(70) NOT NULL UNIQUE,
90.   `description` TEXT NOT NULL,
91.   PRIMARY KEY (`teamID`)
92. );
93. -- -----
94. -- Table structure for table `team_User`
95. CREATE TABLE `vtbh_Team_Users` (
96.   `teamID` INTEGER NOT NULL,
97.   `UserID` INTEGER NOT NULL,
98.   `status` INT(2) NOT NULL DEFAULT 0,
99.   PRIMARY KEY (`teamID`,`UserID`),
100.   CONSTRAINT FK_TeamUser_Team FOREIGN KEY (`teamID`) REFERENCES `vtbh_team` (
`teamID`),
101.   CONSTRAINT FK_TeamUser_User FOREIGN KEY (`UserID`) REFERENCES `vtbh_User` (
`UserID`)
102. );

```

```

103.      -----
104.      -- Table structure for table `message`
105.      CREATE TABLE `Vtbh_Message` (
106.        `messageID` INTEGER NOT NULL AUTO_INCREMENT,
107.        `msg` LONGTEXT NOT NULL,
108.        `DATE` DATETIME NOT NULL,
109.        `isPostTo` INT(2) NOT NULL DEFAULT 0 COMMENT 'Specify to who can see the
        comment. 0. Just the owner. 1. Public. 2. Friends. 3. Specific team. 4. specific list
        of colleagues',
110.        PRIMARY KEY (`messageID`),
111.        `postBy` INTEGER NOT NULL,
112.        `postToTeam` INTEGER,
113.        CONSTRAINT FK_Message_User FOREIGN KEY (`postBy`) REFERENCES `Vtbh_User` (`
        UserID`),
114.        CONSTRAINT FK_Message_Team FOREIGN KEY (`postToTeam`) REFERENCES `vtbh_Team
        ` (`teamID`)
115.      );
116.      -----
117.      -- Table structure for table `Comment`
118.      CREATE TABLE `Vtbh_Comment` (
119.        `commentID` INT(11) NOT NULL AUTO_INCREMENT,
120.        `msg` LONGTEXT NOT NULL,
121.        `DATE` DATETIME NOT NULL,
122.        PRIMARY KEY (`commentID`),
123.        `postBy` INTEGER NOT NULL,
124.        `messageID` INT(11) NOT NULL,
125.        CONSTRAINT FK_Comment_User FOREIGN KEY (`postBy`) REFERENCES `vtbh_User` (`
        UserID`),
126.        CONSTRAINT FK_Comment_Message FOREIGN KEY (`messageID`) REFERENCES `vtbh_Me
        ssage` (`messageID`)
127.      );
128.      -----
129.      -- Table structure for table `Vtbh_Boundaryaccess`
130.      CREATE TABLE `Vtbh_Boundary_Access` (
131.        `messageID` INTEGER NOT NULL,
132.        `postAccessTo` INTEGER NOT NULL,
133.        PRIMARY KEY (`messageID`,`postAccessTo`),
134.        CONSTRAINT FK_Boundaryaccess_User FOREIGN KEY (`postAccessTo`) REFERENCES `
        vtbh_User` (`UserID`),
135.        CONSTRAINT FK_Boundaryaccess_Message FOREIGN KEY (`messageID`) REFERENCES `
        vtbh_Message` (`messageID`)
136.      );

```

Appendix C – Some source code

The following code is the Login.xhtml file

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <ui:composition xmlns="http://www.w3.org/1999/xhtml"
3.                 xmlns:ui="http://java.sun.com/jsf/facelets"
4.                 xmlns:f="http://java.sun.com/jsf/core"
5.                 xmlns:h="http://java.sun.com/jsf/html"
6.                 template="/WEB-INF/templates/default.xhtml">
7.     <ui:define name="content">
8.         <h1>
9.             Welcome to <br/>
10.            Virtual TeamBuilder Helper!</h1>
11.
12.        <div>
13.            <p>This application is a practical work of my MSc Dissertation</p>
14.            <br/>
15.            <h3>User Authentication</h3>
16.        </div>
17.
18.        <h:form id="reg">
19.            <h2>Please, introduce your username and password</h2>
20.            <h:panelGrid columns="3" columnClasses="titleCell">
21.                <h:outputLabel for="usernameInput">Username: </h:outputLabel>
22.                <h:inputText id="usernameInput" value="#{loginBean.username}" required
23.                ="true" />
24.                <h:message for="usernameInput" errorClass="invalid"/>
25.                <h:outputLabel for="passwordInput">Password: </h:outputLabel>
26.                <h:inputSecret id="passwordInput" value="#{loginBean.password}" requir
27.                ed="true" />
28.                <h:message for="passwordInput" errorClass="invalid"/>
29.            </h:panelGrid>
30.            <p>
31.                <h:panelGrid columns="2">
32.                    <h:commandButton value="Accept" action="#{loginBean.verify}" />
33.                    <h:messages styleClass="messages" errorClass="invalid" infoClass
34.                    ="valid" warnClass="warning"
35.                    globalOnly="true"/>
36.                </h:panelGrid>
37.            </p>
38.        </h:form>
39.
40.        <br/><br/><br/>
41.    </ui:define>
42.    <ui:define name="aside">
43.        <h3> Teambuilding and virtualization</h3>
44.        <p>Team Building is the cooperative process that a group of individuals
45.        uses to solve both physical and mental challenges</p>
46.        <p>There are many different kinds of teams, the next classification is
47.        taking in consideration the work environment: </p>
48.        <ul>
49.            <li><b>Project team</b></li>
50.            <li><b>Working teams </b></li>
51.            <li><b>Distributed teams or virtual teams</b> </li>
52.        </ul>
53.    </ui:define>
54. </ui:composition>
```