Supporting Distributed Software Teams with 3D Virtual Worlds

Design and prototype of a 3D virtual world environment for virtual collaboration in software engineering

Master's Thesis

at

Graz University of Technology

submitted by

Mirza Hadžić

Supervisor: Univ.-Doz. Dipl.-Ing. Dr.techn. Christian Gütl

Institute for Information Systems and Computer Media (IICM),
Graz University of Technology
A-8010 Graz, Austria

March, 2010

Unterstützung von verteilten Softwareteams mittels 3D virtuelle Welten

Entwurf und Prototyp einer virtuellen 3D-Welt Umgebung für die virtuelle Zusammenarbeit im Software-Engineering

Masterarbeit

an der

Technischen Universität Graz

vorgelegt von

Mirza Hadžić

Betreuer: Univ.-Doz. Dipl.-Ing. Dr.techn. Christian Gütl

Institut für Informationssysteme und Computer Medien (IICM),
Technische Universität Graz
A-8010 Graz, Österreich

März, 2010

Abstract

Today, software engineering projects are prevalently performed by development teams that are geographically distributed over two or more locations. However, this trend also introduces a number of complications when it comes to communication between stakeholders, coordination of work, and control of the project. This thesis aims to develop a prototype of an environment in a 3D virtual world which supports distributed software teams. In this environment, virtual team members share a common space, communicate through multiple communication channels and have better team awareness, which ultimately helps them to complete software projects.

The theoretical part of this thesis investigates the changes that have evolved in software engineering and software project management disciplines as a consequence of distribution of software team members. Furthermore, it elaborates the complexities within these teams as well as different technologies that have been employed in order to connect geographically distant co-workers. Finally, the virtual world technology and its opportunities in the context of distributed software projects are presented.

The practical part is based on the findings of the theoretical part of the thesis. Based on the issues of virtual software teams that are discovered, one part of the requirements for a virtual world tool is defined. Together with the requirements of the software process itself, these requirements are implemented into a prototype that should enable distributed software teams to perform their work more efficiently and with greater ease. The prototype is based on Sun's Project Wonderland.

A user study was conducted which shows that environments in virtual worlds can support the software development and project management disciplines in distributed settings. Richer and numerous communication possibilities, different applications and the ability to work with team colleagues in a shared space help distributed software teams to overcome the barriers of physical separation. The only prerequisite for the successful use of such environments is their high performance.

Kurzfassung

Heute werden Software-Engineering-Projekte überwiegend von Entwicklungsteams, die geographisch über zwei oder mehrere Standorte verteilt sind, durchgeführt. Doch diese Entwicklung führt auch eine Reihe von Komplikationen bezüglich der Kommunikation zwischen den Stakeholders, der Koordinierung der Arbeiten und der Kontrolle der Projektabwicklung ein. Das Ziel dieser Arbeit ist es, einen Prototyp einer 3D virtuellen Umgebung, welche verteilte Software-Teams unterstützt, zu entwickeln. In dieser Umgebung teilen die virtuellen Team-Mitglieder einen gemeinsamen Raum, kommunizieren über mehrere Kommunikationskanäle und haben besseres Teambewusstsein, was ihnen letztlich hilft, Software-Projekte abzuschliessen.

Der theoretische Teil der Arbeit untersucht die Veränderungen, die im Software-Engineering und Software-Projekt Management-Disziplinen als Folge der Verteilung von Software-Team-Mitgliedern entwickelt haben. Darüber hinaus erarbeitet dieser die Komplexität innerhalb dieser Teams, sowie verschiedene Technologien die eingesetzt wurden, um geographisch weit voneinander entfernten Kollegen in Verbindung zu treten. Schließlich wird die virtuelle 3D-Welt-Technologie und ihre Möglichkeiten im Rahmen der verteilten Software-Projekte vorgestellt.

Der praktische Teil beruht auf die im theoretischen Kapitel der Arbeit erlangten Ergebnisse. Basierend auf den Problemen der virtuellen Software-Teams, welche entdeckt wurden, wird ein Teil der Anforderungen für einen virtuellen-Welt Tool definiert. Zusammen mit den Anforderungen an die Softwareprozess selbst, diese Anforderungen sind in einen Prototyp umgesetzt, mit deren Hilfe verteilte Software-Teams ihre Arbeit effizienter und mit größerer Leichtigkeit ausführen sollten. Der Prototyp basiert auf Sun's Project Wonderland.

Eine Benutzer-Studie wurde durchgeführt, die zeigt, dass Umgebungen in virtuellen Welten die Software-Entwicklung und Projekt-Management-Disziplinen in verteilten Einstellungen unterstützen können. Reichere und zahlreichere Möglichkeiten der Kommunikation, verschiedene Anwendungen und die Fähigkeit, mit den Teamkollegen in einem gemeinsamen Raum zu arbeiten, helfen verteilten Software-Teams die Hindernisse der physischen Trennung zu überwinden. Die einzige Voraussetzung für den erfolgreichen Einsatz solcher Umgebungen ist ihre hohe Leistung.

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

……. ……..

date (signature)

Dankaussagung

Mein besonderer Dank gilt Herrn Univ.-Doz. Dipl.-Ing. Dr.techn. Christian Gütl, der mir die Möglichkeit geboten hat diese Arbeit zu erstellen und mich über die gesamte Dauer dieser Arbeit umfassend betreut zu haben.

Ein herzliches Dankeschön gilt meiner Familie und ganz besonders meinen Eltern, Mufid und Kerima, die mir meine Ausbildung ermöglicht haben und auf deren Unterstützung ich immer zählen konnte.

Mein Dank gilt auch meinen besten Freunden hier in Graz, die mich mit vielen glücklichen Momenten während meines Studiums beschänkt haben und mich die ganze Zeit unterstützt haben.

Mirza Hadžić

Graz, März 2010

Contents

List of Figures

Chapter 1

Introduction

The globalization phenomenon has had a huge impact on all businesses and industries, and software engineering is no exception. Also, over the last decade, software has become a competitive weapon and the success of almost every business depends on its successful use and its successful development (Herbsleb & Moitra, 2001).

This dependence on software engineering has led to the transformation of the way the software is being developed today. Hence, the software is increasingly being developed by teams whose members are distributed geographically. This distributed development carries many potential opportunities. Some of these opportunities, like cost reduction and shorter times to market, promise economic benefits, whereas others, like shortage of local skilled workforce, essentially force companies into this business model (Mockus & Herbsleb, 2001).

However, the distance between team members imposes many challenges and complexities into the software development process and thus toughens the achievement of these opportunities. As software project management is essential to the success of software engineering endeavors, it is also challenged by the same issues. The most cumbersome issues include inadequate communication and project and process management issues (Herbsleb & Moitra, 2001).

This thesis aims to develop a prototype of a virtual environment to support distributed software engineering projects. The goals of the environment are to provide better communication channels, enhance team awareness, support team meetings, project and knowledge management, and provide functionality needed to complete software engineering tasks.

1.1 Motivation and Objective of the Work

In order to counter the complexities that emerge from geographical distance and to enhance the work conditions for members of distributed teams, a number of communication technologies like e-mail, audio conferences, videoconferences and others, have been developed in the past years. However, due to very limited capabilities of these technologies when it comes to conveying non-verbal and emotional cues, they had very little success (Wainfan & Davis, 2004).

Virtual world technology has many advantages over traditional communication technologies. Moreover, in recent years, many companies have recognized this potential and have started to exploit them for various reasons that range from marketing to providing their workers with a common virtual workspace. Hence, it raises a research interest how this technology could apply specifically to the software engineering. This work aims to develop a prototype of an environment in a virtual world that would enable distant software developers to overcome the barriers and difficulties faced by physical separation.

1.2 Structure of the Work

Chapter 2 introduces the most interesting and complex form of distributed software development, globally distributed software development (GSD). Furthermore, motivations for GSD as well as its issues and challenges will be presented. Finally, a number of propositions for organizations that want to maximize the probability of a successful GSD project will be given.

Due to the fact that software project management is closely related to the software development, it is also a subject to changes in a distributed setting. Hence, Chapter 3 presents distributed project management (DPM). Various types of projects will be defined together with the needs of distributed project managers, enablers of DPM, and several best practices for DPM.

Chapters 2 and 3 aim to present the problem that exists in current software development projects. As software teams present the core of any project, they are the focus of Chapter 4. This Chapter introduces the differences between traditional and software teams with distant team members, or virtual software teams. Besides elaborating the effects of distance on the workflow of a software team, this Chapter will present the issues and problems that software teams face in a distributed environment.

Chapter 5 describes the difference between traditional and virtual collaboration as well as elaborates on the characteristics of collaborative work in general. It also presents different tasks that software engineers need to perform in an average project. Finally, different modes of communication and different technologies used in context of virtual collaboration are described.

Chapter 6 presents the virtual world technology, covering topics such as definition, characteristics, evolution, and state-of-the-art in this technology. Next, it elaborates on the characteristics and requirements of virtual worlds and their application in the virtual collaboration context. In addition, this Chapter describes the opportunities and challenges in the application of virtual worlds and the work that relates to the work in this master thesis.

Chapter 7 begins the practical work in this thesis. It presents the requirements for the prototype of the virtual collaboration environment. It focuses also on conceptual architecture as well as on the visual design of the environment.

Following on the results from previous Chapter, Chapter 8 describes the process of the implementation of the virtual environment. It starts by choosing the specific 3D virtual world in which the environment is to be implemented. Next, it presents the virtual world in more detail as well as enumerates the used applications and tools. Furthermore, the process of setup and implementation is described.

Chapter 9 shows the possible applications and capabilities of the prototype. It describes how virtual software engineers can use richer communication channels, how the environment fosters team awareness. It also elaborates on how virtual teams can organize and hold meetings as well as presents shared applications which software engineers can use for software engineering and project management purposes. Finally, it presents the evaluation of the virtual collaboration environment. The research questions are presented as well as the process evaluation. This Chapter also presents the results of the evaluation and draws the conclusion for the prototype. Finally, Chapter 10 describes the learned lessons and Chapter 11 concludes the thesis and provides an outlook.

Chapter 2

Globally Distributed Software Development

The advent of globalization had a major impact on the world business environments. The globalization changed the way industries perform their business and thus also had an effect on the software industry. This change in software industry displays itself in a new way of conducting software development projects. The current Chapter explains this new trend in software development, the motivations that support it and the challenges it introduces to the software industry.

2.1 Definition and Characteristics

In the last decade, we have been witnesses of increased industrial globalization. (Friedman, 2005) stated that “the world is flat” and also named the ten flatteners that made the world flat:

1. *The fall of Berlin wall on 9.11.1989*
2. *Netscape went public on 9.8.1995*
3. *Work Flow Software (Virtual applications working together over a networking)*
4. *Open-Sourcing*
5. *Outsourcing*
6. *Offshoring*
7. *Supply-Chaining*
8. *Insourcing*
9. *In-forming (Search engines)*
10. *The Steroids*

Globalization has increased the connectivity and integration in the political, cultural, social, economic, and technological systems between nations, corporations, households, and individuals. It has also led to emergence of worldwide production/consumer markets, emergence of worldwide financial markets, spread of political sphere of interests, increase in information flows, and emergence of new way of developing information systems (Gütl & Chang, 2008; GlobalizationWiki, 2009). A wide range of industries has recognized the vast potential that the globalization produced. Numerous examples can be seen in manufacturing, automotive, financial, retail, and other sectors. High-speed telecommunications, accessible travel, global opportunities, and other technological and social advancements have thus enabled companies to expand their reach and holdings worldwide (Fryer & Gothe, 2008). Similarly, the software industry has also begun using economical advantages of globalization as well as the advances in technologies and communications that support software development. As a result we now have a global dispersion of activities which is now an increasing trend in software development. It carries the name of Globally Distributed Software Development or, simply put, Global Software Development (GSD).

So what exactly is GSD? *GSD* or *Global Software Development* is software development that uses teams of developers from multiple geographic locations. Sahay Sundeep defines GSD as "*software work undertaken at geographically separated locations across national boundaries in a coordinated fashion involving real time (synchronous) and asynchronous interaction*" (as cited in Mohagheghi, 2004, p. 2). Parvathanathan, Chakrabarti, Patil, Sen, Sharma, and Johng (2007) describe GSD as almost any software development that involves teams that are spread across geographies and share workflows or deliverables. The term development describes the entire software life cycle, including requirements analysis, design, implementation, testing, deployment, maintenance, and support. Besides the global software development projects, there are two very similar types of projects that are being deployed today. Those are distributed and virtual projects. The differences between these projects will be elaborated in the next Chapter. For now, it is sufficient to know that in all of these projects the team members are not co-located. In contrast to traditional software development, where the teams are co-located, these types of software development request higher efforts concerning (Mohagheghi, 2004):

- *Communication for information exchange*
- *Coordination of groups, activities and artifacts*
- *Control of groups and artifacts such as quality, visibility and management*

Today, a large number of countries around the world take part in GSD (see Figure 1). The most popular destinations include India, China, Israel, Singapore, Japan, Ireland, and other.

To get an even more insight in the large scale of application of global software development, one can take a look at following figures (Mohagheghi, 2004):

- *40% of the Fortune 500 companies use GSD and 185 of these outsourced to India alone.*
- *Upwards to 50 nations are participating in GSD.*
- *IBM, British Airways, Alcatel, British Telecom and General Electric have moved parts of their software development to countries like Ireland and India.*
- *80% of the Irish software industry's output is exported.*
- *Gartner Dataquest has projected that IT outsourcing will reach $159 billion by 2005.*
- *87 000 open source projects are hosted at SourceForge.net on Sept. 2004*.

Figure 1: Worldwide development centers (Mohagheghi, 2004)

The basic difference between GSD and traditional, collocated software development is the number of parties involved (Cho, 2007). Whereas the traditional software development implies two players, namely customers who buy the software product, and suppliers who develop and sell the product, global software development introduces clients as a third player. To get the idea of what the roles of these three parties are, let us take a look at the following example. Let us say, that a customer in Austria ordered a software product from an Austrian company. In the traditional setting, the company would develop the product itself or appoint another Austrian company to do the development for them (outsourcing) or it could appoint the development to a company in some other country (offshoring or GSD). If the company decided to use the

third method, GSD, the company now takes the role of clients and the foreign company acts as a supplier (see Figure 2). It must be noted however, that the client and the supplier don't necessarily have to be two different organizations. Supplier could also be the company's subsidiary in some other country.

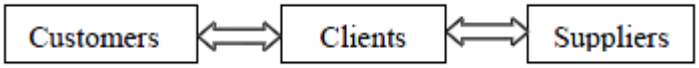

Figure 2: Three main roles in GSD (Cho, 2007)

As shown in Figure 2, the clients reside in the middle of the project, between customers and suppliers. Thus, it is the clients' duty to collect necessary information from customers and forward it to suppliers. They also manage the whole process (and are responsible for the successful execution) and deliver the final product to the customers.

Due to the potential benefits that GSD carries, software development is increasingly a multisite, multicultural, globally distributed undertaking. In the next section, I will present the reasons motivating this new trend in software development.

2.2 Motivations

Although global software development introduces new complexities in regards of communication, coordination, and control of projects, there are many technological, organizational, and economic factors that motivate companies toward this trend. The most cited motivators for GSD include (Cho, 2007; Carmel, 1999; Herbsleb & Moitra, 2001):

- ***Differences in development costs favoring dispersing teams geographically*** – this can be seen as the primary motivation of GSD due to lower labor cost which is much lower in most Asian countries than in the US, Canada, and most European countries. Other countries, like Ireland and Israel, gained presence in the supplier market due to the quality of their developer pool.
- ***Limited pool of trained workforce in technologies that are required to build today's complex systems*** – many countries such as India and China have a large pool of high quality developers and generate many graduates in IT field. In contrast to these countries, many western countries suffer from deficit of developers and don't generate the needed number of graduates. The shortage has to be covered by using GSD.
- ***A "shift"-based work system facilitated by time zone differentials (follow-the-sun development) allowing for shorter times to market*** – this motivator can provide literally 24 hours of development per day which can lead to improved

performance. However, in order to get benefits from it, the software life cycle has to be divided and assigned to different geographic locations.

- ***Advances in infrastructure (e.g., Internet bandwidth available and software development and integration tools)*** – recent improvements in communication technology and the tools that support distributed projects gave a boost to implementation of GSD.

- ***A desire to be "close" to a local market*** – there is a number of business advantages in being close to customers such as using locality specific expertise to customize or localize the products. There are also advantages of supple investment on merger and acquisition opportunities. These opportunities do not exist if the company insists on using traditional, co-located software development.

- ***Faster buildup of virtual corporation and virtual teams*** – it is an easy task to form virtual teams from the developer pool. This is very important if the company wants to utilize market opportunities.

- ***National policies*** – in some countries the government may be a customer requiring suppliers to locate R&D facility in that country as a condition of sale or a favorable tax treatment.

Whereas some companies move to GSD solely for its economic benefits, others are existentially forced to it. An example for a company that wants to utilize the economic benefits of GSD would be a company that attempts to get shorter development time for its product and thus exploits the advantage of different time zones. On the other hand, a company which is not able to acquire a sufficient number of developers for the project at the local level is essentially forced to secure it on a global market.

Although global software development promises huge benefits, there are a vast number of issues and challenges that come with the fact that the project teams are distributed by space and time. In the next section, I will describe some of the most common issues in global software development.

2.3 Issues and Challenges

In recent years, cost reduction was a buzz word in global economy. Countries like Brazil, India, China, and Eastern European nations presently have lower labor rates and outsourcing to them can cut development costs. Some companies from these regions may have expertise in specific technologies and thus be very attractive for partnership. In some cases, companies are not able to acquire sufficient number of employees for the project, or are pressured by time-to-market for a specific product. For these various reasons, companies are attracted to global software development. However, there are significant issues that come with GSD that may not only cancel out these advantages,

but result in huge disadvantages such as going over budget, longer development times, and others.

As presented it the first section, three different parties interact in GSD. The introduction of a third party imposes further differences between GSD and traditional software development. Moreover, together with the distance that separates project teams in different sites, it produces new complexities to the software industry. Several studies show, that multisite development takes much longer than comparable collocated development (Herbsleb, Mockus, Finholt, & Grinter, 2001). Communication and coordination have the biggest impact on this delay.

Herbsleb and Moitra (2001) outline the major dimensions of problems in global software development:

- ***Strategic issues*** – the division of work between defined sites is difficult. This is inherently difficult because the sites often vary in available resources, levels of expertise in various technologies, their infrastructure, and other factors. Ideally, the management could divide the work in such a manner, that the sites operate as independently as possible also assuring easy, flexible, and effective communication between them. A precondition for this ideal scenario would be a highly modular project, what is rarely the case in practice. Another branch of this problem is the organization's attitude towards GSD. Sometimes there is resistance to GSD because of different views between senior and middle management on the intent and perceived benefits of GSD. Moreover, the employees can believe to be threatened by job loss, loss of control, or possible relocation and extensive travel.

- ***Cultural issues*** – in GSD, close cooperation of individuals with different cultural backgrounds is expected. These differences become crucial in respect to project execution in terms of attitudes toward hierarchy, need for structure, sense of time, and communication styles (Hofstede & Hofstede, 1997). For example, some cultures prefer direct communication whereas that kind of communication as seen as rude or unpleasant by someone from a different culture.

- ***Inadequate communication*** – software development involves teams of developers which perform the work on different, but dependent, parts of the project. Thus, it requires much communication in order to coordinate this work and assure that it is carried out within the defined scope, time, and budget. In distributed projects, people don't communicate that much and with that many people at distant sites as they do with people at their own site. Some reasons for this can be found in temporal distance (different time zones), socio-cultural distance (language and culture), and geographic distance (makes the travel difficult) (Carmel & Agarwal, 2001; Holmstrom, Conchuir, Agerfalk, & Fitzgerald, 2006). These issues result in a lack of information about responsibilities and expertise of other project workers. Both formal and informal communication channels are important to different contexts of the project.

The formal, official communication is important for crucial tasks like updating project status, escalating project issues, and defining responsibilities. In order to assure an easy and problem-free execution of these tasks, there is a need for a clear and well-understood formal communication interface. Such an interface could save much time and catch many otherwise unnoticed problems.

The informal communication is important to other aspects of the project. In the traditional, co-located setting, project teams that have worked together over a period of time naturally develop a number of ways to coordinate their work. They also know how their work is related to the work of other teams, and what specific work other teams are performing. The information flows freely through the workspace through various informal interactions such as “corridor talks” or coffee breaks. Through these interactions, team members have an awareness of who has what sort of expertise and competences. This background information enables developers to work together efficiently. However, when the distance between teams is introduced, these important factors are greatly reduced. It is highly likely that the issues that rise daily in a software project go unnoticed or stay unresolved over an extended period of time. The absence of informal communication can lead to unpleasant surprises from distant sites. These result then in misalignments and rework, which lead to extended development time.

The communication issue becomes even more apparent in outsourcing scenarios. In these scenarios, the communication is often filtered or restricted. Reasons for this are the fear of loss of intellectual property or other information about products, market, and customers.

- ***Knowledge management issues*** – the promised benefits of GSD cannot be exploited without effective information- and knowledge-sharing mechanisms. This is essentially important in cases such as when the information from customers and the market needs to be communicated to all of the development teams. Without an adequate knowledge management infrastructure, teams are not likely to find the needed expertise and may miss many reuse opportunities. This is equally true for the documentation. Inadequate documentation can hinder effective collaborative development. Specifically, GSD takes huge importance on updating and revising the existing documentation. This is needed in order to assure that there is no ambiguity and no assumptions are made about what various teams are using and working on.

- ***Project and process management issues*** – the biggest issue represents itself in synchronization between the processes that are exchanged between different sites. In order for the synchronization to function properly, common milestones, clear entry and exit criteria have to be defined. The concurrent development process models that have been developed for concurrent engineering are often hardly applicable to GSD due to volatile requirements, unstable specifications, the unavailability of good tools for collaboration across time and space, and the lack of informal communication. Frequent problem is that the risk management is

being practiced in traditional way, with no relation to the possible issues that come from diverse cultures.

- ***Technical issues*** – some logistical issues, that are trivial in co-located setting, present great problems and produce time loss in distributed development. This is especially true for the network infrastructures that connect different sites as they are often slow and unreliable. This can produce critical problems for the transmission of important data. Other technical issues include incompatible data formats used in different sites as well as different versions of tools.

These problems are barriers that companies have to overcome in order to exploit the benefits of GSD. In the next section we will take a look at some critical success factors of GSD. They represent potential solutions to the issues presented in this Chapter.

2.4 Critical Success Factors

In the last section, some of the most common pitfalls of globally distributed software development have been presented. During last several years, many researchers identified some counter-measures that would help companies mitigate these issues. Due to the fact that the proposed solutions vary from author to author, they will be presented here separately.

Fryer and Gothe (2008) suggest following five critical factors for success (see Figure 3):

- ***Coordination and oversight for all sites*** – the authors argue that it is highly important to "think global" from the start and to plan out exactly how the organization is going to support the company's business model. It must be decided how the roles and responsibilities are distributed across the sites as well as where the management and decision-making will take place. The tools for the development and the infrastructure needed at each site should be identified. Moreover, consistent and reliable measures should be defined in order to successfully track progress and status of the project.

- ***Well-defined and consistent workflows*** – in order to reduce any ambiguity about how the work should be done, well-defined and documented processes need to be defined and be consistently followed across all sites. Naturally, the teams have to be educated on these processes.

  The bounds or parameters should be consistent and common for all processes in order to have consistent touch points and deliverables across and between teams. The company should define the artifacts to be produced, the metrics to be tracked, the measures of success, and the mechanisms for oversight and governance. The authority and responsibilities (who has to do

what, what has to be delivered to whom, what standards have to be met) have to be clearly defined.

If the circumstances allow it, the processes and workflows need to be automated and enforced in the tools that are used across sites. The automation of the processes and workflows could be especially helpful in settings with time zone differences.

Figure 3: Critical success factors in GSD (Fryer & Gothe, 2008)

- ***Inventory and information Management*** – companies need to manage their assets and artifacts and to assure that appropriate and needed access rights are defined. This involves network access and availability of the assets over WAN or Web, security permissions for appropriate actions, and the ability to find the artifacts.

  A versioning mechanism should be used for the artifacts in order to assure tracking and tracing of changes, and to assure traceability between the correct versions of different artifacts.

- ***Clear and accessible communication*** – this could be the most important of all success factors as it is the key ingredient for solid team building. Collaboration, team building, and social networking are facets of successful projects. As discussed in previous section, collaboration in collocated teams happens informally in the hallway or “at the water cooler” and formally in face-to-face meetings, whereas distributed teams require harder work to ensure effective collaboration.

Companies can also develop communication plans that define how to keep all team members informed and working toward a common goal. The communication should be frequent and open and there should be a great focus on building trust between team members. Companies have access to a wide range of collaboration mechanisms such as e-mail, teleconferences, and various notes or documentation facilities that are built into development tools. Today, distributed projects take advantage of chat and instant messaging, Web conferences, screen sharing, wikis, and team blogs.

Trust is very important for effective communication and information exchange between team members. However, it is not easy to establish trust in distributed teams. The authors advise companies to create online profiles of team members in a central location. These profiles should include pictures and even audio clips of team member's voice. The purpose of these profiles is that others can link faces and other personal information to the name and voice of a team member. The best way to build trust among distributed teams is to provide periodic face-to-face meetings. However, these are often too expensive and could go over budget.

In case of distributed projects where the teams are separated by different time zones, the meetings should be scheduled considering this difference. If no appropriate time can be found for all teams, the meetings should be scheduled at alternating times, allowing teams to take turns attending at convenient or inconvenient hours. Holidays in different regions should also be taken into account when scheduling meetings. These methodologies ensure equal treatment of all teams and thus reduce potential team conflict.

In order to surpass language and cultural barriers textual records of vocal communications can be made. This way, team members with spoken language problems can read and reread the written documents and better understand decisions and context. The management should be aware of all cultural and work habit differences across the sites.

- ***Flexible and adaptable development infrastructure*** – the IT infrastructure (including networks and development tools) should be configured in such a way that it enables support to a variety of distributed organization scenarios. The selected tools and platforms should provide the capability and flexibility for supporting required topologies. The authors specifically name following considerations:

    - *Client access to servers via LAN, WAN, or Web*
    - *Security and the ability to restrict access to artifacts as needed*
    - *Ability to interact and integrate with other tools and platforms in a heterogeneous operating environment*
    - *Support for team collaboration*
    - *Ability to support necessary languages and code pages*

In the “Global Software Development Handbook” Sangwan, Bass, Mullick, Paulish, and Kazmeier (2007) list the following potential solutions to issues in GSD:

- ***Reduce ambiguity*** – whereas the ambiguity is a problem for any project, for GSD it gets even bigger. In co-located projects, the informal communication usually solves ambiguity issues. As we have seen in previous section, the informal communication is greatly reduced in distributed settings and thus cannot leverage possible ambiguities. Ambiguity leads to assumptions which are not always instantly apparent. Sometimes it can take a while to see conflicting assumptions and that can cause problems for the project in form of re-planning, redesign, and rework. Due to the complexities that exist in distributed projects, these tasks are not easy to execute and can ultimately lead to abortion of the project.

  The ambiguities about organizational processes, the management practices, the requirements, or the design can exist. The situation gets even more complex if the teams don’t have sufficient experience and domain knowledge. In order to tackle this problem, companies should establish conventions for how different teams work together that would assure that the teams can extract the meaning from project artifacts. As an example for such conventions, the authors state that “*processes should have clearly defined rules of engagement, requirements should be modeled so they are easily understood, architecture should be elaborated with dependencies among modules identified and components specified with well-defined interfaces, work packages should be created for individual teams with clearly stipulated task assignments, and, above all when communication occurs between teams to seek some clarification, one must ensure that answers to questions are correctly understood*“.

- ***Maximize stability*** – the instability can lead to the same problems for the project as ambiguity. The stability can be achieved by delaying the initiation of the development phase until the specification of requirements and design are stabilized as much as possible. It can also be achieved through additional prototypes, UI mock-ups, frameworks, or customization of development environments. All these precautions reduce the possibility of change requests that are identified as a major risk to GSD projects and need 2.4 times longer to resolve in distributed environments (Herbsleb & Mockus, 2003).

- ***Understand dependencies*** – the volume, frequency, and type of coordination in a GSD project is defined by the dependencies between the tasks. In order to ensure the needed and important coordination in the project, the management needs to be aware of all the dependencies that exist. The source of dependencies can be of technical and temporal aspects of the project. Technical dependency would be if one team needs to develop a subsystem that requires a complex image processing algorithm and temporal dependency can come from the planning of the implementation.

- ***Facilitate coordination*** – the teams that need to coordinate because of dependencies that exist between them have to be able to do so accordingly and

efficiently. Although the communication is the main mean of coordination, the teams can also coordinate via processes, management practices and via other means. The problem is that these coordination mechanisms are overhead for the companies and often don't get implemented even though they reduce risks that come from deficient coordination.

- ***Balance flexibility and rigidity*** – because of complexities of GSD and its difference to collocated development, the practices of projects should be more flexible and more rigid. Distributed teams differ from one another in terms of experience levels, domain knowledge, development processes, culture and other facets. Because of these differences the overall development process should be flexible enough in order to ensure comfortable work for all development teams. However, Sangwan, Bass, Mullick, Paulish, and Kazmeier (2007) argue that some aspects of the process need to be rigid enough to ensure that „*particular aspects of the project are well defined and the processes followed as the normal safety nets are in place, for things such as instructions are understood, architectures are complied with, requirements are achieved, configuration management (CM) processes are adequately defined, integration and test procedures are appropriate, etc*". This rigidity is necessary in order to enable the control of the project (progress monitoring, ensuring deadlines are met, quality control).

Finally, (Cho, 2007) discusses four additional success factors for globally distributed software development:

- ***Use common process*** – the communication between clients and suppliers can be better and problems related to coordination and control of projects can be reduced if the common process is used on both sides. The common process is to be established in a systematic way in many aspects of software development process. Both sites should fully understand the process and follow it. This is true from the start to the end of the project, from the planning process of the project to reporting and delivery process.

- ***Choose and divide project strategically*** – the company should have a strategic mechanism in how to divide the work and appoint certain portions to which sites as this is an inherently difficult process. Clients are usually responsible for customer interfacing, installation, system integration, and acceptance testing whereas suppliers are responsible for development of main functions and unit testing. When assigning the tasks to suppliers, clients should choose the culturally neutral tasks and tasks that require less cultural understanding. The level of expertise, available resources, technology, and infrastructure in the supplier site should also be taken into account. By such strategic division many problems that result from cultural and social differences can be mitigated.

- ***Enhance the diversity*** – there are potential benefits to GSD that emerge from the diversity in many areas of software development process. Utilizing diversity in gender, management, and opinion increase the productivity in GSD. Also, the

geographic distance in GSD can be reduced by using a variety of people to work together in interlinking fashion and using various opinions from different levels of employees on both sites.

- ***Educate employees*** – a good portion of employees' time is spent on training and education in many global companies. Due to the specifics of GSD, the employees should also get a cultural education in addition to technical training. If such education is followed at both sites, the cultural barriers can be lowered and trust and confidence between the teams at different sites can be raised.

The implementation and applicability of these success factors naturally depends on the specifics of the given project. Their implementation can be seen as overhead for the project but is in fact an investment that pays off in quality and risk mitigation. The absence of such measures makes the project vulnerable for the pitfalls discussed in previous section.

2.5 Summary

The business environment in which software projects are being performed has changed greatly as a result of globalization process. As a consequence, there is a new trend in software development called Global Software Development or shortly GSD. Although it can have many forms and variations, its main characteristics are the geographical dispersion of software team members.

GSD is currently being applied across the globe due to a large number of benefits it promises. The most important are cost reduction, access to sufficient numbers of trained workforce, shorter times to market, advances in infrastructure, closeness to markets, and faster organizational buildup. Conversely, there are also a number of issues and challenges that plague the success of GSD projects. Problems in strategy, culture, technology, communication, knowledge management, and process and project management have caused many software projects to crash and burn. However, due to its necessity and importance, GSD is a hot topic in fields of research. Recent such research has reaped rewards in form of a numerous success factors whose following promises successful globally distributed software projects.

Changes in business environment have also influenced a closely related discipline to software development. Project management has been identified as a key necessity for the reduction of numerous unsuccessful software projects. The impacts of global trends on this discipline are presented in the next Chapter.

Chapter 3

Distributed Project Management

In recent years project management has been identified as a crucial factor for successful execution of software engineering projects. Project management methods are now being applied around the world to maximize the chances that software products are delivered on time, within budget, and that they correspond to required quality and scope requirements. However, due to the fact that organizations are applying global development strategies and that project teams are distributed in different places, and often time zones and organizations, the project management discipline faces new challenges. The focus of this Chapter is the project management of distributed software projects, its enablers, requirements and some new guidelines that have been defined for organizations that run distributed projects.

3.1 Motivation and Definition of Project Management

From the beginning of the software industry, software development projects have had issues when it comes to fulfilling obligations in terms of budget, development time, quality, and scope. All of these issues are standing in the way of success of corporate management that is more than ever under the pressure of increasing personal and material costs, shareholder expectations, and inflation. The new issue of Standish Group's "Chaos Report" (2009) announces an even further decrease in project success rates. According to this report, 32% of all projects succeed and are delivered on time, on budget, with required features and functions. Moreover, 44% are challenged and are late, over budget, and/or with less than the required features and functions and 24% fail (cancelled prior to completion or delivered and never used). In order to try to reduce this great number of failed and troubled projects, formal approaches to software process improvement (Herbsleb, Carleton, Rozum, Siegel, & Zubrow, 1994) and the application of formalized project management methods to plan, monitor, and control cost, time, and quality (Project Management Institute, 1996) have been developed.

In recent years project management has gained solid reputation. It is an area of expertise that has been recognized by organizations and that is becoming increasingly crucial to the achievement of their strategic objectives. Now, however, project management is one of the fastest growing professional disciplines. To illustrate this fact, one can take a look at the memberships of the PMI (Project Management Institute). They have grown from 71 in 1969 to 5000 in 1984 to more than half a million in 2009 (PMI, 2009). A huge boost to the application of project management was the issue of the Project Management Body of Knowledge (PMBOK) guide. The PMBOK (Project Management Institute, 1996) guide defines formal project management methodologies that have evolved from the need to monitor and control large and complex projects, ensure quality while maintaining budgets and schedules. In addition to the cost, time, and quality, Project Management Body of Knowledge incorporates methods for managing integration, scope, resources, risk, people, and communication. The aim of the methodology is to point out that projects operate within an organizational context with resource constraints and are influenced by multiple, often conflicting stakeholder demands. Today, project teams are often managed by formal project management methods, derived from the PMBOK guide.

In a software engineering project, *software development process* is defined as a process that directly causes the achievement of project results (software products). The project management process is seen as a process for planning, monitoring and control of the software development process and leads indirectly to the achievement of project results (DIN, 2009). In this engineering model, the software development process is seen as a sub-process of the project management process.

Project is defined as a "*temporary endeavor to create a unique product or process*" (Project Management Institute, 1996). Projects are distinguished from the on-going operational tasks by following characteristics (Kapur, 2004):

- *a specific start date and end date*
- *novel enterprise*
- *objectives that need to be fulfilled unconditionally*
- *demands on personal and material resources*
- *differentiation to other enterprises*
- *multifunctional arrangement (projects span across multiple lines of function)*

Project Management Body of Knowledge (Project Management Institute, 1996) defines *Project Management* as the application of knowledge, abilities and techniques to plan activities intending to meet necessities and exceed expectations of people involved, in relation to the project. In order to meet these necessities and exceed expectations the project manager has to balance many concurrent demands such as quality, cost, deadline, and scope, different and sometimes conflicting needs from different stakeholders and shareholders. The activities of the software development project manager have to be adjusted towards planning of quality, productivity, and risk reduction and executing the development of the product (Project Management Institute, 1996).

The project management literature takes notice of various kinds of projects. These different kinds of projects evolve from the number of organizations and locations involved in their implementation. The most frequent categories are (Binder, 2007):

- ***Traditional projects*** – a large majority of the team members are working for the same organization and in a single location.
- ***Distributed projects*** – team members are working for the same organization, but in many locations.
- ***International projects*** – similarly to distributed projects, team members are working for the same organization in many locations. However, they are located across country borders.
- ***Virtual projects*** – in virtual projects team members are dispersed geographically and work for different organizations.
- ***Global projects*** – these projects can be seen as a combination of virtual and international projects. They include people from different organizations that work in different countries around the globe.

Except traditional projects, all other project types have some similar and some different complex challenges that mainly arise from the distance between project team members. However, due to the greatest number of complexity factors involved, global projects are faced with most challenges. The complexity of global projects emerges from a combination of following dimensions (Binder, 2007):

- ***Number of distant locations*** – when the project team is collocated (in a single room or different rooms), team members can benefit from easily organized face-to-face meetings, influence of body language and social interaction. In the case of global projects, the team members are located at least in two different countries. The distance between these countries is prevalently such that travel is required for face-to-face meetings. Project manager needs to implement communication strategies and enforce the use of information communication technologies (ICT) such as phone, video conferencing, e-mail, in order to tackle the communication issues that arise in these projects.
- ***Number of different organizations*** – this factor influences the project in cases where project team members work for multiple companies or even when they work for multiple departments of the same company. Project managers have to adapt their people and leadership skills in order to assess different policies, procedures, and organizational cultures.
- ***Country cultures*** – the diversity between project team members that results from different customs and traditions of different countries and regions can be the source of misunderstandings and conflicts. It can also bring diversity to the work environment, thus reducing the group thinking, improving the collective creativity and increasing motivation. It is the task of the project manager to apply

rules and practices to take advantages of the cross-cultural communication and to avoid the potential disadvantages.

- ***Different languages*** – due to the different countries in which the project team members work, it is very likely that they will speak different languages. Companies that operate on the international level usually establish a common language for the exchange of information. The problems can emerge from the different levels of the selected language's knowledge that various team members have. The project manager has to make sure that every stakeholder has the same information conveyed and that misunderstandings are cleared.

- ***Time zones*** – a side-effect of the geographical dispersion of project team members is that they can also be separated by different time zones that are present in different countries. This makes it difficult or even impossible to organize meetings in common office hours. The dimension of time zones can be used to an advantage in pursuing the "follow-the-sun" model which was discussed in the last Chapter. Conversely, different working times can have negative effects on the coordination and control of the project and thus lead to delays. In order to reduce the possibility of such problems occurring, the project manager must define procedures and communication rules among people in different time zones where there is low overlapping of working hours.

This thesis will not focus on the issues that result from cultural, languages, and time zone differences. It will also not go into addressing different policies and cultures that exist in different organizations. Hence, the focus of this work are the projects where the project team members can have multiple overlapping working hours and speak the same language and poses the knowledge of it to communicate effectively and can use it in such a way that misunderstandings are not likely to happen. Moreover, throughout this work the term distributed project will be used to describe such projects in which team members are primarily separated by geographic location. They can, however, come from different organizations, but do not have to.

3.2 Requirements of Distributed Project Managers

Today already less than 15% of projects are currently managed where 75% of team members operate in a co-located environment (Keys). In other projects team members are dispersed geographically and rely heavily on the use of information technology to support the communication and collaboration among them. In contrast to traditional projects, the issue with distributed projects is the collaboration across time. Beise states that this issue also effects the traditional project management as it is widely accepted that distributed teams cannot be managed using traditional paradigms (as cited in Smith, Bohner, & McCrickard, 2005, p. 301). The evolution of project teams to more diverse, global, geographically dispersed environments presents new challenges to traditional PM approaches.

Together with the project teams, the project management has also evolved. It now has some new requirements that need to be addressed. Binder (2007) lists a number of requirements for the global project management. Many of these are also applicable on distributed projects and include requirements for following project management areas:

- ***Management of distributed teams:***

  - How can the project manager manage conflicts among team members working in different locations (to him)?
  - How can the project manager establish trust among virtual team members, particularly when they work for different organizations?
  - How can the project manager develop leadership skills that can be effective for team members in other locations?
  - How can the project manager perform team-building activities if the budget does not allow it for all team members to be in the same location during the project initiation phase?
  - How can the project manager provide coaching to other project managers and coordinators working in different locations (to him)?

- ***Communication across borders:***

  - How can the project manager have more efficient and shorter meetings over distance?
  - How can the project manager adapt his company templates for meeting over distance?
  - How can the project manager track the project tasks and deliverables assigned to people located in other locations?
  - How can the project manager conduct special meetings, like brainstorming, coaching and knowledge transfer over distance?

- ***Distributed organizations:***

  - How can the organization adapt its structure and culture to succeed in a distributed setting?
  - What is the best way to structure the project team?
  - What types of professionals work well over distance, and how to select them?

- ***Collaborative tools:***

  - Which tools can the company deploy to enhance communication in distributed environments?
  - Which tools can the project manager employ to control and coordinate virtual teams working on my project?
  - Which tools can the project manager use to improve the quality of communication between project managers, which enable him to monitor the milestones without creating administrative overheads?
  - How can the company deploy these tools at all organizational levels?
  - How to foster the adoption of the tools?

The methodologies of distributed project management need to address these requirements in order for the companies to become successful with the implementation of distributed projects.

3.3 Enablers of Distributed Project Management

As presented in the previous section, the distributed project management presents some new challenges that are not covered by the traditional project management practices. Nidiffer and Dolan (2005) argue that the evolution toward distributed project management requests improved processes, methods, and tools to input and share common data (technical, financial, project, and communication data). The following subsections summarize their findings on the topic of enablers of distributed project management.

3.3.1 Tools and Technologies

The advent of the Internet can be seen as the most important milestone when it comes to connecting distributed locations. Together with the maturing of collaborative tools it has greatly influenced the success of complex projects. Software industry has also released a number of project management tools that enable specialized tasks such as tracking requirements, schedules, and budgets to be distributed among multiple sites. Software engineering tools are also following along and adapting to the distributed collaboration setting. Moreover, many companies are deploying Web-based repositories (project Web sites, portals, and workspaces) for sharing and storing files both within and across corporate firewalls.

The structured collaboration tools act as an enabler for the project management hard skills (tracking of requirements, schedules, and budgets) in complex projects. The issue with these tools is that they do not enable critical soft skills such as defining the

business value, determining requirements, building teams, resolving issues and others. This is characterized as the greatest constraint for distributed or virtual teams. In order to overcome this problem, organizations are employing tools such as instant messaging, Web conferencing, whiteboards, and desktop videoconferencing. In contrast to structured collaboration tools, these tools can enhance communication because they enable more frequent collaboration between distant team members and because they are designed for frequent, ad hoc use.

Various technologies have helped organizations with multiple locations in terms of security of data, increased productivity and coordination. Organizations are employing digital environments for secure sharing of files and databases and using workflow functionality to minimize lag time between tasks and thus increase productivity. Maturing communication and collaboration tools are supporting geographically separated team members. There are still some issues with some of the tools like desktop videoconferencing. These issues are presented in security concerns and limited network capacity that limit the use of such advanced tools. However, progress is being made in these segments as well.

The technology for the support of distributed work is available to organizations. One of the remaining challenges is incompatible data formats and exchanges where trade-offs among the alternatives have to be made.

3.3.2 Reevaluation of Processes and Procedures

The currently applied processes that organizations use in distributed work environments have to be reevaluated. Some of the existing processes are inappropriate, controlling, or confining, while others need to be more formal in order to be affective.

Process improvement models such as CMM Integration (Capability Maturity Model) also need to be adapted to the distributed environment as they are primarily developed for application in a single organization. In this model each organization interprets the propositions for its own business case and applies the practices to its own processes. Even though the proposed standardization of the processes can be done uniquely on project level, organizations create a base for reporting, analysis and measurement of success that can be applied on all projects in the organization. The compliance with one standard enables organizations to implement common, shared processes across all business units. This allows a minimal amount of training for the personnel that works across different business units.

However, the organization’s adherence to a single standard causes problems in the distributed environment. The problem is that the partners and suppliers also have to be incorporated in the organization’s process and follow the guidelines themselves.

3.3.3 People and Cultural Change

People are at the core of each project. In addition to tools and technologies used in projects the human element greatly influences the success of any project, especially if the project is distributed. Today it is not rare to have project teams composed from people from different cultures. These differences display themselves in how conflicts are resolved, how people communicate, how they respond to authority and so on. Therefore, distributed project management needs new human interaction skills. The focus should go from project management to project leadership. The project manager has to become a motivator and manager of complex interconnected relationships and forget the role of the dictator. The project manager has to have negotiation as his core competence in order to address the needs of enlarged and diverse number of stakeholders. There is a need for cultural change if the organizations are to manage distributed work accordingly because a wide range of management activities relies on soft skills.

3.4 Best Practices

In today's global economy, software development projects have become very complex and influenced by a number of factors that make a project implementation a pretty delicate undertaking. The whole spectrum of industries, not just the software industry, is in a great need of professional and effective project management, as presented in previous sections. Recent researches have shed a little light on the nature of distributed projects and the relationships that exist among the distributed team members. All of this research has made it possible to identify some common best practices that companies can adapt to their own business and use it to evolve their project management practice.

Garrett (2004) identifies several best practices for the project management in a global setting and next subsections reflect and lean onto his work. The following best practices, except the one that focuses on cultural differences, are of importance for all projects where the team members are geographically separated.

3.4.1 Develop Distributed Project Management Methodology

The companies that want to meet and exceed their customer's needs and expectations through distributed projects have developed a high-level project management methodology. These methodologies are often composed of standard project management processes and processes that are successfully applied within the company. A frequent problem is that companies develop or buy a PM methodology for distributed settings, but do not make an effort and spend time to keep it current and accurate.

In order for a distributed project management methodology to be successful, it has to be:

- Applied across the whole organization
- Widely acknowledged at all levels of the organization
- Routinely practiced by the project management community in the organization

3.4.2 Support Structure

The support structure is an essential for a successful distributed project management. The level of the support varies from company to company but the most usual elements include:

- Executive sponsorship – a certified project management executive who has the knowledge and experience and acts as a champion of project management both within and outside the company.
- Distributed project management office – an experienced, trained, and certified team of project management professionals. They support the project managers in the field and provide them with the needed tools, resources, training, and support.
- Regional and/or local project management office (PMO) – besides the central distributed project management office, there should be a PMO in every location that is involved in the project. The purpose of these offices is the support of local project managers in tasks that are specific to the location.
- Web-based project management best practices (knowledge management) – the best practices from past projects should be documented and be easily available and key-word searchable to all project managers in the organization.
- Project management networking – the company should support the networking of project managers to share expertise, experience, and knowledge. This can be accomplished by teleconferences, video conferences, net-meetings, Web portals, and appropriate face-to-face meetings.
- Accurate and real-time project-level data – the company should employ an information system that is accessible from all locations, which supply real-time critical project information.

3.4.3 Distributed Project Portfolio Management

According to the survey of the Center for Business Practices (CBP) performed in 2003, most companies still had a low level of project portfolio management maturity. Some of

the common problems include allowing lower priority projects to take vital resources needed by higher priority projects, not placing best talent on the highest priority projects, failing to determine the projects of highest priority, failing to acquire status information on key projects, and not terminating failing projects. Today most companies are implementing multiple projects concurrently. Naturally, the projects are not equally important. Thus, it is crucial that companies run a project portfolio in order to understand which projects are of highest priority. This way, they can provide these key projects with appropriate resources and support.

3.4.4 Multi-Cultural Awareness Training

In the current highly distributed economic environment, the project managers often need to relocate to other countries or remotely support customer's projects. Moreover, team members need to collaborate with other team members from other countries. Thus, it is essential that companies provide a multi-cultural awareness training in order to minimize communication problems that evolve through cultural differences and by that support their projects.

3.5 Summary

As presented in this Chapter, apart from software development, project management is another discipline related to software engineering that is greatly influenced and challenged by today's trends in global economy. Software development projects in a distributed setting require additional efforts concerning project management.

A project can have various forms. Traditional projects are performed by mostly collocated and single organizational team members. Distributed projects include single organizational team members that operate from different geographical regions. A derivation of this kind of project is an international project in which case these geographical regions surpass national borders. In virtual projects, multi organizational team members work from different geographical locations. Finally, global projects involve multi organizational workers from different countries around the world. This thesis, however, focuses on distributed projects in terms of physical dispersion of team members regardless of the fact if they come from one or more organizations and if they are located in one or more countries.

Distributed project managers face many challenges and risks on the road to a successful project. In order to maximize the probability of a successful project, distributed project managers require new management possibilities in terms of management of distributed teams, cross-site communication, distributed organization, and collaborative tools.

Due to the fact that people are in the center of every software project, it is highly important to get a grasp onto everyday tasks, relations, and problems of software team

members. The next Chapter takes a closer look on software teams in a virtual setting. It presents the differences between traditional and virtual software teams as well as their most important characteristics. In addition, the impact of distance is investigated and the issues that result from this impact are presented.

Chapter 4

Virtual Software Teams

In software engineering the work is always being performed by teams. As companies are following the trends of globalization and following the promises of lower development costs, availability of highly trained workforce, or some other motivator presented in the second Chapter of this work, they are relying on virtual software teams to do the work. This Chapter presents the characteristics and specifics of virtual software teams. It also presents some of the most common issues that virtual teams face as well as describe the most significant factor that introduces these issues.

4.1 Social Aspects of Software Development

The software development discipline would not be possible without humans who do the tasks of requirement specification, analysis, design, implementation, testing, and evaluation. As such, the success of software development depends on the human factor involved in it, specifically on the complex relationships that exist among the people that collaborate in order to deliver the product.

Some even argue that software development is essentially a social discipline and give psychological views to programming and software development. Curtis suggest that cross-scientific research settings should be created more to better understand the group and personal psychological factors that have essential role in software development. Furthermore, Sawyer and Guinan argue that team-level social processes may be a better predictor for team performance than the production methods (as cited in Piri, 2008, p. 265).

John, Maurer, and Tessem (2005) argue that theories from group psychology to management science can provide insights into how software engineering teams can improve their work practices by not only considering technical choices.

Hence, the importance of social factors in software development is enormous. Because of this, organizations need to investigate relationships between team members and give special notice to the development teams and the complexities and problems they face every day.

4.2 Virtual vs. Traditional Software Teams

Software engineering is technical as well as social discipline. Wheatear an organization is implementing traditional, distributed, virtual, or global software development project, the crucial building block of the project are the developer teams. Cohen and Bailey (1997) define a *team* as "*a collection of individuals who are interdependent in their tasks, who share responsibility for outcomes, who see themselves and who are seen by others as an intact social entity embedded in one or more larger social systems (for example, business unit or the corporation), and who manage their relationship across organizational boundaries*" (p. 241).

In the traditional, co-located software development, the work is performed by traditional or face-to-face teams. Therefore, a *traditional team* would be a collection of collocated individuals who perform tasks and have responsibilities mentioned in the above definition of a team.

Similarly, *virtual software teams* are the work units of distributed, virtual, or global software development. Lipnack and Stamp argue that virtual teams have the same goals and responsibilities as traditional teams (as cited in Casey & Richardson, 2006a, p. 66). However, they operate across time, geographical locations and organizational boundaries and are linked by communication technologies. O'Brien formally describes a virtual team as "*a team whose members use the Intranet, Intranets, Extranets and other networks to communicate, coordinate and collaborate with each other on tasks and projects even though they may work in different geographical locations and for different organizations*" (as cited in Casey & Richardson, 2006, p. 66). However, the most important distinction between virtual and traditional teams is that the members of a virtual team are distributed across geographical locations. Both works from Grenier and Metes and from Simons (as cited in Wong & Burton, 2000, p. 341) found that, in contrast to traditional teams, virtual teams are very dynamic because they are prevalently formed as the need arises and disassembled when the task is complete.

4.3 Characteristics of Virtual Teams

Due to the fact that virtual teams are assembled and disassembled very dynamically, there is very little prior team history and work roles and responsibilities of team members vary with each new virtual team they are appointed to. Savage (as cited in Casey & Richardson, 2006, p. 66) points out that the structures of virtual teams are typically non-

hierarchical and decentralized. Moreover, virtual team members are prevalently dependent on lateral and informal information exchange to perform the tasks.

Wong and Burton (2000) argue that the virtual team has to manifest following characteristics:

- It is a set of culturally and organizationally differentiated members
- The members are grouped temporarily
- The members are physically dispersed
- The members are connected by weak lateral ties
- The members are engaged in performing non-routine tasks

The above characteristics would be the characteristics of the ideal virtual team. In practice, however, there are few such teams. For example, there are teams where the members are geographically distributed, but are culturally and organizationally homogeneous. In other cases, team members may come from different cultures and organizations, but be physically co-located. The consequence of this fact is that the virtuality of a team is determined in degrees rather than in kind. According to Wong and Burton (2000), this virtuality has three characteristics:

- ***Virtual Team Context*** – the virtual team context is characterized by *low team history, novel tasks, and physically distributed members*. Works from McDonald, and Lipnack and Stamps (as cited in Wong & Burton, 2000, p. 341) reported that one of the biggest advantages of virtual teams over traditional teams is that its members can be assembled quickly in order to utilize emerging opportunities, and disassembled when the job is finished. Thus, low team history is argued by the fact that virtual teams tend to have no history of collaboration. Also, different knowledge and capabilities of people have to be leveraged in order to exploit emerging market opportunities. Novel tasks are a side-effect of the nature of these opportunities. In order to utilize them, virtual teams must perform non-routine tasks and have non-routine responsibilities. They also have to perform them under time-pressured environments. Furthermore, the members of virtual teams are prevalently not co-located but dispersed around the world. They are connected only by various information technologies.
- ***Virtual Team Composition*** – virtual team members are characterized by the *heterogeneity in their cultural and organizational backgrounds*. Virtual teams are often composed of culturally and organizationally diverse members. Nowadays, as a result of the globalization and improvements in information technology, organizations are enabled to form virtual teams that connect members from different countries and organizations. Lipnack and Stamps (as cited in Wong & Burton, 2000, p. 341) found that due to the unique cultural and organizational backgrounds of team members, the mix of their knowledge and talents maximizes the potential of the team to take advantage of market opportunities.

- ***Virtual Team Structure*** – McGrath claims that the structure of a group describes the nature and the strength of patterns of relationships among individuals in work groups (as cited in Wong & Burton, 2000, p. 342). As for the relationships between members in virtual teams, they are often lateral but weak. Virtual team members tend to be connected by lateral communication ties because of the physical distance between the members and the nature of the work they are performing. Thanks to the lateral structure, the team members have an efficient flow of information and are able to coordinate their task activities, despite the physical distance between them. However, Wong and Burton (2000) found that these ties tend to be weak because *"the lack of face-to-face interactions, the span across cultural and organizational boundaries, and the lack of prior history of cooperation prevent the time, the mutual confiding and the emotional support required for the formation of strong ties"*. Due to the weak ties among the members in a virtual team, the members are more likely to treat each other formally and less likely to reciprocate requests from one another. Hence, because of the cultural and organizational barriers and the shortage of prior work history, the relationships connecting virtual team members are likely to be lateral but weak.

The coordination dynamics within the team greatly depends on the levels of virtuality characteristics it possesses.

4.4 Distance as Key Complexity Factor

The physical distance that is imposed on team members working in a distributed environment is found to have the greatest influence on the collaboration issues in virtual software teams. To distributed project management, distance itself introduces barriers and complexities. For virtual teams however, the distance has negative influence on other factors such as coordination, visibility, communication, and cooperation (Casey & Richardson, 2006b). If the issues that rise in these areas are neglected, they can cause additional barriers and complexities to the project (see Figure 4).

It has been known that physical proximity of co-workers has a great influence on collaboration. Kraut et al. found that collaboration is more effective and probable if people in the building are located closer to each other (as cited in Brander & Mark, 2002, p. 226). Allen claims that the frequency of communication among team members decreased with distance (as cited in Herbsleb, Mockus, Finholt, & Grinter, 2000, p. 320). Furthermore, he stated that in cases where the engineers' offices were about 30 meters or more apart, the frequency of communication dropped to almost the same low level as in cases where the offices are separated by many miles.

In order to combat the complexities introduced by the distance between members and aid virtual collaboration, software industry has been developing a number of computer supported cooperative work (CSCW) tools. These tools are far from perfect

but research that is being conducted on virtual work helps developers improve their possibilities. The topic of CSCW will be presented in the next Chapter.

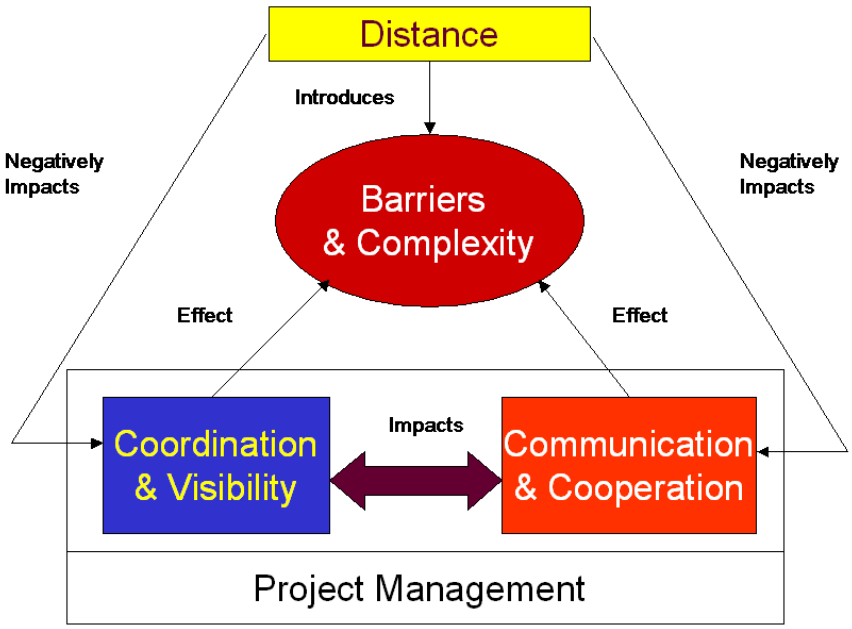

Figure 4: Virtual software team environment (Casey & Richardson, 2006b)

4.5 Issues within Virtual Software Teams

The distance between the members in a virtual team, the lack of face-to-face contact, and the cultural and organizational diversity complicate the work of virtual teams.

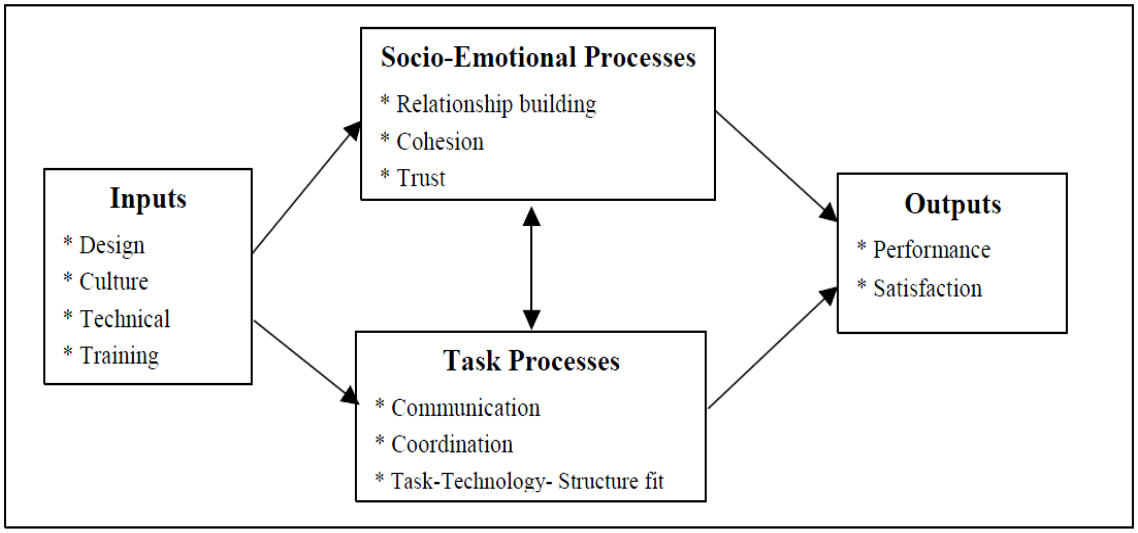

Figure 5: Categories of the life cycle model (Powell, Piccoli, & Ives, 2004)

Powell, Piccoli, and Ives (2004) summarize the results of current researches on virtual teams and present the issues that the virtual teams face and the following subsections are based on their work. The issues that virtual team members are facing are organized around the lifecycle model researched by Saunders (as cited in Powell, Piccoli, & Ives, 2004, p. 8). The life cycle model includes four general categories of variables: inputs, socio-emotional processes, task processes, and outputs (see Figure 5).

4.5.1 Inputs

Inputs stand for the design and composition characteristics of the virtual team and the benefits of resources, skills, and abilities with which the team begins its work (Powell, Piccoli, & Ives, 2004). The most commonly researched inputs are design, culture, technical expertise, and training.

4.5.1.1 Design

The development of a shared language and shared understanding by team members depends greatly on the design of the virtual team and the structuring of its interactions. This is even more crucial early on in the team's life. There are a number of various designs of virtual teams. Some incorporate different levels of face-to-face interaction, planning of activities and the use of communication media, and the articulation of goals, structures, norms, and values (Powell, Piccoli, & Ives, 2004).

The studies that researched differences between traditional and virtual teams inform that traditional teams generally outperform virtual teams with respect to the ability to orderly and efficiently exchange information and to perform effective planning (DeMeyer; Galegher & Kraut; as cited in Powell et al., 2004, p. 8). The probability of the success of a virtual team can be greatly improved by team-building exercises (Kaiser, Tullar, & McKowen; as cited in Powell et al., 2004, p. 8), establishment of shared norms (Suchan & Hayzak; Sarker, Lau, & Sahay; as cited in Powell et al., 2004, p. 8), and the specification of a clear team structure (Kaiser et al., as cited in Powell et al., 2004, p. 8). Some authors point out the need for periodic face-to-face meetings during project planning under the limitation of the use of electronic communication. This is due to the fact that discussion and team interaction in virtual environments can take longer and be confusing, thus leading to poorer comprehension and understanding. By organizing early face-to-face meetings during the team's launch phase, organizations can improve the team's project definition (Ramesh & Dennis, 2002; as cited in Powell et al., 2004, p. 8) and foster socialization, trust, and respect among team members (Maznevski & Chudoba, 2000; Suchan & Hayzak, 2001; Robey, Khoo, & Powers, 2000; as cited in Powell et al., 2004, p. 8). This way they can also enhance the effectiveness and quality of subsequent electronic communication.

By enabling knowledge sharing (either by face-to-face meetings or electronic communication), designs can establish a common understanding and language. The establishment of a common understanding and language helps the team members to solve ambiguous tasks communicating electronically (Majchrzak, Rice, King, Malhotra, & Ba; as cited in Powell et al., 2004, p. 9). On the other hand, the absence of shared understanding and language bares with it a number of possible communication problems. Crampton names such problems as: failure to communicate, unevenly distributed information, difficulty understanding the importance of information to various team members, and difficulty interpreting the meaning of silence or non-reply by others (as cited in Powell et al., 2004, p. 9).

A design of team interaction that employs the setting of goals and strategies leads to the establishment of shared mental models. Different goal and strategy decisions are found to improve the performance of virtual teams.

4.5.1.2 Cultural Differences

As projects are being deployed around the world, they often include team members that come from different cultural backgrounds. The cultural differences and their effect on project success have been studied on numerous occasions. The most important issues that lead from these differences are the coordination difficulties (Maznevski & Chudoba, 2000; Kayworth & Leidner, 2000; Johansson, Dittrich, & Juustila, 1999; Robey et al., 2000; as cited in Powell et al., 2004, p. 9) and the creation of obstacles to effective communication (Sarker, Lau, & Sahay, 2001; Kayworth & Leidner, 2000; as cited in Powell et al., 2004, p. 9). These negative effects are present not only in global virtual teams but also in teams where there are subtle differences among team members from different regions of the same country (Robey et al., 2000; as cited in Powell et al., 2004, p. 9).

The negative effects of cultural differences can be surpassed by actively understanding and accepting the differences (Sarker et al., 2001; Robey et al., 2000; as cited in Powell et al., 2004, p. 9). However, cultural differences have a lesser impact then the distance between members when it comes to project management challenges such as setting goals, budgets, schedules, resources, and identifying needs (McDonough, Kahn, & Barczak; as cited in Powell et al., 2004, p. 9).

4.5.1.3 Technical Expertise

The technical expertise of virtual team members has a great impact on team performance and individual satisfaction. The performance and individual satisfaction with the team experience are negatively impacted by a lack of technical expertise and the inability to cope with technical problems. Hollingshead, McGrath, & O'Connor found that the novelty of the team affects the team members less than the novelty of the technology being used (as cited in Powell et al., 2004, p. 9). The absence of technology

related uncertainty and technological challenges foster the development of high trust among the team members (Jarvenpaa & Leidner; as cited in Powell et al., 2004, p. 9).

4.5.1.4 Training

Various researches have shown that consistent training among all team members increases team performance (Van Ryssen & Godar, 2000). Moreover, team members require training not only in the usage of technology, but in effective communication using the virtual medium (Townsend et al., as cited in Van Ryssen & Godar, 2000, p. 58). However, virtual teams in which team members possess diverse technology skills may have difficulties if they cannot resolve differences and agree on one specific technology skill for the execution of a task (Sarker & Sahay, 2002; as cited in Powell et al., 2004, p. 9).

In order to foster cohesiveness, trust, team work, commitment to team goals, individual satisfaction, and higher perceived decision quality, companies can provide team members with early and uniform training (Warkentin & Beranek, 1999; Tan, Wei, Huang, & Ng, 2000; as cited in Powell et al., 2004, p. 9). Companies are also deploying formal mentoring programs. The goal of these programs is to cultivate relational development and help new members to feel connected to other team members (Suchan & Hayzak, 2001; as cited in Powell et al., 2004, p. 9).

4.5.2 Socio-Emotional Processes

Relationship building, cohesion, and trust are the most important processes within virtual teams. Their existence has positive effects on team performance. However, they are very hard to realize when the team members are separated by physical distance.

Relationship building includes interaction processes designed to increase feelings of inclusiveness or belonging to the team that are hypothesized to foster cohesion and trust (Powell et al., 2004). Research has found that there is a positive link between socio-emotional processes and outcomes of the virtual team project. It has also shown that virtual teams are confronted with unique challenges when it comes to meeting socio-emotional needs of virtual team members (Sarker et al., 2001; Maznevski & Chudoba, 2000; Lurey & Raisinghani, 2001; as cited in Powell et al., 2004, p. 9).

4.5.2.1 Relationship Building

Another difference between virtual and traditional teams is that virtual teams are often more task focused than social focused. However, over time, the amount of the task focus usually lessens (Chidambaram & Bostrom; Walther; Walther & Burgoon; as cited in Powell et al., 2004, p. 10). Virtual team members also generally have weaker

relational links to their co-workers (Burke & Chidambaram, 1996; Warkentin, Sayeed, & Hightower, 1997; as cited in Powell et al., 2004, p. 10). This problem rises from the fact that virtual teams rely significantly on electronic communication and from difficulties that are present with this kind of communication. Thus, many authors have found that face-to-face communication early in the project supports the formation of closer interpersonal relationships between team members (Maznevski & Chudoba, 2000; Robey et al., 2000; as cited in Powell et al., 2004, p. 10). If the budget and deadlines allow it, the team members should physically meet early in the project. These meetings should focus only on relationship building. Such meetings strengthen the socio-emotional development of the team (Robey et al., 2000; as cited in Powell et al., 2004, p. 10) and support later success by enhancing learning and improving performance (Kaiser et al.; as cited in Powell et al., 2004, p. 10).

If face-to-face meetings are not possible, the relationship building can be encouraged by other means. One way to foster relationship building is to focus on the exchange of social communication. Virtual teams that send more social communication achieve higher level of trust (Jarvenpaa & Leidner, 1999; as cited in Powell et al., 2004, p. 10) and better social and emotional relationships (Robey et al., 2000; as cited in Powell et al., 2004, p. 10). Social conversations between team members can also foster relationship building and improve social bonds if they emphasize commonalities between members of different cultures (Sarker & Sahay, 2002; as cited in Powell et al., 2004, p. 10). Effective team leaders can stimulate relationship building by scheduling regular chat sessions with all team members present (Kayworth & Leidner, 2000; as cited in Powell et al., 2004, p. 10).

4.5.2.2 Cohesion

Cohesion in a virtual team fosters better performance (Lurey & Raisinghani, 2001; Maznevski & Chudoba, 2000; as cited in Powell et al., 2004, p. 10) and greater satisfaction among team members (Chidambaram; as cited in Powell et al., 2004, p. 10). It has been identified as one of the differences between successful and unsuccessful virtual teams. Cohesion was the focus of several studies that compared virtual and traditional teams. However, the results have been mixed. Warkentin et al. found that the development of cohesion in virtual teams was obstructed by the use of collaborative technologies (as cited in Powell et al., 2004, p. 10). Hence, traditional teams were found to have higher team cohesiveness. In contrast to this study, other studies have found that even though virtual teams start with lower cohesion, their members exchange enough social information over time and develop strong cohesion (Chidambaram; Chidambaram & Bostrom; Chidambaram, Bostrom, & Wynne; Walther; as cited in Powell et al., 2004, p. 10).

4.5.2.3 Trust

It is a big challenge to develop trust in virtual teams because team members can hardly assess teammates' trustworthiness if they never met them. Moreover, trust must

develop quickly because the life of many virtual teams is relatively limited (Jarvenpaa & Leidner, 1999; as cited in Powell et al., 2004, p. 10). The development of trust is essentially important because it is crucial for the successful completion of virtual team projects (Sarker et al., 2001; as cited in Powell et al., 2004, p. 10).

Even though it is difficult to develop trust in virtual teams, early research has found that short-lived teams are in fact able to develop high trust. However, they do not develop trust by following the traditional model of trust development but by following a swift trust model (Jarvenpaa & Leidner, 1999; as cited in Powell et al., 2004, p. 10). The swift trust model claims that, when they don't have sufficient amount of time to slowly build trust, team members assume that teammates are trustworthy and begin working as if the trust was already developed (Meyerson, Weick, & Kramer, 1996; as cited in Powell et al., 2004, p. 10). During the project they seek for confirming or disconfirming evidence about this trustworthiness. Virtual teams that show high trusting behaviors experience significant social communication, predictable communication patterns, substantial feedback, positive leadership, enthusiasm, and are also able to deal with technical uncertainty (Jarvenpaa & Leidner, 1999; as cited in Powell et al., 2004, p. 10).

The perceived integrity of other team members is especially important in the development of trust early in a team's life. On the other hand, the perception of other member's benevolence helps the maintenance of trust over time (Jarvenpaa, Knoll, & Leidner; as cited in Powell et al., 2004, p. 10). Face-to-face meetings with the focus on developing a strong foundation of trust between members can also be used to instantiate high trust virtual teams (Suchan & Hayzak, 2001; as cited in Powell et al., 2004, p. 11). Besides face-to-face meetings, communication training can also be used to develop high trust between virtual team members (Warkentin & Beranek, 1999; as cited in Powell et al., 2004, p. 11).

4.5.3 Task Processes

Task processes are defined as "*the processes that occur as team members work together to accomplish a task or goal*" (Powell et al., 2004, p. 11). In the task processes category there are major issues regarding communication, coordination, and task-technology-structure fit.

4.5.3.1 Communication

Communication is an essential part of any virtual team process. Moreover, Hulnick states that "*if technology is the foundation of the virtual business relationship, communication is the cement*" (as cited in Powell et al., 2004, p. 11).

Past research on traditional teams suggests that successful co-located teams can communicate effectively and share information crucial to project completion in a timely manner (Ancona & Caldwell, 1992; Bordia, 1997; as cited in Powell et al., 2004,

p. 11). However, the communication in a virtual setting is confronted with serious challenges that evolve from the nature of virtual environment. Such challenges include time delays in sending feedback, lack of a common frame of reference for all members, differences in salience and interpretation of written text, and assurance of participation from remote team members (Crampton, 2001; Mark, 2001; as cited in Powell et al., 2004, p. 11).

In contrast to traditional teams, virtual teams usually suffer from absence of an important component of team communication, namely, nonverbal communication. Due to the importance of communication to virtual teams, it has been the most studied aspect of virtual work. These studies report that traditional teams often communicate more effectively than their virtual equivalents (Galegher & Kraut, 1990; Burke & Chidambaram, 1996; as cited in Powell et al., 2004, p. 11). Saunders states that due to the physical separation between them, virtual team members are heavily dependent on information and communication technologies (as cited in Powell et al., 2004, p. 11). However, technology is very likely to restrain the communication process. Sproull and Kiesler argue that this happens because electronic media are intrinsically leaner than face-to-face communication and convey a limited set of communication cues (as cited in Powell et al., 2004, p. 11). Hence, the teams that perform the work in the virtual setting affront greater difficulties to orderly and efficiently exchange information than their equivalents in the traditional setting.

Even though technical challenges have the greatest influence, they are not the only cause of restricted communication. Information exchange runs into problems also when some team members are co-located and others are dispersed. In such settings dispersed members prevalently assume that co-located team members are talking and sharing information that is not communicated to them. Also, private exchanges have been found to cause friction between team members (Sarker & Sahay, 2002; as cited in Powell et al., 2004, p. 11). Similarly, ineffective leadership and cultural differences have also been identified as the negative influence on communication effectiveness (Kayworth & Leidner, 2000; Sarker & Sahay, 2002; as cited in Powell et al., 2004, p. 11).

In spite to all the difficulties of communicating in a virtual environment, virtual team members must effectively exchange information if they are to achieve their objectives and successfully complete their tasks. That is why the mitigation of communication difficulties and the development of information-sharing culture were the focus of many studies. The results of these studies inform that the frequency and predictability of communication, and the extent to which feedback is provided on a regular basis, improves communication effectiveness. This then leads to higher trust and improves team performance (Maznevski & Chudoba, 2000; Kayworth & Leidner, 2000; Tan et al., 2000; Jarvenpaa & Leidner, 1999; as cited in Powell et al., 2004, p. 11). Contrarily, unpredictable communication patterns are said to cripple the coordination and success of virtual teams (Johansson, Dittrich, & Juustila, 1999; as cited in Powell et al., 2004, p. 11). The most frequent unstable communication pattern includes team members leaving for an extended period of time and failing to communicate the absence to other members previously (Van Ryssen & Godar, 2000; Sarker & Sahay, 2002; as cited in Powell et al., 2004, p. 11).

In regard to the extent of communication, virtual team members communicate more frequently than their traditional counterparts (Galegher & Kraut, 1990; as cited in Powell et al., 2004, p. 11). In addition, members of female-only virtual teams communicate more than members of male-only or mixed gender virtual teams (Savicki, Kelley, & Lingenfelter, 1996; as cited in Powell et al., 2004, p. 11). Also, studies have found that more effective communication improves cultural understanding and vice versa (Van Ryssen & Godar, 2000).

4.5.3.2 Coordination

Cheng defines coordination as the degree of functional articulation and unity of effort between different organizational parts and the extent to which the work activities of team members are logically consistent and coherent (as cited in Powell et al., 2004, p. 11). Even though coordination has a great influence on the performance of virtual teams there are significant challenges that virtual teams face as they try to coordinate their work across time zones, different cultures and divergent mental models (Kayworth & Leidner, 2000; Sarker & Sahay, 2002; Warkentin, Sayeed, & Hightower, 1997; as cited in Powell et al., 2004, p. 12). Furthermore, collaboration norms need to be developed for the team to be able to consistently and coherently bring together team member's contributions (Sarker et al., 2001; as cited in Powell et al., 2004, p. 12).

In order to get leverage on challenges to effective coordination in the virtual setting, the research has focused on investigating interventions and approaches designed to improve virtual team coordination. Face-to-face meetings have been identified as a huge help in mitigating various issues in the virtual environment. If they are feasible, they can also have positive influence on coordination activities and drive a project forward (Maznevski & Chudoba, 2000; as cited in Powell et al., 2004, p. 12). On the contrary, if periodic face-to-face meetings cannot be held, organizations can develop coordination protocols and communication trainings. Such activities support the improvement of coordination and collaboration (Warkentin & Beranek, 1999; as cited in Powell et al., 2004, p. 12). Another way that has shown itself useful when it comes to improving coordination between virtual team members is the minimization of cultural barriers (Robey et al., 2000; as cited in Powell et al., 2004, p. 12).

4.5.3.3 Task-Technology-Structure Fit

The possible combination between different technologies available to virtual teams and the tasks they need to perform plays a significant role in the life of a virtual team. Studies suggest that the technology for the completion of a task is chosen according to the individual preferences, individual experience with the technology and its ease of use, the need for documentation, and the urgency of the task (Robey et al., 2000; as cited in Powell et al., 2004, p. 12). For instance, face-to-face meetings or phone calls have shown themselves as best adapted for ambiguous tasks, managing conflicts, managing external resources, brainstorming, and for setting strategic direction (Powell et al., 2004). On the other hand, electronic communication is the best choice when it comes to

execution of more structured tasks or monitoring project status (Majchrzak et al.; as cited in Powell et al., 2004, p. 12). In settings where virtual team members are not able to attend synchronous meetings (i.e. because of different time zones), a shared language can be successfully developed in order to help members overcome the limitations and adapt the technology to complete ambiguous tasks (Hollingshead, McGrath, & O'Connor; as cited in Powell et al., 2004, p. 12).

Regardless of the availability of various technologies, effective virtual teams are often able to adapt the technology and accord it to the communication requirements of the awaiting task (Maznevski & Chudoba, 2000; as cited in Powell et al., 2004, p. 12). The availability of different technologies for the completion of tasks is said to foster more satisfaction and better performance from virtual team members (Kayworth & Leidner, 2000; as cited in Powell et al., 2004, p. 12).

The adaptability of virtual team members to the different team structure was also the focus of many studies. Sarker et al. (2001) found that virtual teams experience distinct stages of team development just as traditional teams (as cited in Powell et al., 2004, p. 12). In addition, in spite the fact that members of virtual teams need time to adapt to the technology and new team form, they are prevalently able to do so in a satisfying manner (Chidambaram et al.; Chidambaram; Sharda et al.; as cited in Powell et al., 2004, p. 12). Majchrzak et al. and Qureshi and Vogel stated that virtual team members adapt themselves to the technology, organization/social environment, and/or team structures (as cited in Powell et al., 2004, p. 12).

4.5.4 Outputs

The outputs of virtual teams have also been the focus of many researches. They include the performance of virtual teams as well as the member's satisfaction with the virtual team experience.

4.5.4.1 Performance

The researched on performance also compared traditional and virtual teams. The results were mixed. Sharda et al. found that virtual teams are more effective than traditional teams (as cited in Powell et al., 2004, p. 13). However, Warkentin et al. (1997) argue that virtual teams cannot outperform traditional teams (as cited in Powell et al., 2004, p. 13). In addition, the majority of studies conducted on this topic found no significant difference between the two types of teams (Lind, 1999; Galegher & Kraut, 1990; Burke & Chidambaram, 1996; as cited in Powell et al., 2004, p. 13).

Other research conducted on the performance of virtual teams focused on more specific aspects such as decision quality, number of generated ideas, and time the members needed to reach a decision. Archer, and Chidambaram and Bostrom found no significant differences between the decision quality of virtual and traditional teams (as

cited in Powell et al., 2004, p. 13). Similarly, the studies of Archer, Lind, and Sharda et al. also found that virtual teams do not differ much from traditional teams when it comes to the number of generated ideas (as cited in Powell et al., 2004, p. 13). However, Chidambaram and Bostrom did find that virtual teams generated more ideas than their co-located counterparts (as cited in Powell et al., 2004, p. 13). When it comes to time needed for decision making, virtual teams needed more time to make a decision because of the constraints in the virtual environment (Galegher & Kraut, 1990; as cited in Powell et al., 2004, p. 13).

The results of several studies found following contributors to the successful performance of a virtual team:

- Training (Tan et al., 2000; as cited in Powell et al., 2004, p. 13)
- Strategy and goal setting (Malhotra, Majchrzak, Carman, & Lott; Kaiser et al.; as cited in Powell et al., 2004, p. 13)
- Development of shared language
- Team building (Kaiser et al.; as cited in Powell et al., 2004, p. 13)
- Communication (Kayworth & Leidner, 2000; Maznevski & Chudoba, 2000; Suchan & Hayzak, 2001; as cited in Powell et al., 2004, p. 13)
- Coordination and commitment of the team (Maznevski & Chudoba, 2000; as cited in Powell et al., 2004, p. 13)
- Appropriate task-technology fit (Maznevski & Chudoba, 2000; as cited in Powell et al., 2004, p. 13)
- Competitive and collaborative conflict behaviors (Montoya-Weiss, Massey, & Song; as cited in Powell et al., 2004, p. 13)

4.5.4.2 Satisfaction

Warkentin et al. (1997) found that members of traditional teams were more satisfied with their experience that the members of virtual teams (as cited in Powell et al., 2004, p. 13). Archer, however, found no significant difference between two kinds of teams (as cited in Powell et al., 2004, p. 13). The difference between satisfied and unsatisfied virtual team members was also studied. Training (Tan et al., 2000; as cited in Powell et al., 2004, p. 13) and the use of more communication methods (Kayworth & Leidner, 2000; as cited in Powell et al., 2004, p. 13) are identified as possible prerequisites for a satisfied virtual team.

4.6 Summary

Because software development is both social and technical discipline, the aspect of team members is inherently important. Virtual software teams represent a group of software engineers who are involved in a distributed software project and collaborate toward its goal. Virtual team members have to use various communication technologies in order to collaborate and coordinate their work.

The main reason of complexity in distributed projects and workflows of virtual team members is the geographical distance between various development sites. The distance has negative effects on coordination, communication, visibility, and cooperation. Neglecting negative effects can lead to various kinds of issues that hinder the success of virtual teams.

The issues that face geographically distributed team members fall into four categories: inputs, socio-emotional processes, task processes, and outputs. Every category includes several aspects of a virtual team. Inputs involve virtual team design, team culture, training, and technical expertise. Aspects of relationship building, cohesion, and trust fall into the category of socio-emotional processes. Task processes include communication, coordination, and task-technology-structure fit. Finally, performance and satisfaction of team members represent the outputs category.

Research findings on issues of these aspects give a more clear view and insight on problems that distributed co-workers face as well as on reasons why these problems emerge. This can be highly useful for development and creation of new virtual collaboration tools that support virtual teams.

The next Chapter focuses on collaboration in a virtual setting. Characteristics of collaborative work are presented and the difference between traditional and virtual collaboration is elaborated. The activities that team members have to perform are presented afterwards. Finally, the Chapter on virtual collaboration presents the tools that support collaborative work in a virtual environment as well as the different modes of virtual collaboration.

Chapter 5

Virtual Collaboration

As more and more organizations implement virtual team strategies, they rely on the information and communication technology to link geographically dispersed team members. Just as team members in co-located environments collaborate to accomplish their tasks and objectives, virtual team members must collaborate virtually in order to deliver the end product. The software industry has identified the need for virtual collaboration some years ago and has given birth to a wide range of tools that address this challenge. The goal of these tools is to enable people that live in different geographical locations to work effectively on their common assignment.

5.1 Traditional and Virtual Collaboration

The software that is being developed today involves mainly large software systems that cannot be developed by a single person. The tasks of requirements specification, analysis, and design mostly require that more persons are involved in order to minimize the possible mistakes that would result in a failed project and unusable software. Furthermore, the tasks of implementation and testing of the software cannot be completed in an appropriate time frame if a single software engineer is assigned for a task. Because of this, the software engineering projects require that teams of software engineers work on separate but usually interdependent portions of the project and coordinate their work in order to deliver a functional, usable end product. Hence, the members of software teams are involved in a collaborative activity (see also Chapter 2; Chapter 3; Chapter 4).

Collaboration is defined as "*a recursive process where two or more people or organizations work together in an intersection of common goals by sharing knowledge, learning and building consensus*" (Collaboration, Oxford English Dictionary, 1989). In traditional environments, in order for two or more people to work together and share knowledge and other resources, they need to be located in the same geographic

location. Moreover, they need to perform this work in such a setting where they can meet each other regularly, as the need arises. In this manner, they can communicate to exchange ideas, coordinate interdependent tasks, and perform other project related work.

However, as stated previously, distributed projects are stripped of this commodity. In distributed settings the team members have to collaborate virtually. Wainfan and Davis (2004) define *virtual collaboration* as "*collaboration by people working together who are interdependent in their tasks, share responsibility for outcomes, are geographically dispersed, and rely on mediated, rather than face-to-face communication*" (p. 1). The purpose of virtual collaboration in distributed software engineering projects is to produce a software system of high quality that has advantages over the traditional counterpart in either development costs and/or development time. Another purpose of virtual collaboration is to connect qualified workers in cases when organizations are not able to acquire enough workforces in their own geographic region.

Virtual collaboration takes place in a telecommunications network (Wainfan & Davis, 2004). In such network, each node contains one or more people. The nodes can be connected by a wide range of communication media. These communication media will be the focus of subsequent sections.

5.2 Characteristics of Collaborative Work

Moran and Anderson argue that designers of systems that support collaborative work need to learn a lot from people working together and collaborating together in conventional settings (as cited in Churchill & Snowdon, 1998, p. 5). In addition, Churchill and Snowdon (1998) list following characteristics of collaborative work:

- **Transitions between shared and individual activities** – collaborative work requires the interleaving of individual and group effort. The studies conducted by Bellotti and Rogers, Harper, Heath and Luff, Hutchins and Klausen, and Suchman inform that because of this interleaving, collaborative work involves considerable complex information exchange (as cited in Churchill & Snowdon, 1998, p. 6). In order for collaborations to be successful, much explicit and tacit communication needs to take place between the collaborators. Individuals must have shared understandings of the specifics of the task that is currently being performed. They have to know what it is that is currently being worked on and what has been done previously in order to reach the task goals.
- **Flexible and multiple viewpoints** – there is often a need to use multiple representations of a task according to different points of view and different subtasks. Individual collaborators may demand multiple representations in order to grasp different aspects of their task. Others, on the other hand, can require specific representations that provide information specific to their tasks. For example, in software engineering process, software designers require use case

diagrams, whereas programmers need complete class diagrams as a prerequisite for their tasks. Each of them requires information about the same subject, only the viewpoint and abstraction level are different.

- **Sharing context** – it is crucial for collaborative activities that collaborators share the same context. Even though 'shared context' can mean many things (shared knowledge of other's current activities, shared knowledge of other's past activities, shared environment, shared artifacts, etc.), it ultimately leads to shared understandings. One of the greatest stimulators of shared understandings is shared space facilities. Moreover, it is more difficult to foster shared context in environments where actions are not physically collocated and co-temporal.

- **Awareness of others** – similar to 'shared context', 'awareness' also has several meanings. One aspect of awareness is that it is knowledge of task related activities. Dourish and Bellotti see awareness as "understanding of the activities of others, which provides a context for your own activity" (as cited in Churchill & Snowdon, 1998, p. 7). Hence, awareness can be seen as an audit of activity. In this context, awareness has high importance for groupware where activities are often asynchronous and the tasks are performed by interleaving members. Another important context of awareness for virtual collaboration and groupware is the feeling of 'co-presence'.

- **Negotiation and communication** – Wardhaugh emphasizes the importance of negotiation through talk for collaboration (as cited in Churchill & Snowdon, 1998, p. 8). The importance of negotiation relates not only to task related content, but also to task structure and allocation. Verbal and nonverbal cues have influence on these negotiations. However, nonverbal cues (facial expression, body posture and gesture) are crucial for most conversations as they carry from 60%-90% of the information in the verbal message being communicated (Hewstone et al.; Kendon; as cited in Churchill & Snowdon, 1998, p. 8).

These characteristics have a great importance for groupware design and can be the judgment point when it comes to success or failure of a groupware system.

5.3 Virtual Meetings and Tasks

The primary two types of activities that both traditional and virtual software teams perform are tasks and team meetings. These activities are the focus of following subsections.

5.3.1 Tasks

Software engineering projects mostly include phases of requirements analysis, design, implementation, and testing. As Object-Oriented paradigm is dominant in software engineering, it is also the focus of this work.

Hence, the tasks that virtual software teams perform in each phase are (Stiller & LeBlanc, 2002):

- **Analysis:**
  - specification of refined requirements
  - creation of scenarios
  - creation of primary class list
  - creation of class diagrams
  - creation of use case diagrams
  - structured walk-through

- **Product design:**
  - creation of object diagrams
  - refinement of class diagrams
  - creation of user interface mock-ups
  - creation of state machines

- **Class design:**
  - creation of collaborative diagrams
  - creation of sequence diagrams
  - creation of object diagrams
  - refinement of class diagrams
  - creation of class skeletons
  - informal walk-through

- **Implementation:**
  - definition of an implementation plan
  - coding

- **Testing:**
  - definition of a test plan
  - creation of a test analysis report
  - integration of the system
  - system delivery and demo

5.3.2 Meetings

Meetings, on the other hand, are more complex activities in both face-to-face and virtual setting. Meetings usually involve the whole software team, whereas tasks involve only the team members that have been appointed with it. In addition, events in meetings occur simultaneously on multiple levels. The most frequent activities in meetings are brainstorming, voting, and ranking (Beise, Evaristo, & Niederman, 2003). The purpose of meetings is mostly to further progress on one or more tasks. In order to achieve this goal, speech acts are used. These speech acts can also have other effects, such as establish authority, reinforce social norms, and support or weaken alliances and social relationships.

The communication in meetings is found to have various purposes. These include coordination, organization, motivation, affiliation, information dissemination, and off-topic comments (Scheerhorn, Geist, & Teboul; as cited in Beise, Evaristo, & Niederman, 2003, p. 10). These purposes are closely tied in a face-to-face meeting. In virtual meetings, however, due to the distribution of attendants, the ties between these purposes have to be constructed on purpose if the meeting is to succeed. One of the issues apparent in virtual meetings is the movement from discussion to resolution of an issue. In face-to-face meetings, there are many blatant and subtle signals that let participants know when to make this movement. These signals are not present in the virtual setting and have to be consciously provided in order to move from one to another meeting activity.

Hayes (2004) states that a groupware tool that is used for virtual team meetings can make a meeting more effective through the use of following features:

- **Profile** – basic information such as date, time, location
- **Participants** – records attendees and can be updated in real time
- **Agenda** – every participant can contribute to, review, and prepare before the meeting
- **Minutes** – can easily be added and keeps a record of meeting for all participants
- **Actions** – can be assigned, recorded, continuously tracked, and are visible for all participants

5.4 Computer-Supported Cooperative Work (CSCW) and Groupware

The communication media that connects geographically dispersed team members and allows them to collaborate virtually has been the focus of *Computer-Supported Cooperative Work (CSCW)* studies. CSCW is "*a multidisciplinary field which focuses primarily on the nature of group work and how it can best be supported with computer-*

based information and communications systems" (Kurbel, as cited in Knotts, Dror, & Hartman, 1998, p. 623). Furthermore, the research field of CSCW addresses how collaborative activities and their coordination can be supported by means of computer systems (Carstensen & Schmidt, 1999).

Groupware, on the other hand, although sometimes mistakenly referred to as a synonym for CSCW, describes computer software used to support the communication and coordination needs of individuals working in groups (Lundgren, as cited in Knotts, Dror, & Hartman, 1998, p. 623). Other synonyms for groupware include collaboration technologies, collaborative software, and workgroup support systems. Thus, groupware can be seen as a basis and tool of CSCW.

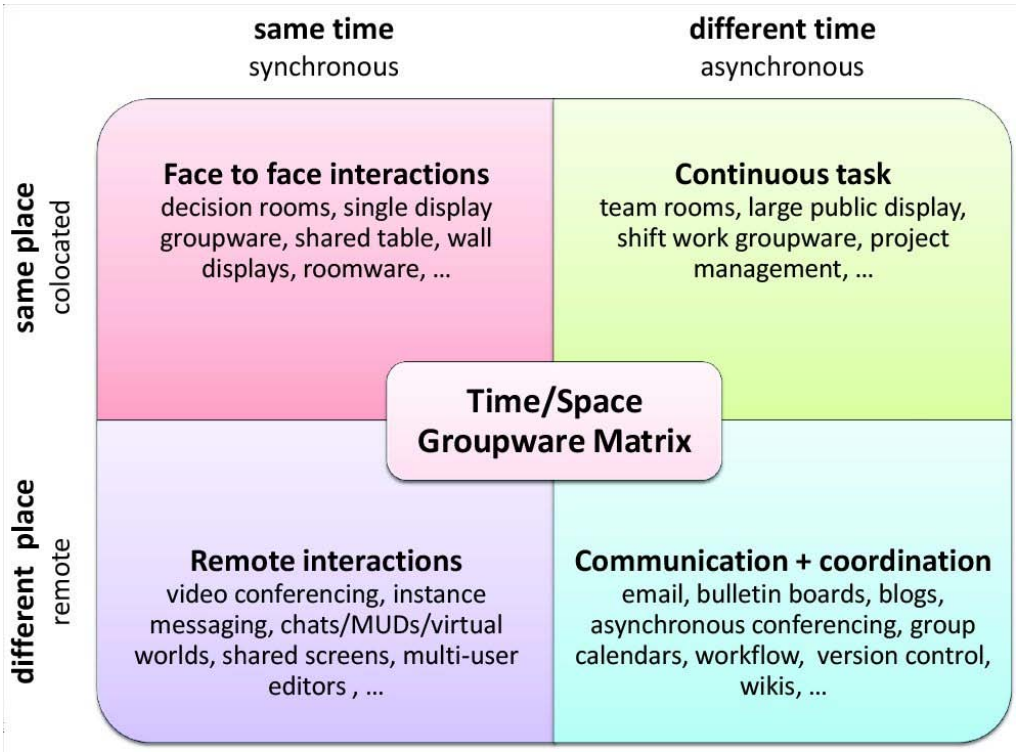

Figure 6: CSCW matrix (Systemic Logic, 2009)

Groupware systems are usually categorized according to the CSCW matrix (see Figure 6). According to the CSCW matrix, there are four categories of groupware systems (Systemic Logic, 2009):

- ***same time/same place*** – collaborators are co-located and perform the work simultaneously (e.g. decision rooms, single display groupware, shared table, wall displays, public computer displays).
- ***different time/same place*** – collaborators are co-located but perform the work shift wise (e.g. team rooms, large public display, shift work groupware, project management).

- ***same time/different place*** – collaborators are physically distributed and perform the work simultaneously (e.g. video conferencing, audio conferencing, instant messaging, chats, virtual worlds, shared screens).
- ***different time/different place*** – collaborators are physically distributed and perform the work shift wise (e.g. email, bulletin boards, blogs, asynchronous conferencing, group calendars, workflow, version control, wikis).

In the context of distributed projects, same time/different place and different time/different place (remote synchronous and remote asynchronous) groupware has the greatest importance, even though it is common that two or more members of a virtual team are co-located and hence benefit from other two types of groupware. The purpose of groupware is to provide an alternative to traditional face-to-face collaboration for both interdependent tasks that require coordination and for team meetings.

However, the development of groupware is accompanied with a great number of expensive failures. Having done the research on this problem, Grudin offers following challenges for developers of groupware systems (as cited in Knotts, Dror, & Hartman, 1998, p. 624):

1. Close the gap between those who benefit directly from the adoption of groupware and those who must do additional work as a result of the adoption of groupware.
2. Ensure that the critical mass of users required for the success of a groupware system is involved in its adoption and use
3. Do not disrupt normal group social processes any more than necessary.
4. Provide as much flexibility as possible within the groupware system to address the kind and number of exceptions that typically occur and are easily managed during normal group interaction.

5.5 Modes of Virtual Collaboration

As expressed previously, virtual collaborators can be connected through a wide range of communication media. In this respect Wainfan and Davis (2004) list three principal modes of virtual collaboration: audio-conferencing, videoconferencing, and computer-mediated communication.

One of the essential differences between traditional and virtual collaboration, besides the location of collaborators, is the type of communication that occurs within them. Traditional collaboration uses face-to-face communication and virtual collaboration uses mediated communication. The problem with mediated communication is that it limits nonverbal, paraverbal, and status cues. It also reduces the “richness” of the information communicated. Krauss and Bricker also found that the discussion in

mediated communication tends to be less social and more task-oriented than in face-to-face communication (as cited in Wainfan & Davis, 2004, p. 19).

Figure 7 shows the placement of videoconferencing, audio-conferencing, computer-mediated communication (e-mail, chat, and web-conference), and face-to-face communication when it comes to the synchronicity and the presence of nonverbal and paraverbal cues. Ultimately, all communication media have limitations simulating face-to-face communication. However, the research that is being performed on the topic of virtual collaboration identifies challenges and issues of mediated communication as well as gives propositions that help improve the communication media and thus support virtual team members by simulating co-located environment as much as possible.

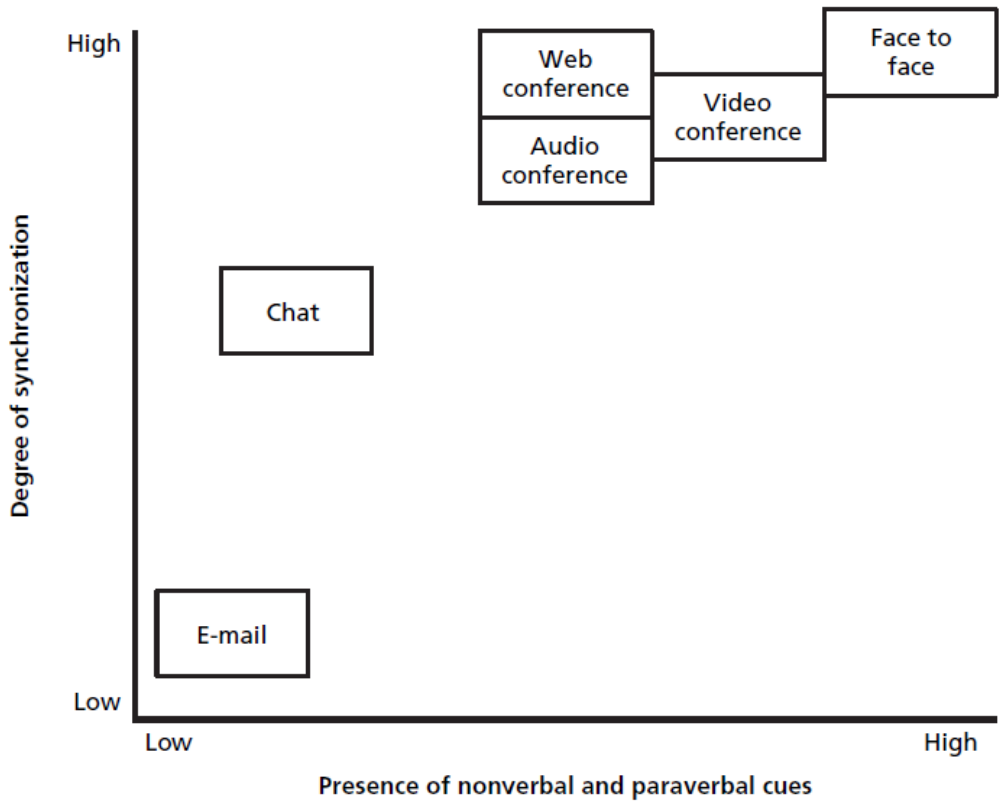

Figure 7: **Placement of Communication Media, by Synchronization and Cues (Wainfan & Davis, 2004)**

The following subsections represent short summaries of Wainfan and Davis (2004) on the three communication modes and their issues.

5.5.1 Videoconferencing

In videoconferencing (VC), participants see a video image of another collaborator or multiple images of other collaborators. Hence, VC incorporates real-time images and voices of other participants. In some cases, VC can include other shared images or text. Examples of VC include group videoconferencing in dedicated rooms and desktop videoconferencing. This mode can also involve subgroups meeting face-to-face in the same room.

Even though VC image quality has significantly improved over the last decade, the users still encounter problems when involved in VC. Moreover, it is still difficult to maintain eye contact due to image resolution and the distance between the camera and the monitor, and it is challenging to interpret body language and gestures. This is particularly true when the number of participants increases. In such cases, the monitor is filled with representations of other users. Daft and Lengel found that VC participants may find it difficult to identify a remote speaker, detect movements, attain mutual gaze, and gain floor control (as cited in Wainfan & Davis, 2004, p. 19).

Another problem that faces participants in VC is its small audio delay. Finn, Sellen, and Wilbur discovered that it can literally frustrate participants and seriously disrupt their ability to reach mutual understanding (as cited in Wainfan & Davis, 2004, p. 21). Tang and Isaac report that participants in their study had difficulties in managing turn-making and coordinating eye-contact because of the audio delay (as cited in Wainfan & Davis, 2004, p. 21). Finally, they turned off the audio and used a phone call via speakerphones. In this setting, the audio arrived before the video, the quality was poorer, and the speakerphone allowed only one party's sound to be transmitted at a time. This stresses the fact that for most tasks, audio quality and responsiveness are more important than video quality when it comes to participant satisfaction (Gale; Ochsman & Chapanis; Adrienssen & van der Velden; as cited in Wainfan & Davis, 2004, p. 22). The hope still remains for VC with the advantages and improvements of technology that could mitigate these issues.

5.5.2 Audio-Conferencing

In audio-conferencing (AC), participants are engaged in voice communication with one or a number of other people. Unlike VC, AC does not possess useful real-time video images of other users. Examples include phone calls, conference calls, and also conference calls where the participants share views of images or documents. Similar to VC, AC makes it possible for subgroups to meet face-to-face in the same room.

The downside of AC is that it removes all visual cues about other participants. Isaacs and Tang report that this reduces the ability to show understanding or agreements, forecast responses, enhance verbal descriptions, manage extended pauses, express attitudes through posture and facial expression, and provide nonverbal information (as cited in Wainfan & Davis, 2004, p. 23).

Harmon found some typical challenges participants face when adapting to audio-only communication (as cited in Wainfan & Davis, 2004, p. 23). These include users looking and gesturing at the speakers, having problems with turn-taking, identifying speakers, and interpreting the discussion. Another highly frequent problem that faces both VC and AC collaboration is the formation of local coalitions. The participants in these coalitions prevalently agree more with those in the same room than with those on the other end of the line. Moreover, in AC there is a greater possibility for disagreement with those on the other end of the communication link.

5.5.3 Computer-Mediated Communication

Contrary to VC and AC, computer-mediated communication (CMC) is typically text-based. In this communication mode, text, images, and other data are exchanged via computer and participants do not have real-time voice or video images of others. Lately, CMC is increasingly incorporating drawings, photos, and other images such as happy faces or “emoticons”. Examples of CMC involve e-mail, chat rooms, discussion boards, text messaging, shared databases, and group support systems (GSS). Whereas VC and AC are primarily synchronous, CMC possesses representatives of both synchronous and asynchronous communication. Synchronous CMC includes tools such as chat rooms or instant messaging. Asynchronous CMC involves e-mail, discussion boards, and shared databases.

Due to the prevalently text-based nature of CMC, many context cues are eliminated entirely. The difficulty of this elimination presents itself in the fact that participants can hardly tell if the collaborator at the other computer is paying attention, understanding, agreeing, surprised, shocked, confused, or even in the room. In addition, participants can get frustrated and feel increased time pressure by having to type a response when involved in CMC collaboration.

Generally, when involved in CMC collaboration, relative to face-to-face groups, virtual collaborators:

- have difficulty reaching consensus
- have greater equality of participation
- take longer to reach a decision
- show differences in influence, particularly relating to status
- exhibit lower inhibition
- are more likely to be polarized

When it comes to discussions, CMC alters the methods that individuals use to influence each other. Opposed to face-to-face collaboration, CMC collaborators make more explicit proposals, defer less to high-status members, and are less inhibited than are face-to-face collaborators (Siegel et al.; as cited in Wainfan & Davis, 2004, p. 27).

5.6 Summary

In order for a software project to deliver a high quality and successful product, software team members need to collaborate on the project. The process where two or more people or organizations work together on a common goal is called collaboration. Consequently, if these people or organizations are dispersed in more geographical locations, then they are involved in virtual collaboration.

Collaborative work presupposes transition between shared and individual activities, flexible and multiple viewpoints, context sharing between coworkers, awareness of others, negotiation, and communication.

In a software development project, team members are primarily involved in two collaborative activities: software engineering tasks and team meetings. In order to bridge the distance between collaborators and make the performance of these activities feasible, various groupware systems have been developed in recent years. These tools fall into four different categories depending on the factors of time (synchronous and asynchronous) and place (collocated and remote). These groupware systems are computer software that supports the communication and coordination between team members and is not to be mistaken with CSCW. CSCW is a field that focuses not only on software but on the nature of group work and the possible application of computer-based information and communication systems in this context.

Three modes of virtual collaboration are audio-conferencing, videoconferencing, and computer-mediated communication. Every one of these modes introduces some constraints to virtual collaborators in terms of coordination and communication. Besides these three virtual collaboration modes presented previously, there is another mode that is becoming increasingly used for bringing distributed teams together. This fourth mode combines audio-conferencing, CMC and a slight derivation of videoconferencing and will be the focus of the next Chapter.

Chapter 6

3D Virtual Worlds

The previous Chapter was concerned with the collaboration in the virtual setting. It dealt with the general characteristics of collaborative work, listed the tasks that virtual teams have to perform, presented the research field of CSCW and tools that support virtual teams in their efforts. Moreover, previous Chapter enumerated three different modes of virtual collaboration. As stated previously, this enumeration entails a fourth mode of virtual collaboration. This fourth mode is represented by virtual environments or virtual worlds. This Chapter dwells deeper into the characteristics and the phenomenon of 3D virtual worlds. It begins with the definition and evolution of virtual worlds, mentions the most popular ones, and presents opportunities and challenges of virtual worlds in context of virtual collaboration.

6.1 Definition and Characteristics of Virtual Worlds

In last decade, virtual worlds have grown extremely popular. Maybe the most interesting example of the virtual world phenomenon, Second Life, has contributed the most to this massive popularity. From 2006 to now, the population of Second Life has increased from 2 million to more than 9 million. Furthermore, Second Life is a real business ground where users spend more than a million dollars every day in a virtual economy (Hendaoui, Limayem, & Thompson, 2008).

There are a vast number of different virtual worlds. Hence, various definitions evolved ranging from religion, fantasy to computer science and gaming. However, the following definitions of virtual worlds correspond the best with the context of this thesis. Franceschi, Lee, and Hinds (2008) define a virtual world as "*a set of computer rendered images that comprise a simulated environment in which users interact through the use of avatars*." It is important to note that this simulated environment is represented as a 3-dimensional. *Avatars* are graphical representations of users in a virtual world. The word

"Avatar" comes from a Hindu philosophy and means incarnation on a different level (Avatar, 2008; as cited in Sivan, 2008, p. 16).

According to Owens, Davis, Murphy, Khazanchi, and Zigurs (2009), a virtual world is "*an instantiation of a metaverse - a fully immersive 3D virtual space in which people interact with one another through avatars and software agents*."

Castranova (2001) also lists the most important characteristics of a virtual world and defines it as "*a computer program with three defining features:*

- *Interactivity: it exists on one computer but can be accessed remotely (i.e. by an internet connection) and simultaneously by a large number of people, with the command inputs of one person affecting the command results of other people.*
- *Physicality: people access the program through an interface that simulates a first-person physical environment on their computer screen; the environment is generally ruled by the natural laws of Earth and is characterized by scarcity of resources.*
- *Persistence: the program continues to run whether anyone is using it or not; it remembers the location of people and things, as well as the ownership of objects.*"

In addition, Prentice (2008) added another feature to this list. He argues that virtual worlds are also characterized by *Representation*, describing how our online persona is represented through an avatar in virtual worlds.

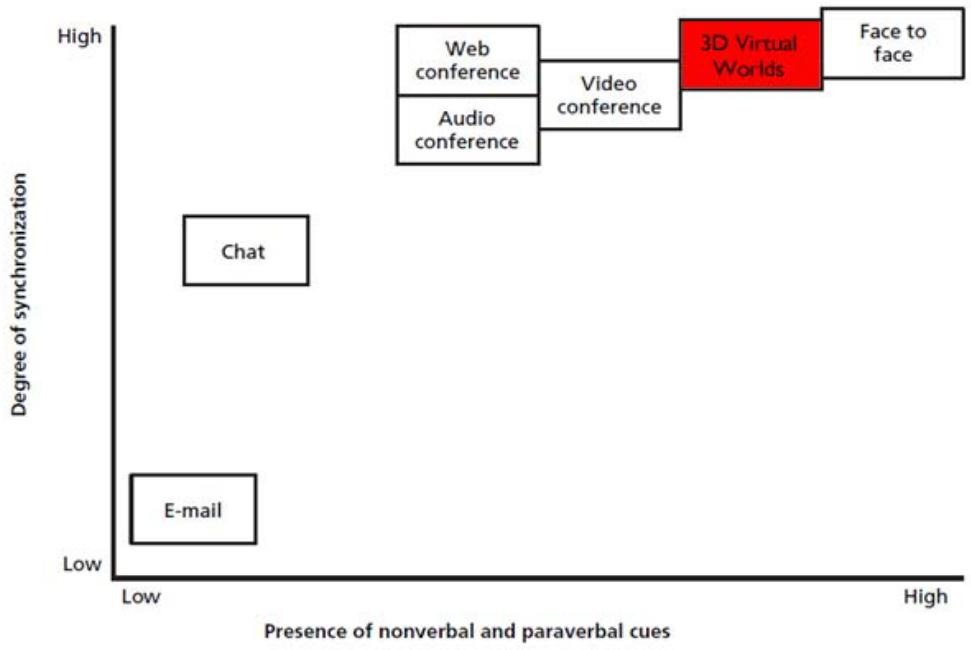

Figure 8: Placement of virtual worlds in relation to other communication modes (adapted from Wainfan & Davis, 2004)

Following these three definitions, the following definition can be derived. A virtual world is an interactive, physical and persistent computer program that uses rendered images to create a simulated 3D environment in which users can interact through the use of their graphical representations or avatars.

Moreover, virtual worlds possess additional interesting characteristics. They resemble the real world, but without its physical limitations (Bainbridge, as cited in Owens et al., 2009, p. 34), support multiple users, and incorporate social networking capabilities (Franceschi, Lee, & Hinds, 2008). With these possibilities, virtual worlds provide an environment suitable for the transformation of education, learning, organizational communication, and virtual project management (Owens et al., 2009). In addition, when it comes to synchronicity and presence of nonverbal and paraverbal cues, 3D virtual worlds exhibit better characteristics than other communication modes (see Figure 8).

6.2 Evolution of Virtual Worlds

Sivan (2008) argues that a virtual world is created thanks to two ingredients: virtual reality and "gaming worlds". Burdea and Coiffet define virtual reality as "*a simulation in which computer graphics is used to create a realistic-looking world. Moreover, the synthetic world is not static, but responds to the users' inputs*" (as cited in Sivan, 2008, p. 2). Gaming worlds, however, suppose a large number of online games that have been around since 1978 (Sivan, 2008). The rest of this section enumerates and describes the games that had the most impact on the development of virtual worlds.

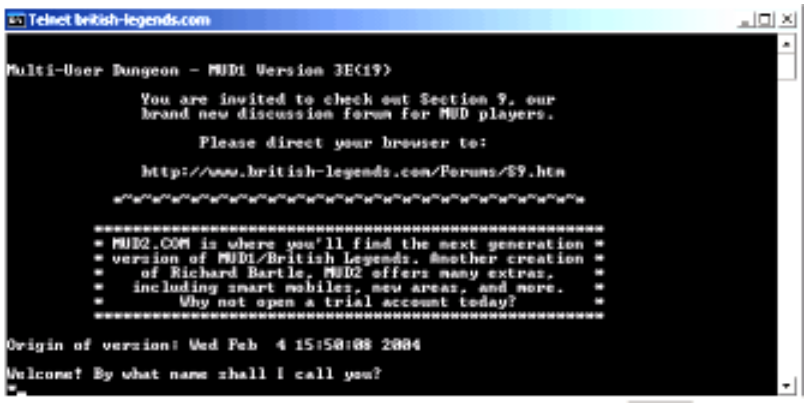

Figure 9: Game screen in MUD (MUD, 2009)

The development and hype of these games began with the deployment of the ARPA-Net in 1969 by the United States Defense Department, which would later become the Internet. The first such game was *MUD (Multi User Dungeon)*[1] which was developed

[1] http://www.british-legends.com/

by Roy Trubshaw and Richard Bartle in 1978 at the University of Essex. This game is recognized as the world's first virtual world and had two versions (Virtual World Timeline 2009; MUD, 2009). MUD was text based and though it contained no graphics, it possessed a rich imaginary world (Sivan, 2008). Figure 9 shows a typical game screen in MUD.

During 1981, Dr. John Taylor and Dr. Kelton Flinn from University of Virginia wrote *Island of Kesmai*[2] which was launched in 1984. In 1985, Island of Kesmai was available on CompuServe. Upon logging into CopmuServe and choosing to play Island of Kesmai, the user was able to create a character that represented him in the game. Next, the user entered the chat room and from the chat room he could enter the virtual world (see Figure 10). Island of Kesmai is seen as one of the first successful multi-user online role-playing games and also one of the first to have a graphical two-dimensional user interface (Virtual World Timeline, 2009).

Figure 10: Game scene from Island of Kesmai (Island of Kesmai, 2009)

Habitat (see Figure 11) was another two-dimensional virtual world, developed in 1987. The importance of this virtual world lies in the fact that it was the first virtual world that used the term "Avatar" (Virtual World Timeline, 2009).

The year 1994 introduced the first avatar-based virtual 3D chat called *WorldsChat*[3]. It was created by Dave Gobel. WorldsChat enabled users around the world to communicate in a 3D virtual environment. Just a few months later, in 1995, *Active Worlds*[4] was launched. It incorporated further technological advantages in form of voice chat and instant messaging (Virtual World Timeline, 2009).

[2] http://www.legendsofkesmai.com/
[3] http://www.worlds.net
[4] http://www.activeworlds.com/

Designer Mike Sellers and two programmers, Andrew and Chris Kirmse started the development of *Meridian 59*[5] *(M59)* in 1995 and launched the game in 1996. According to Colker, this game started the recent explosion of virtual worlds (as cited in Costranova, 2001, p. 6). The users' avatars were located in a virtual world represented by a town and an open field. The users were able to manipulate the environment by the use of keyboard and mouse commands to their avatar (Costranova, 2001; Virtual World Timeline, 2009).

Figure 11: A screenshot from Habitat (Gamasutra, 2009)

Whereas M59, according to contemporary standards, could support a quite small number of simultaneous users, current virtual worlds can support several thousand on a single server (Castronova, 2001). In 1997, the first virtual world on this scale was launched. It was called *Ultima Online*[6]*.* This virtual world (see Figure 12) is a 3D graphical world and it attracted 100,000 users, what was amazing at that time (Sivan 2008). Ultima Online is still played online today and had a huge impact on development of MMORPG (Massively Multiplayer Online Role-Playing Game) genre (Virtual World Timeline, 2009).

Driven by the popularity of Ultima Online, Sony/Verant Interactive launched *EverQuest*[7] in 1999. EverQuest is a 3D fantasy-themed MMORPG which is unique for several reasons. The camera was able to roam around instead to be fixed to the eyes of the avatar, and the game implemented a real economy, where users could buy and sell virtual items in the world. Another interesting effect that EverQuest manifested is the power of people. For whatever reason the users started playing it, they continued

[5] http://meridian59.neardeathstudios.com/
[6] http://www.uoherald.com
[7] http://everquest.station.sony.com/

because of their friends. EverQuest also bounded people in the game as it allowed small groups to play together (Castranova, 2001; Virtual World Timeline, 2009; Sivan 2008).

Figure 12: A scene from Ultima Online (Ultima Online, 2009)

In 2002, a game called *Sims Online*[8] was released by Electronic Arts. This was the first virtual world that was not based on killing and adventure (Castronova, 2001). Sims Online introduced tools that enabled creativity in 3D creation and thus evolved and expanded the idea of "user-created content" (Sivan, 2008).

Figure 13: Game scene in World of Warcraft (WOW, 2009)

[8] http://thesims.ea.com/

MMORPG genre reached its highest peak in 2004 when Blizzard released *World of Warcraft*[9] *(WOW)*. This is by far the most successful MMORPG from both business and gaming aspects. The extension of WOW released in 2007, named as The Burning Crusade, sold 2.4 million copies in North America and Europe during the first 24 hours of distribution and made a profit of 100 million dollars in just one day. As WOW is built as a massive parallel world it showed for the first time that it is technically possible to run a world with millions of users (Sivan 2008; Virtual World Timeline, 2009). Figure 13 shows a sample game screen in WOW.

From this point onward, a wide range of virtual worlds with different possibilities and target groups appeared. The currently most popular and the most interesting virtual worlds are presented in the next subsection.

6.3 Currently Most Popular Virtual Worlds

As elaborated in the previous section, virtual worlds evolved mostly from online multiplayer games. From simple text-based games, over time, virtual worlds integrated first 2D and later 3D graphics and developed various interactive possibilities to their players. These included chat, voice chat, avatar customization and other. Although virtual worlds have their roots in games, today's virtual worlds take a different course and focus on subjects like distance teaching and learning, virtual collaboration, and social networking (Gütl, Chang, Kopeinik, & Williams, 2009; Scheucher, Belcher, Bailey, Dos Santos, & Gütl, 2009; Churchill & Snowdon, 1998; Virtual World Timeline, 2009). The rest of this Chapter presents and elaborates on the currently most popular and interesting virtual worlds.

6.3.1 Second Life

Second Life (SL)[10] is currently the most popular virtual world. SL is an Internet-based virtual world created entirely by its users who are called "Residents". The development of SL began in 1999 and is accountable to a company called Linden Lab. SL was launched on June 23, 2003 and till 2006 was operating on the sidelines of the hardcore gaming worlds like World of Warcraft (Sivan, 2008). In late 2006 and early 2007 SL received international attention thanks to mainstream news media (Virtual World Timeline, 2009). This huge jump in popularity is depicted in Figure 14.

In SL, residents can communicate, collaborate, participate in individual and group activities and explore the world through motional avatars. Residents can also create and trade virtual items and services with each other. For this purpose SL has a Marketplace that supports transactions worth millions of US dollars every month. The trade is performed in Linden dollar currency, but SL provides several online exchanges

[9] http://www.worldofwarcraft.com/index.xml
[10] http://secondlife.com/

where residents can convert these to US dollars (Virtual World Timeline, 2009). Residents can build and personalize their avatars, create or buy private virtual spaces (lands), and objects (houses and clothes). This is all possible through a powerful and easy-to-use interface. As for communication possibilities, residents can choose from text-based chat, instant messaging and recently added voice chat (Hendaoui, Limayem, & Thompson, 2008). A sample environment in SL can be seen in Figure 15.

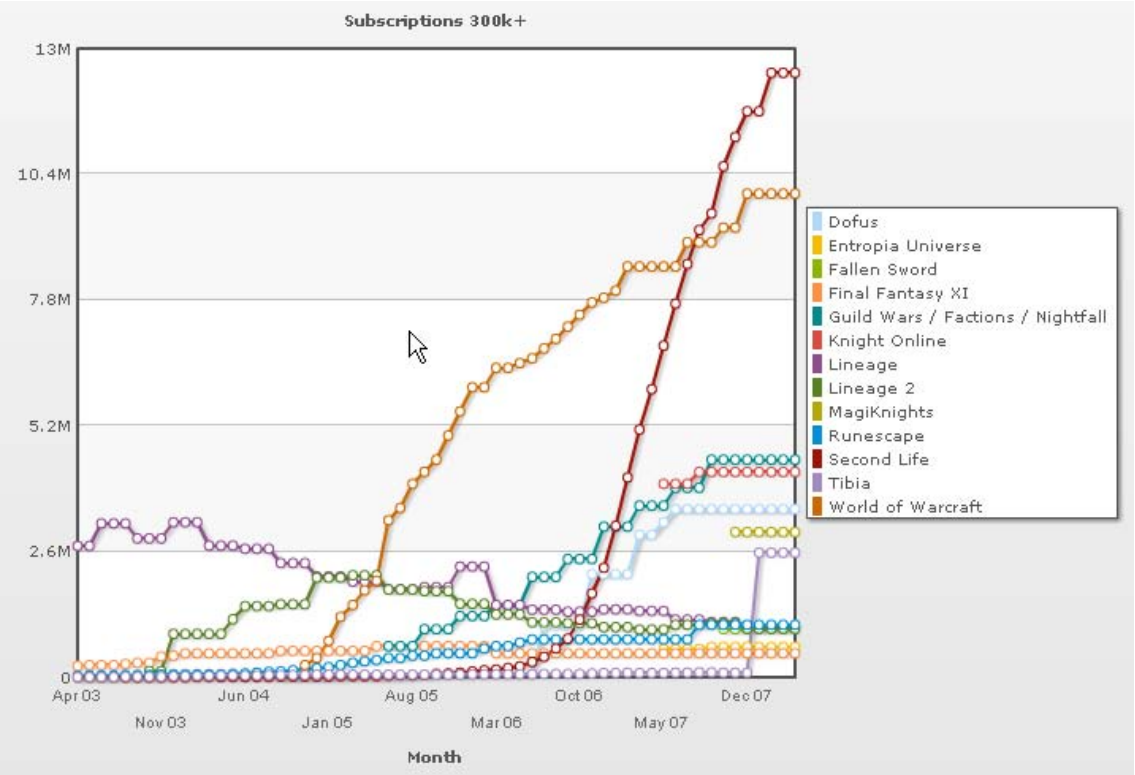

Figure 14: The growth of WOW and SL compared to other virtual worlds (Sivan, 2008)

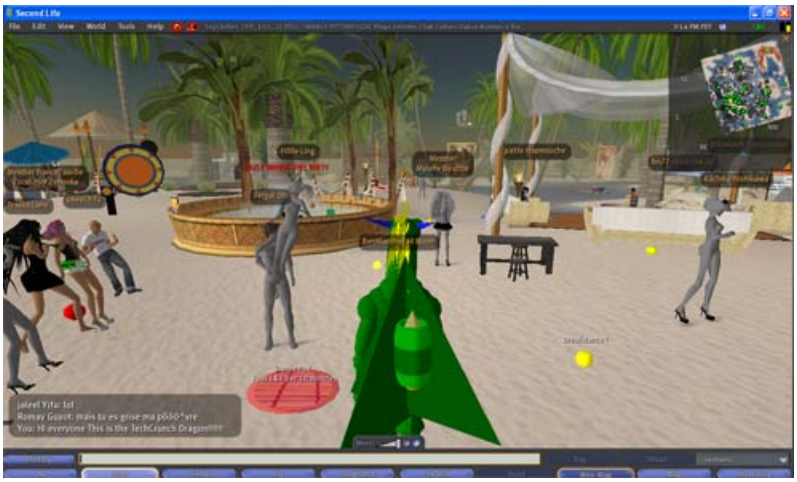

Figure 15: A sample scene from SL (CrunchBase, 2009)

Linden Lab also incorporated an innovation in SL in form of a scripting language, called Linden Scripting Language. With this scripting language users can program their virtual objects within SL (Second Life, 2009a).

The potential of SL was also recognized by a vast number of companies and organizations. IBM, STA Travel, Sears, and BP are just some examples of companies that purchased islands in SL. In this way, innovative companies are exploring the possibilities of virtual worlds for commercial uses (Owens et al., 2009).

Today, SL has nearly 16 million registered users worldwide and thus the biggest user community (Second Life, 2009b). Moreover, Gartner Group predicts that *"eighty percent of active Internet users will have a 'second life' by the end of 2011"* (Gartner, 2009).

6.3.2 There

There[11] is a 3D social world that was released in 2003 by There.inc. The company branched in 2005 to form Makena Technologies and Forterra Systems. Today, There has nearly two million users and its creators define it as *"the destination of choice for anyone who wants to experience the power of chat combined with the fun of online games"* (There, 2009).

Figure 16: A screenshot from There (Brown & Bell, 2004)

There is primarily used for socializing and has less role-playing then MMORPGs and defines itself as a service providing a shared experience that allows people to interact in an online society (Virtual World Timeline, 2009). The most important characteristic of There is objects. Objects have a dual role: they are shared 3D objects

[11] http://www.prod.there.com/info/

that can be manipulated by all users and also objects that can be bought and sold through a 2D web interface. Further, There incorporates a flexible overlapping chat (see Figure 16) with a wide range of conversation topics (Brown & Bell, 2004).

Although There is regarded to as an online game, it has no definite goal. However, it supports a number of multiplayer games such as buggy races, treasure hunts, paintball and others. Avatars in There are created by the user upon the registration and can express emotional gestures (Brown & Bell, 2004; Virtual World Timeline, 2009).

6.3.3 Project Wonderland

Sun Microsystems Inc[12] also recognized the potential of virtual worlds and released *Project Wonderland*[13]. Project wonderland is *"a 100% Java and open source toolkit for creating collaborative 3D virtual worlds. Within those worlds, users can communicate with high-fidelity, immersive audio, share live desktop applications and documents and conduct real business"* (Wonderland, 2009a). This virtual world is completely extensible and developers are able to create completely new worlds as well as extend and provide new functionality to existing worlds. A sample virtual world created in Project Wonderland can be seen in Figure 17. The latest version of this virtual world is version 0.5. This version has a better architecture and its functionality is organized in modules. Through the module system, Project Wonderland's functionality can be extended easily than in previous versions.

Project Wonderland 0.5 relies on several open-source projects for key technologies (Wonderland, 2009b; Wonderland, 2009c):

- *Project Darkstar*[14] - provides the scalable, persistent server software architecture
- *jVoiceBridge*[15] – provides realtime immersive stereo audio with distance attenuation
- *jMonkeyEngine*[16] – provides the scene graph on which the 3D world is built

One of the most important features in Project Wonderland is application sharing. In this virtual world, users can run and share various applications. These applications include 2D and 3D applications such as the multi-user PDF Viewer and the SVG White board (WonderlandWiki, 2009c).

[12] http://www.sun.com/
[13] https://wonderland.dev.java.net/
[14] http://www.projectdarkstar.com/
[15] https://jvoicebridge.dev.java.net/
[16] http://jmonkeyengine.com/

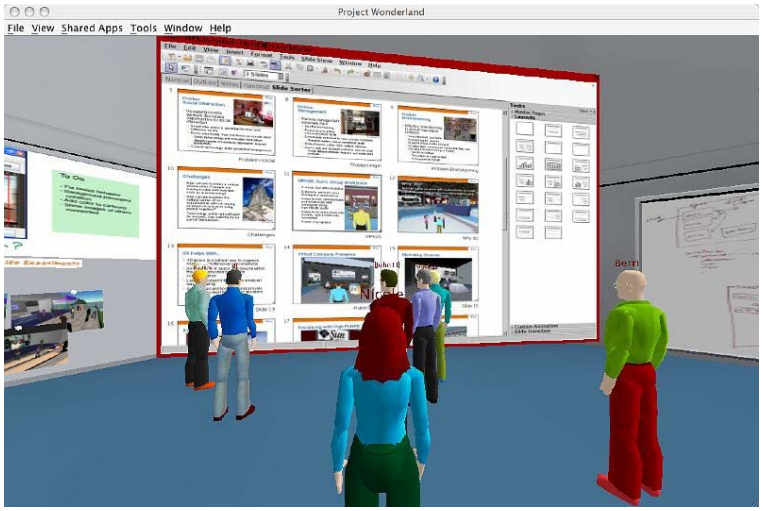

Figure 17: A screenshot from Project Wonderland (SunBlog, 2009)

6.4 Factors of Virtual Worlds

Virtual worlds that are interesting in context of this thesis are virtual worlds where users can participate in more profound and meaningful actions than those in gaming worlds. Sivan (2008) defines these worlds as “Real Virtual Worlds”. These worlds are an aggregation of four factors which represent a 3D3C definition of virtual worlds (Sivan, 2008):

- **3D World** – a dynamic world with visible objects such as avatars, houses, and cars. The world has similar surroundings to those in real life. Hence, the world has land, a sky, a sun, gravity, water, and fire. The avatars can move freely in the world and users can view the world from different points of view.
- **Community** – the creation of TV and the Web (at its early point) had a negative effect of human socialization. This solitude began to evaporate with the advent of technologies such as e-mail, chat, and multi-player worlds. In such surroundings, users get involved in virtual communities.
- **Creation** – in current virtual worlds, the users are able to create their own content and thus impact the look and feel of the world. This was the greatest invention and technological achievement of Second Life.
- **Commerce** – new trends in virtual worlds incorporate commercial possibilities and perform transactions that are worth millions of dollars (Virtual World Timeline, 2009). This was also a Linden innovation where users can purchase and sell objects in Second Life for an imaginary currency and later exchange that currency for a real one.

6.5 Collaborative Virtual Environments

Various companies have recognized the potential of virtual worlds. Moreover, they expect that the visual, aural, and spatial dimensions of virtual worlds will enrich electronic interaction between virtual collaborators. The potential lies in the fact that these dimensions are absent in lean communication channels such as chat, e-mail and others (Kharif, as cited in Kahai, Carroll, & Jestice, 2007, p. 61). As elaborated in 6, 6.2, and 6.3, virtual worlds provide auditory and visual communication channels. In this way, virtual worlds combine the modes presented in section 5.5: the users can have an audio conference by using voice chat communication, they can use CMC through tools like regular text chat, and they can engage in a videoconference by seeing the avatars of other users while communicating. Hence, virtual worlds excel upon previous communication modes and provide greater possibilities for virtual collaboration.

The advantages of virtual worlds for virtual communication and collaboration when compared with other communication options have been recognized by a large number of organizations. For example, several companies constructed environments in Second Life and Project Wonderland in which their employees collaborate on various projects (Hendaoui, Limayem, & Thompson, 2008; MPK20, 2009). Moreover, the application of virtual worlds in context of collaborative learning activities and knowledge transfer was also studied (Gütl, Chang, Kopeinik, & Williams, 2009). Through this trend of virtual world usage, another subgroup of virtual worlds emerged. These are called collaborative virtual environments (CVEs). *"CVE is a distributed, virtual reality that is designed to support collaborative activities. [...] systems intended to support individual and collaborative activities must be designed with the user's social and cognitive task characteristics and requirements considered explicitly"* (Churchill & Snowdon, 1998, p. 1).

According to Biuk-Aghai (2001), a CVE should minimally involve following characteristics:

- **Action** – an operation that a user can perform within a CVE (e.g. open an artefact)
- **Artefact** – a document or other object which is located in, or can be accessed from, the CVE (e.g. a text file)
- **Communication channel** – a medium that enables users to communicate with each other (e.g. chat, forum, etc.)
- **CVE** – the virtual environment itself, which provides users with a means to collaborate and communicate
- **User** – a representation of the human user who uses the environment to collaborate with other users (e.g. avatar)

Following the requirements imposed above, virtual worlds such as Second Life and Project Wonderland provide the environment suitable for effective virtual

collaboration. In addition, Churchill and Snowdon (1998) state that in order to support collaborative and cooperative activities, virtual environments have to provide task appropriate information representation and communication tools, and appropriately designed landscapes that foster needed activities.

When it comes to the fulfillment of requirements imposed by the nature of collaborative work itself, presented in Chapter 5, virtual environments offer a wide range of possibilities and enhancements (Churchill & Snowdon, 1998):

- Virtual environments should enable users to switch easily between individual and collaborative tasks by providing the possibility to alter representations, navigate easily, and to communicate quickly and efficiently as the need arises.
- Communication and collaboration is fostered when people share a common space. Benford et al. argues that a shared virtual space that follows shared conventions provides a natural and intuitive way to navigate and to interact with individuals and groups (as cited in Churchill & Snowdon, 1998, p. 10).
- Flexible and negotiated interactions as well as flexibility in the design of shared data representations are supported by the flexibility of virtual environments.
- The communication is richer because of the presence of gestures and non-verbal cues provided by graphical representations of users in a virtual environment (avatars).

6.6 Opportunities and Challenges

Virtual worlds offer a wide range of possibilities that are not possible with other technologies. Users can create an environment (world) of their choice and also perform specific tasks like displaying slide shows, listening to recordings, modeling and many more. These environments are able to support virtual collaboration better that other technologies. This advantage comes from vividness and the technology capabilities that can enhance interaction and collaboration of virtual team members (Owens et al., 2009).

Compared to other communication modes, outlined in section 5.5, virtual worlds offer much more advantages. In contrast to videoconferencing that offers communication through what is referred negatively as “talking heads”, virtual worlds offer greater scope for use of body movements and spatial orientation through the use of avatars (Churchill & Snowdon, 1998). Also, Scheucher et al. (2009) found that users preferred the avatar interaction over the standard video chat. The advantage of virtual worlds over audio conferencing is that virtual worlds provide visual cues that are absent in audio conferencing (section 5.5.2). Finally, virtual worlds provide greater awareness of others and context cues, what is not the case with CMC (section 5.5.3). Hence, one of the greatest advantages of virtual world technology is the presence of avatars through whose capabilities in current virtual worlds users can show gestures, exchange non-

verbal cues, recognize who is who, know who is doing what, and know where who is (Churchill & Snowdon, 1998).

According to Owens et al. (2009), following features are distinctive for virtual world technology and provide opportunities for virtual collaboration:

- 3D life-like conversation and an immersive environment for interaction
- purposeful nonverbal communication (even the ability to touch)
- the ability to control avatar appearance, avatar behavior, and the environment

Virtual Worlds can also help in reducing the negative effects imposed by geographic distance and cultural differences. Geographic distance does not have such a strong effect because avatars are performing the work together in a shared space and are able to interact with each other. Cultural differences can be reduced by the possibility to create a graphical representation (avatar) that is independent of racial or cultural differentiations. This also helps virtual teams develop the all important trust which can additionally be supported by the use and control of verbal and nonverbal communication cues, concurrent use of multiple communication channels and the environment where team members can socialize and engage in non-formal communication (Owens et al., 2009).

Distributed project management can also benefit from virtual world technology. Owens et al. (2009) list following opportunities that virtual worlds make possible for the management of distributed projects:

- immediate feedback during team communication
- development of trust through multiple channels of communication
- elimination of geographic boundaries
- the ability to see one another's artifacts as team members work on them

Just as with any technology, the possibilities of virtual worlds also come with some challenges. Owens et al. (2009) enumerate following challenges of virtual world technology:

- **Client software and hardware** – each team member has to download client software that requires memory and graphics. In addition, virtual world audio capabilities are not robust, and desktops have to be high end.
- **Learning curve** – it can take a considerable effort to learn to operate within the environment in the way that would exploit its full capabilities.
- **Balancing worlds** – project managers need to realize that a virtual world is not a substitution for the real world. Some cases could still require traditional face-to-face meetings.

- **Acceptance** – this is the deal breaker of the technology and confidence must be present that the project manager and employees are in fact going to use the technology.
- **Distractions** – people's avatars could wander off and drift away.
- **Norms of behavior** – some users could differ from usual norms of behavior (i.e. not convey their thoughts when using virtual world technology).
- **Uncertainty of behavior** – there is no control over the behavior of other people's avatars.
- **Representation** – working through a virtual persona might request adjustments from some users.
- **Security** – virtual worlds are often public places with limited security features.

6.7 Related Work

Due to the fact that the focus of this master thesis lies on software engineering and the possible application of 3D virtual worlds in its distributed scenarios, it is important to see what the current application of 3D virtual worlds in this context looks like. Unfortunately, even though the 3D virtual worlds have been examined and created for the use in various contexts, such as learning and knowledge transfer (Gütl et al., 2009), at this moment there could not be found much about their application for software engineering. The only example of the application of virtual worlds in this context is Sun's *Virtual Workplace MPK20*[17].

The purpose of this virtual world environment is to support the majority of Sun's workers, who work remotely. The creators state that *"in this 3D world, employees can accomplish their real work, share documents, and meet with colleagues using natural voice communication"* (MPK20, 2009). This world is built by using Project Wonderland which provides most functionality for the environment. However, besides preconfigured functionality that comes with Wonderland, MPK20 also integrates Sun Lab's *Porta Person*[18]. This integration enables bridging the virtual and physical worlds in meetings. The Porta-Person is present in the real world and consists of a camera, speakers, a microphone, and a computer. These entities are all found on a remote-controlled, rotating platform which provides the remote control of the Porta-Person. In addition, MPK20 provides Team Rooms and Live Applications that simulate and provide a workplace for software teams similar to those found in the real world.

[17] http://research.sun.com/projects/mc/mpk20.html
[18] http://research.sun.com/projects/mc/porta-person.html

6.8 Summary

Virtual worlds represent the fourth mode of virtual collaboration. A virtual world is an interactive, physical and persistent computer program that uses rendered images to create a simulated 3D environment in which users can interact through the use of their graphical representations or avatars.

Virtual worlds originated from virtual reality and online multiplayer gaming worlds. Although virtual reality has a significant impact to the evolution of virtual worlds, their development is closer tied to that of online games. These games started as simple text-based games and over time evolved in huge 3D environments where players can jointly pursue different quests and goals.

Today, virtual worlds have moved from gaming context and focus now on more advanced topics such as socializing, distant teaching and learning, and virtual collaboration. Examples of such worlds include Second Life, There, and Project Wonderland. These more profound virtual worlds have four common factors: 3D world, creation, community, and commerce.

The richness of communication in virtual worlds has also made them interesting for virtual collaboration. Virtual worlds that specialize in virtual collaboration are called collaborative virtual environments (CVEs) and fulfill the requirements of collaborative work. A CVE should minimally provide action, artifact, communication channel, CVE, and a user. Through a number of different communication channels, avatars, and a shared common space, virtual worlds excel over other three communication modes and entail a number of opportunities for virtual teams, virtual collaboration, and the management of distributed projects.

This Chapter concludes the theoretical part of this thesis. The practical part begins with the next Chapter which investigates the requirements of a virtual collaboration tool in a 3D world. The requirements are enumerated based on the findings of the Chapters in the theoretical part. Their fulfillment should provide distributed software teams with an optimal environment for the execution of software projects in a distributed setting.

Chapter 7

Requirements and Design

Previous sections represent the theoretical basis for the practical work of this thesis. The issues and challenges in current software engineering and project management have been outlined. Further, virtual teams have been presented together with the phenomena of virtual collaboration. Finally, last Chapter presented virtual worlds as a state of the art technology that provides a number of possibilities and enhancements for virtual collaboration. Based on the findings of these sections, this Chapter summarizes and defines the scope of functionality for a virtual world collaboration tool.

7.1 Functional and Non-functional Requirements

The purpose of this master thesis is to create an environment in a virtual world where distributed software team members could come and collaborate on software engineering projects. This virtual environment will provide the functionality and capabilities which will confront the issues and challenges of distributed software teams which have been described in the theoretical part of this thesis.

Based on the findings in Chapters 2, 3, 4, and 5, two main groups of requirements can be derived. The first group is concerned with the issues raised by the nature of distributed work and has three subgroups: communication, team awareness, and project and knowledge management. The second group is primarily focused on the tasks imposed by the software development process and is arranged according to the phases and corresponding tasks defined in section 5.3.1.

The goal of this system is to provide an optimal environment for distributed software development teams. In this environment, they would have access to richer communication channels, would have better sense of the team, and would additionally have all the tools and applications that they need in order to successfully complete a software project. Hence the following requirements specify such optimal system.

The functional requirements are bundled in following groups and subgroups and are defined as follows:

A Communication

A.1 **Synchronous communication:** team members have to be able to communicate synchronously with other team members in both cases when they are online and offline in order to discuss current tasks and issues.

A.2 **Asynchronous communication:** in cases where one or more team members are absent for whatever reason, present team members have to be able to communicate with them asynchronously.

A.3 **Social and informal communication:** the system has to enable and foster social, informal and non-task related communication between team members as this kind of communication helps resolve issues and improves knowledge transfer (see section 2.3).

A.4 **Exchange of nonverbal cues:** the system has to enable the exchange of nonverbal cues (i.e. body language, voice tone) between distributed developers. In this way, the system should provide richer information exchange (see section 5.5).

B Team awareness

B.1 **Presence awareness:** team members have to know whether other team members are present or not. This would reduce possible frustration among virtual team members.

B.2 **Absence awareness:** in case some team members are absent, other users should know the reasons for this absence in order to mitigate possible frustration by for example, thinking other team members are deadbeats.

B.3 **Activity awareness:** team members should know if present colleagues are actively working on the project at any given time.

B.4 **Team structure:** the system should provide developers with the information about their team roles.

B.5 **Expertise and responsibilities:** developers need to have information about the expertise and responsibilities of other team members in the current project.

B.6 **Contact information:** for cases such as when a team member is absent but his knowledge or insight are needed in the moment, the developer has to have access to contact information of his work colleagues.

C Project and knowledge management

C.1 **Team meetings:** the system has to enable developers to organize and participate in team meetings.

C.2 **Project management:** team members have to be able to manage their project progress and tasks.

C.3 **Shared data:** the system enables developers to share and exchange artifacts and documents on which they are collaborating and which are of relevance for their project (such as various media, graphics, pdf documents, and others).

C.4 **Versioning system:** as team members are often collaborating on the same parts of the project, the system needs to manage different versions of the same data.

C.5 **Access to other CMC media:** in cases where one or more team members do not have access to the virtual environment, other team members need to be able to communicate with them via other CMC means (such as email, discussion boards, and others).

D Software process relevant requirements[19]

D.1 Analysis

D.1.1 **Word processor:** the system has to provide users with a text editor tool where users can create deliverables such as requirements specification, informal scenarios and others.

D.1.2 **UML tool:** the system has to provide users with a UML tool so that team members can create analysis deliverables such as Use Case Diagrams.

D.2 Design

D.2.1 **UML tool:** the system has to provide users with a UML tool so that team members can create design deliverables such as object, class, sequence, and collaborative diagrams.

D.2.2 **Graphical tool:** team members have to have access to a graphic tool in order to create user interface mock-ups.

D.2.3 **CASE tool:** developers have to be able to use a CASE tool for creation of class skeletons.

[19] The requirements define the development phase relevant functionality of the system needed by developers in each software development phase. However, two comments are worth to outline. First, the requirements specification can be executed by using various tools that range from text editors to specialized input forms. For the purpose of the prototype in this thesis, a word processor will be used for the completion of such tasks. Second, in this thesis, software testing supposes a simple testing of written classes and the resulting program and does not include advanced topics such as testing on different platforms. Hence, the same CASE tool that is used for the implementation will also be used for the testing of the code in this prototype.

D.3 Development

D.3.1 Word processor: in order to create an implementation plan, the system has to provide developers with a text editor.

D.3.2 CASE tool: developers have to be able to use a CASE tool in order to code the software designed in the previous phase.

D.4 Testing

D.4.1 Word processor: in order to define a test plan and test analysis report, team members need to have a text editor.

D.4.2 CASE tool: the system has to provide developers with a CASE tool in order to conduct system testing.

D.5 Phase wide requirements

D.5.1 Internet access: team members need to have internet access for the purpose of information acquiring.

D.5.2 Presentation tool: the system needs to provide developers with a presentation tool for the creation of presentations after deliverables.

Due to the large scope of functional requirements, only a part of functional requirements are going to be implemented in the prototype. Hence, for the purpose of this thesis, it is supposed that virtual software developers perform all of their project related work entirely within the virtual world. Consequently, the prototype for this thesis does not require the versioning system and the integration of CMC media.

The non-functional requirements are requirements that are not directly concerned with the functions the system needs to provide. They are more constraints on the services and functions that the system should offer. As they prevalently concerned with the system as a whole, their unfulfilling can result in an unusable system (Sommerville, 2007).

The following requirements represent the critical constraints on the system:

- **Performance** – the system has to show high responsiveness to the actions of users. The system should not freeze or hold often as it would render it as unusable for the virtual teams.

- **Usability** – the system should provide easy navigation and movement of the users. In addition, all the functionalities should be intuitive and easy to use.

- **Reliability** – the system has to be highly reliable, as it is the main mean of executing the project. Hence, it should be available at least 99% of the time.

7.2 Conceptual Design

Due to the nature of the tool that is going to be developed, which includes the virtual world as a single place that provides various functionalities for a number of users, the client-server architecture is an appropriate choice.

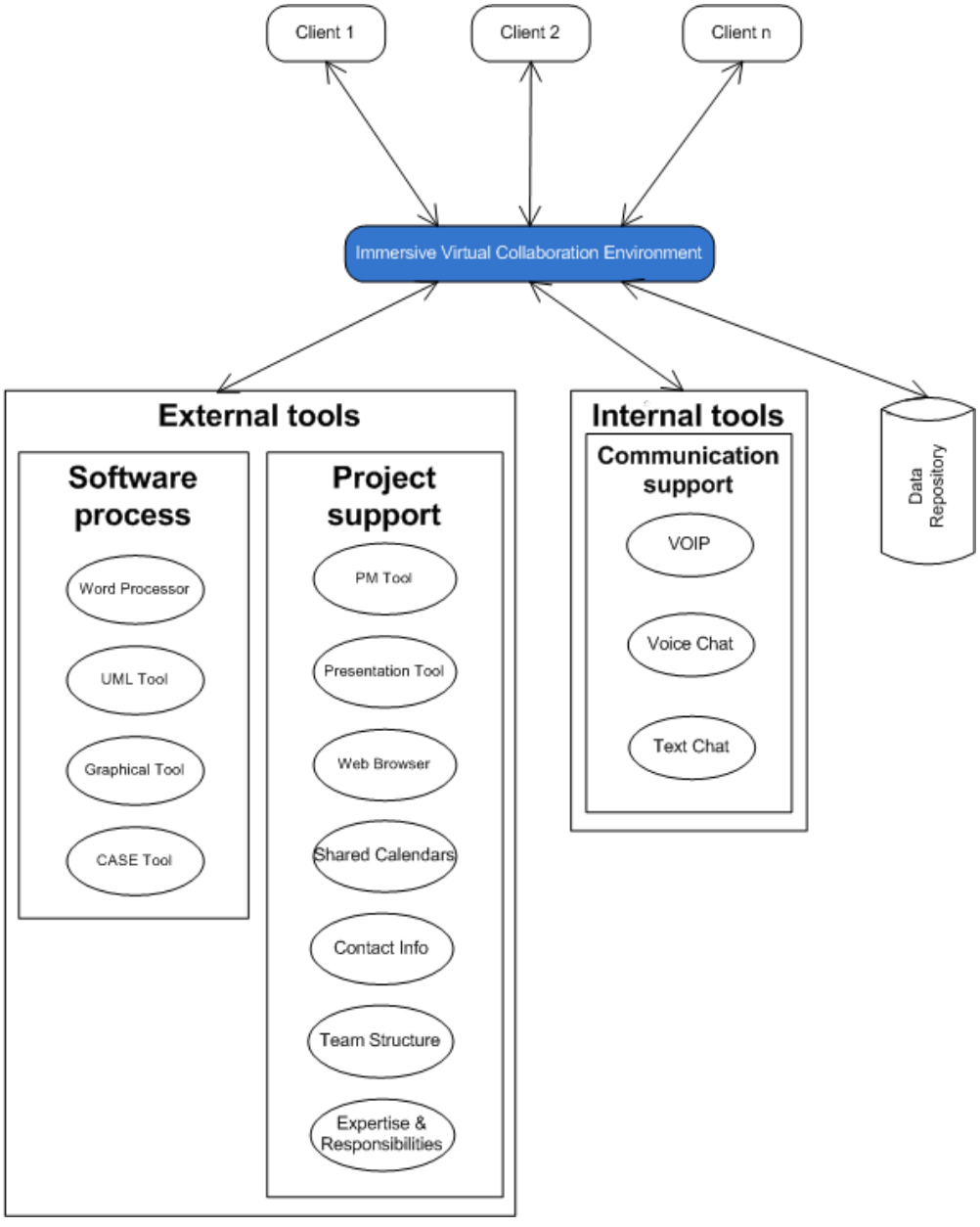

Figure 18: Conceptual Architecture

Clients connect to the server that provides the required functionality (see Figure 18). The server provides different tools for collaboration as well as a central data repository for storing of created project related data. The server provides external and

internal tools. Internal tools are used for communication support and consist of a Voice over IP (VOIP) tool, voice chat and text chat. In this way, collaborators have multiple communication channels. The external tools are grouped in two groups. One group supports the software process and represents all tools that virtual teams need for software engineering (i.e. CASE tool, UML tool, word processor, and graphical tool). The second group acts as support for the execution of the project itself. It consists of a PM tool, web browser, presentation tool, shared calendars that provide information about the presence or absence of team colleagues (Hayes, 2004), and other information such as information about responsibilities, experience, and contacts. All the data that is created by using these applications is stored centrally in the data repository.

7.3 Design of the 3D Environment

Due to the fact that the author of this thesis, during the time of his studies, was mostly engaged in projects performed by groups of four software developers, this team size is chosen for the design of the environment. Hence, a software team in this thesis consists of three developers and a project manager. It must be noted, however, that the project manager can act also as a developer and can take active part in the project.

Another interesting aspect of the 3D environment design is the information visualization. In this case, the most interesting technique is the room metaphor. By using this technique, certain tasks can be allocated to certain rooms and users can move to specific rooms as needed. Hence, this gives users another way to structure and organize their work (Young, 1996).

The functional requirements enumerated in 7.1 provide the base for the creation of rooms. Obviously, the environment needs a room per software development phase. Additionally, after observing other requirements, the environment requires a room where team members can socialize, a room where they can hold meetings, and a room where activities such as project management and team organization and information take place. Hence, the environment should entail following rooms:

- **analysis room**
- **design room**
- **implementation room**
- **testing room**
- **social room**
- **organization room**
- **meeting room**

The rest of this subsection will present single rooms and the applications that these rooms should poses, based on the requirements from 7.1.

7.3.1 Analysis Room

The analysis room consists of a 3D space and applications that are needed in this development phase (see Figure 19). As this phase is heavily focused on requirements and use cases, this room has a text editor and an UML tool for every team member. In addition, web browser and a presentation tool are very useful for every project and have a place in this room.

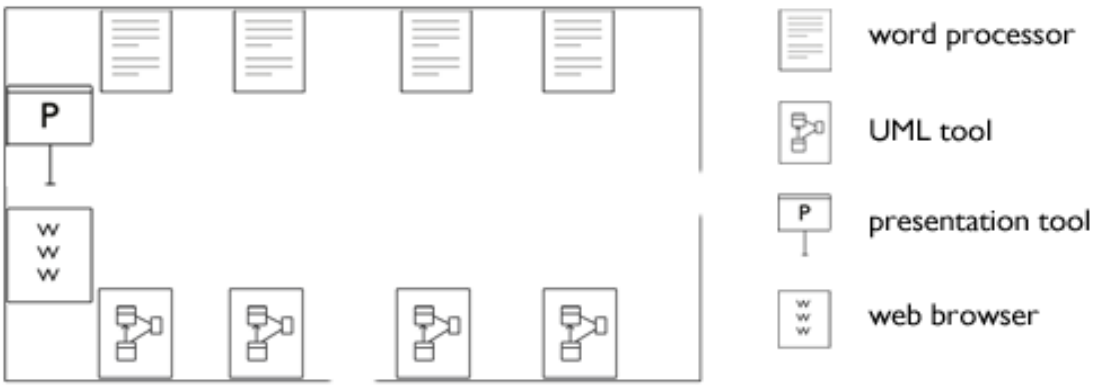

Figure 19: Analysis room and its applications

7.3.2 Design Room

Similarly to the analysis room, design room also consists of applications that are required for the completion of phase related tasks. Hence, the design room provides for each team member an UML tool for the creation of various design diagrams. In addition, two CASE tools for the development of class skeletons, two graphics tools, a web browser and a presentation tool are available (see Figure 20).

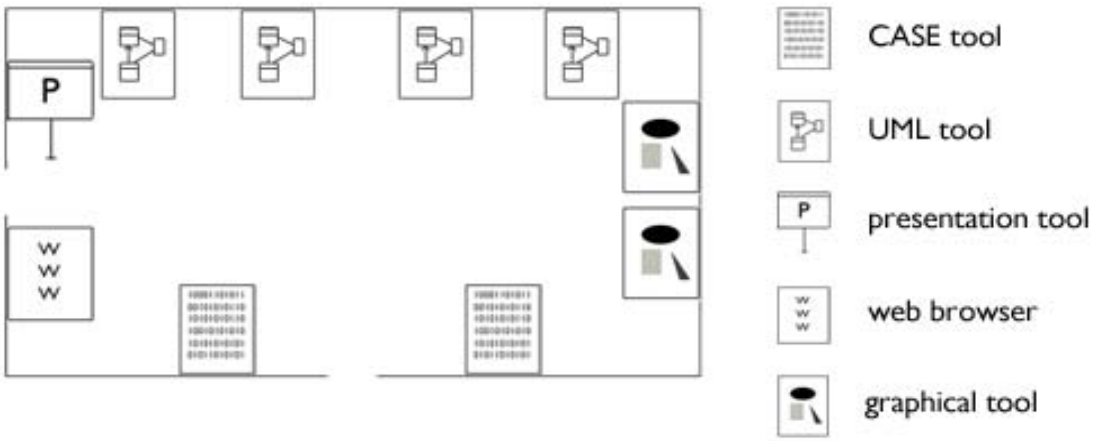

Figure 20: Design room and appropriate applications

7.3.3 Implementation Room

The implementation phase is focused on the coding part of the project. Due to this focus, implementation room provides software developers with four CASE tools. Also, in order to write an implementation plan, team members can use existing text editors. Similarly to previous rooms, this room also has a web browser and a presentation tool (see Figure 21).

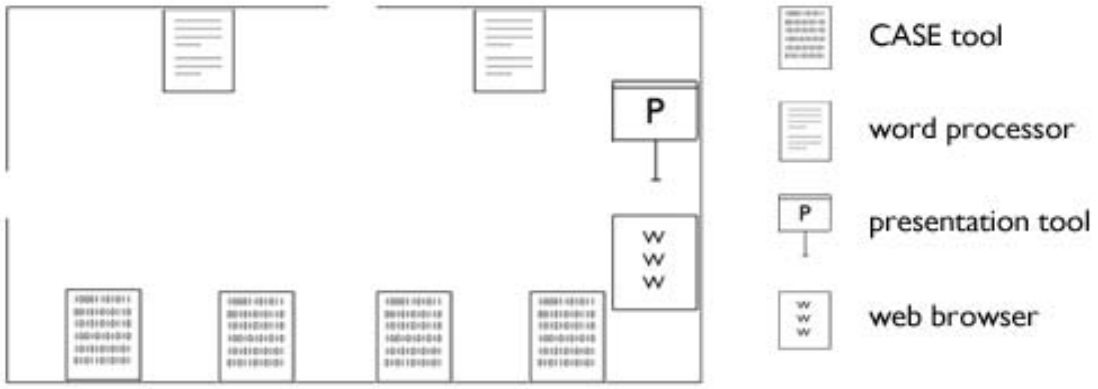

Figure 21: Implementation room with applications

7.3.4 Testing Room

Similar to the implementation room, the testing room provides the same tools. However, these tools are there for different purposes (see Figure 22). Four CASE tools will be used for various tests of the product, whereas the text editors provide a means for the creation of test plans. A web browser and a presentation tool serve the same purpose as in all other rooms.

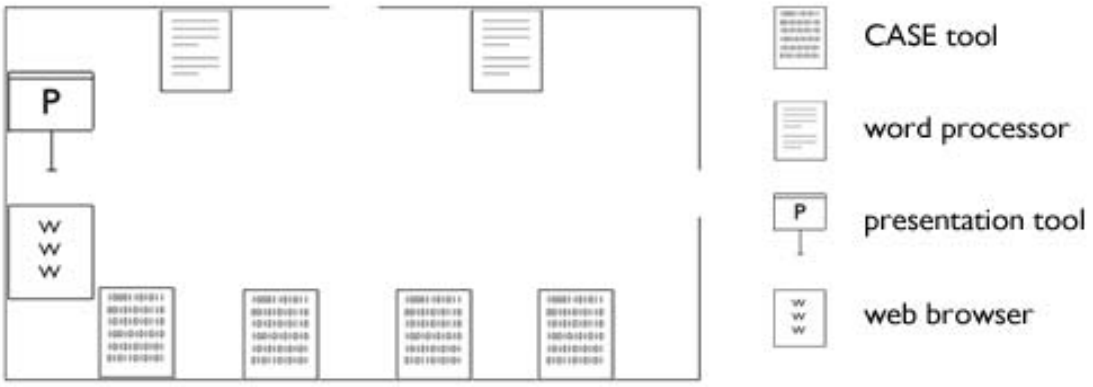

Figure 22: Testing room with tools

7.3.5 Organization Room

Based on the requirements presented in 7.1, the organizational room provides team members better team awareness and also data and tools which are related to the organization and execution of the project (see Figure 23).

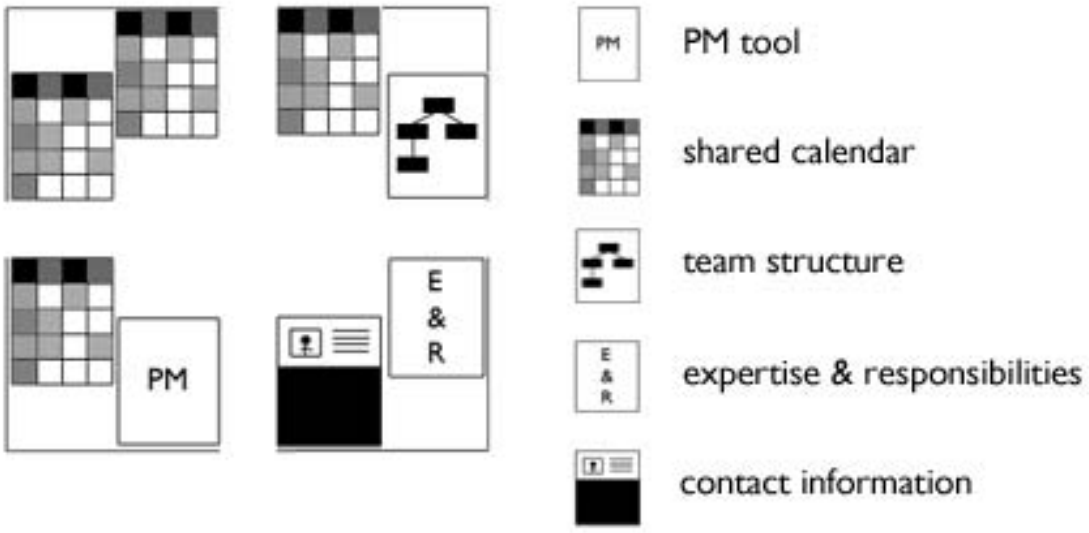

Figure 23: Organizational room with appropriate tools

Each team member has an own calendar which is available to all of his/her colleagues. This should be convenient in cases when a team member is not present in the world and when colleagues need to know the reason behind this absence or when he/she will be back in the world and available for collaborative activities. This room also displays contact information, team structure, expertise and responsibilities of each member for a particular project. Through this team members should acquire better awareness of the team and also know who to contact in order to resolve burning issues in the project. Finally, there is a project management tool that helps the team to track their progress, gives overview of the tasks and deliverables and provides additional information needed for the execution of the project.

7.3.6 Meeting Room

As meetings are usually composed of discussions and presentations, besides a presentation tool, this room will provide a text editor. For the purpose of this prototype, the text editor will provide multiple functions. Besides the use for taking notes during the meeting, it will also be used for manipulation of the meeting agenda. For various information needs, a web browser is also located in this room (see Figure 24).

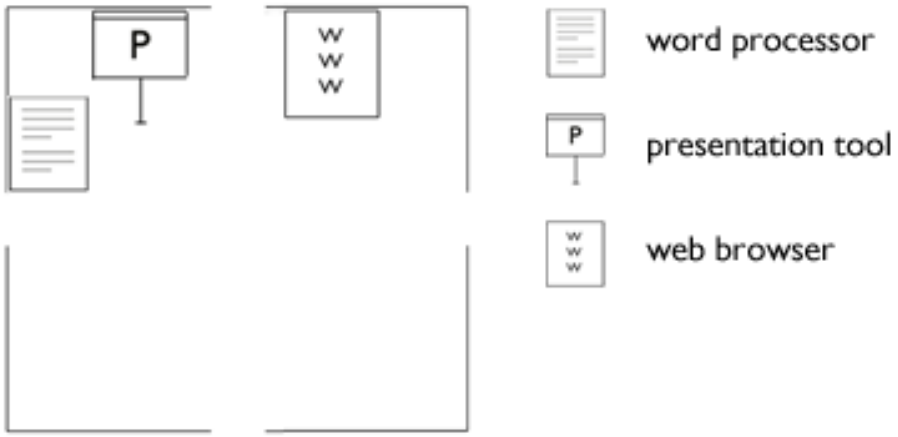

Figure 24: The meeting room

7.3.7 Social Room

The social room will not possess any specific application as these are essentially unneeded. Conversely, it will resemble an environment usually found in such rooms in the physical world. Artifacts such as coffee machines, water coolers, plants and furniture should accomplish this goal easily.

7.3.8 Bringing it all together

After single rooms and their applications have been presented, it is now time to take a look at the big picture. The layout of the complete environment is depicted in Figure 25. Even though the figure is assembled from single room layouts presented above, there are two interesting things to point out. Firstly, the social room is placed in the middle of the environment. Secondly, the phase rooms are located in such a way that the movement through them resembles both the waterfall and incremental software process. This is depicted by the light blue arrow.

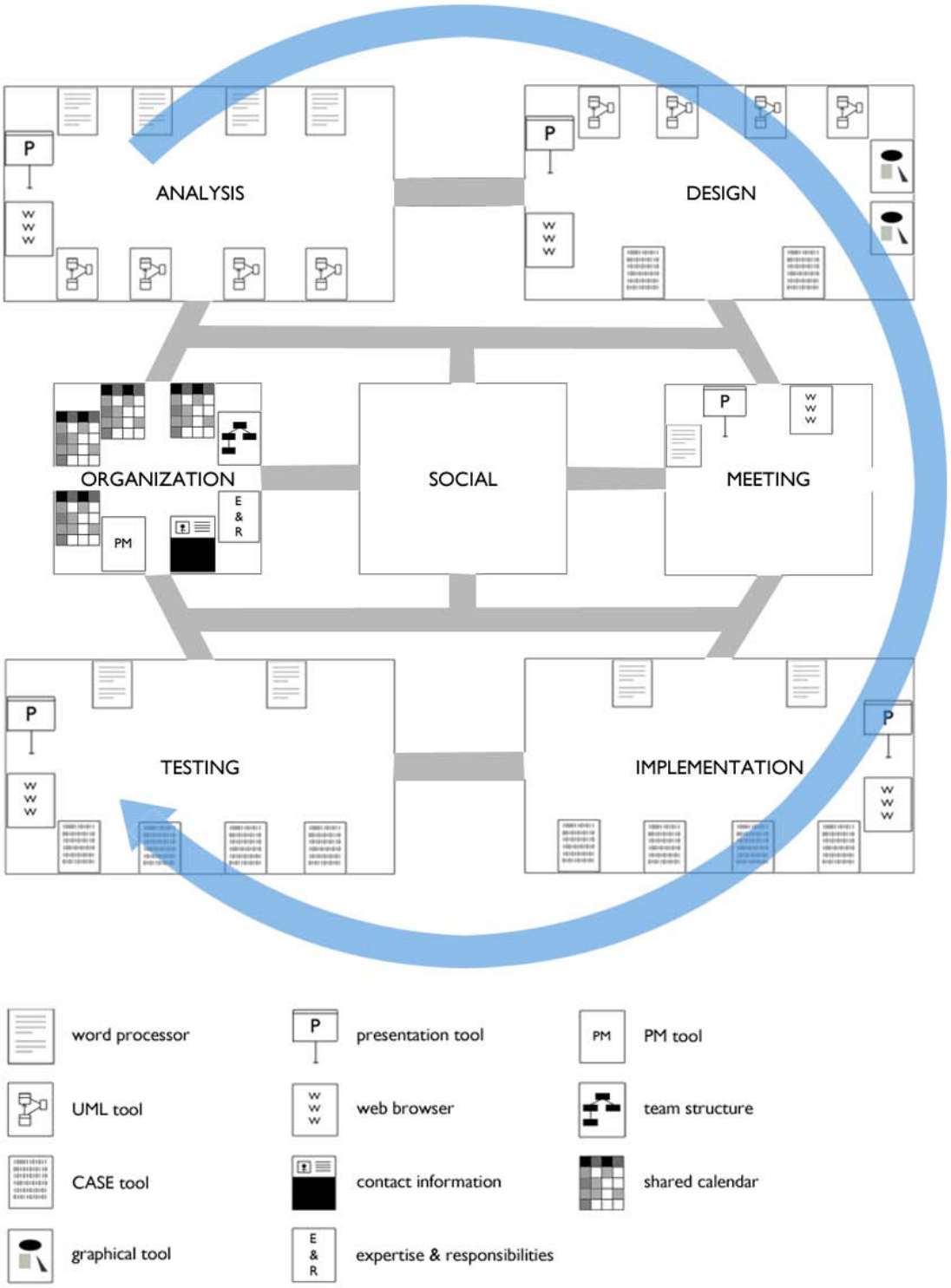

Figure 25: Layout of the virtual environment

7.4 Summary

This Chapter presented the requirements for a virtual collaboration environment for software engineering. The requirements are grouped in four major categories: communication, team awareness, project and knowledge management, and software process. In software process category, the requirements are based on the specific needs of every development phase.

The application's architecture is built around the client-server model. A number of clients (i.e. software developers) connect to the server. The server then provides various tools and artifacts that support the software development process. These range from the 3D environment itself, to multiple communication channels, and finally to specific tools needed in the development process (such as an UML, CASE and other tools).

The collaboration environment is built around the room metaphor. There are a total of seven different rooms. Four of them are based on the four main phases of the software process: analysis, design, implementation, and testing. The remaining three act as a support to the process and are occupied with project management, knowledge transfer, and meetings.

The next Chapter is concerned with the implementation of this virtual environment. The decision about which 3D virtual world will be chosen for the implementation will be brought. It will present various tools and applications whose functionality will be used to fulfill the requirements and also describe the way in which the environment was implemented.

Chapter 8

Development of the Virtual Environment

Based on the requirements and the design elaborated in the previous Chapter, this Chapter is focused on the implementation of the virtual collaboration environment for software engineering. The topics include choosing the virtual world in which the environment will be implemented, used tools and applications, the specific architecture of the environment, the process of implementation and usage possibilities.

8.1 Immersive Collaborative Virtual Environment

In order to develop the virtual collaboration environment, the choice has to be made about which virtual world to use for the development. In addition, the chosen virtual world needs to be presented in more detail and the most important characteristics that find use in the development process need to be elaborated. These points represent the focus of this subsection.

8.1.1 Decision-making Process

For the implementation of the previously presented virtual environment, the choice between the most popular virtual worlds (see section 6.3) has to be made. The author of this thesis has chosen Project Wonderland, version 0.5 for following reasons:

- Project Wonderland is 100% Java, hence it is platform independent

- Project Wonderland enables easy to use application sharing across different platforms and this fact makes it very useful for the implementation of the environment

Now that the virtual world has been chosen, the next subsection will elaborate in more detail how this virtual world functions and what its mechanisms are.

8.1.2 Project Wonderland v0.5 Overview

Project Wonderland was briefly introduced in section 6.3.3. In addition to this introduction, this section elaborates on its architecture, server administration, modules and application sharing mechanisms.

The server administration in Project Wonderland v0.5 is done via the Web-Based Administration UI (see Figure 26). Through this interface, called *Server Admin,* every aspect of the world run in Wonderland can be managed. Server Admin is composed of a number of web pages that manage different parts of Wonderland. The pages include: *"Manage Server"*, *"Edit Placemarks", "Manage Apps", "Manage Content", "Manage Groups", "Manage Modules", "Manage Worlds",* and *"Monitor Server"*.

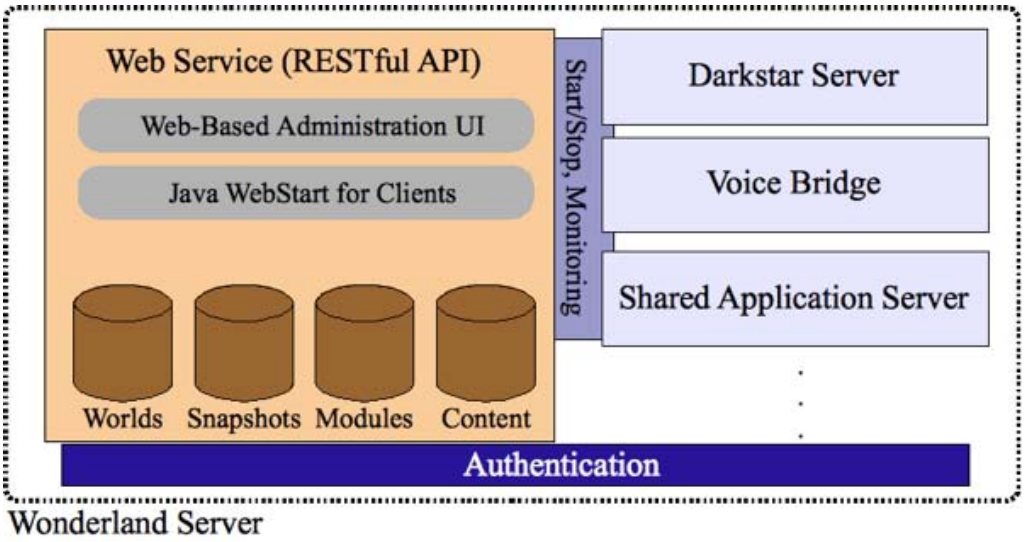

Figure 26: Project Wonderland v0.5 Server Architecture (WonderlandWiki, 2009b)

The "Manage Server" page enables the monitoring and management of four Wonderland-relevant servers: Web Administration Server, Darkstar Server, Voice Bridge Server and Shared Application Server. From this page, the user can start, stop, and restart the servers, as well as edit their settings and see the logs. The "Edit Placemarks" page is responsible for the management of worldwide placemarks. Placemarks represent specific locations in the world that have been saved for future usage. By

clicking on a desired placemark, the user can teleport his avatar to a specific location (WonderlandWiki, 2010).

Through the “Manage Apps” page, the admin can manage shared applications within the world. Wonderland provides the ability to run and share different X11 Applications in the world (WonderlandWiki, 2010). The applications are run by the *Shared Application Server (SAS)*. The only precondition for application sharing is that these are run within the world which is run on a Solaris or Linux server machine. Besides some applications already integrated in Wonderland, such as the multi-user PDF viewer and the SVG Whiteboard, users can share additional applications which are installed on the server machine (WonderlandWiki, 2009c).

The “Manage Content” page is responsible for the management of the content (e.g. 3D models) that is present in the world. The “Manage Groups” page enables the management of different security groups (WonderlandWiki, 2010).

Through the “Manage Modules” page, the admin can manage different modules in the world. *Modules* represent the main way of extending Wonderland and adding new functionality (WonderlandWiki, 2010). Developers can organize new content in modules and integrate it with Wonderland through a module system that manages all the modules within Wonderland. Besides new functionalities, the core functionalities are also mostly organized in modules, what enables an easier management and updates (WonderlandWiki, 2009b). For the purpose of the prototype in this thesis, three modules are particularly interesting. The *“Virtual Phone”* module enables users to make phone calls via VOIP. The *“Text-Chat”* module provides a text chat within Wonderland through which the users can communicate by exchanging text messages. Finally, the *“Voice-Chat”* module provides the users with the capability to engage in audio conversations.

The “Manage Worlds” page provides management features for the initial worlds and creation of world snapshots. Finally, the “Monitor Server” displays the statistics on the current performance of the server (WonderlandWiki, 2010).

As for security, Wonderland provides a module that enables the authentication of users. The authentication module[20] first needs to be downloaded and installed using the “Manage Modules” page. Upon restarting the server, the administrator needs to login with default username *“admin”* and default password *“admin”*. The administrator is then able to change these credentials with much safer content. After login, the “Manage User” page appears on the left and is used to register other users of the Wonderland world. The administrator sets up the user ID and the password for other users (WonderlandAuth, 2010).

Project Wonderland uses cell architecture. Hence every volume of space in the world, such as avatars and 3D objects, is represented through cells. The cells, and hence the 3D world, are organized around a tree hierarchy (see Figure 27). Each cell

[20]http://www.projectwonderland.com/index.php?option=com_docman&task=doc_details&gid=190&Itemid=87&cat=add_ons&Itemid=87&Itemid=87

holds information concerning its position and orientation in the world (WonderlandWiki, 2009a).

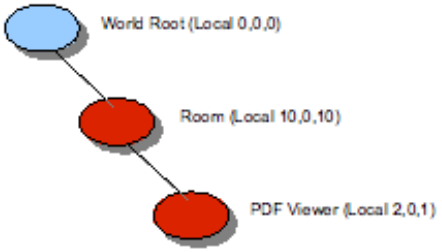

Figure 27: Cell hierarchy (WonderlandWiki, 2009a)

Whereas the *Cell* class maintains the data model for its state, the *CellRenderer* class creates the visual rendering of the cell. A novelty in version 0.5 is the *CellComponents* class. Unlike the approach in version 0.4, where the characteristics concerning the dynamics and communication of a cell were hard coded, the version 0.5 provides the possibility of dynamical addition (and withdrawal) of features to a cell through the *CellComponents* class. In addition, every cell has a server and client representation. Cell represents the cell on the client side, and *CellMO* (MO stands for *ManagedObject*) represents the cell on the server side (WonderlandWiki, 2009a).

8.2 Used Tools and Applications

Due to the great amount of time this thesis already required in the theoretical part, it has been decided not to develop the functionality from scratch but to integrate existing software applications for this purpose. This section presents the tools and applications that are used within the prototype for a virtual collaboration environment for software engineering in order to fulfill the requirements listed in section 7.1. All used software is open source.

Following tools and applications are used within the virtual environment to provide the necessary functionality:

- **OpenOffice.org Writer**[21] – *Writer* is used as a word processor component of the OpenOffice.org package. Its main use is in the phases of software development for the purpose of writing requirements specifications, scenarios, implementation and testing plans. Besides these, it has a use in the meeting room where team members manage their meeting agenda and take notes during the meeting.

[21] http://www.openoffice.org/product/writer.html

- **OpenOffice.org Impress**[22] – *Impress* is a presentation component of the OpenOffice.org software package. It is used for the creation and presentation of various presentations that are needed throughout the project.

- **OpenOffice.org Draw**[23] – *Draw* is the graphics editor within OpenOffice.org package. It is used within the virtual environment for the creation of user interface mockups and similar purposes.

- **Mozilla Firefox v3.5.6**[24] – for the purpose of finding information on the Internet, the virtual environment uses the web browser *Firefox.*

- **Poseidon for UML CE v6.0**[25] – *Poseidon for UML* is freeware application for the creation of models with the Unified Modeling Language (UML). Hence it is used in the analysis and design rooms for creation of UML diagrams. Besides the UML modeling, Poseidon for UML incorporates features such as round trip engineering and the generation of documentation (Gentleware, 2010).

- **NetBeans IDE v6.8**[26] – as a CASE tool, the virtual environment uses *NetBeans*. NetBeans is an integrated development environment (IDE) used for creation of web, enterprise, desktop, and mobile applications using the Java platform, PHP, JavaScript and Ajax, Ruby and Ruby on Rails, Groovy and Grails, and C/C++. NetBeans is written entirely in Java and can run on every platform that has the *Java Virtual Machine (JVM)* installed on it. It is the first IDE that offers complete support for the complete Java EE 6 specification. NetBeans provides features such as an Ant-based project system, Maven support, refactoring, and version control (NetBeans, 2010).

- **GanttProject**[27] – for project management purposes, the virtual environment uses *GanttProject*. It is a cross-platform desktop tool written in Java for project scheduling and management. GanttProject enables the creation of Gantt charts that enable the creation of work breakdown structure, dependencies, and definition of milestones. From the Gantt chart, an appropriate PERT chart can be created. It also provides features such as the assignment of human resources to specific tasks. The allocation of single resources can be seen on the Resource Load chart. As for the supportive features, GanttChart enables the export of charts in different formats such as PNG images, PDF and HTML reports, as well as import and export between Microsoft Project formats (GanttProject, 2010).

[22] http://www.openoffice.org/product/impress.html
[23] http://www.openoffice.org/product/draw.html
[24] http://www.mozilla-europe.org/en/firefox/
[25] http://www.gentleware.com/products.html
[26] http://netbeans.org/
[27] http://www.ganttproject.biz/

8.3 Architecture

The architecture is based on the client-server model (see Figure 28). Four Wonderland clients connect to the Wonderland server. The Wonderland server uses the jMoneyEngine to provide the scene graph for the 3D environment and jVoiceBridge to provide VOIP. SAS (Shared Application Server) is used by the server in order to run the external applications such as NetBeans, Poseidon for UML, and others. Finally, the module system is employed by the Wonderland server and accommodates the internal applications (i.e. Virtual Phone, Text Chat, Voice Chat) for the users.

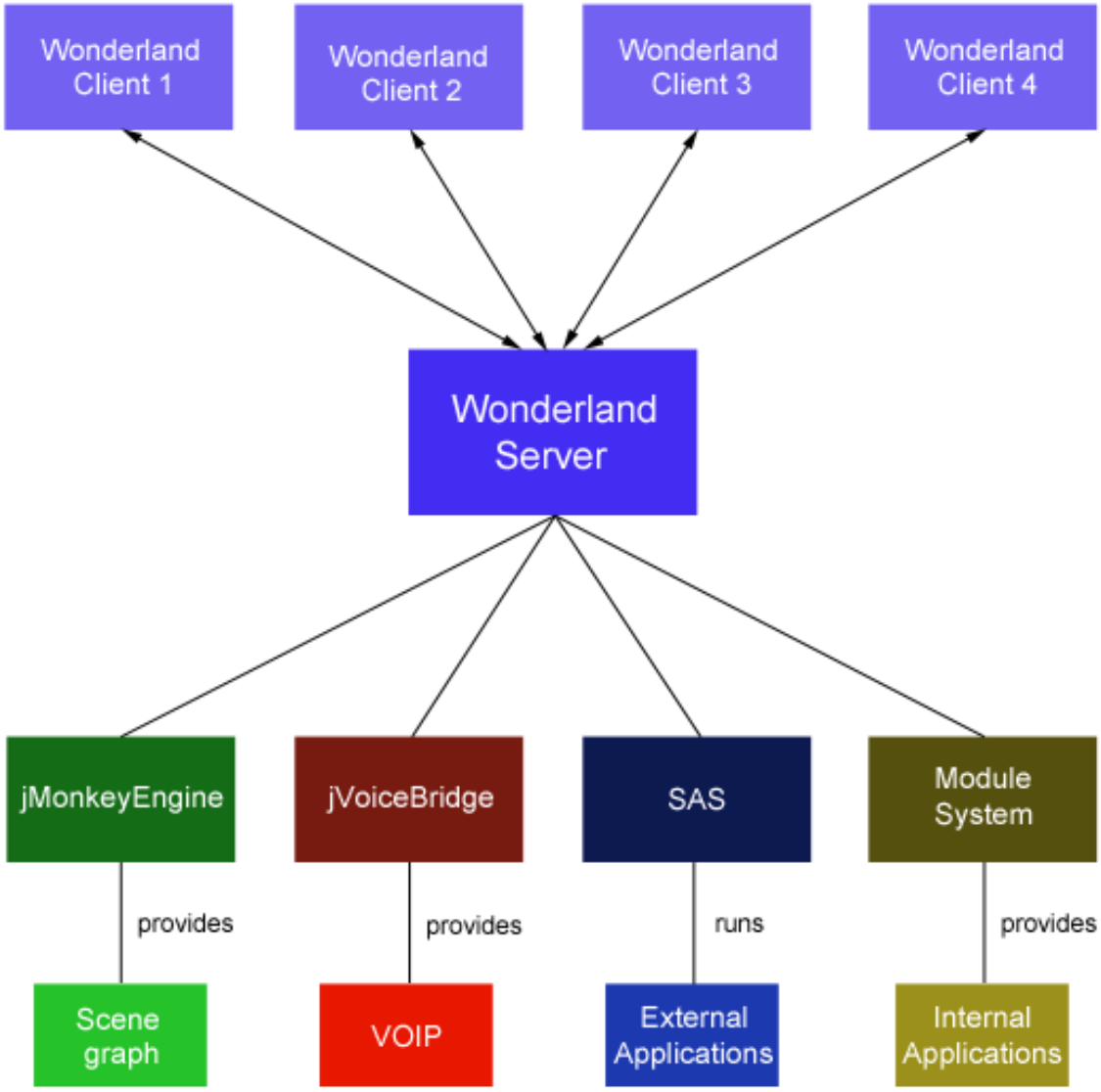

Figure 28: The software architecture of the virtual environment

8.4 Implementation and Integration

This section describes the process by which the virtual environment was implemented in Project Wonderland 0.5. It describes the way the Project Wonderland 0.5 was installed and run. Then it elaborates on the implementation of the 3D environment. Finally, the process of integration and deployment of shared applications is described.

8.4.1 Installing and Configuring Wonderland 0.5

Because the prototype requested shared applications to be used, Wonderland needed to be installed on a Linux or Solaris platform. The author of this thesis chose to use a virtualized Linux environment on a Windows 7 host because he had experience in working with such configuration. For the virtualization *VMWare Workstation v7*[28] was used to virtualize the Linux *Ubuntu 9.04* image. Upon the successful booting of the operating system, Sun's *JDK (Java Development Kit) 6* was installed on the system. After the successful installation of the Java environment, a binary of Project Wonderland 0.5 User Preview 2 was downloaded from Wonderland's homepage[29].

The server is started on the local machine by issuing "`java -Dwonderland.webserver.host=localhost -jar Wonderland.jar`" command in the Terminal window. The right IP address for the `localhost` argument was obtained by following next steps and was the only way to connect to the Wonderland server from the host system:

1. The file *etc/hosts* is opened and the IP address for localhost is commented out. The file is saved.
2. The Wonderland server is then started with the command "`java -jar Wonderland.jar`".
3. Upon the successful server start, the command log prints out the IP address on which the server is started. In this specific case, the IP address was 192.168.37.133.
4. The file *etc/hosts* is opened again and the localhost address is re-enabled.
5. The Wonderland server is then started with the command "`java -Dwonderland.webserver.host=192.168.37.133 -jar Wonderland.jar`" in the Terminal window.

[28] http://www.vmware.com/
[29] https://wonderland.dev.java.net/binary-builds.html

6. After the server undergoes its startup process, it issues the following message:

```
------------------------------------------------------------
Wonderland web server started successfully.
Log files are in /home/user/.wonderland-server/0.5-dev/log
Web server running on http://192.168.37.133:8080/
------------------------------------------------------------
```

7. The web server is accessed from a web browser in the Windows 7 host machine on http://192.168.37.133:8080/ (see Figure 29).

8. The Server Admin is started by clicking on the Server Admin button within the web server (see Figure 29).

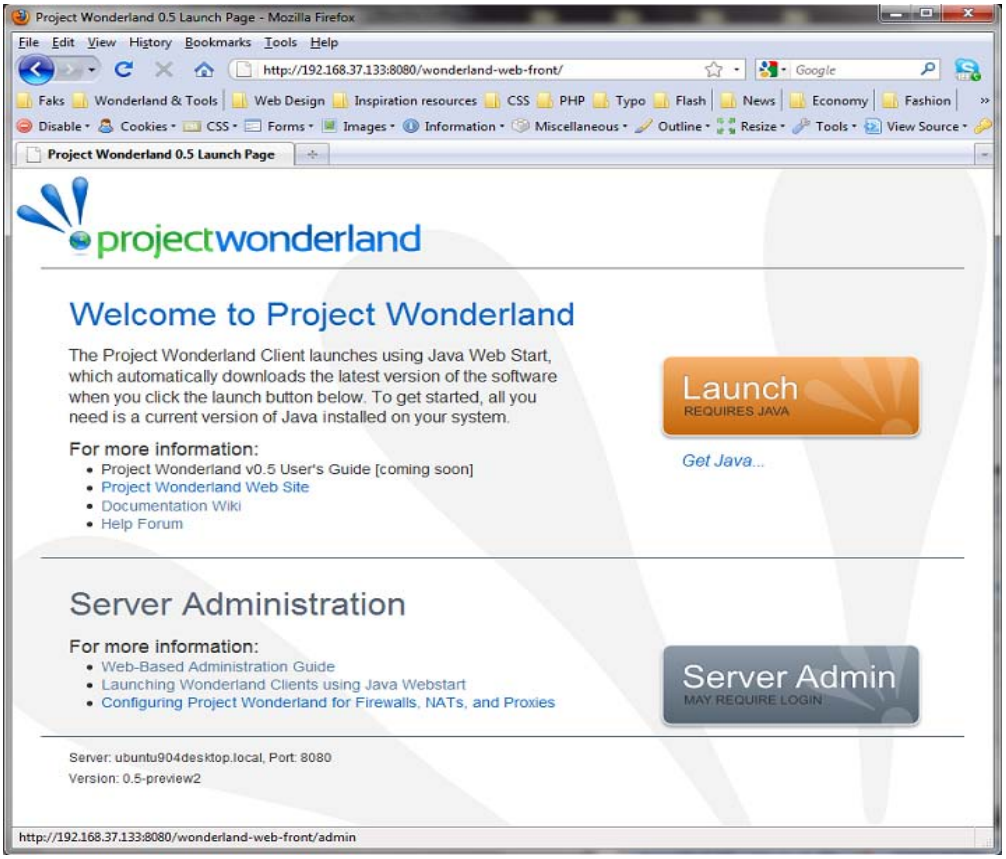

Figure 29: A screenshot of Wonderland's web server

9. By navigating to the "Manage Worlds" page and clicking on the "make current" action beside the "Empty World", Wonderland creates an empty world that is ready for the development (see Figure 30).

10. From the web server's home page, the Wonderland client is started by clicking on the "Launch" button (see Figure 29).

11. Upon the successful startup, a screen with an empty world containing only the sky and the avatar are displayed (see Figure 31). Wonderland is now ready for development.

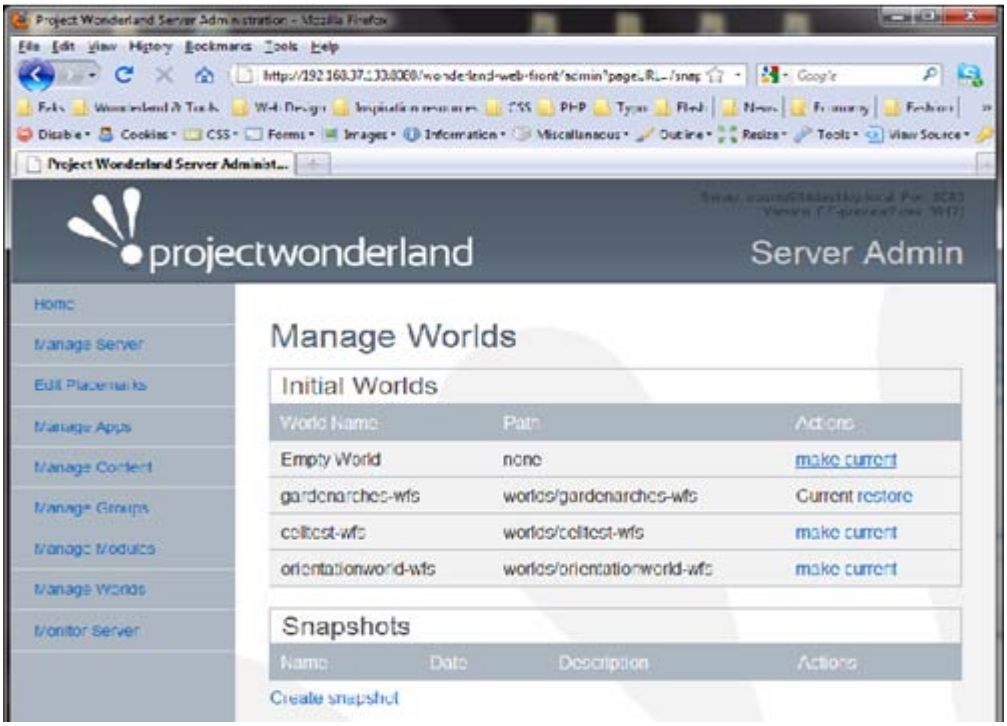

Figure 30: A screenshot of the “Manage Worlds” page

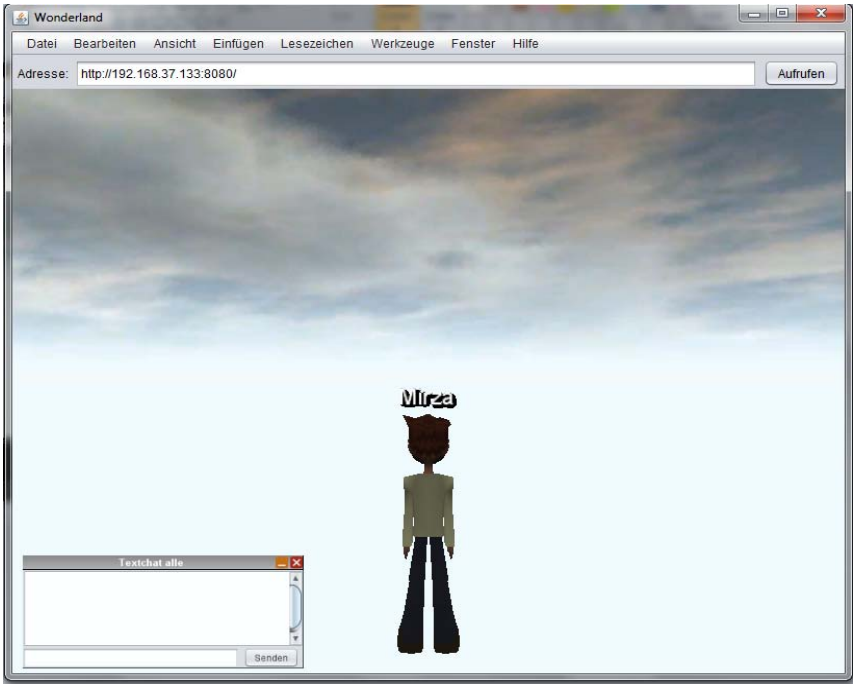

Figure 31: A screenshot of an empty Wonderland world

8.4.2 Creating the Virtual Environment

The first step in creating a virtual 3D world in Wonderland is creating the floor and the skyline for the world. For the purpose of importing 3D objects, Wonderland provides a drag-and-drop functionality. The 3D objects need to be in *Google Earth*[30] *(kmz)* format. The 3D models for the floor and the skyline for this prototype have been downloaded from the *Google 3D Warehouse*[31]. This prototype used the *GrassCircle-100m.kmz* floor model for the floor and *ChrisGnomeMtSkyline-250m.kmz* model for the skyline. Both models are created by Nicole Yankelovich[32]. After the import of the models into Wonderland and some positioning, the virtual world takes the form shown in Figure 32.

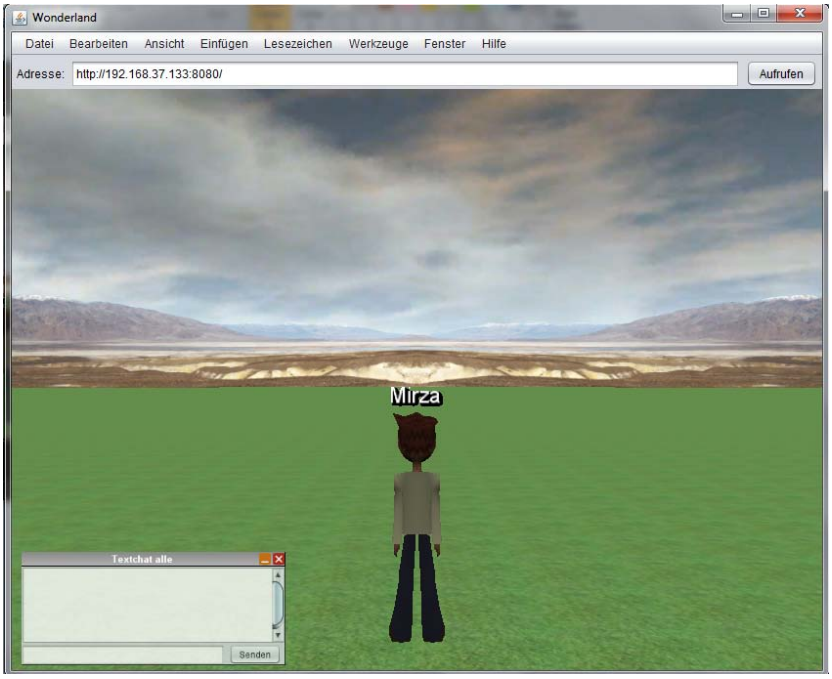

Figure 32: Virtual world with the floor and the skyline

The next step in creating the virtual environment is modeling of rooms presented in section 7.3. This task was performed in *Google SketchUp 7.0*[33]. In order to support a better awareness in team members, it has been decided that the room walls be transparent. In this way team members can see what their team colleagues are currently working on and where they are. Due to this and the fact that Wonderland has some issues concerning the light effects on imported models, the walls were modeled separately from other room elements (see Figure 33). Finished models were exported in .kmz format and imported in Wonderland via drag-and-drop functionality.

[30] http://earth.google.com/
[31] http://sketchup.google.com/3dwarehouse/
[32] http://sketchup.google.com/3dwarehouse/search?uq=10926082699640160839&scoring=m
[33] http://sketchup.google.com/

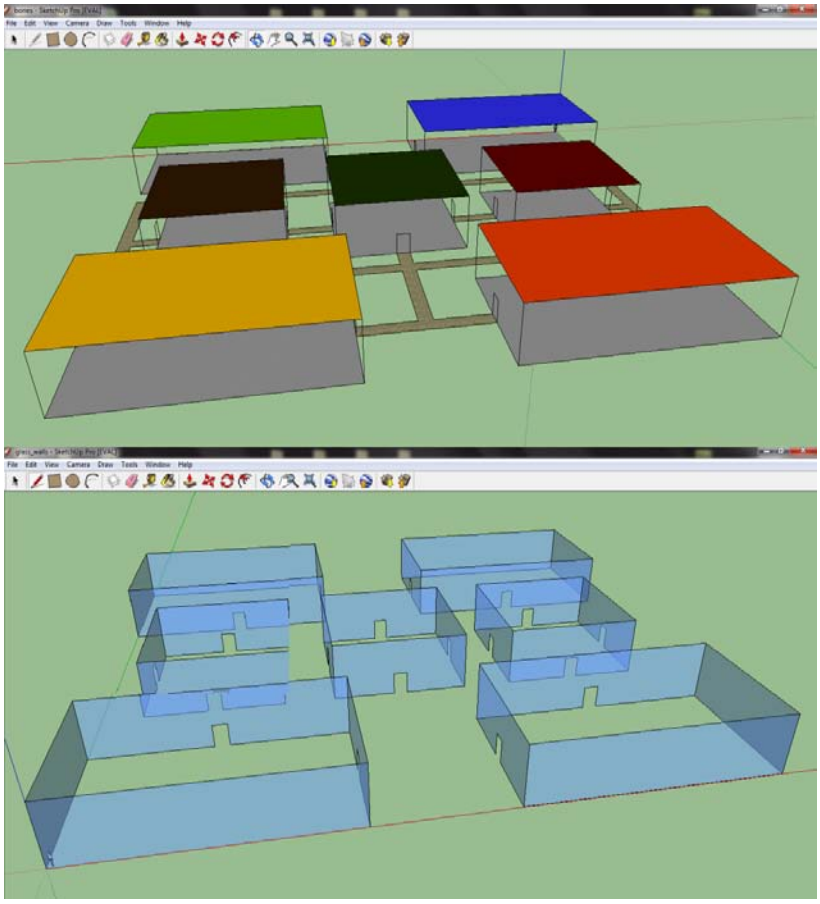

Figure 33: Two 3D models for rooms

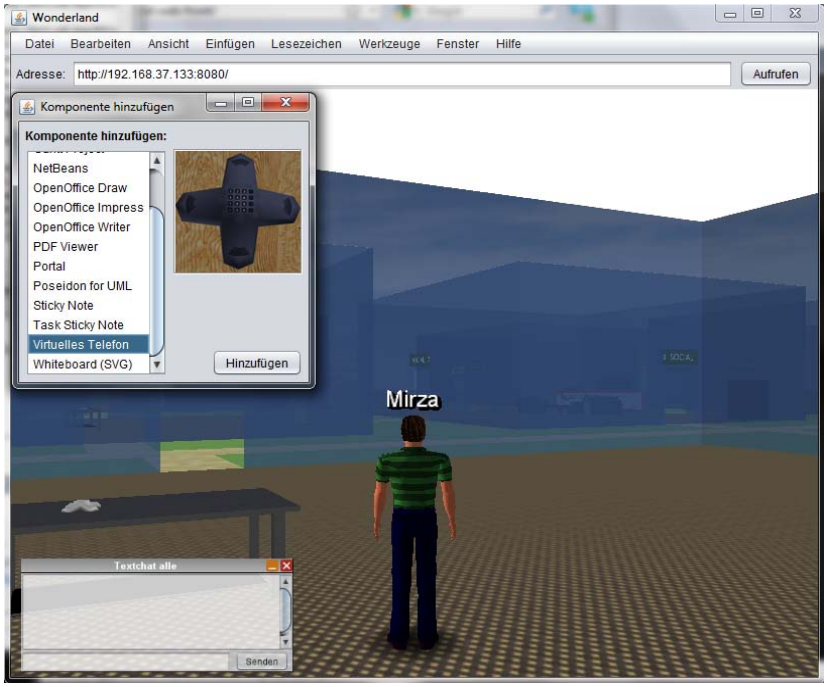

Figure 34: Inserting the “Virtual Phone“

In order to support easier navigation and localization of team members within the virtual environment, labels were created as .gif files and placed above corresponding doors on virtual rooms. Also, for the purpose of this prototype, the representations of the team structure, team member's expertise, responsibilities, contact information, and calendars are realized as graphic elements placed in corresponding rooms. Finally, furniture and similar objects were imported from Google 3D Warehouse and arranged across the rooms and every room was equipped with a *"Virtual Phone"*. It is inserted in-world, from the *"Insert Component"* menu (see Figure 34).

8.4.3 Importing Shared Applications

The primary prerequisite for the use of shared applications within a virtual world in Wonderland is that these are installed on the server machine, in this case on a virtualized Ubuntu 9.10 guest operating system. Once the applications have been installed, they have to be registered with Wonderland server.

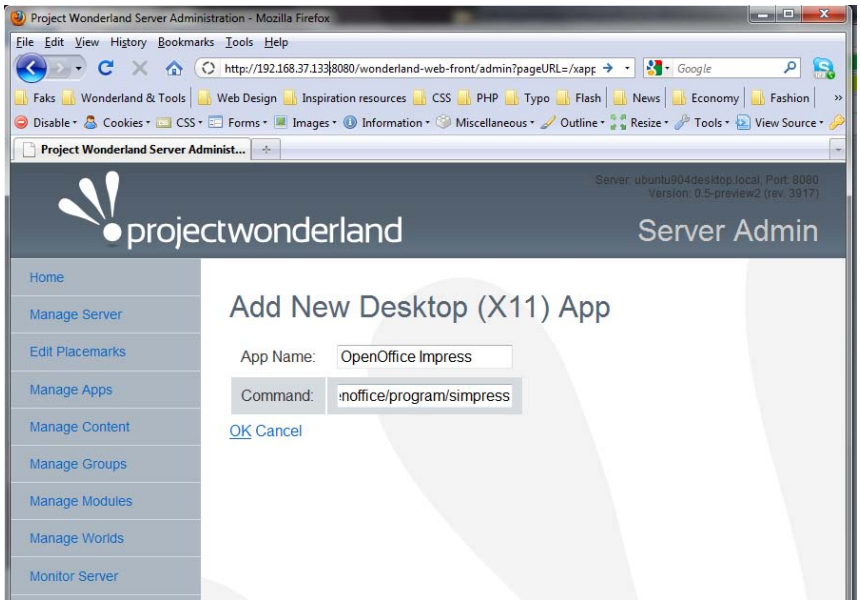

Figure 35: Adding new shared applications in Wonderland

The registration of applications is performed on the Web Server Admin page by navigating on the "Manage Apps" page. By clicking on the *"Add App"* button, the new page with a form for the addition of an application opens. In this form, one has to enter the values for parameters *"App Name"* and *"Command"* (see Figure 35). The *"App Name"* parameter represents the label that will be used for this application within Wonderland's *"Insert Component"* menu and the *"Command"* parameter requests the path to the application's executable. After the parameters have been filled in, the *"OK"* button is pressed in order to confirm the process of registration of the application. The

applications are then inserted in Wonderland from the *“Insert Component”* menu (*Insert->Component*).

8.5 Implementation Issues

Although the implementation of the virtual collaboration environment was pretty straight forward, a few issues have been encountered. These issues are related to Wonderland’s high requirements concerning computer resources as well as its buggy rendering of transparent surfaces.

Due to performance issues that emerge from running multiple shared applications, it has been decided not to run multiple instances of the same application in one room. Hence, the analysis room, for example, does not contain four instances of the word processor (see section 5), but only one.

Because the implementation was performed on a virtualized Linux platform and that the system disposed with a total of 2 GB of RAM, the development of the environment was confronted with continuous delays and freeze-outs because of the high memory demands. These issues hindered the import and manipulation of 3D models and had even more serious effects on the use of shared applications. Hence, the performance and usability of shared applications in this configuration does not fulfill the high demands for an environment where software teams execute their project work. However, better performance can be provided on a host Linux platform with better computer resources.

Another issue presented itself in the way Wonderland renders transparent surfaces. The walls of the single rooms, which were created as transparent in Google SketchUp, rendered solid when the avatar was positioned under some angles. Additionally, server administrators could deny the implementation of the virtual collaboration environment for security reasons. They could see issues in running shared applications on their server as well as providing user accounts for the server.

8.6 Summary

This Chapter has described the implementation process of the virtual collaboration environment for software engineering. Project Wonderland was chosen for the implementation due to its platform independency and application sharing mechanisms. For the purpose of this thesis, the most important notion of Wonderland is its web based server administration. The administration is performed through a web interface called Server Admin where the management of the server, modules, content, placemarks, security groups, and shared applications is provided.

The virtual environment takes advantage of different external applications that provide the functionality needed by software teams for the task completion purposes.

These applications include OpenOffice.org Writer, Impress, Draw, Mozilla Firefox, Poseidon for UML, NetBeans IDE, and GanttProject.

The architecture of the environment is based on the client-server model. Basically, four Wonderland Clients connect to the Wonderland Server which provides them with appropriate services and functionalities. The Wonderland Server itself uses the functionality of different software packages: jMonkeyEngine is used to provide the scene graph for the 3D world, jVoiceBridge provides the VOIP functionality, SAS enables the use of external applications, and finally internal module system provides internal applications such as voice and text chat.

In order to use shared applications the environment had to be implemented on a Linux or Solaris platform. For the purpose of the prototype in this thesis, a virtualized Linux Ubuntu 9.04 image was run by VMWare. After the installation of JDK 6 and the download of Wonderland 0.5 binary, the Wonderland server was run from the Ubuntu guest. Next, the Server Admin was started where the empty world was chosen. Then, from the Windows host, the Wonderland client was run.

As the next step, the environment was created. It was assembled from different 3D models. The rooms were created in Google SketchUp and entailed transparent walls for better awareness. Room labels were created with a graphics editor. Elements such as floor, skyline, and furniture were downloaded from Google's 3D Warehouse. The 3D models were then imported into Wonderland via Wonderland's drag-and-drop functionality and placed appropriately. Information such as team structure, shared calendars, contact information, and responsibilities were also placed in the same manner in corresponding rooms. Finally, within Wonderland, every room was equipped with a Virtual Phone.

The final step in the implementation process was the installation of shared applications with Wonderland. First, the applications were installed on the server machine (i.e. the Ubuntu guest). Second, the applications were registered on the Server Admin. This was done on the "Manage Apps" page where the label for Wonderland and the full path for the applications were entered. Having completed this process, the applications are inserted in the world from the "Insert Component" menu.

Now that the process of implementation of the virtual environment has been presented, the possibilities and usage scenarios within this virtual world need to be elaborated. This is the focus of the next Chapter. It will describe the communication possibilities of the environment, the possibilities for better team awareness, knowledge and project management functionality, and finally the tools needed in the software process.

Chapter 9

Application Point of View

After the virtual collaboration environment for software engineering is implemented, it is now time to see how distributed software teams can use it for their benefit. This Chapter presents different functionalities which team members can perform in the environment as well as verifies the requirements found in section 7.1.

The functionality of the virtual collaboration environment prototype created for the purpose of this thesis will be presented through a usage scenario. This scenario involves four team members that are distributed geographically. Mirza is located in Graz and is both the project manager and a developer in the project. Eva and Anton are working as developers in an office in Vienna. Finally, Carlos is a software developer who works and lives in Barcelona, Spain. The functionality of the environment is presented following the groups of requirements presented in Chapter 7.

In order to start using the virtual environment, the team members first start the Wonderland Client from the "Launch Page" (see Figure 29). When the login page appears, each member enters the user ID and password provided by the system administrator. Shortly afterwards, the team members see their avatar in the world.

9.1 Communication

In the virtual environment team members can use multiple communication channels. They can engage in audio conversations thanks to Wonderland's realtime immersive stereo audio and also text chat with one another via the text chat functionality (see Figure 36). Audio conversations can be conducted in two ways. Besides the standard conversation in which team members speak and hear other developers with respect to the distance between them, team members can also initiate a voice chat session with their colleagues. This conversation can be private but can also be made public for other team members to hear (see Figure 37). These communication channels help team members communicate, discuss urgent issues, and perform project related.

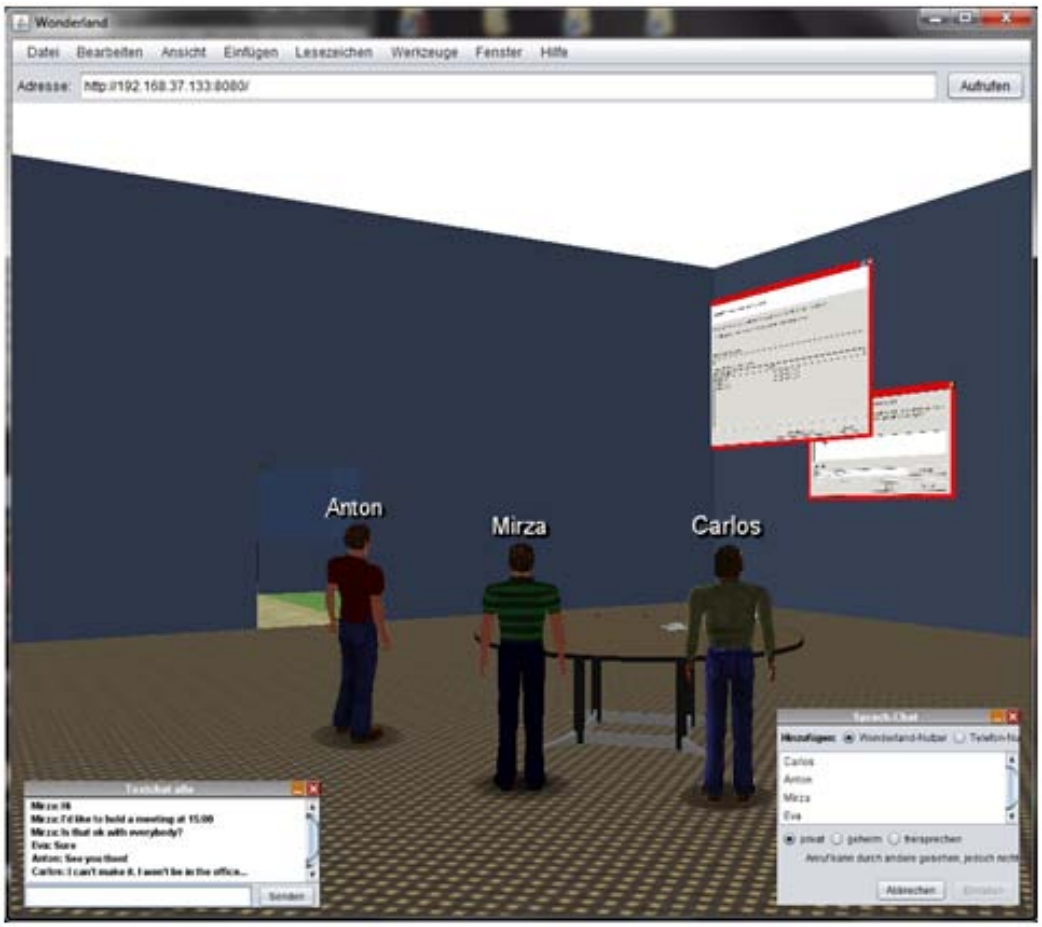

Figure 36: A screenshot from the virtual environment with the Text Chat in the left corner and the Voice Chat in the right corner

Figure 37: Voice Chat

In cases when team members want to communicate with Carlos, and he is not present in world, team members can take the advantage of the Virtual Phone. In order to

initiate the conversation, team members can click on the Virtual Phone object on the table and call his mobile phone number (see Figure 38). Team members are then able to communicate synchronously with Carlos even though he is not present in world.

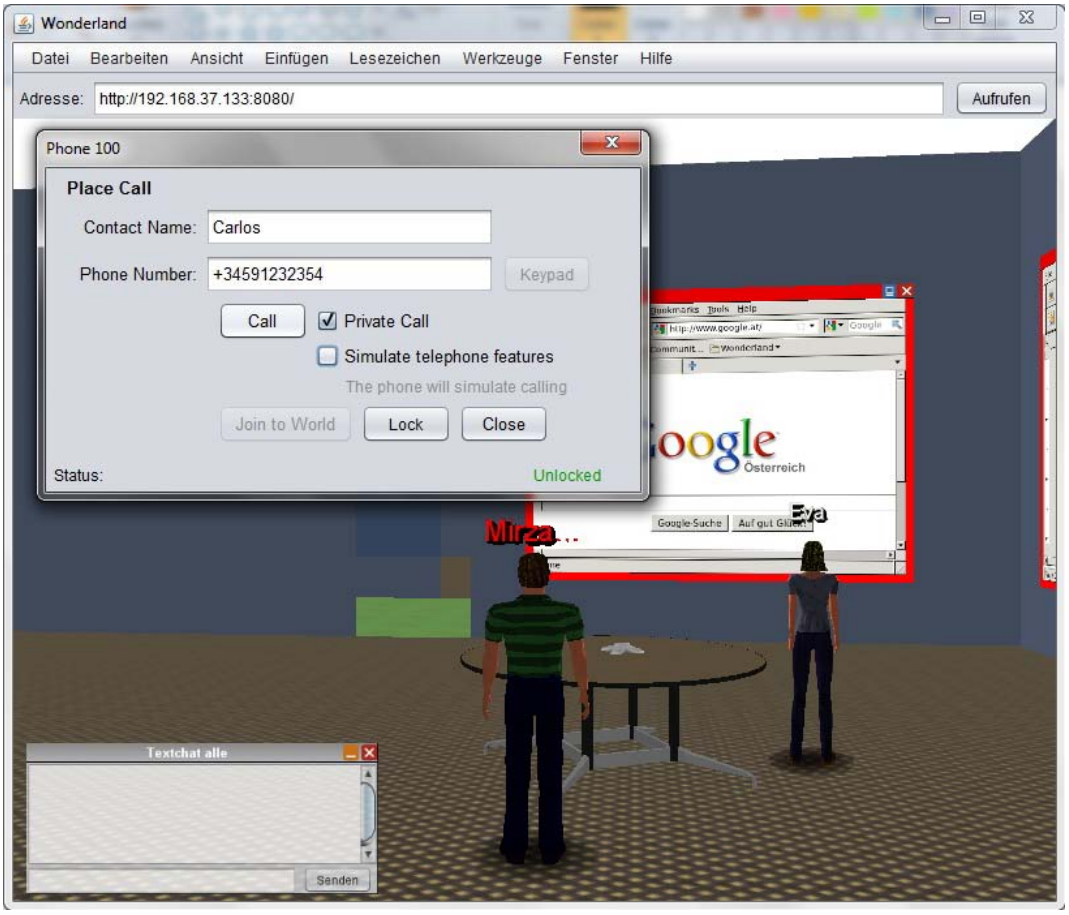

Figure 38: Starting a phone call via Virtual Phone

In cases when a virtual team member is not present in world and his colleagues need to communicate with him, asynchronous communication needs to be applied. For example, the virtual team is in the implementation phase of the software project and Mirza does not understand what the purpose of Carlos' code is. Mirza can then leave a message or a question for Carlos in the Implementation Room by using Wonderland's *Sticky Notes* object (see Figure 39). When Carlos returns to the virtual environment, he can see the question and answer it.

The virtual collaboration environment provides a space where team members can socialize and engage in informal communication. This is realized through the Social Room which resembles an environment similar to the ones in real world where software developers converse near water coolers and coffee machines (see Figure 40). Also, the environment supports the exchange of nonverbal cues between virtual software developers. Voice tone is easily picked up through audio conversations and body language can be initiated by using Wonderland's *Gestures* functionality. By clicking on

one of potential gestures, the avatar displays certain body language actions (see Figure 40).

Figure 39: Leaving a question for Carlos in the Implementation Room

Figure 40: Social environment and the performance of Gestures

9.2 Team Awareness

When it comes to team awareness, virtual team members can benefit from multiple capabilities. By observing the avatars of team colleagues, virtual software developers can see what they are currently working on and where they are located. This is additionally supported through the use of transparent room walls. In must be noted, however, that the transparency in Wonderland has some issues under certain angles and does not render properly (what can be seen in Figures in this section).

Additionally, the Organization Room greatly fosters team awareness. If, for example, Anton's avatar was not present in the environment, other team members can easily consult his shared calendar and find out the reason for his absence as well as when he will be back in world (see Figure 41). As seen in Figure 41, the Organization Room displays also images containing the team structure, expertise and responsibilities of virtual software developers in a particular software development project. In terms of this thesis the team structure information need not be especially helpful due to the small size of the development team, but could be very helpful in projects with larger teams where the division of roles is more complex. As for expertise and responsibilities, the information can be helpful in many ways. For example, if a certain task is reaching its deadline and is no progress is being made toward its completion, the project manager can see which team member is responsible for its fulfillment and take appropriate action. Also, if a virtual software developer needs information about a specific technology, she can take a look at the expertise information and find the colleague who she needs to consult.

Figure 41: A shared calendar, team structure, and expertise and responsibilities image in Organization Room

Contact information is also displayed in the Organization Room (see Figure 42). In cases when team members have to be contacted by other communication means, this information is particularly useful. An example can be seen in Figure 38 where virtual team members need Carlos' phone number in order to communicate with him during the team meeting.

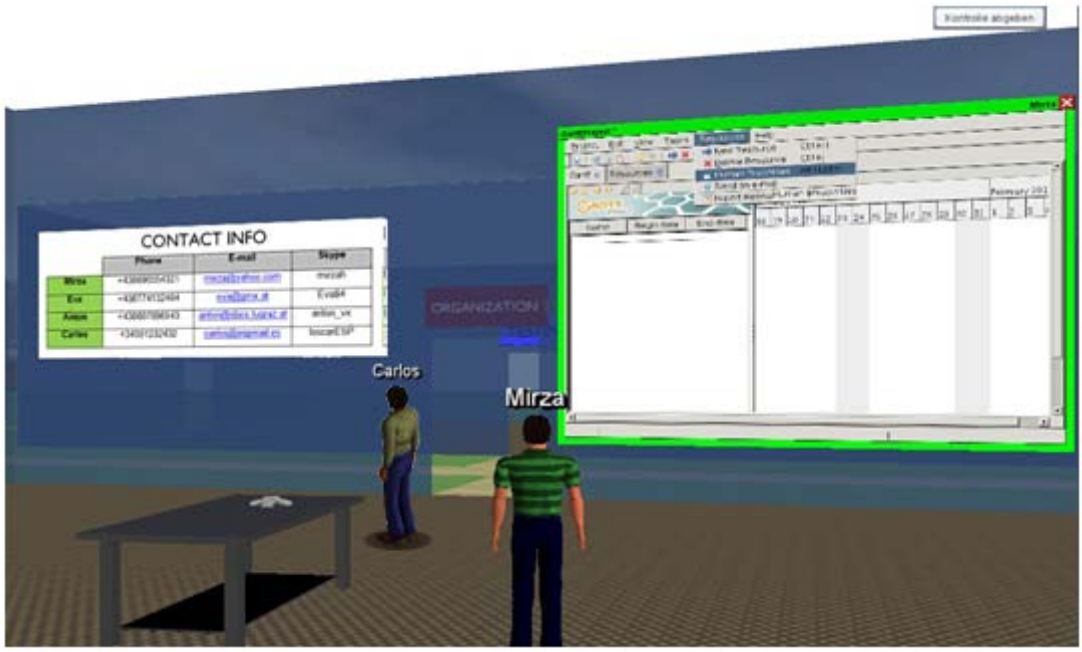

Figure 42: Contact information and the GanttProject application in the Organization Room

9.3 Project and Knowledge Management

Project management represents a top requirement for the virtual collaboration environment. Hence, the Organization Room provides the virtual software team with a project management tool (see Figure 42). The project manager can use this application to run and control the execution of the project.

When it comes to data sharing, the virtual environment provides two possibilities. First, the users can import the data into the world where it can be accessed by other team members. Second, due to the fact that the most project relevant data is created through shared applications, this data is stored centrally on the server machine and can be accessed from the environment at any time.

Team meetings are essentially important in software engineering projects. This environment supports them in the Meeting Room. In this room, virtual software teams can hold meetings, hold presentations, and manage meeting relevant data such as the meeting agenda. As seen in Figure 43, Mirza is creating an agenda for an upcoming meeting by using the shared Writer application. Additionally, team members can

organize team meetings by using multiple communication channels. An example can be seen in Figure 44 where team members are negotiating a specific meeting time.

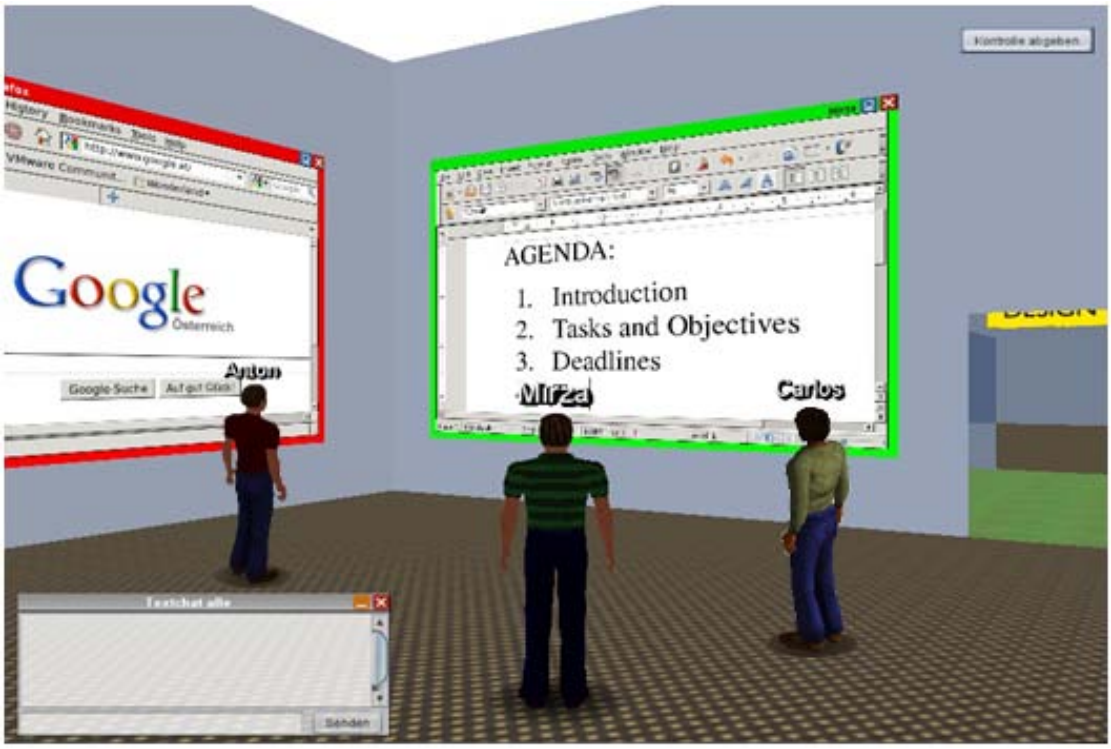

Figure 43: Creating a meeting agenda in a shared Writer application

Figure 44: Organizing a team meeting via Text Chat

As argued throughout Chapter 8, shared applications represent the most important functionality in this prototype. In order to use a launched shared application, a developer has to hold the “Shift” key and click on the shared application’s window. The red frame around the application turns green and is ready for the input. The shared application can in the same way be used simultaneously by multiple team members. As shown in Figure 43, Mirza’s avatar is in control of the OpenOffice.org Writer. There is, however, another way to use shared application which will be presented in the next section.

9.4 Software Process

The software process is supported by the shared applications found in the Analysis, Design, Implementation, and Testing Room. Besides the standard way of using shared Applications in Wonderland, these can also be used in front view. This is done by right-clicking on application window and selecting “Front View” from the menu. As seen in Figure 45, the application displays itself similarly to the way it displays when launched on a normal desktop. The same effect can be achieved by clicking on the blue screen icon in the top right corner of the application window (see Figure 45). In order to leave the application the developer needs only to press the “Release control” button in the top right corner of the Wonderland window (see Figure 43 and Figure 45).

By following these steps, virtual software developers can use all applications presented in 7.3 in order to perform the tasks required in the project. By using these applications, every created document is stored on the server and can be accessed from Wonderland when needed. Another advantage for the collaboration is the fact that all team members can use shared application and conduct work simultaneously.

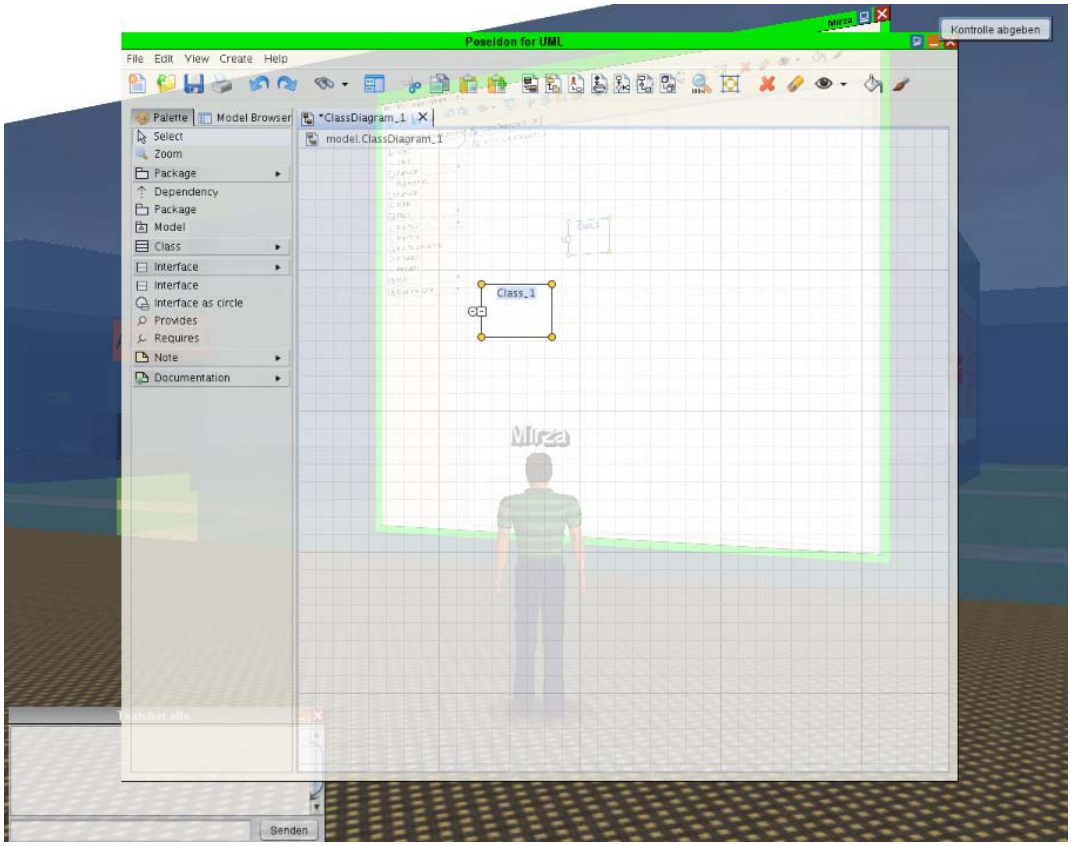

Figure 45: Using a Poseidon for UML application in Front View

9.5 Usability

As presented in section 7.1, it is very important that the users be able to navigate easity through the environment. In this prototype, navigation is supported by two means. As introduced in previous section, the rooms in this prototype contain labels that are found above every door of the room (see Figure 46). This should support better navigation of virtual teams in the virtual environment. An additional functionality in Wonderland that helps teams when it comes to navigation is *Placemarks.* As shown in Figure 46, Mirza's avatar is located in the Analysis Room. If he has to attend a meeting, he is able to immediately teleport himself to the Meeting Room by selecting a predefined placemark.

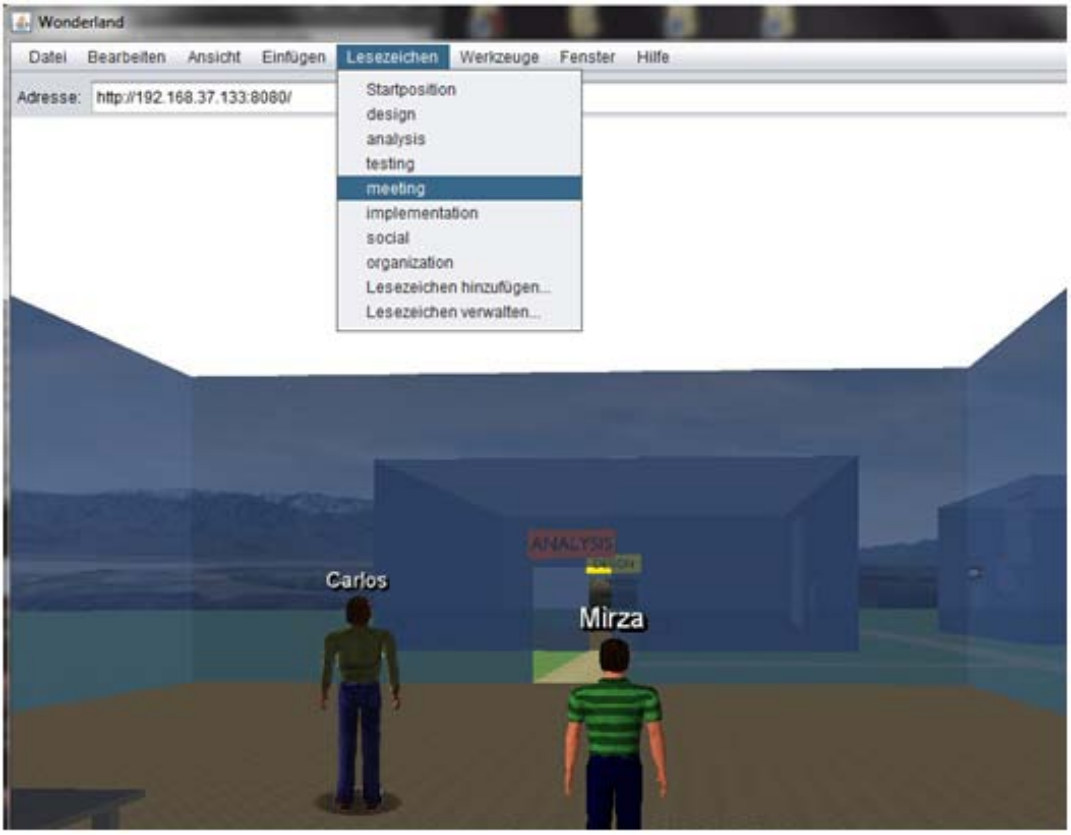

Figure 46: Appearing at a desired spot in the world by using Placemarks

9.6 User Study

The prototype described above was developed in order to support distributed software development and project management by providing richer communication, better team awareness, providing mechanisms for meeting, knowledge and project management, and supporting the software development process. In order to find out if the environment supports the above goals, a user study was conducted.

The overall goal of the study is to address the following research questions:

R1: *Does the prototype support the software development process?*

R2: *Does the prototype support the project management process of as meta-process of the software development process?*

The subjects of the study are both female and male software engineering students at TU Graz aging from 18 to 30. The survey was conducted at the authors place on the development environment. Hence, the study was performed on a notebook running Windows 7 and virtualized Linux Ubuntu 9.04. Five students enrolled in the study. As for the content of the study, it was the same for all participants. Every participant first answered a pre-questionnaire. After finishing the pre-questionnaire, the participants were shown the virtual collaboration environment. The functionality of the environment was fully presented to the participants. Afterwards, they also used the environment themselves. Finally, having concluded the experiment, the subjects of the study completed a post-questionnaire which asked for the impressions about the virtual collaboration environment.

9.6.1 Pre-survey

The pre-survey assesses the students' background using computer tools for communication, organization and coordination of work with their peers, using mobile devices, using 3D multi-user environments, their experience and knowledge about software development and project management in a distributed setting. The question in the pre-questionnaire are organized in three categories: (1) demographic data, (2) software development process and (3) software project management process as meta-process of the software development process. Of the 5 participants, there was one female and four male students.

The demographic questions of the pre-study show that participants frequently used different tools and applications to communicate, organize and coordinate work with their colleagues in virtual settings. Four out of five participants use Skype and MSN for communication with colleagues. As for the organization and coordination of work, four out of five students use e-mail and three out of five use MSN. Also, the pre-questionnaire reveals that participants rarely or never use 3D virtual environments or have no understanding about them. One participant used Second Life occasionally, whereas one subject of the study misunderstood what 3D virtual environments are. When asked about mobile phone use, all students admitted using mobile phones every day.

Additional parts of the pre-survey were concerned with questions about software development and project management in both traditional and distributed setting. Whereas the participants had little or no experience in project management and provided no useful input in this context, they all had some level of experience when it comes to software development. Two participants had one year of experience, and

others had two, three and four years of experience. However, they needed further explanations about the meaning and purpose of distributed software development. The participants viewed communication, organization, coordination, and inability to get help easily as the main problems in this domain. One of the students was not confident about the ability of 3D virtual environments in supporting distributed software development and project management. Additionally, all of the participants thought that there is a difference in education and training software development and project management using immersive collaborative virtual environments.

9.6.2 Post-survey

The post-questionnaire evaluates the prototype. The participants are asked for their impressions about the virtual collaboration environment in general and in context of application in distributed software development, distributed project management and their training and education. In addition, the students were asked about things they did and did not like, as well as suggestions for the improvement of the prototype in all contexts.

When asked about their impression of the applicability of the virtual environment for distributed software development and its training and education, the participants graded the environment with a mean of 4.6 on the scale from 1 to 5 where 1 was represented the worst and 5 the best impression. The participants then identified the things they did and did not like about the environment. The results confirmed most assumptions about the environment. The preferred things differed from one participant to another. Three students liked the presence of different facilities for communications, whereas one liked the possibility to work with other people live, and one student liked separate rooms for different software development phases. On the negative front, all participants disliked the slow performance and essential need for its enhancement. In addition, one of the participants disliked Wonderland's graphics and one did not like the fact that he was not able to enlarge windows of shared applications. In the context of training and education of software development, one participant liked that *"also a large group can keep in touch and help each other"*. Another participant liked the fact that the users are able to collaborate on the same document.

The environment was graded slightly worse in context of the application for distributed project management with a grade of 4.2 on the same scale. Although most participants liked the same things in context of software development and project management, two students liked the ability to track the progress of the project live and another student liked the presence of the PM tool. The environment received a grade of 4.4 in the domain of training and education for project management. Finally, the participants graded the virtual collaboration environment in general with a grade of 4.4 what is very encouraging. Concluding the user study, the prototype supports both research questions with the precondition of high performance of the prototype.

9.7 Summary

In the virtual environment, the virtual team members can communicate via multiple communication channels. These include a text chat, voice chat, and audio communication with respect to distance. They can also call absent team members by using a Virtual Phone, as well as communicate asynchronously over Sticky Notes. By using these different communication means, virtual teams can cooperate on different project related tasks. The Social Room imitates an environment from the physical world, where software developers can communicate informally near artifacts such as a coffee machine or a water cooler. In addition, Wonderland's Gestures provide the capability to display and exchange nonverbal cues in communication between virtual team members.

In order to support and benefit to the team awareness, the environment possesses the Organization Room which displays important information such as calendars of every software developer, team structure, expertise, responsibilities, and contact information of every virtual team member. Moreover, the virtual software developers can observe the avatars of their colleagues and know if they are present and what they are currently doing.

Project management is supported by the project management application in the Organization Room. Team members can share data by importing it into the environment. Also, Through the use of shared applications, all documents and data are stored centrally on the server machine. In this way, Wonderland world acts as an interface for accessing different project related data. The Meeting Room provides a place where virtual teams can meet, hold presentations and manage meeting relevant documents. The meetings can be organized via different communication channels such as voice or text chat.

The environment has four rooms that hold shared applications that support the software process. The shared applications that are present in different rooms can be used by multiple team members simultaneously and also in a front view, similar to the view a user has on a desktop. Virtual software developers can navigate through the virtual environment by seeing the labels placed above the doors of different rooms. In addition, they can also teleport their avatars to a desired room by using Wonderland's Placemarks functionality.

The user study shows confidence about the application of the prototype for distributed software development and project management. The users were satisfied with the ability to work together in the same place, the presence of different communication ways, the ability to track the project progress as well as the notion of rooms in the environment. The study also shows that the performance of the virtual environment needs to be better in order to support its usability and acceptance.

Chapter 10

Lessons Learned

The author improved his knowledge in working with virtualized operating systems and working in Linux. He gained additional experience in finding and eliciting software requirements as well as creating software architectures. Additionally, 3D modeling was also a new thing with which the author had to acquaint himself. Finally, the architecture and functionality of Project Wonderland were learned. The knowledge was also extended by the use of different functionality modules, communication channels, and shared applications that played a major role in this prototype.

The author also learned a lot about global software development as the most complex form of distributed software development. Companies take part in GSD because of the promising benefits of cost reduction, larger workforce, shorter times to market, advances in infrastructure, vicinity to market, faster buildup, and national policies. However, there are significant challenges that stay in the way of these benefits. Difficult division of work, cultural and technical issues, inadequate communication, knowledge, project and process management issues all represent possible stumbling blocks. As it is closely related to the software engineering, distributed project management is also a very complex discipline with challenges concerning the management of distributed teams, communication, organization and the choice of collaborative tools.

Virtual teams are faced with challenges that result from the distance imposed between them. Because of distance, virtual teams have problems with relationship building, cohesion, trust, communication, coordination and other categories of the life cycle model. When it comes to the collaboration, virtual teams need to perform software engineering tasks and virtual team meetings. A wide range of communication media such as audio and video conferences, e-mail and chat have been used to support virtual teams with less success. When compared to face to face communication, this communication media has lower presence of nonverbal and paraverbal cues and lower

degree of synchronization. However, virtual worlds exhibit better characteristics for virtual collaboration because of their richer communication channels. The greatest advantage of virtual worlds is that they provide a shared common space for virtual teams and the use of avatars as graphical representations of single virtual software engineers.

The requirements for a prototype of the virtual collaboration environment for software engineering include requirements regarding the richer communication between team members, better team awareness, support for meetings, knowledge and project management, and the ability to perform software process related tasks. However, in order to effectively be able to use the environment for such purposes, it has to have great performance, be highly reliable, and the users have to be able to use it with ease. In addition, the room metaphor is extremely helpful when it comes to organization and structuring of the work.

Project Wonderland is written in Java and provides a web interface for the server administration. The worlds can be assembled in Wonderland very easily through the drag-and-drop functionality. Besides a large number of finished 3D models, new models can be created in Google SketchUp. Wonderland provides many prebuilt modules that can be inserted from the Insert Components menu together with registered shared applications. However, Wonderland has performance issues when running a large number of shared applications and the development in virtualized environment is confronted with serious performance issues.

In the virtual collaboration environment, virtual software developers can communicate through multiple communication channels, socialize and exchange nonverbal and paraverbal cues. Team awareness can be supported by observing avatars and their activities and providing information about expertise, responsibilities, team structure, contacts and daily obligations. Shared applications can provide the functionality needed for software engineering and project management. They can be used in normal mode and in front view. Moreover, virtual team members can collaborate on the same document simultaneously. The navigation of users can be supported through adequate labeling of the rooms and functionality that enables them to teleport to desired locations.

The prototype developed in this thesis can be used to support the software development process and project management process in distributed settings. However, high performance must be ensured in order to foster the usability and acceptance of such environments. In addition, possible conflicts with server administrators regarding the security need to be resolved.

Chapter 11

Conclusion and Outlook

This master thesis aimed to create an environment in a 3D virtual world that would support distributed teams in software engineering projects. The need for a better communication, coordination, and control medium exists in both software engineering as well as software project management. The main goal was to counter different challenges that were elaborated in the theoretical part of this thesis which result from the geographical distance between team members. While most of the issues are non-task related, such as insufficient cohesion, trust, and team awareness, task related issues such as communication and software process activities played a major role in an attempt to create an environment that would help distributed software engineers overcome these barriers. Furthermore, in the theoretical part it was shown how 3D virtual worlds excel over other media used in recent years for the purpose of virtual collaboration regardless of the context in which it was used.

The identified issues and challenges represented the base for the elicitation of requirements for the virtual collaboration environment. These were bundled in four groups: communication, team awareness, project and knowledge management, and software process. These groups were then the base for rooms in the design which followed the room metaphor. Hence, the Organization Room is responsible for team awareness and project and knowledge management, the Social Room serves a partial purpose for the communication (informal communication), the Meeting Room provides a place where team members can meet, and lastly, there is a single room for every main phase of the software process: Analysis, Design, Implementation, and Testing Room.

The virtual environment was implemented in Sun's Project Wonderland which is implemented entirely in Java. It uses a number of open source application that provide the virtual environment with the required functionality. The environment is created from 3D graphical models. Rooms in the environment were created in Google SketchUp, while other graphical elements were either created in graphical editors or downloaded from Google 3D Warehouse. Possible application scenarios have been presented and have shown how distributed software engineering teams can use the virtual environment

to overcome distance issues. Finally, a user study was conducted that showed that the developed prototype supports the software development and project management processes.

11.1 Future Work

The prototype developed in this thesis can be enhanced in several ways. For once, it was developed according to the specific set of chosen requirements. Hence, new features that would support scenarios where some users are performing their work outside the virtual environment could be developed. This enhancement would provide the environment with version control, integration of other CMC technologies, and VNC. In addition, all shared applications could be developed as internal modules. For example, contact information, responsibilities and expertise could be provided through an interactive module where team members could manage relevant information in the world.

In addition, advances towards better performance have to be made. Only through this would such kind of environment find wider use among virtual software development teams. Finally, any issues and resistance from server administrators need to be mitigated for the successful launch and beneficial use of these kinds of environments.

Bibliography

Beise, C., Evaristo, R., & Niederman, F. (2003). Virtual meetings and tasks: from GSS to DGSS to project management. *Proceedings of the 36th Annual Hawaii International Conference on System Sciences* (str. 9-18). Washington, DC, USA: IEEE Computer Society.

Binder, J. (2007). *Global project management: communication, collaboration and management across borders.* Aldershot, Hampshire, England: Gower Publishing Ltd.

Biuk-Aghai, R. (2001). Visualizing Structural and Behavioural Aspects of Virtual Collaboration. *Tenth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises.* Massachusetts, USA: IEEE.

Brander, E., & Mark, G. (2002). Why Distance Matters: Effects on Cooperation, Persuasion and Deception. *CSCW'02 , November 16–20*, 226-235.

Brown, B., & Bell, M. (2004). CSCW at play: 'There' as a collaborative virtual environment. *Proceedings of the 2004 ACM conference on Computer supported cooperative work* (str. 350-359). Chicago, Illinois, USA: ACM.

Carmel, E. (1999). *Global Software Teams: Collaborating across Borders and Time Zones.* Upper Saddle River, NJ: Prentice Hall.

Carmel, E., & Agarwal, R. (2001). Tactical Approaches for Alleviating Distance in Global Software Development. *IEEE Software , March/April 2001*, 22-29.

Carstensen, P., & Schmidt, K. (1999). Computer Supported Cooperative Work: New Challenges to Systems Design. U I. Kenji, *Handbook of Human Factors.* Tokyo.

Casey, V., & Richardson, I. (2006b). Project Management within Virtual Software Teams. *Proceedings of the IEEE international conference on Global Software Engineering.* Washington, DC, USA: IEEE Computer Society.

Casey, V., & Richardson, I. (2006a). Uncovering the reality within virtual software teams. *Proceedings of the 2006 international workshop on Global software development for the practitioner: International Conference on Software Engineering.* New York, NY, USA: ACM.

Castronova, E. (2001). Virtual Worlds: A First-Hand Account of Market and Society an the Cyberian Forntier. *CESifo Working Paper* .

Cho, J. (2007). Globalization and global software development. *Issues in information systems , 8* (2), 287-290.

Churchill, E., & Snowdon, D. (1998). Collaborative virtual environments: An introductory review of issues and systems. *Virtual Reality , 3* (1), 3-15.

Cohen, S. G., & Bailey, D. E. (1997). What Makes Teams Work: Group Effectiveness Research from the Shop Floor to the Executive Suite. *Journal of Management , 23* (3), 239-290.

Collaboration, Oxford English Dictionary. (1989). U J. A. Simpson, & E. S. Weiner, *Oxford English Dictionary, Second Edition.* Oxford: Oxford University Press.

CrunchBase. (2009). *Second Life*. Preuzeto 15. October 2009 iz CrunchBase: http://www.crunchbase.com/product/second-life

DIN. (2009). *DIN 69904:2000-11 „Projektwirtschaft; Projektmanagementsysteme; Elemente und Strukturen“.* Deutsche Institut für Normung.

Franceschi, K., Lee, R., & Hinds, D. (2008). Engaging E-Learning in Virtual Worlds: Supporting Group Collaboration. *Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008).* Waikoloa, Big Island, HI, USA: IEEE Computer Society.

Friedman, L. T. (2005). *The world is flat: A brief history of twenty-first century.* New York, NY, USA: Farrar, Straus and Giroux.

Fryer, K., & Gothe, M. (2008). Global software development and delivery: Trends and challenges. *Retrieved June 7, 2009, from http://www.ibm.com/developerworks/rational/library/edge/08/jan08/fryer_gothe/index.html* .

Gamasutra. (2009). Preuzeto 13. October 2009 iz Gamasutra: http://www.gamasutra.com/images/200901_habitat/screenshot.png

GanttProject. (2010). *GanttProject Home*. Preuzeto 12. January 2010 iz http://www.ganttproject.biz/

Garrett, G. A. (2004). Global Project Management: Best Practices. *Contact Management , April 2004*, 8-17.

Gartner. (2009). Preuzeto 14. October 2009 iz Gartner Newsroom: http://www.gartner.com/it/page.jsp?id=503861

Gentleware. (2010). Preuzeto 13. January 2010 iz http://www.gentleware.com/products.html

GlobalizationWiki. (2009). *Globalization*. Preuzeto 13. July 2009 iz Wikipedia: http://en.wikipedia.org/wiki/Globalization

Gütl, C., & Chang, V. (2008). Ecosystem-based Theoretical Models for Learning in Environments of the 21st Century. *iJET International Journal of Emerging Technologies in Learning* , 1-11.

Gütl, C., Chang, V., Kopeinik, S., & Williams, R. (2009). 3D Virtual Worlds as a Tool for Collaborative Learning Settings in Geographically Dispersed Environments. *Conference ICL.* Villach, Austria.

Hayes, B. (2004). Project Management Marries Collaboration – A New Technology for Distributed Project Teams. *PMI Global Congress Proceedings.* Prague: PMI.

Hendaoui, A., Limayem, M., & Thompson, C. (2008). 3D Social Virtual Worlds: Research Issues and Challenges. *IEEE Internet Computing , 12* (1), 88-92.

Herbsleb, J. D., & Mockus, A. (2003). An empirical study of speed and communication in globally distributed software development. *IEEE Transactions on Software Engineering , 29* (6), 481-494.

Herbsleb, J. D., & Moitra, D. (2001). Global software development. *IEEE Software , 18* (2), 16-20.

Herbsleb, J. D., Carleton, A., Rozum, J., Siegel, J., & Zubrow, D. (1994). *Benefits of CMM-Based Software Process Improvement: Initial Results.* Pittsburgh, PA, USA: Carnegie-Mellon Software Engineering Institute.

Herbsleb, J. D., Mockus, A., Finholt, T. A., & Grinter, R. E. (2001). An Empirical Study of Global Software Development: Distance and Speed. *Proceedings of the 23rd International Conference on Software Engineering: International Conference on Software Engineering.* Toronto, Ontario, Canada: IEEE Computer Society.

Herbsleb, J. D., Mockus, A., Finholt, T. A., & Grinter, R. E. (2000). Distance, Dependencies, and Delay in a Global Collaboration. *Proceedings of the 2000 ACM conference on Computer supported cooperative work: Computer Supported Cooperative Work.* New York, NY, USA: ACM.

Hofstede, G., & Hofstede, J. G. (1997). *Cultures and Organizations: Software of the Mind—Intercultural Cooperation and Its Importance for Survival, revised ed.* New York, NY, USA: McGraw-Hill.

Holmstrom, H., Conchuir, E. O., Agerfalk, P. J., & Fitzgerald, B. (2006). Global Software Development Challenges: A Case Study on Temporal, Geographical and Socio-

Cultural Distance. *ICGSE '06. International Conference on Global Software Engineering*, (str. 3-11). Florianopolis, Brazil.

Island of Kesmai. (2009). Preuzeto 13. October 2009 iz Island of Kesmai: http://homepage2.nifty.com/09/game/Legends/AboutLOK.html

John, M., Maurer, F., & Tessem, B. (2005). Human and Social Factors of Software Engineering – Workshop Summary. *ACM SIGSOFT Software Engineering Notes , 30* (4), 1-6.

Kahai, S., Carroll, E., & Jestice, R. (2007). Team Collaboration in Virtual Worlds. *The DATA BASE for Advances in Information Systems , 38* (4), 61-68.

Kapur, G. (2004). *Project management for information, technology, business, and certification.* Pearson/Prentice Hall.

Keys, S. *Enterprise project management: Challenges of the 21st century.* Primavera Systems Inc.

Knotts, G., Dror, M., & Hartman, B. (1998). A project management tool for computer-supported cooperative workduring project planning. *Proceedings of the Thirty-First Hawaii International Conference on System Sciences. 1*, str. 623-631. Kohala Coast, HI, USA: IEEE Computer Society.

Mockus, A., & Herbsleb, J. D. (2001). Challenges of Global Software Development. *Proceedings of the Seventh International Software Metrics Symposium (METRICS'01)*, (str. 182-184). London, England.

Mohagheghi, P. (2004). *Global software development: Issues, solutions, challenges [PDF document].* Retrieved from http://www.idi.ntnu.no/grupper/su/publ/parastoo/gsd-presentation-slides.pdf .

MPK20. (2009). Preuzeto 17. October 2009 iz MPK20: Sun's Virtual Workspace: http://research.sun.com/projects/mc/mpk20.html

MUD. (2009). *Multi User Dungeon*. Preuzeto 12. October 2009 iz http://www.british-legends.com/

NetBeans. (2010). *NetBeans IDE 6.8 Release Information*. Preuzeto 12. January 2010 iz http://netbeans.org/community/releases/68/

Nidiffer, K. E., & Dolan, D. (2005). Evolving Distributed Project Management. *IEEE Software , 22* (5), 63-72.

Owens, D., Davis, A., Murphy, J., Khazanchi, D., & Zigurs, I. (2009). Real-World Opportunities for Virtual-World Project Management. *IT Professional , 11* (2), 34-41.

Parvathanathan, K., Chakrabarti, A., Patil, P. P., Sen, S., Sharma, N., & Johng, Y. (2007). *Global Development and Delivery in Practice: Experiences of the IBM Rational India Lab.* New York, NY, USA: IBM.

Piri, A. (2008). Challenges of Globally Distributed Software Development – Analysis of Problems Related to Social Processes and Group Relations. *2008 IEEE International Conference on Global Software Engineering*, (str. 264-266).

PMI. (2009). *Project Management Institute.* Retrieved from: http://www.pmi.org/.

Powell, A., Piccoli, G., & Ives, B. (2004). Virtual Teams: A Review of Current Literature and Direction for Future Research. *The DATA BASE for Advances in Information Systems , 35* (1), 6-36.

Prentice, S. (2008). *Virtual Worlds - What to Expect in 2009.* London: Virtual Worlds Forum 08.

Project Management Institute. (1996). *PMBOK - A Guide to the Project Management Body of Knowledge.* Sylva, NC, USA: The Project Management Institute.

Sangwan, R., Bass, M., Mullick, N., Paulish, D. J., & Kazmeier, J. (2007). *Global Software Development Handbook.* Boca Raton, Florida, USA: Auerbach Publications.

Scheucher, B., Belcher, J. W., Bailey, P. H., Dos Santos, F. R., & Gütl, C. (2009). Evaluation Results of a 3D Virtual Environment for Internet-accessible Physics Experiments. *Conference ICL.* Villach, Austria.

Second Life. (2009a). *Second Life*. Preuzeto 14. October 2009 iz http://de.secondlife.com/

Second Life. (2009b). *The Marketplace*. Preuzeto 14. October 2009 iz Second Life: http://secondlife.com/whatis/marketplace.php

Sivan, Y. (2008). 3D3C Real Virtual Worlds Defined: The Immense Potential of Merging 3D, Community, Creation, and Commerce. *Journal of Virtual Worlds Research , 1* (1).

Smith, J. L., Bohner, S. A., & McCrickard, D. S. (2005). Project Management for the 21st Century: Supporting Collaborative Design through Risk Analysis. *Proceedings of the 43rd annual Southeast regional conference – Volume 2: ACM Southeast Regional Conference.* New York, NY, USA: ACM.

Sommerville, I. (2007). *Software Engineering, Eighth Edition.* Amsterdam: Addison-Wesley.

Stiller, E., & LeBlanc, C. (2002). *Project-Based Software Engineering: An Object-Oriented Approach.* Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

SunBlog. (2009). *Experimenting with World Creation (Project Wonderland)*. Preuzeto 15. October 2009 iz Dana in Geeksville: http://blogs.sun.com/DanaInGeeksville/entry/experimenting_with_world_creation_project

Systemic Logic. (2009). *Alternative Software Collaboration Tools (ASCTs)*. Preuzeto 15. November 2009 iz Systemic Logic: http://www.systemiclogic.com/?p=617

The Standish Group. (2009). *Standish Newsroom - CHAOS 2009*. Preuzeto 2. July 2009 iz http://www1.standishgroup.com/newsroom/chaos_2009.php

There. (2009). *Company*. Preuzeto 14. October There iz There: http://www.prod.there.com/info/company

Ultima Online. (2009). Preuzeto 13. October 2009 iz Online-Games: http://www-users.mat.umk.pl/~rembol/eng/screens.html

Van Ryssen, S., & Godar, S. H. (2000). Going International Without Going International: Multinational Virtual Teams. *Journal of International Management , 6*, 49-60.

Virtual World Timeline. (2009). Preuzeto 12. October 2009 iz Virtual World Timeline Project: http://www.dipity.com/xantherus/Virtual_Worlds

Wainfan, L., & Davis, P. K. (2004). *Challenges in Virtual Collaboration: Videoconferencing, Audioconferencing, and Computer-Mediated Communications.* Santa Monica, CA, USA: RAND Corporation.

Wonderland. (2009a). Preuzeto 15. October 2009 iz Project Wonderland: https://lg3d-wonderland.dev.java.net/

Wonderland. (2009b). *About Project Wonderland*. Preuzeto 15. October 2009 iz Project Wonderland: http://wiki.java.net/bin/view/Javadesktop/ProjectWonderlandAbout#VisionAndGoals

Wonderland. (2009c). *Project Wonderland Roadmap and Release Estimates*. Preuzeto 18. October 2009 iz Project Wonderland: http://wiki.java.net/bin/view/Javadesktop/WonderlandRoadmap#Release_0_5_July_2008_Major_Arch

WonderlandAuth. (2010). *Project Wonderland v0.5: Configuring Authentication*. Preuzeto 19. February 2010 iz http://wiki.java.net/bin/view/Javadesktop/ProjectWonderlandAuthentication05

WonderlandWiki. (2010). *Project Wonderland v0.5: Web-based Server Administration* . Preuzeto 27. January 2010 iz http://wiki.java.net/bin/view/Javadesktop/ProjectWonderlandServerAdministration05

WonderlandWiki. (2009b). *Project Wonderland v0.5: Working with Modules*. Preuzeto 12. November 2009 iz Wonderland Wiki: http://wiki.java.net/bin/view/Javadesktop/ProjectWonderlandWorkingWithModules05

WonderlandWiki. (2009c). *Using Shared X11 Applications*. Preuzeto 15. October 2009 iz Wonderland Wiki: https://wonderland.dev.java.net/user-guide-05/app-sharing.html

WonderlandWiki. (2009a). *Wonderland Cell Architecture*. Preuzeto 12. November 2009 iz Wonderland Wiki: http://wiki.java.net/bin/view/Javadesktop/ProjectWonderlandCellArch

Wong, S., & Burton, R. M. (2000). Virtual Teams: What are their Characteristics, and Impact on Team Performance? *Computational & Mathematical Organization Theory , 6*, 341-343.

WOW. (2009). Preuzeto 13. October 2009 iz World of Warcraft: http://wirelessdigest.typepad.com/photos/uncategorized/world_of_warcraft_1.jpg

Young, P. (1996). *Three Dimensional Information Visualisation*. Preuzeto 20. October 2009 iz http://vrg.dur.ac.uk/misc/PeterYoung/pages/work/documents/lit-survey/IV-Survey/index.html#sec2

Appendix A

Questions from the User Study

A1. Pre-questionnaire

A) Demographic Questions

A1) Age:

- ☐ 18 to 24
- ☐ 25 to 30
- ☐ greater than 30

A2) Gender:

- ☐ female
- ☐ male

A3) What is your main activity:

- Following a study program [ ]
- Teaching [ ]
- Professional work [ ]

A4) What computer tools to you mainly use to communicate with your peers following your main activities (give at least 3):

A5) What computer tools to you mainly use to organize and coordinate work with your peers following your main activities (give at least 3):

A6)

a) How often do you use mobile devices

- Never [ ]
- Occasionally [ ]
- Daily [ ]

b) What tools are most helpful (give at least 3):

c) Give at least 3 application domains where mobile devices can be used:

A7)

a) How often do you use 3D multi-user environments

- Never [ ]
- Occasionally [ ]
- Daily [ ]

b) Which one:

c) Give at least 3 application domains where such environments can be used:

B) Software Development Process

B1) How is your expertise in Software Development? (Please mark the appropriate number)

No expertise 2 1 0 1 2 High expertise

B2) How many years have you experiences in software development?

B3) What are you main activities?

B4) How experienced are you in distributed software development? (Please mark the appropriate number)

No experience 2 1 0 1 2 Great experience

B5) What are the main problems in distributed software development?

B6) How can technology support the distributed software development process?

B7) What tools can be used?

B8) How should in general the software development process be educated and trained?

B9) How should the distributed software development process be educated and trained?

B10) Do you agree that immersive collaborative virtual environments (virtual worlds) can be used for the distributed software development process? (Please mark the appropriate number)

I disagree 2 1 0 1 2 I agree

B11) What components should be provided?

B12) Do you agree that immersive collaborative virtual environments (virtual worlds) can be used to educate the software development process? (Please mark the appropriate number)

I disagree 2 1 0 1 2 I agree

B13) What components should be provided?

B14)

a) Do you think there is a difference educating and training of distributed software development using immersive collaborative virtual environments?

Yes [ ] No [ ]

b) If yes, what are the differences and how can it be considered in the virtual environment?

C) Software Project Management Process as meta-process of the Software Development Process

C1) How is your expertise in Project Management? (Please mark the appropriate number)

No expertise 2 1 0 1 2 High expertise

C2) How many years have you experiences in project management?

C3) What are you main activities?

C4) How experienced are you in distributed project management? (Please mark the appropriate number)

No experience 2 1 0 1 2 Great experience

C5) What are the main problems in distributed project management?

C6) How can technology support the distributed project management?

C7) What tools can be used?

C8) How should in general the project management be educated and trained?

C9) How should the distributed project management be educated and trained?

C10) Do you agree that immersive collaborative virtual environments (virtual worlds) can be used for the distributed project management? (Please mark the appropriate number)

I disagree 2 1 0 1 2 I agree

C11) What components should be provided?

C12) Do you agree that immersive collaborative virtual environments (virtual worlds) can be used to educate the project management? (Please mark the appropriate number)

I disagree 2 1 0 1 2 I agree

C13) What components should be provided?

C14)

c) Do you think there is a difference educating and training of distributed project management using immersive collaborative virtual environments?

Yes [ ] No [ ]

d) If yes, what are the differences and how can it be considered in the virtual environment?

A2. Post-questionnaire

D) Evaluation of the tool

D1) What is the overall impression of the tool appropriate for distributed software development? (Please mark the appropriate number)

Very bad 2 1 0 1 2 Very good

a) I liked most

b) I disliked most

c) What should be improved

d) Would you like to use such an environment performing your main activities? (Please mark the appropriate number)

No 2 1 0 1 2 Yes

D2) Do you agree that the tool is appropriate to education and train the software development process? (Please mark the appropriate number)

I disagree 2 1 0 1 2 I agree

a) I liked most

b) I disliked most

c) What should be improved

d) Would you like to use such an environment performing your main activities? (Please mark the appropriate number)

No 2 1 0 1 2 Yes

D3) What is the overall impression of the tool appropriate for distributed project management? (Please mark the appropriate number)

Very bad 2 1 0 1 2 Very good

a) I liked most

b) I disliked most

c) What should be improved

d) Would you like to use such an environment performing your main activities? (Please mark the appropriate number)

No 2 1 0 1 2 Yes

D4) Do you agree that the tool is appropriate to education and train project management? (Please mark the appropriate number)

I disagree 2 1 0 1 2 I agree

a) I liked most

b) I disliked most

c) What should be improved

d) Would you like to use such an environment performing your main activities? (Please mark the appropriate number)

No 2 1 0 1 2 Yes

D5) My overall impression of the tool is (Please mark the appropriate number)

Very bad 2 1 0 1 2 Very good

Appendix B

CD-Rom