Master's thesis

Flexible Learning Environment in Virtual 3D Worlds

Version of 09 February 2012

submitted by

Karl Haas, BSc
Mat-Nr 0630884

supervised by

Dipl.-Ing. Dr.techn. Univ.-Doz. Christian Gütl
Institute of Information Systems and Computer Media, Graz University of Technology, Austria

and

Assoc. Prof. Dr. Vanessa Chang
Curtin Business School, Curtin University Perth, Australia

Institute of
Information Systems and Computer Media
Graz University of Technology

Masterarbeit

Flexible Lernumgebung in Virtuellen 3D Welten

Version vom 9. Februar 2012

eingereicht von

Karl Haas, BSc
Mat-Nr 0630884

betreut durch

Dipl.-Ing. Dr.techn. Univ.-Doz. Christian Gütl
Institut für Informationssysteme und Computer Medien, Technische Universität Graz, Österreich

und

Assoc. Prof. Dr. Vanessa Chang
Curtin Business School, Curtin University Perth, Australia

Institut für
Informationssysteme und Computer Medien
Technische Universität Graz

Abstract

The rapid technological progress and the constantly changing social and societal conditions result in new requirements for learning processes. New information and communication technologies enable computer-aided learning approaches like E-Learning or M-Learning. Recent research results show the high potential of virtual worlds regarding innovative learning approaches with a very high degree of flexibility and possibilities for interaction and collaboration.

This work analyses requirements for virtual environments to support teaching and education. An extensive literature survey and analysis of current research results show the demand for collaboration between learners, adaptability of the learning environment and reusability of learning content to use existing infrastructures in an effective way.

The spatial separation of learners can cause feelings of isolation and a lack of community in case of an increased use of computer-based learning systems. The possibility of interaction and collaboration in virtual worlds can help to reduce those negative impacts and can have a positive effect on motivation and productivity.

Based on the free availability and the high degree of extensibility, OpenSimulator was analysed in detail and finally selected as reference platform. Apart from the missing guarantee regarding stability, because of the early development stage of the simulator, OpenSimulator represents a good alternative to the commercial platform Second Life. As the communication protocols and scripting languages of OpenSimulator and Second Life are compatible, a huge variety of practical examples and client applications are available.

A web application which configures and manages flexible learning environments in virtual 3D worlds has been developed. Teachers have the opportunities to customise learning spaces without knowing the technical details of the virtual world. The fully automated generation of the learning environment reduces the technical entry barriers. Provided learning tools encourage the collaboration of learners. The state of each learning session is saved automatically by the application and classes can be continued at any time later. The access to the learning spaces is restricted by a simplified user management.

Kurzfassung

Der rasante technologische Fortschritt und sich ständig ändernde soziale und gesellschaftliche Rahmenbedingungen führen zu neuen Anforderungen an Lernprozesse. Neue Informations- und Kommunikationstechnologien ermöglichen computergestützte Lernansätze wie E-Learning oder M-Learning. Jüngste Forschungsergebnisse zeigen das hohe Potenzial von virtuellen Welten für innovative Lernansätze mit einem sehr hohen Grad an Flexibilität und Möglichkeiten zur Interaktion und Zusammenarbeit.

Diese Arbeit analysiert Anforderungen an virtuelle Umgebungen um Lehre und Bildung zu unterstützen. Eine umfangreiche Literatur-Recherche und Analyse aktueller Forschungsresultate zeigt die Notwendigkeit der Zusammenarbeit von Lernenden, Anpassbarkeit der Lernumgebung und Wiederverwendbarkeit von Lerninhalten zur effektiven Nutzung der vorhandenen Infrastrukturen.

Die räumliche Trennung der Lernenden kann bei erhöhter Verwendung von computergestützten Lernsystemen zu einer zunehmend stark empfundenen Isolation und zu einem erhöhten Mangel an Gemeinschaft führen. Die Möglichkeit der Interaktion und Zusammenarbeit in virtuellen Welten kann helfen diese negativen Auswirkungen zu mindern sowie die Motivation und Produktivität positiv beeinflussen.

Auf Grund der freien Verfügbarkeit und dem hohen Grad an Erweiterbarkeit wurde OpenSimulator im Detail analysiert und als Referenzplattform ausgewählt. Abgesehen von der nicht gewährleisteten Stabilität auf Grund des frühen Entwicklungsstadiums des Simulators, stellt OpenSimulator eine vielversprechende Alternative zur kommerziellen Plattform Second Life dar. Durch die Kompatibilität der Kommunikationsprotokolle und Skriptsprachen von OpenSimulator und Second Life stehen eine Vielzahl von Praxisbeispielen und Client-Anwendungen zur Verfügung.

Es wurde eine Web-Applikation entwickelt, die es ermöglicht flexible Lernumgebungen in virtuellen 3D Welten zu konfigurieren und zu verwalten. Lehrende können Lernräume individuell anpassen ohne technische Details der virtuellen Welt kennen zu müssen. Die voll-automatisierte Generierung der Lernumgebungen in der virtuellen Welt reduziert die technische Einstiegsbarriere. Zur Verfügung gestellte Lerntools fördern die Zusammenarbeit der Lernenden. Der Zustand der Lerneinheit wird automatisch gespeichert und kann zur Fortsetzung des Unterrichts zu einem späteren Zeitpunkt wiederhergestellt werden. Der Zutritt zu Lernräumen wird mittels einer vereinfachten Benutzerverwaltung eingeschränkt.

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.

Place	Date	Signature

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

Ort	Datum	Unterschrift

Contents

List of Figures

List of Tables

List of Listings

List of Abbreviations

ALE	Adaptive Learning Environment
API	Application Programming Interface
AW	Active Worlds
BDV	Best Digital Village
CVE	Collaborative Virtual Environment
CL	Collaborative Learning
CLE	Collaborative Learning Environment
CoXR	Collaborative Cross-Reality
CSCL	Computer-Supported Collaborative Learning
DBR	Design-Based Research
DBMS	Database Management System
DLL	Dynamic Link Library
FAS	Federation of American Scientists
HTTP	Hypertext Transfer Protocol
LMS	Learning Management System
FOSS	Free and Open-Source Software
GUI	Graphical User Interface
GWT	Google Web Toolkit
HTML	Hypertext Markup Language
ICT	Information and Communication Technology
IDE	Integrated Development Environment
IICM	Institute for Information Systems and Computer Media
IP	Internet Protocol
JSON	JavaScript Object Notation
LSL	Linden Scripting Language
MMORPG	Massively Multiplayer Online Role-Playing Game
MOODLE	Modular Object Oriented Dynamic Learning System
MOAP	Media On A Prim
MUD	Multi-User Dungeon
MUVE	Multi-User Virtual Environment
OS	OpenSimulator
OSSL	OpenSim Scripting Language
OW	Open Wonderland
REST	Representational State Transfer
SL	Second Life

RMS	RoomManager Server
RMC	RoomManager Client
SLOODLE	Simulation Linked Object Oriented Dynamic Learning Environment
SOA	Service Oriented Architecture
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UUID	Universally Unique Identifier
VCUHK	Virtual Campus of the Chinese University of Hong Kong
VLE	Virtual Learning Environment
V3DW	Virtual 3D World
VE	Virtual Environment
VoIP	Voice over IP
VR	Virtual Reality
VW	Virtual World
WoW	World of Warcraft
WRC	World Realisation Component
X3D	Extensible 3D
XML	Extensible Markup Language

Chapter 1

Introduction

These days, the main stakeholders of educational approaches are confronted with a variety of issues which are caused by ever-changing political, economical, social, technological and legal situations. The goal aspirations and actual needs of learning groups become increasingly dynamic and learning groups tend to be geographically dispersed. This physical distance between learners causes negative learning experiences like isolation, loneliness or a lack of community. (Gütl, Chang, & Freudenthaler, 2010)

> *"The predominantly one-way nature of the media used in second generation distance education results in little or no student/student and student/teacher communication. This lack of communication leads to students feeling isolated."* (Jones, 1996)

This work focuses on how modern Information and Communication Technology (ICT) can support distance education and reduce the negative effects caused by the physical distance between learners and teachers. Furthermore the opportunities and drawbacks of Virtual 3D Worlds (V3DWs) in the context of education are analyzed.

> *"Traditional education forms have limitations that virtual worlds can overcome. [...] Virtual worlds allow users with specific needs and requirements to be able to access and use the same learning materials from home, as they would be receiving if they were in the presentation. This can help users to keep up to date with the relevant information and needs while also feeling as though involved."* (B. Chen, Huang, & Lin, 2009)

In order to analyze the educational features of V3DWs a web based configuration tool for flexible learning settings has been developed. The application tries to enhance the educational learning capabilities of virtual worlds by providing a simple way of creating and managing such flexible learning settings.

1.1 Background and Motivation

Previous research work at the Institute for Information Systems and Computer Media (IICM) at the Graz University of Technology has analyzed the educational features of the commercial V3DW Second Life. Apart from its powerful virtual world features, which are described later in this work in detail, Freudenthaler (2011); Kopeinik (2010) experienced some major restrictions when trying to extend the educational features of the virtual world. Amongst others, they mention the following limitations:

- The possibilities for in-world developments are highly restricted.
- Significant restrictions have been experienced due to missing document sharing possibilities.
- Because of its commercial background, working with Second Life is not free of charge.
- Object manipulation is very limited, to given an example objects can not be moved further than 10 meters.

In order to overcome these financial and technical restrictions, the primary goal of this work was to analyse the educational features of an open source alternative to Second Life, the open source project OpenSimulator. By choosing a non commercial project, initial and variable costs for land and object ownership can be avoided. Due to the modular structure of OpenSimulator, it is possible to extend the functionality via plug-in modules. (Dafli, Vegoudakis, Pappas, & Bamidis, 2009)

Based on the gained experience with Second Life, a tool to manage flexible learning settings within OpenSimulator should be realised. The application should provide a simple Graphical User Interface (GUI) to create and manage learning settings within the virtual world. Teachers without high technical skills should be able to create virtual 3D world learning settings, without knowing the technical in-world object creation, and handling functionalities. Furthermore the tools should provide a simple mechanism to save the state of a learning setting in order to be able to continue the session in the future.

In addition the application should address the need for collaboration between the stakeholders of the learning session. The support for interactive tools (e.g. communication board) in OpenSimulator has been analyzed and some examples like an interactive flip chart have been developed. The access control capabilities have been investigated and a basic access permission system has been implemented.

A web application has been build in order to manage the learning settings. The in-world object realisation has been provided by adding a new module to the plug-in architecture of the virtual world server.

1.2 Structure

The first part of this work deals with the theoretical background and summarises relevant related work. Chapter 2 gives an overview about education and technology enhanced education, virtual 3D worlds and learning in virtual worlds. It provides a more detailed motivation about how V3DWs can support education and it describes which requirements such a world has to meet in order to provide an educational benefit.

Chapter 3 describes the architecture of OpenSimulator in detail. It gives an overview about how objects can be created and how media can be provided within the virtual world. The development and extension mechanisms of the virtual world server are analysed and known limitations of the open source platform are discussed in this chapter.

Chapter 4, which is the last chapter of the theoretical part, summarises the current research work in the fields of learning in virtual worlds and extension of the OpenSimulator virtual world server. It gives a brief overview about the architecture of the solutions and the findings of the research.

The second major part of this work contains a description of the developed flexible learning setting management application. Chapter 5 describes the requirements of the prototype in detail. The design of the application is discussed and the implementation details are explained. The configuration options of the application and the usage scenarios are documented in order to illustrate the functionalities of the solution. The gained experience during the work on the prototype is listed at the end of the chapter.

The final chapter concludes the work by summarising the results of the work and by providing an outlook about possible future work on this topic.

Appendix A contains additional detailed implementation information. Appendix B explains in detail how to install the developed application. Finally appendix C lists the content of the enclosed CD ROM.

Chapter 2

Education in Virtual Worlds

Because of the ever-changing political, social, economical, technological and environmental situations, educational approaches have changed over the last century. Learning becomes a day-by-day routine which is no longer concentrated in the first stages of human life. It must be seen as an integrated activity. (Chang & Gütl, 2008)

This chapter deals with the question why we need virtual learning environments and which potentials flexible learning settings in virtual 3D worlds have.

2.1 E-Education

Over the last decades, technology has influenced educational approaches. Children who grew up with Information and Communication Technologies (ICTs) use several media simultaneously for communication, learning and entertainment. (Chang & Gütl, 2008)

Figure 2.1 shows the evolution of learning types.

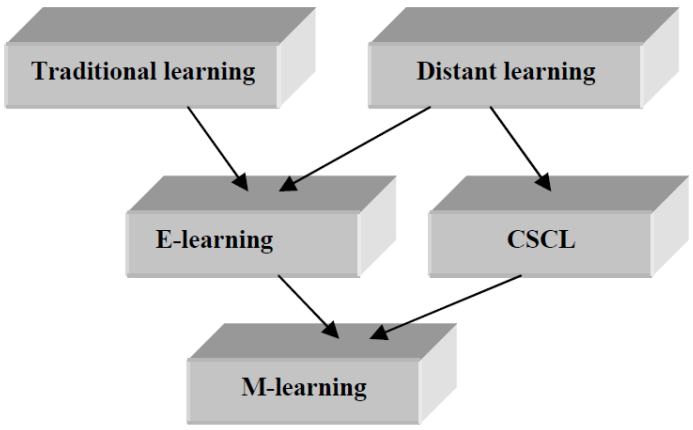


Figure 2.1: Learning type evolution (Yordanova, 2007)

2.1.1 Traditional Learning

Learning is a very broad topic involving many different aspects. As Mehlenbacher, Holstein, Gordon, and Khammar (2010, p. 240) mentioned, *"General human learning occurs anytime learners interact with their environment and, more specifically, we tend to assume learning is facilitated best when an instructor designs or presents learning tasks to a learner or learners, an act or event that requires social dynamics and that assumes a surrounding learning environment and artifacts"*.

People use different styles of learning. Layman, Cornwell, and Williams (2006, p. 429) introduces the Felder-Silverman learning styles which can be used to characterise the way in which students absorb and retain information as follows:

- *Active-Reflective*
  Active learners learn best by experimentation and working with others, while reflective learners learn more by thinking things out on their own.
- *Sensing-Intuitive*
  Sensors prefer information gathered through experience and are attentive to details, while intuitors prefer abstract concepts and are bored by details, preferring innovative thoughts instead.
- *Visual-Verbal*
  Visual learners absorb information best through pictures, graphs, and charts, whereas verbal learners prefer written or spoken explanations.
- *Sequential-Global*
  Sequential students learn in orderly, incremental steps with one point or fact connecting to the next, whereas global learners have trouble learning fact-by-fact and learn in cognitive leaps after accumulating all the facts.

2.1.2 Distance Learning

Albion (2009) mentioned that originally distance in eduction was meant as the geographical distance between the teacher and the learner. A similar definition of distance education is given by Wains and Mahmood (2008) as *"a form of teaching/learning in which the participants are separated by physical distance, time and resources"*. Moore (1993) identified distance in distance education as *"a pedagogical concept addressing the psychological and communications space that separates learner and teacher rather than merely geographic separation"* (as cited in Albion, 2009, p. 665).

Despite the advantages of distance education over face to face learning, Wains and Mahmood (2008) mentioned the following problems related to classical distance learning approaches:

- Used technologies like print media, Internet or TV and radio broadcasts are not flexible enough.

- Students have to study at the designated schools and at specific times.
- The used technology is costly and not common among the masses.

Research has shown that distance learning causes negative learning experiences like isolation or a lack of community. Gütl (2011, p. 280) found that, *"Collaborative online learning can improve or even overcome these problems"*. Sufficient support of ICT is needed in order to satisfy requirements of distance learners. (Gütl et al., 2010)

2.1.3 E-Learning

According to Balasundaram (2011, p. 624), *"E-learning, in general refers to the approach of facilitating and enhancing the teaching and learning process by means of tools and technologies. The growth of technologies has made it possible to define the educational activities in a variety of ways using personal computers, CD ROMs, Digital Television, Mobile Devices and the Internet"*.

E-Learning systems are not only used in the traditional learning context, but also in professional trainings and by businesses. They may be used to support face to face, collocated and traditional classrooms as well as distance learning. The combination of pedagogical approaches of traditional learning and ICT enables a distant form of education. Some problems require a hybrid form of education - E-Learning. (Costa & Aparicio, 2011; Yordanova, 2007)

As listed by Costa and Aparicio (2011, p. 37), E-Learning systems are also known as:

- Computer Learning Content Information Management System (CLCIMS)
- Course Management System (CMS)
- Learning Content Management System (LCMS)
- Learning Management System (LMS)
- Learning Platform (LP)
- Learning Support System (LSS)
- Managed Learning Environment (MLE)

In traditional E-Learning systems the accessibility of the content is provided by an activity of the lecturer. The major activities which are provided by E-Learning systems are content development, making the content available to the learners, conducting tests and exams and performing assessment and providing certifications. (Ferretti, Mirri, Muratori, Roccetti, & Salomoni, 2008; Balasundaram, 2011)

2.1.4 E-Learning 2.0

Based on modern Web 2.0 technologies, ICT-based learning environments have emerged from content-centered, centralised and static systems to enhanced approaches. These ap-

proaches have been known as E-Learning 2.0. Unlike the content-centered approach of traditional E-Learning approaches, it is seen as a people-centric approach. (Chang & Gütl, 2008)

As mentioned by Chang and Gütl (2008), E-Learning 2.0 approaches have the following characteristics:

- The roles of teachers and students blur.
- Collaborative nature of learning with a strong focus on content sharing, reuse, adaptation and personalisation.
- Pre-existing knowledge is transferred to recipients.
- A platform which is built on knowledge.

The Web has evolved from an information repository to a place for dynamic user interaction, social networking and community building. E-Learning 2.0 is the new paradigm of online learning in and through networked communities. It provides the potential for global knowledge sharing. By the use of Web 2.0 technologies new evolving E-Learning contents can be dynamically created, aggregated, classified, syndicated and shared by students. Such technologies are for example blogs, podcasts, wikis or media sharing applications. (S.-J. Chen, Hsu, & Caropreso, 2009; Ferretti et al., 2008)

As outlined by Pollini, Giusti, and Napoletano (2011) *"Two main features have been identified that make Web 2.0 technologies suitable for facilitating online collaborative learning: one of the features is the relatively simple and intuitive use of Web 2.0 tools, which enable learners to easily contribute and experiment in online learning communities, the second one is that Web 2.0 technologies contribute to change the role of learners from being passive recipients of knowledge to active participants in the construction of knowledge".*

The participation of people to the content creation and customisation of content is recognised as a great potential for E-Learning. On the other hand the complexity of collaborative 2.0 tools represents a risk of exclusion for people with disabilities. (Ferretti et al., 2008)

2.1.5 Computer-supported Collaborative Learning

> *"Collaborative learning can be seen as an abstract concept involving joined intellectual effort by the learning community built of students and teachers in a wide range of settings. The learning group mutually searches for understanding or meaning, explores solutions or creates some sort of product."* (Gütl, 2011, p. 280)

Konstantinidis, Tsiatsos, and Pomportsis (2009, p. 280) defined Collaborative Learning (CL) as a *"general term used for the description of educational practices based on the simultaneous cognitive and mental effort of multiple students or/and educators. Students*

share a common goal, depend on each other and are mutually responsible for their success or failure".

Vygotsky (1986) found, that *"learning and developing is a social, collaborative activity"*. According to (Tate et al., 2010, p. 63), *"successful collaborations occur in an information space that offers access to appropriate informational facilities and resources"*. Tate et al. (2010) further found, that in a collaborative informational process the collaborator applies his knowledge and skills in order to add value. Woo and Reeves (2007) mentioned that *"interaction is an essential ingredient of any learning environment"* (as cited in Albion, 2009, p. 664).

De Lucia, Francese, Passero, and Tortora (2009) grouped Computer-Supported Collaborative Learning (CSCL) into two categories which are:

- *Asynchronous*
  Due to the fact that students work at different times, there is no problem with time zones or holidays. For this group, international collaborative work is well supported by the features offered by Learning Management Systems (LMSs).
- *Synchronous*
  Enables students to work simultaneously together by using tools like audio or video conferencing, multi-user domain or dimension, instant messaging, etc.

Previous research has shown that collaborative learning activities are more effective compared to individualistic instructional learning approaches. Konstantinidis et al. (2009, p. 280) found, that *"probably, the major advantage of collaborative learning compared to other educational practices (e.g. personalised learning) is the interaction with others"*.

According to Bouras and Tsiatsos (2006); Konstantinidis et al. (2009) a Collaborative Learning Environment (CLE) has the following characteristics:

- The users of the environment have different roles and privileges.
- The educational interaction transforms the virtual space into a communication space. Thus communication channels have to be provided.
- Multiple media varying from simple text to 3D graphics are used to represent the information.
- Users can interact with each other and with the environment.
- The system must be able to support various technologies.
- The environment supports the capability of implementing multiple learning scenarios.
- The environment should have common features with a physical space, recognisable real world elements can be visualised.

Collaboration in the context of learning has also some challenges to face. Developing collaborative learning settings is time consuming and expensive and the relationship between teacher and student can be disorientating. Researchers found, that unguided and

or unstructured collaboration does not necessarily result in learning. (Gütl, 2011; Konstantinidis et al., 2009)

As cited in Farley (2009), Caeiro-Rodríguez, Llamas-Nistal, and Anido-Rifón (2005) found three timing issues associated with collaborative learning:

- *Synchronisation patterns*
  An activity undertaken by one learner occurs in temporal relation to another activity undertaken by a second learner according to the conditions of the collaboration.
- *Scheduling patterns*
  To determine times when an event will occur or a product will become available (e.g. deadlines).
- *Allocation patterns*
  To determine how much time is spent on each task.

2.1.6 M-Learning

Yordanova (2007) introduces M-Learning or Mobile Learning as *"learning that is wireless and ubiquitous"*. Mobile technologies provide the necessary mobility and flexibility of learning activities and the support of different services related to educational processes. Figure 2.2 shows the main elements of M-Learning which are mobile technologies, mobile devices, wireless protocols, wireless language, and wireless applications. (Yordanova, 2007)

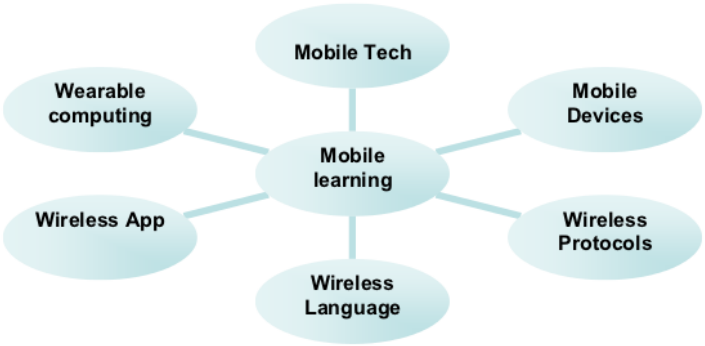


Figure 2.2: Elements of M-Learning (Yordanova, 2007)

M-Learning is a type of E-Learning which uses wireless and mobile technology for the learning experience. Devices like laptops, tablet PCs or cell phones are used to deliver learning content to students. Due to the increased computing power of such devices, software which was once known as to be used on computers only, can now be used on mobile devices. In M-Learning, learning can happen any time and anywhere. A permanent connection to the Internet is not needed. (Wains & Mahmood, 2008)

Yordanova (2007) identified the following technical and social issues of the application of M-Learning in education:

- Older users encounter problems when they have to use mobile devices for education.
- The displays of mobile devices are limited.
- New technologies are more vulnerable of attacks by intruders. Thus user data privacy and learning material security must be considered in effective mobile learning systems.

2.2 Virtual 3D Worlds

A Virtual Environment (VE) is an environment which is produced from automated rules. Those rules allow users to modify the environment to some degree. In the VE each user is regarded as a separate being, usually identified by an anthropomorphic character, the so called avatar. Further the VE is shared, the actions take place in real time, it supports multiple users simultaneously and is persistent. (Konstantinidis et al., 2009)

As defined by Fox, Kelly, and Patil (2010, p. 88) Virtual Worlds (VWs) are *"online, persistent, shared environments that allow multiple users to create content and experiences, collaborate, and communicate with other users generally through the use of an avatar"*.

Bartle (2010) defines a VW as *"an automated, shared, persistent environment with and through which people can interact in real time by means of virtual self"*. The VW provides the set of rules which define what each visitor can do within the world. The changes a visitor makes in the VW are persistent, which means that the world continues to exist whilst the visitor is not connected to the VW. Players can interact with the VW and each other. The world generates feedback for events of users in real time. Each player or visitor is uniquely identified by their character, the so-called Avatar. (Bartle, 2010)

> *"V3DWs help to recreate real world scenarios or to create complete new ones either with real world physical phenomena or different ones. V3DWs are composed by static or interactive objects and avatars as the digital representatives of the users acting and interacting."* (Gütl, 2011)

A term which is closely related to V3DW is Virtual Reality (VR). According to De Troyer, Kleinermann, Pellens, and Bille (2007, p. 3), *"Virtual Reality (VR) is a technology to simulate environments and create the effect of an interactive three-dimensional world in which objects have a sense of spatial and physical presence and can be manipulated by the user as such. VR has gained a lot of popularity during the last decennia due to e.g., games and applications such as Second Life"*.

There are two classes of virtual reality applications. Whilst the first class is single user oriented, the second class supports more than one user. In this Multi-User Virtual Environment (MUVE), collaboration between the users is supported. For example a user is represented by an avatar which can communicate to the avatars of other users. (Gütl, 2011)

Another important term which should be mentioned in this context is Collaborative Virtual Environment (CVE). Snowdon, Churchill, and Munro (2001) defined CVEs as *"computer-based, distributed, virtual space or set of places. In such places, people can meet and interact with others, with agents or with virtual objects. CVEs might vary in their representational richness from 3D graphical spaces, 2.5D and 2D environments, to text-based environments. Access to CVEs is by no means limited to desktop devices, but might well include mobile or wearable devices, public kiosks, etc."*.

According to Farley (2009), *"a Multi-User Virtual Environment (MUVE) is a computer-, server- or Internet-based virtual environment that allows participants to move around and use various forms of communication (text chat, voice chat or instant messaging)."*

The characteristics of virtual 3D worlds or MUVEs are similar to those of computer games. The main difference is that there is no explicit goal the user has to achieve in the VW. (Albion, 2009)

2.2.1 History of Virtual 3D Worlds

1978 the development of Multi-User Dungeon (MUD) started by Roy Trubshaw and Richard Bartle at the University of Essex, England. MUD was a text based multi-player development of early adventure games. Players wrote their own VWs by using the MUD system. The first VWs inspired by MUD, which were not native to the University of Essex, had been developed in 1985. In 1987 *AberMUD* has been written by Alan Cox and due to the fact that it was written in C it could be executed under Unix. After some years *AberMUD* had been installed on thousands of UNIX machines. Inspired by AberMUD the VW *TinyMUD* has been developed by Jim Aspnes in 1989. In *TinyMUD* users were able to create content from within the world. In 1990 Stephen White wrote MOD ("MUD, Object-Oriented") which was a scripting language with the purpose of object creation. This period was dominated by VWs executed on university computers. The worlds have been used and played for free. (Bartle, 2010)

Starting with 1995 the first commercial VWs appeared. They proved that they could be lucrative and the game industry began to take notice. 1997 the online role-playing game *Ultima-Online* appeared and was a great success. Within a year of launch 100.000 players were playing. The first fully V3DW with a big success was *EveryQuest. The Sims* in 2002 and *Star Wars Galaxy* in 2003 became the first franchised VWs. In the following years dozens of new VWs appeared. According to Fox et al. (2010) in 2009 nearly 200 VWs have been located by the Federation of American Scientists (FAS). Gütl (2011) estimates that *"80 percent of active Internet users will have some sort of online presence in virtual worlds by end of 2011"*. (Bartle, 2010)

In the past years researchers began to explore how VWs can be adapted for the use of learning. First approaches typically involved VR equipment which had caused higher costs compared to traditional 2D learning environments. Even though these research had been theoretically important, the practical impact on classroom practices has been low. (Jacobson, Kim, Miao, Shen, & Chavez, 2010)

Currently the global players are Second Life and World of Warcraft (Bartle, 2010). In the following section those applications and some relevant alternatives are introduced.

2.2.2 Second Life

Second Life (SL) is a social world which has no embedded game-play and no priori defined goals. It is not a game and there are no competitors. It is a persistent, computer-generated virtual world which has been developed by Linden Labs in 2003. Figure 2.3 shows an university building within SL. All its content is created by the users and it can be seen as a platform providing a VW with open-ended possibilities. (Bartle, 2010; Vrellis, Papachristos, Bellou, Avouris, & Mikropoulos, 2010)

Figure 2.3: Second Life University (Second Life, 2011)

As outlined by Vilela et al. (2010), *"although there are many 3D virtual worlds and virtual world platforms, Second Life has been the most used for adult-oriented education and training."* SL has become the most popular social VW. (Varvello, Ferrari, Biersack, & Diot, 2011)

Linden Labs provides an open communication protocol between the server and the client application. The virtual space within SL is based on areas of 256x256 meters. Each so-called region is supported by a server, which can have its own technical specification. A region can be either public or private. Whilst a public reagion belongs to Linden Labs, private regions are owned by individuals or companies. Users can control their viewpoint independent of their avatar's movement. Objects that stand on the ground and objects that hover are supported by SL. Companies have invested millions of dollars in order to build their own virtual products and advertisement within the virtual space. (Vilela et al., 2010; Varvello et al., 2011)

Varvello et al. (2011) identified the following two innovative features of SL:

- *World building*
  It is possible to participate in the deployment of the virtual environment through the creation of user-generated objects.

- *Virtual economy*
  Users can buy and sell objects, services and lands.

Some of the major drawbacks of SL are caused by its commercial background. A key issue is the fact, that the world and its content is hosted on servers which are located in facilities of Linden Lab, which is the hosting company of SL. (Vilela et al., 2010)

2.2.3 Open Simulator

OpenSimulator is an open source multi-user 3D application server. It is designed to be very flexible and to allow virtual world developers to customise their worlds. OpenSimulator implements most of the functionalities of SL based on the open protocol of the SL server. It is freely available and can be installed at any computer, which allows organisations to run their own server. (OpenSimulator Wiki, 2011f; Vilela et al., 2010)

OpenSimulator Wiki (2011f) provides the following list of features of the VW server:

- Supports online, multi-user 3D environments as small as 1 simulator or as large as thousands of simulators.
- Supports 3D virtual spaces of variable size within one single instance.
- Supports multiple clients, accessing the same world at the same time via multiple protocols.
- Supports real-time physics simulation with multiple engine options including Open Dynamics Engine (ODE).
- Supports clients that create 3D content in real time.
- Supports in-world scripting using a number of different languages, including Linden Scripting Language (LSL), OpenSim Scripting Language (OSSL), C# and VB.NET
- Provides unlimited ability to customise virtual world applications through the use of scene plug-in modules.

The application which has been developed during this work is based on OpenSimulator. Thus the architecture and features of the VW server are described in detail in chapter 3.

2.2.4 Open Wonderland

Open Wonderland (2011a) introduces Open Wonderland (OW) as *"a 100% Java 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 collaborate in an education, business, or government context. Wonderland is completely extensible; developers and graphic artists can extend its functionality to create entirely new worlds and add new features to existing worlds"*.

Open Wonderland (2011b) provides a list of the most significant features of OW. Each installation includes a different set of features depending on the applications and utilities installed on the server. (Open Wonderland, 2011b)

One of the major goals of OW is a completely extensible environment. The environment should be robust enough for organisations to build a virtual place which is able to conduct real business or education. (Open Wonderland, 2011a)

2.2.5 Active Worlds

As introduced by ActiveWorlds (2011), *"Activeworlds offers a comprehensive platform for efficiently delivering real-time interactive 3D content over the web. Activeworlds' 3D content is dynamic, visually compelling and most importantly provides users a richer, more exciting online experience"*.

The Active Worlds (AW) platform consists of proprietary 3D server software, browser and content authoring tools. It is a unique multi-user 3D virtual reality browser and chat software. The world can scale up to 65000 simultaneous users and unlimited virtual real estate. The objects of the VW are stored on a web server and once downloaded they are stored in a cache system. (Tatum, 2000)

> *With over 1,000,000 downloads of the Active Worlds browser, and over a million unique users in the main Active Worlds Universe, due in large part to its ease of use and versatility, Active Worlds appeals to a broad and diverse global audience.* (Tatum, 2000)

Tatum (2000) addressed the following features of the AW platform:

- Streaming of 3D contents.
- Real-time authoring.
- Most 3D formats are easily converted into AW supported formats.
- All content objects (eg. audio, video and image files) can be hyperlinked to the Internet.

Several content-driven universes are hosted by Activeworlds.com. Users can purchase their own servers and develop VWs based upon their interests. Public building worlds are offered where users can generate their own content for free. (Tatum, 2000)

2.2.6 World of Warcraft

Blizzard Entertainment (2011b) introduces World of Warcraft (WoW) as *"an online game where players from around the world assume the roles of heroic fantasy characters and explore a virtual world full of mystery, magic, and endless adventure"*. Figure 2.4 shows

Figure 2.4: World of Warcraft screenshot (Blizzard Entertainment, 2011a)

a screenshot of the virtual game. World of Warcraft (WoW) is the leading VW game with about 14 million players. It is a so called Massively Multiplayer Online Role-Playing Game (MMORPG) where people play roles of unique characters in a persistent online world shared by thousands of other players. WoW is available in nine languages and it brings people of different ages and economic classes together. (Varvello et al., 2011; Blizzard Entertainment, 2011b; Bardzell, Bardzell, & Nardi, 2011)

> *"WoW is much more specific: a fantasy world with systematic representations of social life (e.g., historical-cultural identities, race/gender representations, governance structures, and holidays) and distinctive activities, such as practicing (virtual) medieval crafts and organizing groups of up to 40 people to defeat cartoon monsters in elaborate theatrical settings."* (Bardzell et al., 2011)

In their work Nardi and Harris (2006, p. 158) conclude, that *"the design of World of Warcraft and the player culture that have developed within the game provide an innovative space in which strangers collaborate and can become friends. At the same time, WoW joins a long tradition of card and board games in which family and friends of different ages and genders may play together"*.

2.3 Learning in Virtual 3D Worlds

According to Zender, Dressler, Lucke, and Tavangarian (2009), *"Virtual learning is used to simulate face-to-face learning, to create new and innovative learning material and to interconnect distant learners"*. Virtual Learning occurs within a VE. As mentioned by Blascovich et al. (2002), *"virtual environments can be viewed as synthetic sensory infor-*

mation that leads to perceptions of environments and their content as if they were not synthetic" (as cited in Gütl, 2011, p. 281).

2.3.1 Motivation

Nowadays many students have to undertake considerable paid employment during their studies. Not least for that reason, many students are not able to attend all scheduled lectures. They need flexibility in order to proceed with their studies. (Albion, 2009)

> *"Virtual worlds have the potential to revolutionise the way we learn, teach, and conduct research."* (Thomas & Brown, 2009)

Because V3DWs support multiple communication channels, the communication within the V3DW can improve the knowledge transfer. The feeling of being part of the VW can increase motivation and productivity. The learning outcome can be increased by the feeling of belonging to a community. The barriers between students, tutors and instructors are reduced within a V3DW and collaboration between students provokes their activity and stimulates motivation. Learning becomes more student-oriented which increases the likelihood that students will remember the learned content. Konstantinidis et al. (2009, p. 281) surmised, that *"the most important factor in designing a CVE is the catering for communication and interaction between the participating students and educators"*. (Gütl, 2011; Konstantinidis et al., 2009)

As outlined by Dafli et al. (2009), *"recent technological developments have enabled the use of virtual three dimensional platforms for educational use"*.

> *"In recent years there has been significant growth in the use of V3DWs for e-learning and distance education. Virtual learning environments support teaching and learning in an educational context, offering the functionality to manage the presentation, administration and assessment of coursework."* (Callaghan, McCusker, Lopez Losada, Harkin, & Wilson, 2009b)

As stated by Burkle and Kinshuk (2009, p. 320), *"recently, virtual worlds have become an important part of teaching and training, transforming the way people work and learn. As technology allows more and more content to be virtual, so improves the possibility of better learner engagement [...] Virtual reality has also started to transform the way students have access to content, entertainment, and knowledge, making content portable and therefore, transforming the physical limits of the classroom"*.

Thomas and Brown (2009, p. 19) conclude in their work, that *"virtual worlds are illustrating a shift in the way learning is happening"*. To couple VW technologies with a new generation of tools like YouTube, Wikipedia and Facebook can offer spectacular new ways to create, search, display and share information and knowledge. (Fox et al., 2010)

Land and Bayne (2006) mentioned, that *"many educators and learners find text-based online learning environments disembodied and less real than conventional face-to-face classes"*. Encouraging interactivity between learners, adding of new media or application that supports a more direct interaction between teacher and learner could enhance the sensed distance. MUVE have the potential to improve the felt distance by representing the user by its avatar. (Albion, 2009)

In his work Scopes (2009) shows, that V3DWs are able *"to supplement blended learning and to bridge the gap between asynchronous online course participation and physical classroom experience"* (as cited in Gütl et al., 2010). According to Jacobson et al. (2010, p. 113), *"the affordances of highly interactive game-like systems are well suited to support many of the recommendations emerging from learning sciences and educational research related to how people learn"*.

2.3.2 Requirements

Earlier in this chapter the importance of collaboration in the learning process has been pointed out. This section addresses additional requirements which are faced by virtual training and learning applications. The requirements for virtual learning environments depend on different factors like the involved people, the object of the training and pedagogical expectations. (Gerbaud, Gouranton, & Arnaldi, 2009)

Adaptability

One important requirement of virtual training or learning applications is adaptability. Nowadays virtual training applications don't satisfy this ability. A virtual learning environment has to support the following two levels of adaptability. First the application settings must be adjustable to the corresponding training needs and second it must be possible to create a reactive VW that fulfils the requirements of the training session. (Gerbaud et al., 2009)

Hornik, Johnson, and Wu (2007) mentioned, that there is a "*need to flexibly design Virtual Learning Environments (VLEs) such that the technology support of learning is flexible and adaptable to match users learning conception*" (as cited in Gütl et al., 2010, p. 2). There are already many learning tools based on V3DWs available, but they do not satisfy the required flexibility. (Gütl et al., 2010)

Reusability

According to Dafli et al. (2009) *"considering the newest concept of sharing educational material by means of repurposing its characteristics/attributes or content, one needs to consider the way application scenarios stemming from one such platform are transformable into another platform or made to fit a different audience or purpose"*.

The initial set-up of a new virtual environment is time and cost consuming. Pape, Anstey, Dolinsky, and Dambik (2003, p. 1041) found, that *"the ability to dynamically bring objects and their behaviours into a virtual world, even when these objects were originally created as part of a completely different virtual world; the objects would be automatically able to interact in the new environment without any coding modifications"*. With this ability a system would speed the creation of significant shared worlds. (Dafli et al., 2009; Pape et al., 2003)

Reusability of the learning environments is a significant agenda for the design of learning. To develop a mechanism for reuse and adoption would protect the substantial investment necessary for the creation of new Adaptive Learning Environments (ALEs). Further it should allow for interoperability in different environments and aggregation of content by subsequent users. (Farley, 2009)

Dafli et al. (2009) emphasises *"the need to incorporate educational metadata descriptions so that the educational context is also interoperable"*. Experience has shown the importance of standard metadata to exchange and repurpose virtual environments. Dynamically loaded modules can promote the re-use of code. In theory an application can be build by simply bringing together already existing modules. (Dafli et al., 2009; Pape et al., 2003)

2.3.3 Suitable Platform Selection

As already mentioned, nowadays there are various different V3DWs available. A challenging task is the selection of the most suitable platform for the learning settings. Most platforms provide three main components: the illusion of 3D space, avatars that represent the individual users and communication channels which allow the users to communicate. The CVEs have been designed for different purposes like commercial gaming (e.g. World of Warcraft), socialising (e.g. Second Life) or educational and working environments (e.g. OpenWonderland). (Konstantinidis et al., 2009; Dafli et al., 2009)

According to Dafli et al. (2009), *"the choice of the proper 3D platform for the right e-learning application has become a challenge. There exist four such platforms that may be used in e-learning applications, namely, Second Life (www.secondlife.com), Active Worlds (www.activeworlds.com), OpenSim (www.opensimulator.org) and 3D Torque Game Engine (www.garagagames.com). At the same time, there is the need for an inexpensive and reliable tool in terms of design and implementation of a learning environment"*.

Konstantinidis et al. (2009) presents a two step selection process. In the first step the most popular platforms are evaluated against their collaborative features. In the second step the features of the platform with the highest level of collaborative support are validated against the design principles presented in Bouras, Guannaka, and Tsiatsos (2008) which are:

- *Design to support multiple collaborative learning scenarios*
  A useful collaboration tool has to support the execution of various E-Learning scenarios.

- *Design to maximise the flexibility within a virtual space*
  It should be possible to quickly adopt the learning settings according to the needed functions within a collaborative online session. To improve the educational value, virtual environments must fulfil the teacher's expectations for flexibility.
- *Augmenting user's representation and awareness*
  A combination of gestures, mimics and the different communication channels help the users to demonstrate what they are talking about. Thus the look and behaviour of the avatar must be as realistic as possible.
- *Design to reduce the amount of extraneous load of the users*
  The main goal of the environment is the support of the learning process. Participation in the learning process should be easy for each user. The most important commands should be available on a graphical user interface.
- *Design a media-learning centric virtual space*
  The VW should support different communication channels and media formats in order to enhance the communication between the users.
- *Ergonomic design of a virtual place accessible by a large audience*
  The VW should be accessible and usable by a wide range of individuals no matter which backgrounds or level of expertise in ICT they have.
- *Design an inclusive, open and user-centred virtual place*
  The idea of workspace awareness should, additionally to the knowledge of the other's interaction, also include knowledge of the state of the workspace and the user's own actions.
- *Design a place for many people with different roles*
  A variety of different roles with different access rights should be supported by the E-Learning system. For example in an E-Learning scenario there could be moderators, tutors and learners. Within the VW it should be possible to differentiate the roles.

2.3.4 Limitations and Drawbacks

Adapting existing VW technologies for education and learning does not only have advantages. There are some limitations the current development has to face. In their work Fox et al. (2010) mentioned the following limitations and barriers:

- Due to the fact that a VW is based on broadband network it is limited by bandwidth constraints. Bandwidth constraints, processing limitations and further technological limitations make it difficult to provide stunning visual details and effects.
- Currently there is no such search engine available, that allows for search of content of a VW and there is no metadata standard which allows to make the content searchable.
- One of the biggest limitations of VWs is the user interface. The hand of an avatar is very limited and the progress in haptics has been slow in coming.

- A huge number of graphic formats, identity management and other software technologies which are essential parts of VWs have been developed in parallel. The competing formats are often non-interoperable. In the recent years a few dominant 3D graphic formats have emerged and standard interfaces for powerful physics simulations have been developed. Open source platforms like DSpace[1] or Fedora Commons[2] are used to improve VWs.
- One of the biggest problems is the fact that a big number of non-interoperable VW platforms exist and each new VW wants to become the global player in this field of interest.
- It is not possible to create a profitable business model for putting educational material into VWs.

Few tools have been created to help harness these materials for education and training purposes and no coherent version has emerged to organise, store, peer-review, or deliver them in a consistent and trusted way. Yet developed products are rapidly made obsolete by advancing technologies and often the development is done without reusing existing components. This requires a huge amount of time and could be done much more efficient. (Gerbaud, Mollet, Ganier, Arnaldi, & Tisseau, 2008; Fox et al., 2010)

Zender et al. (2009) found, that VW environments are mostly exclusive and connections to other learning environments are rare. The available solutions are stand-alone with a lack of universal methodologies. (Zender et al., 2009)

Developing a virtual learning world will require a huge amount of work and people with different skills. It would require extensive collaboration and effort to build the needed components, a combination of VWs and Web 2.0 technologies can reduce the needed effort. (Fox et al., 2010). Henckel and Lopes (2010, p. 106) outlined, that *"the critical task of creating 3D objects for a simulation model is still a manual process, which can be time consuming"*.

According to (Gütl, 2011), *"90 percent of commercial VW projects fail partly because they focus on technology rather than on the users‘ needs"*.

2.4 Discussion

This chapter summarised the impact of modern technologies and ICT on learning. New learning paradigms like E-Learning and M-Learning arose with the fast improving technological equipment and because of the new requirements for learning techniques caused by the ever changing political, social, economical, technological and environmental situations.

Some critical requirements for successful distant learning have been identified. Amongst others collaboration, adaptability and reusability are very important characteristics of suitable learning platforms.

[1] http://www.dspace.org/
[2] http://www.fedora-commons-org

A brief introduction of V3DW has been given and the potentials of VWs for collaboration and learning have been introduced. Many different VR solutions have been developed in the recent years. The global commercial players like SL and WoW are pleased about the huge number of active users. Due to the drawbacks of commercial solutions like the costs for land ownership, free available open source platforms have been developed. In the following chapters this work tries to answer the question if such free platforms are serious alternatives for education and collaboration within VEs.

For the practical part of this work, the VW platform OpenSimulator has been chosen. The following chapter introduces the architecture and development features of this simulator solution in detail.

Chapter 3

OpenSimulator

Since the final goal of this work was to build a new flexible learning environment management tool based on the Virtual World (VW) platform OpenSimulator (OS), this chapter takes a closer look at the capabilities of the VW, the support for interaction with so-called off-world components and known limitations and hurdles in OS.

> *"OpenSimulator is an open source multi-platform, multi-user 3D application server. It can be used to create a virtual environment (or world) which can be accessed through a variety of clients, on multiple protocols. OpenSimulator allows virtual world developers to customise their worlds using the technologies they feel work best."* (OpenSimulator Wiki, 2011f)

Figure 3.1: OpenSimulator - image of the moment (OpenSimulator Wiki, 2011f)

Figure 3.1 shows a screen-shot of a virtual space in OS. The platform has been based on the mechanisms and open communication protocols of the 3D virtual environment

Second Life (SL) which has been created by Linden Labs in 2003. SL offers different communication channels like synchronous text chat, visual representation and voice enabled synchronous audio communication. Thus SL is a suitable environment for distant online learning communication. Although SL offers amazing VW platform features, there are numerous drawbacks for its educational use. Due to its commercial purpose, the source code is not available to the public. Further it's not possible to backup the content of a user, there is a lack of restrictions on content, downtimes occur to suit US time zones, a mix of under and over 18s is not possible in the same area and viewer upgrades are forced frequently. (Dafli et al., 2009)

According to Quintella, Soares, and Raposo (2010, p. 46), *"even being a success and the current major reference in terms of virtual environments on the Web, Second Life does not have a good integration with web pages and has much of its infrastructure controlled by the company that created it, which limits the expandability and derivation of this product in new proposals."*

> *"Out of the box, OpenSimulator can be used to simulate virtual environments similar to Second Life, given that it supports the core of SLs messaging protocol. As such, these virtual worlds can be accessed with the regular SL viewers. However, OpenSimulator is neither a clone of Second Life's server nor does it aim at becoming such a clone. On the contrary, OpenSimulator lacks support for many of the game-specific features of Second Life (on purpose), while pursuing innovative directions towards becoming the bare bones, but extensible, server of the 3D Web."* (OpenSimulator Wiki, 2011f)

The major advantage of OS in comparison to SL is the fact that it is open source. The provided functionality can be used and extended. Spending money in booking virtual lands and objects is not required unlike it is in SL. (Dafli et al., 2009)

As identified by Dohi and Ishizuka (2009), the OS server has the following advantages compared to the SL server:

- It is open source.
- Avatars can be controlled event-driven.
- Server and client can run standalone without an Internet connection.
- The official SL server restricts the number of objects and the size of the used scripts.
- The official SL viewer can connect it.

3.1 Architecture

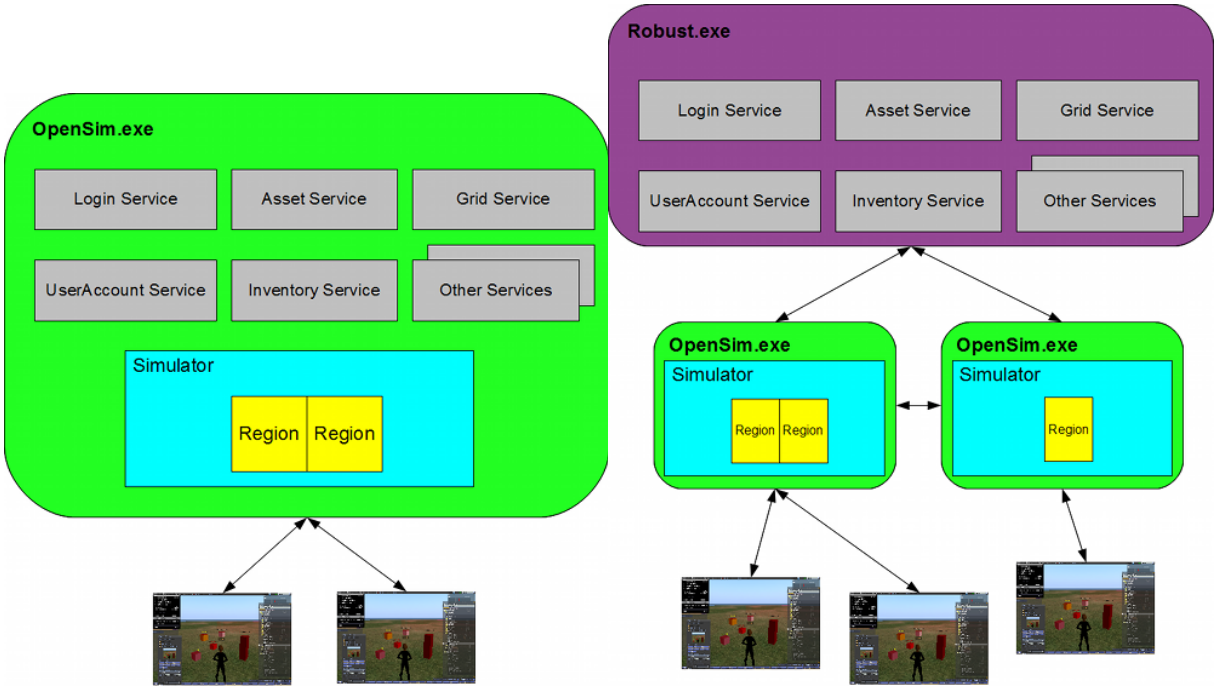
As outlined by OpenSimulator Wiki (2011f), *"OpenSimulator is written in C#, running both on Windows over the .NET framework and on *ix machines over the Mono framework. The source code is released under a BSD License, a commercially friendly license to embed OpenSimulator in products".*

OS can be executed in a standalone mode or can be connected to other OS instances through the build-in grid technology. The standalone mode can be used for off-line development. OS provides a modular plug-in architecture to extend its functionality in order to suit the specific application requirements. (Dafli et al., 2009)

In OS each object has an Universally Unique Identifier (UUID) which means that each user, each primitive, the inventory and all other objects can be identified by an unique value. (OpenSimulator Wiki, 2011e)

3.1.1 Services

The OS server provides five major services which are known as User, Grid, Asset, Inventory and Messaging server (UGAIM). Each region that wants to talk to the OS viewer must provide these services. In standalone mode each region provides its own UGAIM servers which are running in one single process. In grid mode each service runs within a separate process. The processes can be executed on different machines. Figure 3.2 shows these two different architectures. (B. Chen, Huang, Lin, & Hu, 2010; OpenSimulator Wiki, 2011e)



(a) OpenSimulator standalone architecture **(b)** OpenSimulator grid architecture

Figure 3.2: OpenSimulator architecture (OpenSimulator Wiki, 2011a)

Each service has a specific responsibility. OpenSimulator Wiki (2011e) introduces the purpose of the single services as follows:

- *UserServer*
  This service is responsible for authentication of the user to the grid. It creates a

session identifier for the client and assigns an UUID. The session identifier is used to authenticate requests to the other servers in the same grid.

- *GridServer*
  Authenticates regions to the grid and assigns an UUID to them.
- *AssetServer*
  Each new asset gets a new UUID and is immutable. Each modification of an asset results in the creation of a new asset with a new UUID.
- *InventoryServer*
  This service links UUIDs together. It connects users to assets. Since also folders have their own UUID, the server builds a structure by connecting UUIDs.
- *MessagingServer*
  The service provides the communication capabilities of the simulator. It keeps track of who is able to listen to whomever and it manages other message channels. This service is not as essential as the other four services but without the service no communication would be possible.

3.1.2 Regions

As mentioned before the GridServer manages the authenticated regions. OpenSimulator Wiki (2011c) introduces a region in OS as *"what you see when you log into OpenSim. It is the physical place (well, virtual physical space) where avatars move and interact. It is a square patch of land which may contain an island, mountains, a plain, buildings, etc., or just an ocean"*.

As listed by OpenSimulator Wiki (2011e), regions have the following responsibilities:

- Regions run the physics and the scripts.
- They keep track of objects in the scene and any observers connected.
- A region sends scene updates to all of the observers.

The region is a memory space and behaviour simulator. It shares its state with observers. Region servers cache many things to reduce the perceptual lag for the client. (OpenSimulator Wiki, 2011e)

3.1.3 Viewers

OS does not provide a build-in viewer application. Because the OS platform is based on the open communication protocols of SL, compatible SL viewers can be used to access the simulator - OS utilises the SL protocols. The so-called clients perform the three-dimensional rendering of the VW and eventually provide caching mechanisms for the located and rendered objects of the region. (OpenSimulator Wiki, 2011b; Childers, 2009; Varvello et al., 2011)

Various different viewer applications are available on the Web. OpenSimulator Wiki (2011b) provides a list of currently available and supported clients which amongst others are:

- *Linden Client*[1]
  The official SL viewer provided by Linden Labs.
- *Hippo Viewer*[2]
  A fork of version 1.23 of the Linden Client with OS specific enhancements.
- *RealXtend*[3]
  A fork of the 1.23 Linden Client that has specialised features (not fully supported).
- *Cool VL Viewer*[4]
  One of the oldest third-party viewers around (former name: Cool SL Viewer) which is fully compatible to OS.
- *Phoenix Viewer*[5]
  Based on the now discontinued Emerald Viewer (which was based on the 1.23 LL viewer), one of the most popular viewers. Combines many improvements and features from different viewers. Not fully supported by OS.
- *Imprudence*[6]
  Innovative viewer with a high focus on OS (good OSSL support). Highly popular amongst users on OSGrid.

3.2 Content

This section describes how to create content in the VW. It gives an introduction about how to use the basic building functionalities of the viewer applications. Furthermore media streaming concepts of the simulator are described.

3.2.1 Basic Objects

The content of this section is based on the SL Wiki page *Building Tools*[7] which provides detailed information about how to use the SL viewer (version 1.23) to create and manage content of the virtual world. The majority of the available OS viewers are forks of this client and therefore they provide the functionality in a similar way. The viewer provides five different tools to manage the content of the simulator.

[1] http://get.secondlife.com/
[2] http://mjm-labs.com/viewer/
[3] http://opensimulator.org/wiki/RealXtend
[4] http://sldev.free.fr/
[5] http://www.phoenixviewer.com/
[6] http://blog.kokuaviewer.org/
[7] http://wiki.secondlife.com/wiki/Building_Tools

(a) Focus Tool **(b)** Move Tool **(c)** Create Tool

Figure 3.3: Second Life viewer - focus, move and create tool (Second Life Wiki, 2011a)

Focus Tool

The focus tool, as shown in figure 3.3a, changes the camera position in the 3D view. It also changes the mouse pointer to a magnifying glass and it can be used to zoom in and out, rotate around a selected point (Orbit) or to slide the view parallel to the current viewing plane (Pan). (Second Life Wiki, 2011a)

Move Tool

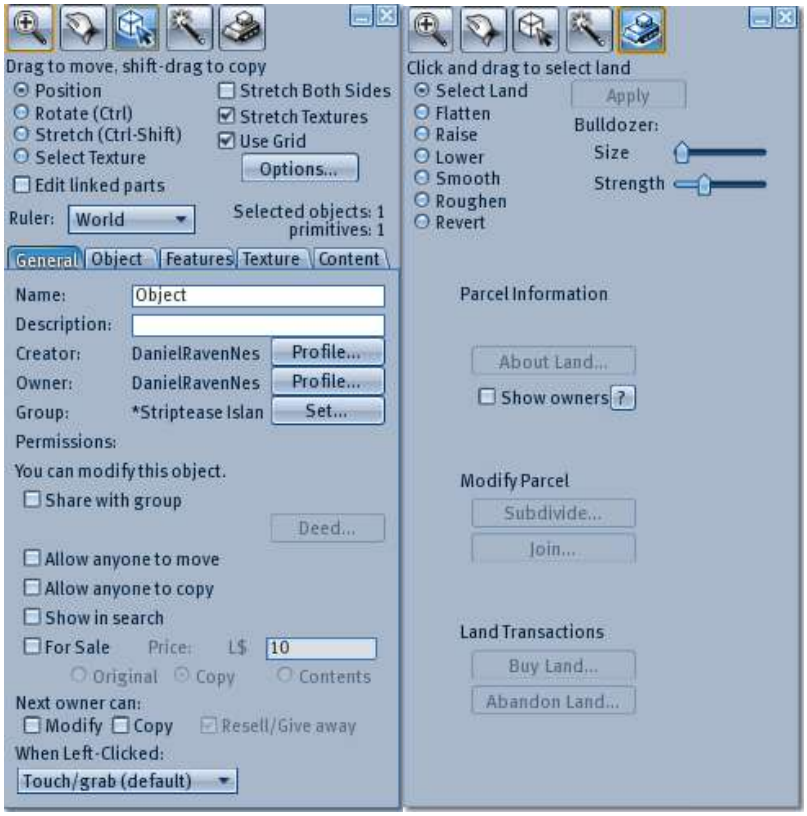
The move tool is used to move an object. As you can see in figure 3.3b, the tool can be used to move the object along the horizontal plane (Move or Spin) or to move the object parallel to the current viewing plane (Lift). Furthermore it can be used to spin the object. (Second Life Wiki, 2011a)

Create Tool

The create tool is used to create basic geometric shapes which are called prims. Figure 3.3c illustrates the tool. By selecting on of the basic shapes and clicking in the 3D view, an object with the selected shape will be created. The size of a new object is 0.5 meter. (Second Life Wiki, 2011a)

Edit Tool

The edit tool can be used to modify the details of an object. As shown in figure 3.4a, the tool offers a lot of functionality which is divided into several tabs. Amongst other functionalities it can be used to modify the size and position of the object, to rotate the object, to manage the textures of the faces of the object, to manage the access permissions, to access the inventory of the object or to manage other features of the object. (Second Life Wiki, 2011a)



(a) Edit Tool **(b)** Land Tool

Figure 3.4: Second Life viewer - edit and land tool (Second Life Wiki, 2011a)

Land Tool

The land tool is used to modify the parcel itself. The terrain map of the simulated world is continuous which means that modifying one part of the world affects also the neighbour parts in order to get a smooth connection of the parts. As illustrated in figure 3.4b several different land operations are available, which are flatten, raise, lower, smooth, roughen and revert. The slider controls size and strength define how much area is affected by one land operation. (Second Life Wiki, 2011a)

3.2.2 Media Streaming

OS provides a streaming mechanism which works like streaming in SL. Streaming is actually not done by the simulator or region servers but by a dedicated streaming server outside of the metaverse. Basically any format which can be played by Quicktime can be streamed in OS. Streaming does not effect the performance of the simulator because it is performed by the streaming service and the OS client. In the settings of the region (World - About Land - Media) the Uniform Resource Locator (URL) of the media can be entered. At the same page the texture on which the content should be streamed can be

chosen. The media will be displayed on all available primitives with the specified texture. Due to this behaviour a special texture has to be used in order to avoid that the media is displayed on other objects. (OpenSimulator Wiki, 2011l)

Scripts can be added to primitives which help to start streaming, load different URLs, view photos an so on. At the OS Wiki page *Streaming Media in OpenSim*[8] the free script *Freeview* is available.

The build-in streaming mechanism has the drawback that just one media can be streamed within one region at a time. The streamed media will be displayed on all primitives which have the configured texture. Thus this mechanism can not be used to stream several different contents at the same time within a region. (OpenSimulator Wiki, 2011l)

3.2.3 Media on a Prim

In order to provide the capability of viewing different media or web contents on the surface of a prim, the new SL viewer 2 provides a functionality called *Shared Media*. According to Second Life Wiki (2011c), the new viewer *"is letting you put Internet content on any prim surface you can change, with new browser-like controls, and without the land restrictions"*. Web-based media content can be added to the surface of any object which the avatar is allowed to modify. (Second Life Community, 2011)

Figure 3.5: Second Life viewer 2 - shared media controls (Second Life Community, 2011)

The viewer provides controls for each object which has a shared media content assigned. Figure 3.5 shows the controls of such a media content. The control buttons are similar to the navigation buttons of a web browser. As described in Second Life Community (2011), the controls have the following functionalities:

- *Back and forward buttons*
  Allows for cycling through previously visited pages.
- *Home button*
  A home URL can be assigned to a shared media content. The home button loads the content of this URL.
- *Refresh and stop button*
  The refresh button causes the shared media object to reload the content of the current URL. During loading operations the button is changed to a stop button. Clicking on the stop button causes the object to abort loading the URL.
- *Address bar*
  Any URL can be entered which will be loaded after pressing enter.

[8] http://opensimulator.org/wiki/Streaming_Media_in_OpenSim

- *Magnifying glass and right arrow button*
  The magnifying glass button positions the camera automatically in front of the media and toggles the control to a right arrow button. The right arrow button returns the camera to the avatar.
- *Window button*
  This button is used to open the current URL in an external web browser.

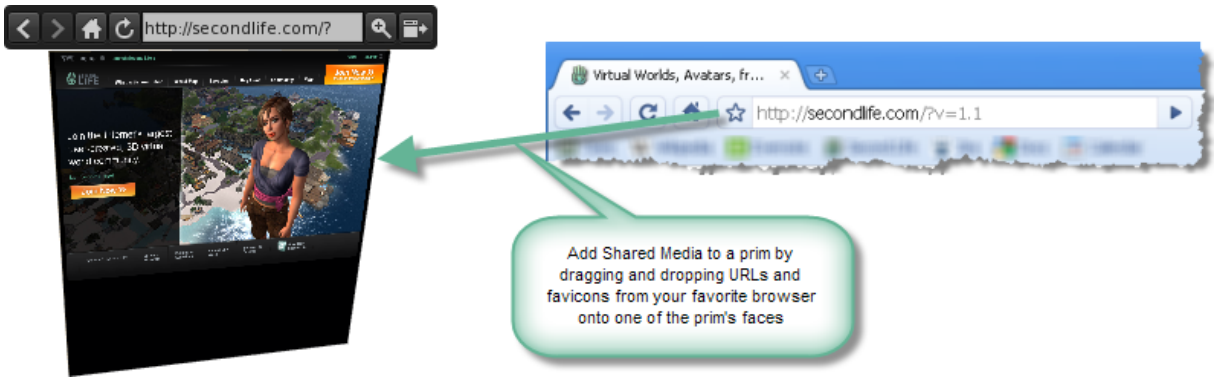


Figure 3.6: Second Life viewer 2 - shared media drag and drop (Second Life Community, 2011)

The shared media objects also support drag and drop which is illustrated in figure 3.6. For example URLs can be dragged from the address bar of a web browser to the media prim.

3.3 Object Permissions

According to OpenSimulator Wiki (2011h), the permission system is based on assigning permission masks to each primitive object. OS uses the LibOpenMetaverse enumeration *PermissionMask*[9] in which the permissions are defined. The permissions Move, Modify, Copy and Transfer can be set in each mask. (OpenSimulator Wiki, 2011h)

As described in OpenSimulator Wiki (2011h), each primitive in OS has the following permission masks:

- *Base mask*
  The base mask specifies the highest amount of permissions a user can have as an owner of an object.
- *Owner mask*
  Specifies the permissions that the owner of the object has.
- *Group mask*
  Specifies the permissions that a group member of the objects group has.

[9] http://lib.openmetaverse.org/docs/0.8/#T_OpenMetaverse_PermissionMask.htm

- *Everyone mask*
  Specifies the permissions that every user has.
- *Next owner mask*
  The owner of the object can change the checkboxes modify, copy and transfer. Depending on these values, the base mask and the owner mask are set when the object is transferred.

The viewer uses the object flags of the primitive to identify its current permissions. The flags are defined by the enumeration *PrimFlags*[10]. The set permissions in the permission masks are automatically set in the flags of the primitive. (OpenSimulator Wiki, 2011h)

3.4 Development

OS provides very flexible extension and customisation mechanisms. This section analyses the features, libraries and tools provided by OS for the purpose of interaction with the off-world and the extension of the basic functionality of the simulator.

3.4.1 Scripting

In SL scripting is very important. The scripting language Linden Scripting Language (LSL) has been invented for scripting in SL. Although it is very slow, it does the job. OS supports the majority of the available LSL functions. A list of currently supported LSL functions is available at the OS Wiki page LSL Status[11]. Additionally to LSL, the scripting languages OpenSim Scripting Language (OSSL) has been introduced and C# scripts are available. In OS the LSL code is translated to .Net code and compiled to native CPU code. This means that LSL scripts are executed much faster than in SL. (OpenSimulator Wiki, 2011k)

New scripts can be added to the inventory (Inventory -> Create -> New Script) and they can be created or modified within the build in script editor of the viewer. Previously written scripts can be added to the prim by dragging it to the content of the object. The viewer displays error information if the script can not be compiled. In order to stop a script, it can be deactivated by unchecking 'Running'. If a script is deleted without prior stopping, it may continue running inside of the prim. (OpenSimulator Wiki, 2011k)

The following sections describe the available functions for scripting in OS.

Linden Scripting Language

As already mentioned, the scripting functionality is based on the scripting concept of SL. There is plenty of documentation available for scripting in SL. A good point to start is

[10] http://lib.openmetaverse.org/docs/0.8/#T_OpenMetaverse_PrimFlags.htm
[11] http://opensimulator.org/wiki/LSL_Status

the LSL tutorial Wiki page[12]. According to Second Life Wiki (2011b), *"a script in Second Life is a set of instructions that can be placed inside any primitive object in the world, but not inside an avatar. Avatars, however, can wear scripted objects"*.

Listing 3.1, which has been copied from Second Life Wiki (2011b), is used to describe the concept of scripting in SL.

```
default
{
      state_entry()
      {
          llSay(0, "Hello, Avatar!");
      }

      touch_start(integer total_number)
      {
          llSay(0, "Touched.");
      }
}

state playing
{
      // this is a state called "playing"
}
```

Listing 3.1: Linden Scripting Language example

Basically each LSL script is based on states and events. Each object has different states in SL, e.g. a door can be open or closed. In SL events are predefined and can be seen as triggers. They can be triggered by objects or avatars which are interacting within the world or by other scripts. A script contains different states and event handlers within these states. The previous example shows a simple script which has an event handler called state_entry for the state default. The script must contain at least one state with one event handler. The state_entry event handler will be executed whenever the object enters the default state. (Second Life Wiki, 2011b)

The SL Wiki page *LSL_Library*[13] provides many LSL scripts which can be used in SL or in OS. Due to the fact that OS supports the majority of all LSL scripts, it is most likely that the available scripts also work without any modification needed. For example the script called *Smooth Sliding Door* has been tested and worked out of the box. The mentioned Wiki page is a good entry point to start learning LSL because the usage of many LSL functions can be found within the scripts.

OpenSim Scripting Language

OS provides additional functionalities as part of the OSSL. The OS Wiki page *Current OSSL Functions Implemented*[14] provides an overview about currently existing OSSL functions. All function names start with the prefix os. LSL and OSSL functions can be combined within scripts in OS.

[12]http://wiki.secondlife.com/wiki/LSL_Tutorial
[13]http://wiki.secondlife.com/wiki/LSL_Library
[14]http://opensimulator.org/wiki/OSSL_Implemented

Listing 3.2 illustrates how LSL and OSSL functions can be used together within a script. The code, which prints a list of all avatars which are currently in the region, originates from OpenSimulator Wiki (2011g).

```
default
{
    touch_start(integer total_number)
    {
        list avatars = osGetAvatarList();
        if (avatars == [])
            llSay(0, "No avatars - just the owner.");
        else
            llSay(0, "Avatars in this sim: " + llList2CSV(avatars));
    }
}
```

Listing 3.2: OpenSim Scripting Language example

3.4.2 Plugins and Region Modules

In order to provide a standardised mechanism to dynamically load modules, the OS developers wanted to use a reusable, cross-platform dynamic assembly loader that could be used as-is. The only candidate found was Mono.Addins. (OpenSimulator Wiki, 2011d)

> *"Mono.Addins is a framework for creating extensible applications, and for creating libraries which extend those applications. Mono.Addins has been designed to be easy to use and useful for a wide range of applications: from simple applications with small extensibility needs, to complex applications which need support for large add-in structures. This new framework intends to set an standard for building extensible applications and add-ins in Mono."* (Mono Project, 2011)

As listed in OpenSimulator Wiki (2011d), Mono.Addins offers the following features:

- Logically splits up module interfaces from implementations in a cross platform way through the use of text-based extension points and addin manifests.
- Lazily loads assemblies only when finally necessary.
- Tracks module dependencies.
- Allows to customize the addin manifest Extensible Markup Language (XML).
- Automatically handles module sharing across application domains.
- Provides support for downloading assemblies from remote repositories.

The Mono.Addins framework is used to dynamically load extensions in OS. Such extensions are application plugins and region modules.

OpenSim Application Plugins

Application plugins act over the whole server. They can be developed by implementing the provided interface *IApplicationPlugin*. During initialisation an instance of the class *OpenSimBase* is passed over to the application plugin, which allows the plugin to access the core functionalities of the server. (Justin Clark-Casey, 2008)

With the exception of region modules, all developed extensions are called plugins and are inherited from the same base class. The class PluginLoader is a small Mono.Addins wrapper which is responsible for loading of the available extensions. (OpenSimulator Wiki, 2011d)

OpenSim Region Modules

As the name already says the region modules act on the level of regions. Region modules are Dynamic Link Librarys (DLLs) which are implemented in C#. They are executed within the heart of the simulator and offer access to all of the simulators facilities. Region modules can subscribe themself for different events like chat messages, login of users or texture transfers. Such modules are sets of classes that implement the interface *IRegion-Module*. The modules cover core functionalities like chat mechanisms and extension functionalities like dynamic texture loading. (Henckel & Lopes, 2010; OpenSimulator Wiki, 2011i; Justin Clark-Casey, 2008)

Region modules are loaded during the start-up of the OS server. The loaded modules remain active until the server shuts down. Region modules are very flexible and can execute a variety of tasks like creating or modifying objects, register for events and so on. The event registration mechanism allows the region module to respond to specific events. (Henckel & Lopes, 2010)

Since version 0.6.9 the new region module mechanism is in place, which separates the modules in shared and non-shared modules. Non-shared modules are modules which are created for each region. Shared modules are created once within a simulator which means that all regions share such a module instance. (OpenSimulator Wiki, 2011i)

3.4.3 OpenMetaverse Library

The OpenMetaverse library is a collection of .NET libraries for interacting with V3DW simulators. It is developed by the Open Metaverse Foundation and provides the protocol, networking and client functionalities. (LibOpenMetaverse Developer Wiki, 2011)

> *"The Open Metaverse Foundation is a non-profit organization founded with the mandate of developing open technologies and open-source software related to the metaverse and virtual worlds. The OMF provides support and guidance to selected projects that espouse the principles of the foundation."* (OpenMetaverse Foundation, 2011b)

The library is a .Net based client-server library that is used for creating and accessing 3D virtual worlds. It is compatible with the SL protocol, thus it can be used for creating clients and automations in SL, OS and other V3DWs which are compatible to the SL protocol. (OpenMetaverse Foundation, 2011a)

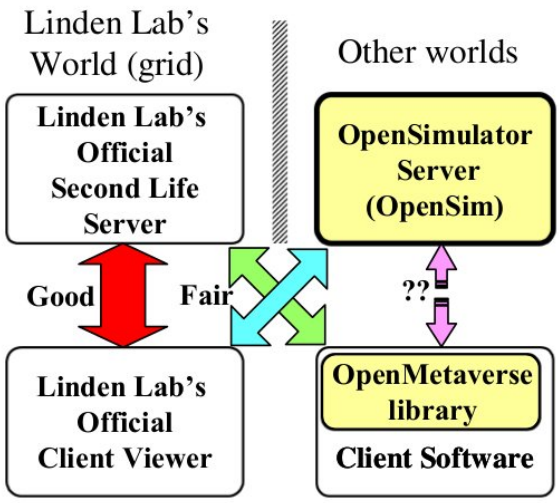


Figure 3.7: OpenSimulator server and OpenMetaverse library (Dohi & Ishizuka, 2009, p. 184)

According to Dohi and Ishizuka (2009, p. 184), OpenMetaverse Library is *"another open source project in order to study how Second Life works. It is also used to develop original client software for the Second Life official server"*. Both OS and the OpenMetaverse library are still an alpha version and have many problems due to their development status. As illustrated in figure 3.7 it is not recommended to combine OS and OpenMetaverse library for a system that should be reliable. (Dohi & Ishizuka, 2009)

Dohi and Ishizuka (2009) mentioned, that for research purposes the combination of the projects has the following advantages:

- Internal code can be accessed and modified.
- No high-speed network Internet connection is needed and the system can be executed standalone.
- It is free of charge.

3.4.4 Remote Access

In order to access the data of the simulator, an Application Programming Interface (API) based on the Representational State Transfer (REST) specification is provided by OS. REST is based on the Hypertext Transfer Protocol (HTTP), it mainly uses the methods GET, PUT, POST, and DELETE. The nature of the payload is not specified by REST. The GET method is used to retrieve the content of a resource, POST adds the payload

of the request as a new entity to the simulator, PUT adds the payload as a new entity if it does not already exist or updates the entity if it does exist and DELETE requests the server to delete the resource identified by the given Uniform Resource Identifier (URI). The OS implementation of the API uses XML as payload format. (OpenSimulator Wiki, 2011j)

To enable the REST interfaces, the server configuration has to be modified. Listing 3.3, which is part of the simulator configuration file *OpenSim.ini*, shows the area of the configuration which belongs to the REST implementation.

```
  [RestPlugins]
     ; Change this to true to enable REST Plugins. This must be true if you wish to use
     ; REST Region or REST Asset and Inventory Plugins
     enabled = true
     god_key = SECRET
     prefix = /admin

 [RestRegionPlugin]
     ; Change this to true to enable the REST Region Plugin
     enabled = true

 [RestHandler]
     ; Change this to true to enable the REST Asset and Inventory Plugin
     enabled = true
     authenticate = true
     secured = true
     extended-escape = true
     realm = OpenSim REST
     dump-asset = false
     path-fill = true
     dump-line-size = 32
     flush-on-error = true
```

Listing 3.3: OpenSimulator - Rest-Plugin configuration

The class *RestHandler* is an application plugin and thus automatically initiated during the start-up of the server. It searches for classes which implement the interface *IRest* and adds all these classes as request handlers. This means, that in order to add a new handler for a specific REST request, a class has to be created which implements this interface. Figure 3.8 shows the basic architecture of the REST integration.

In the latest available release of OS, which was 0.7.0.2, some REST request handlers were provided by the simulator. Handlers for asset, file, appearance and inventory handling have been available but they were not stable or like the inventory and appearance handler, not working at all. Basic user authentication is supported by the OS server. Secure connections are not supported in the used release. The interface *IRest* is internal which means that it is not possible to add a new request handler by adding a new DLL. The class must be added to the core of the project. (OpenSimulator Wiki, 2011j)

As found out by analysing the source code of the project another approach is to extend the provided class *RestPlugin*, which implements the interface *IApplicationPlugin*. Handler methods for each combination of method and request URI can be subscribed. Authentication and secure connection handling is not provided out of the box.

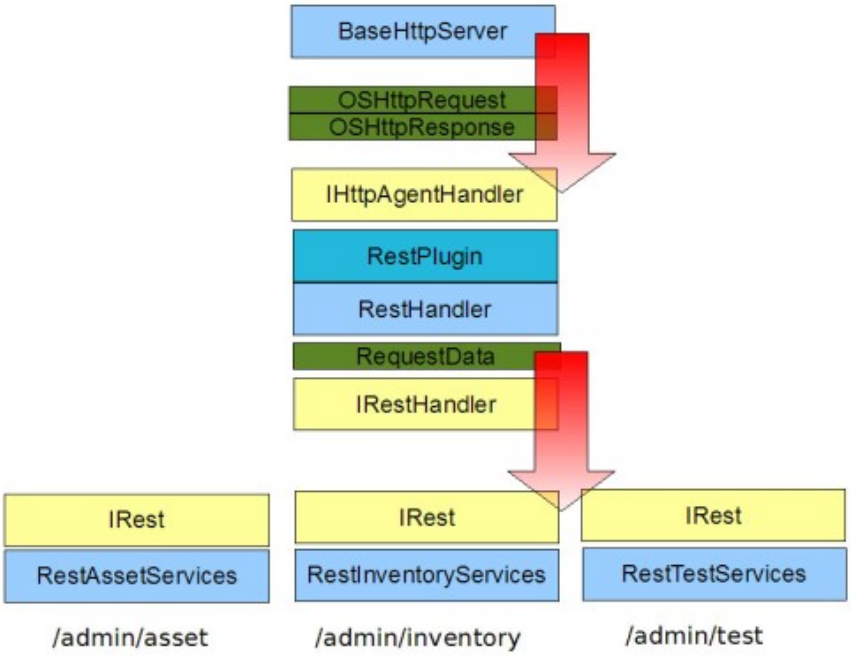


Figure 3.8: OpenSim REST service data flow (OpenSimulator Wiki, 2011j)

3.5 Limitations

The research of the development features of OS has shown powerful and flexible extension mechanisms. Nevertheless the work has also uncovered some current limitations of the simulator which are:

- According to Dafli et al. (2009), *"OpenSim and LSL are in a very mature stage and that means that there is no support for all the available functions of LSL. There is a way of implementing new functions of LSL, however, it is complicated and only addressed to experienced programmers"*.
- OS is still in alpha status and therefore not yet suitable for productive environments. (OpenSimulator Wiki, 2011f)
- The build-in streaming functionality does not allow to stream several different media contents at the same time within a region.
- The REST integration does not seem to be fully developed and usable out of the box.

3.6 Discussion

One of the main goals of the OS server was to be easily extendible. By now the concept to extend the server is to simply implement an interface and add the new class within a DLL to the directory bin of the server. For example region modules or application plugins can be added in this way. This approach has the advantage that it is a very easy task to extend the functionality of the server. One of the drawbacks of this approach seems to be the fact, that it is not possible to configure whether a class will be loaded and used or not. In order to disable a class, the class has to turn itself of (e.g. by configuration of enabled parameters) after initialisation, which seems to be a bit circuitous.

Because of the ported scripting capabilities of LSL and the additional added OSSL functions, OS offers a powerful scripting library which allows the user to customise the behaviour of the created environment. Due to the compatibility to the scripting functions in SL, plenty of documentation and already existing, free to use, source code is available.

The project is still in alpha status which means that the code is changing rapidly and that the implementation is not stable. For example some REST handlers are provided out of the box, but they are not working reliable and the HTTP session can not be secured out of the box.

Because of the possibility to extend the simulator and the free availability of the project, the decision has been made to use OS as the platform for the flexible learning environment prototype which has been developed in the context of this work. The next chapter reviews current research topics which are related to learning in V3DWs and extension approaches of OS.

Chapter 4

Related Work

Earlier in this work, the potentials of V3DWs in the context of learning have been pointed out. Because of the substantial progress of modern ICT, new types of learning like E-Learning and M-Learning arose. Research has shown, that VEs provide powerful features like improved ways of collaboration which can enhance distant education. This chapter analyses current research approaches in the context of education in VWs.

Chapter 3 analysed the capabilities of the open source V3DW OpenSimulator. Apart from the alpha status of the project, the advantages of the open source platform in comparison to commercial products like SL led to the decision to implement a flexible learning setting management tool on top of OS. Thus research approaches which deal with the extension of the OS server have been considered as well later on in this chapter.

4.1 SecondLife Room Manager

During his Master's Thesis, Freudenthaler (2011) has developed a prototype based on the V3DW SL. The goal of the prototype was to create a platform which allows to easily create and manage learning environment settings in the V3DW. According to Gütl et al. (2010) the prototype should meet the following requirements:

- It should be easy to configure, store and reuse learning settings.
- The user interface should be intuitive.
- It should provide mechanisms to extend the library of available learning tools and 3D objects.
- The in-world instantiation of the learning space should be simple and it should provide the functionality to save and restore the state of the settings and the objects within the settings.

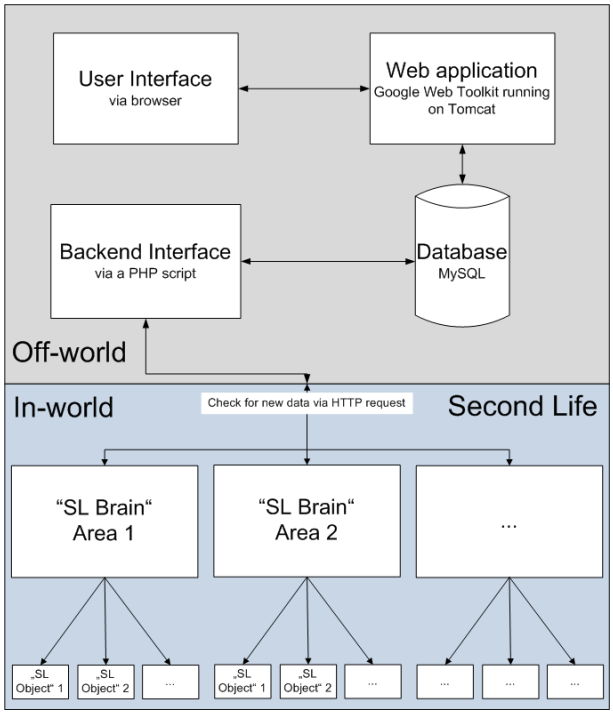


Figure 4.1: SLRoomManager - conceptual architecture (Freudenthaler, 2011)

4.1.1 Architecture

Figure 4.1 shows the conceptual architecture of the implemented prototype. The prototype consists of an off-word and an in-world part. The so-called off-world part, which is implemented as a Web application, is the management application which is used to configure and save the learning settings. The purpose of the in-world part is to create and communicate with the SL objects. To create and control the learning settings within SL a central object called SL Brain has been created. This object retrieves the instructions by usage of HTTP-Requests. (Gütl et al., 2010; Freudenthaler, 2011)

4.1.2 Findings

According to Gütl et al. (2010) the Google Web Toolkit (GWT), which has been used to implement the user interface, is an easy to use framework with a well represented community. SL offers the LSL which is a scripting language within the virtual world. For a skilled programmer the LSL is easy to use and a lot of documentation is available, such as in-word places. (Gütl et al., 2010)

Gütl et al. (2010); Freudenthaler (2011) found in their work the following hurdles and restrictions when it comes to programming in SL:

- SL does not support offline development. Access to the SL land is required to interact with the SL server.

- Object movement actions are very limited. The distance which an object can be moved is restricted by a maximum of 10 meters. Workarounds like stopovers every 10 meters have to be implemented. If an object is moved to an unknown coordinate, the object disappears. To rez (the creation of a virtual object) an object causes time delays which limits the number of rez operations that can be performed.
- In SL each avatar and item has its own SL objects which are called the inventory. In order to move or rez an object it must be in the inventory of the root object.
- Interaction with or access of a single object in SL is just possible by adding a script to the object which means a considerable effort for the programmer.
- The simulation for each user is executed on an own instantiation of the server. Executing actions with many scripts or performing actions on popular islands can cause delays or result in lost data.
- Objects are able to communicate via communication channels. Each communication channel in SL must have its own unique channel ID. It is not guaranteed that no other script which is executed on the same island uses the same communication ID. Using a communication ID which is used by an other script causes transmission faults or the other script can listen to the communication on the specified channel.
- Uploading media like images, sounds or animations as well as ownership of land in SL is not free of charge.

4.2 SLOODLE

Recent approaches try to combine the features of LMSs and V3DWs. The improved interaction capabilities of MUVEs and the content-management qualities of LMSs should lead to platforms which benefit from the advantages of both sides. One such approach is Simulation Linked Object Oriented Dynamic Learning Environment (SLOODLE). (Leidl & Rößling, 2007)

Modular Object Oriented Dynamic Learning System (MOODLE) is an open source E-Learning platform which offers the user the functionality to create and manage teaching material. Those management tools currently don't exist in most VWs. (Callaghan et al., 2009b)

4.2.1 Architecture

According to Callaghan et al. (2009b), *"the SLOODLE open source e-learning software project offers the functionality to link SL and MOODLE and to exchange and synchronise data flow between the two environments"*. Interaction with external applications can be achieved by the usage of LSL. With this approach learning settings in MOODLE can be extended by the usage of virtual 3D world practises. The actions can be automatically

recorded in MOODLE for later review by students or teachers. Figure 4.2 shows the architecture of this approach. (Callaghan et al., 2009b)

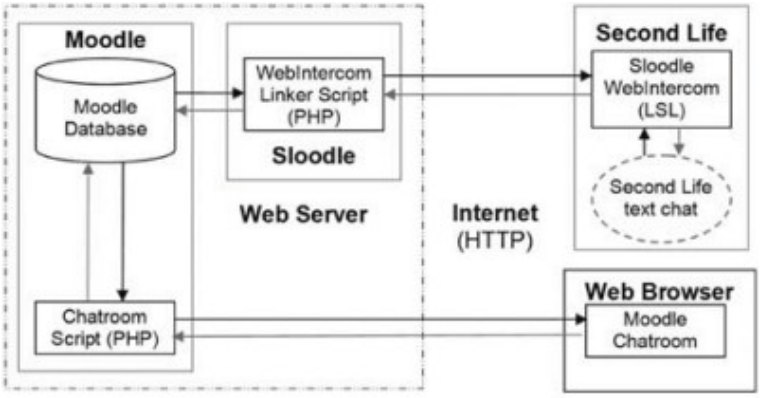


Figure 4.2: Architecture of Sloodle (Callaghan et al., 2009a)

The users can login to the MOODLE system and select a practical task. The details of the task are presented to the user. Then the user can login to SL and teleport to the location of the practical task. Each of this steps is recorded in MOODLE. (Callaghan et al., 2009a)

SLOODLE Blog (2011) provides the following list of features which are provided by SLOODLE:

- *Web-intercom*
  Connects the SL chat and the chat-room of MOODLE.
- *Registration booth*
  Links the avatars of the students to their MOODLE user accounts.
- *Quiz tool and 3D Drop Box*
  Quizzes and 3D modelling tasks are performed in an engaging 3D environment. Reviews and grades can be found in the MOODLE gradebook.
- *Choice tool*
  Students can vote in SL as well as in MOODLE.
- *Multi-function SLOODLE tool-bar*
  Enhances the user interface of SL with classroom gestures and other functionalities like the possibility to write notes which are added to the MOODLE blog.
- *Presenter*
  MOODLE slides and web-pages can be accessed within SL.

4.2.2 Findings

Callaghan et al. (2009b, p. 299) conclude in their work, that *"it is possible to integrate virtual learning environments and virtual worlds to harness relative strengths of each plat-*

form e.g. the course management features of virtual learning environments and the immersive/highly interactive nature of virtual worlds to create engaging learning experiences for students".

The project provides an interface for a VE to control another VE. It does not provide face-to-face lectures which are essential components of immersive learning. (Zender et al., 2009)

4.3 Synchronous Role-based Collaborative Learning

In their work, De Lucia et al. (2009, p. 1025) present *"a system supporting synchronous collaborative learning by naturally enriching Learning Management System services with meeting management and multimedia features. Monitoring and moderation of discussions are also managed at a single group and at teaching level".*

Developed MOODLE plug-ins have been integrated in SL and SL objects have been implemented to support synchronous role-based collaborative activities. Multimedia support has been enhanced with functionalities for navigating multimedia contents. The goal of the work was to integrate SL and the MOODLE platforms in order to enrich the SL environment with communication and management features of MOODLE. Students can select configured learning settings in MOODLE and are automatically teleported to the SL room. Several tools are used to enforce group collaboration like interactive boards. (De Lucia et al., 2009)

Figure 4.3: SecondDMI - virtual classroom (De Lucia et al., 2009)

SecondDMI is a learning environment in SL build by the Department of Mathematics and Informatics at the University of Salerno. It supports didactical experiences in a real classroom replication. Figure 4.3 shows a virtual classroom and a detail of the slide presenter. During the lectures students can exchange text messages and hear the voice of the

presenter. SecondDMI has been equipped with collaborative areas which allow groups to discuss, visualise and record information. (De Lucia et al., 2009)

4.3.1 Architecture

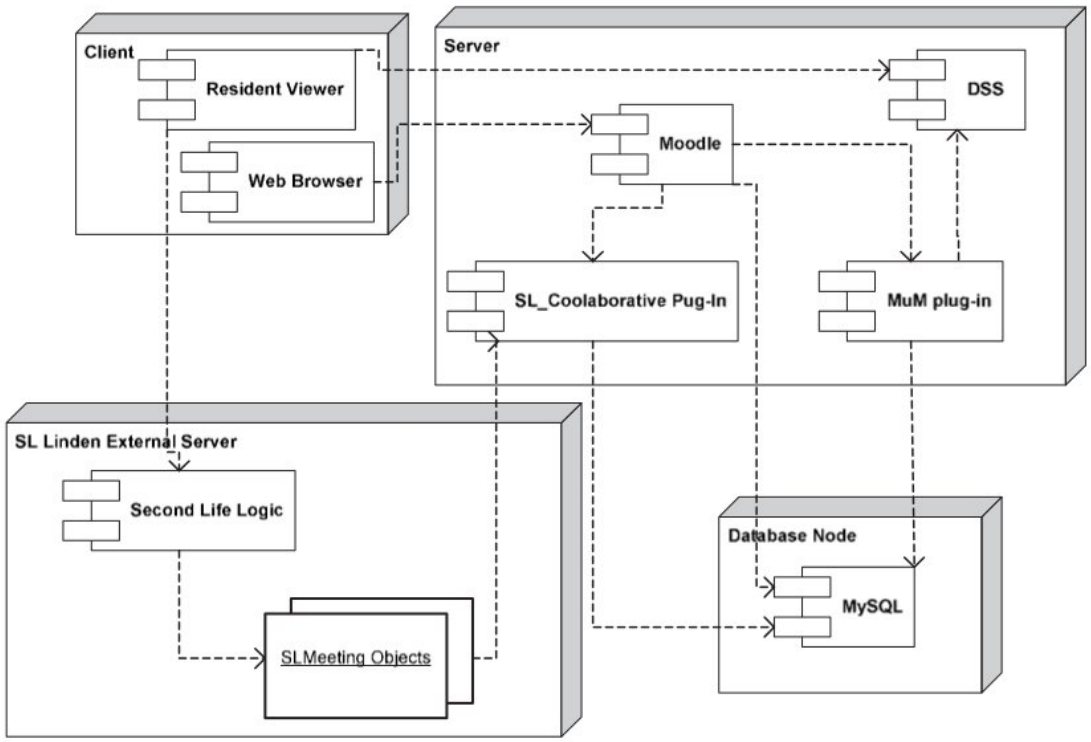


Figure 4.4: SecondDMI - integration architecture (De Lucia et al., 2009)

Figure 4.4 illustrates the basic architecture of the approach. The developed SL_Collaborative plug-in saves the created knowledge conversations and decisions automatically for later reference. The SL objects send the data by using HTTP requests. The functionality of the SL objects is developed by using LSL. The Multimedia MOODLE plug-in (MuM) provides streaming capabilities to both MOODLE and SL. (De Lucia et al., 2009)

4.3.2 Findings

According to De Lucia et al. (2009, p. 1052), *"the proposed system benefits of all the awareness advantages provided by this kind of environment, while providing users with a structured and navigable course content"*. An experiment has shown, that the learning performance obtained with direct face-to-face collaborative environments compared to the proposed setting in SL does not significantly differ. (De Lucia et al., 2009)

4.4 Immersive Learning Prototype

In their work, Zender et al. (2009) analysed the benefits of Immersive Learning in the context of systematic and flexible interconnection of virtual learning and computer-aided face-to-face learning. This fusion leads to learning scenarios that are independent from time and location. Those scenarios adapt to learners and environments with a seamless transition between both worlds. Classroom and virtual learning can be interconnected in order to provide a higher level of flexibility to the user. (Zender et al., 2009)

Zender et al. (2009) found, that previous approaches are unidirectional. The material comes from the VW while learners interact from real classrooms. Face-to-face learning material has not been used as input to enrich virtual lectures. Face-to-face lectures in VWs just provide the voice of the lecturer to the participants. The integration of additional media sources as live lecture slides and a video stream of the lecturer is required. (Zender et al., 2009)

In the integration, the biggest challenge is communication. Questions, discussions and learning material interchange also has to be integrated. Previous work has shown the demand of a concept to integrate existing messaging technologies. The available solutions lack of generality. A systematic fusion concept to handle the diversity of available tools, platforms and content is needed. (Zender et al., 2009)

4.4.1 Architecture

The work is build on a service-based communication concept. An exemplary implementation for multimedia and message exchange between virtual and face-to-face environments has been created. A flexible network architecture to combine all tools, platforms and the processes has been developed. The network should be applicable for scenarios which involve different educational providers. (Zender et al., 2009)

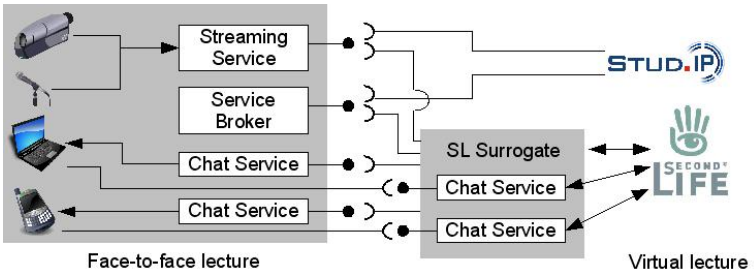


Figure 4.5: Basic architecture (Zender et al., 2009)

The basic Broker architecture model has been found as best suitable due to its scalability, fault tolerance and performance. The requirement of priori knowledge on any E-Learning instance is avoided by this architecture. On the basis of service description (e.g. video streaming of a given lecture) the broker dynamically determines the best suited service of a respective provider (e.g. lecture hall) to fulfil a given request of a consumer (eg. VW).

The user decides what is, where it is and when it is to do and the technical infrastructure supports his activities. It selects and provides the most suitable services available. (Zender et al., 2009)

The implemented prototype connects a face-to-face lecture room with SL and the E-Learning platform Stud.IP. Figure 4.5 illustrates the basic architecture. Web Services are used to connect the components. Because of the use of Service Technology-independent Language (STiL), the prototype is adaptive to multiple Service Oriented Architecture (SOA) technologies. (Zender et al., 2009)

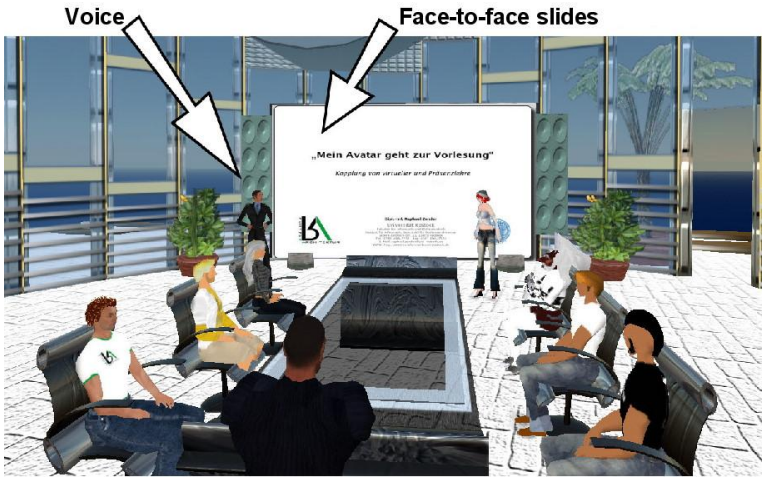


Figure 4.6: Immersive Learning Prototype - learning integration (Zender et al., 2009)

Figure 4.6 shows the result of the recorded face-to-face lecture in SL. Video and voice of the lecturer, the slides of the lecturer, video of the audience as well as recorded video and audio material is available to the participants. (Zender et al., 2009)

4.4.2 Findings

As outlined by Zender et al. (2009), the development of the prototype, which connects face-to-face lectures and E-Learning platforms, has uncovered the following facts:

- The coupling of virtual learning and computer-aided face-to-face learning enriches both environments in terms of pervasive learning.
- SOA is a powerful and well-promising architecture model for pervasive computing applications.
- Because of the loose coupling of SOA, the learning environment can be dynamically modified during the lecture.
- A direct integration of SOA mechanisms into VWs would increase the flexibility.

4.5 TwinSpace

Reilly et al. (2010, p. 119) introduce TwinSpace as *"a flexible software infrastructure for combining interactive workspaces and collaborative virtual worlds. Its design is grounded in the need to support deep connectivity and flexible mappings between virtual and real spaces to effectively support collaboration"*. TwinSpace allows rapid prototyping of applications that span physical and virtual spaces. It provides a flexible infrastructure for exploring and implementing Collaborative Cross-Reality (CoXR) environments. (Reilly et al., 2010)

According to Reilly et al. (2010), their contribution is a novel infrastructure that provides the following four key features:

- *Communication layer*
  Seamlessly links the transaction mechanisms and event-notification of the virtual and physical space together.
- *Common model for physical and virtual spaces*
  Promotes the interoperability of physical and virtual devices, services and representations.
- *Mapping capability*
  Manages how physical and virtual spaces are connected and synchronised.
- *Specialised virtual world clients*
  Clients that participate fully in the larger ecology of collaborative cross-reality devices and services.

TwinSpace provides a generic systems model for CoXR. The sensors and devices in physical rooms can be synchronised to the services offered by the VW interface. Dynamically reconfiguration of the integrations between the spaces is supported by TwinSpace. Multiple interactive rooms can be connected to a virtual space simultaneously. The rooms can map to separate virtual regions or to same regions. (Reilly et al., 2010)

4.5.1 Architecture

Figure 4.7 shows the basic architecture of TwinSpace. OpenWonderland has been chosen as VW platform because of its rich, programmatic control of the VW, the ability to use desktop applications in-world, event-based notifications and well-defined extension points on server and client side. (Reilly et al., 2010)

TwinSpace connects with the VW through a custom service called Realm Bridge which is implemented by using the OpenWonderland API. A one-to-many relationship between VWs and physical space is supported. The Observer-Effector paradigm is used to synchronise VW events with real world events. (Reilly et al., 2010)

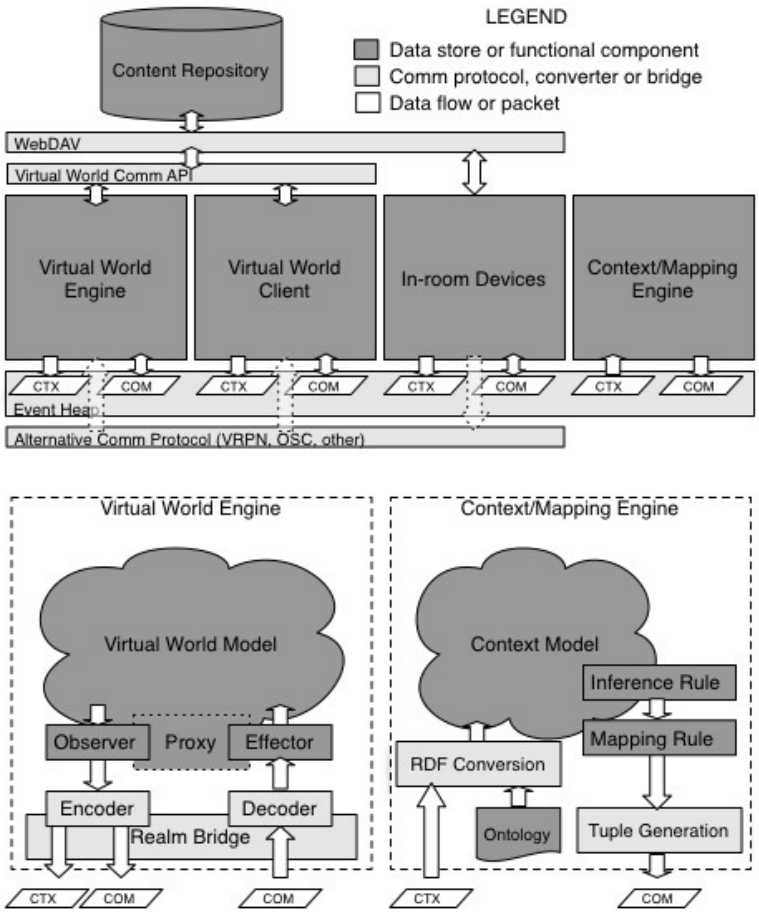


Figure 4.7: Architecture of TwinSpace (Reilly et al., 2010)

4.5.2 Findings

In their work Reilly et al. (2010) identified some relevant characteristics of CoXR systems and OpenWonderland in the context of synchronisation of virtual and real world scenarios which are:

- CoXR systems require more direct access to the VW model than is typically available on a client.
- OpenWonderland provides rich support for document creation in the client API but very little support on the OpenWonderland server.
- In OpenWonderland the server-side transaction and synchronisation services constrain the creation and management of threads.
- Synchronising virtual and real world events is complicated by the fact that the generated events differ.
- It can be difficult to address physical and virtual devices in the same way.

4.6 iSocial

According to Schmidt, Galyen, Laffey, Ding, and Wang (2010, p. 45), the project iSocial *"is developing a 3D virtual environment for youth with autism spectrum disorders to develop social competence"*. iSocial is a 3D, Internet-based, VLE to support social and behavioural outcomes for youth with Autism Spectrum Disorders (ASD). In their work Schmidt et al. (2010) illustrate how Free and Open-Source Software (FOSS) due to its flexibility and community features fits with the iterative nature of Design-Based Research (DBR). The purpose of the development was to expand access to specialised training for developing social competence. (Schmidt et al., 2010)

Schmidt et al. (2010, p. 46) introduced Design-Based Research (DBR) as a *"theory-driven design, wherein the goal is not only the iteration of a product but also the advancement of a design theory for optimal learning and performance within a naturalistic context, usually in relation to the use of technology. In addition DBR addresses specific, complex, and important educational problems by systematically testing designs in context with each implementation and analysis informing the next iteration of the design theory"*. DBR requires the flexibility and ability to control iterations. Thus FOSS seems to fit the requirements of DBR. (Schmidt et al., 2010)

The VW has been build on top of Open Wonderland. Three categories of customisations had to be done: environmental, social and curricular. (Schmidt et al., 2010)

4.6.1 Findings

During their work on iSocial, Schmidt et al. (2010) analysed the potential of FOSS in the context of learning. The following listing summarises their findings:

- FOSS accommodates the needs of DBR to agilely revise, adapt, make changes and re-implement to fit the target context.
- FOSS provides opportunities for designers, developers and users to participate in the community development effort.
- In the development of V3DWs, FOSS is gaining popularity because of the flexibility, customisability and extendibility of the platforms.
- Due to the diversity in implementations of FOSS platforms and the need for highly knowledgeable staff, FOSS software solutions also bring some challenges.
- *Open Wonderland* supports open file formats and multiple tools to create VWs.
- Learning in a V3DW is the ability to have an intrinsically motivating environment.
- In the case of *iSocial* the *Open Wonderland* toolkit allowed for customisation in order to fit the needs of the target population.
- The pairing of FOSS and DBR can be a flexible and powerful method for educational research and development.

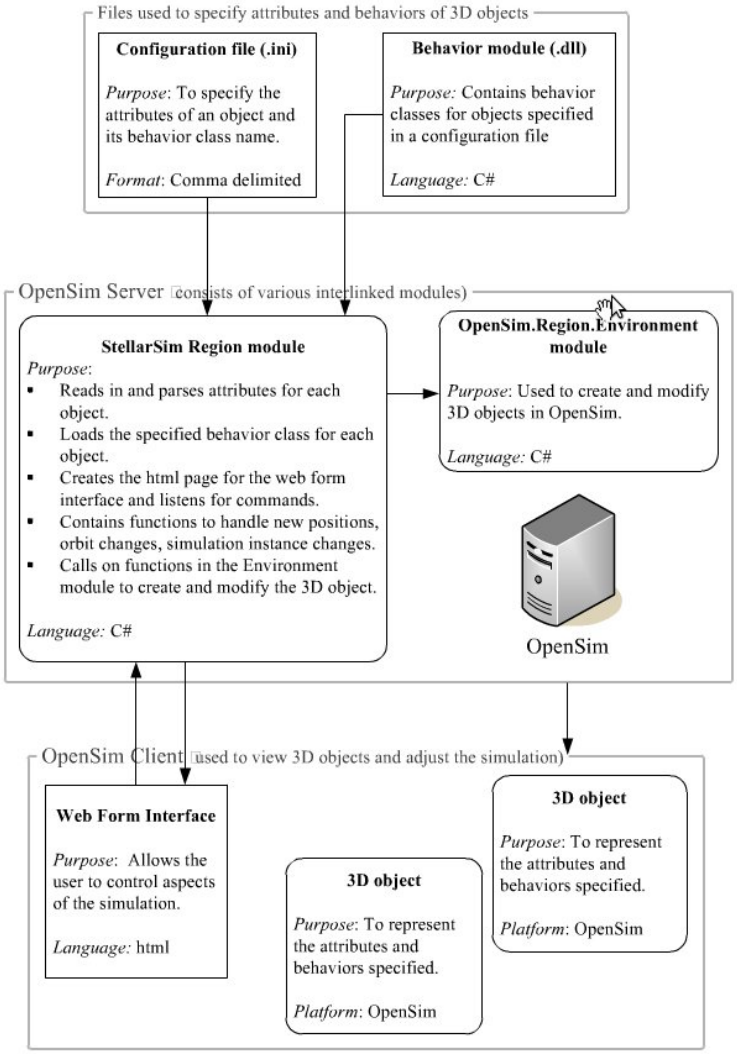


Figure 4.8: Architecture of StellarSim (Henckel & Lopes, 2010, p. 112)

4.7 StellarSim

Creating 3D objects for a simulation model in virtual worlds is a manual process which is time consuming and inflexible. 3D objects must be created and customised manually, textures must be appointed to the objects, scripts have to be written to assign specific behaviour and movement. Programs in use for 3D modelling and simulations by a group of astrophysicists have been analysed and none of them allows automatic population of 3D objects. In their work Henckel and Lopes (2010) present a plug-in architecture framework called *StellarSim* that allows automatic creation of 3D objects. The framework provides methods for input of customised attributes and the assignment of independent behaviour to 3D objects. Their focus was to minimise the effort for the user by importing customised attributes and behaviours of an object into the virtual environment. (Henckel & Lopes, 2010).

4.7.1 Architecture

Because of the already mentioned limitations of *SL*, Henckel and Lopes (2010) based their work on *OpenSimulator*. The framework provides a plugin-architecture to apply the attributes and behaviours of the objects from external files. The attributes and behaviours are specified in configuration files and C# classes. Those files are compiled into DLL files which are loaded and executed by the *StellarSim* framework. The benefits of this approach are that it allows for greater flexibility in the modelling of the objects and the systemic utilisation of the underlying virtual world platform across a variety of applications. (Henckel & Lopes, 2010)

A *OpenSim Region Module* has been developed which reads in attributes for new objects and loads behaviour modules which calculate the position of the objects. It uses modules provided by *OpenSim* to create the 3D objects and to update their position within the virtual world. Figure 4.8 shows the architecture of the framework. (Henckel & Lopes, 2010)

The *StellarSim* module registers for http request events. A web form interface sends a http request which causes the region module to call the appropriate functions. Thus the end user can control the objects within the simulation, for example the position of the object can be set. (Henckel & Lopes, 2010)

4.7.2 Findings

According to Henckel and Lopes (2010), their work with OpenSimulator in order to automatically create and mange objects in the virtual space led to the following findings and results:

- OS includes a direct access to the backend of the VW server for dynamic additions and modifications of 3D objects. Further it offers the use of the system timer for orbit simulations and the registration of events.
- By using the region module approach of OS, the effort to create and modify objects is relatively small.
- StellarSim provides a mechanism for automatic 3D population and ad hoc modifications of the objects. The framework generates an application independent of the objects being modelled. Thus the application can support several settings.

4.8 Virtual Campus of the Chinese University of Hong Kong

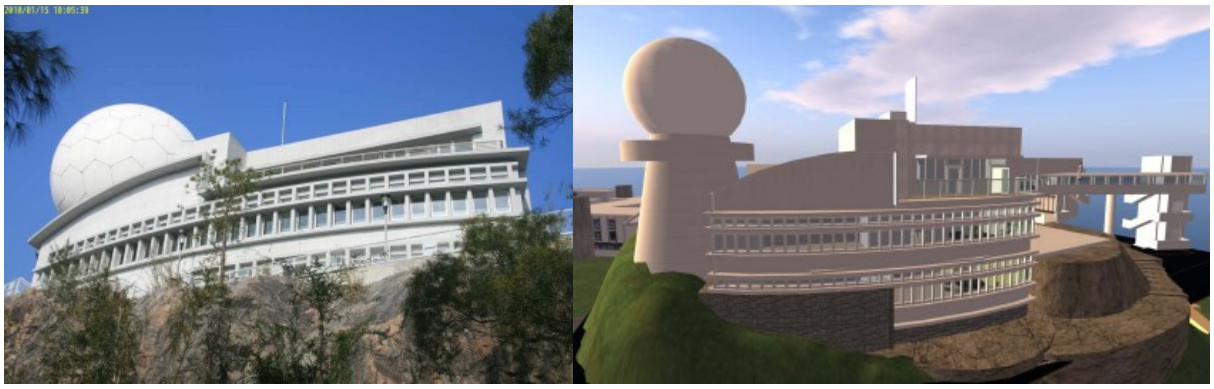
In their work, B. Chen et al. (2010) introduce the Virtual Campus of the Chinese University of Hong Kong (VCUHK) project which builds a 3D virtual campus. The goal of the

project was to integrate the real campus of the Chinese University of Hong Kong into a 3D immersive networked VW. The virtual university should be hosted by the university's own VW servers. (B. Chen et al., 2010)

> *"VCUHK is an ongoing project aiming to explore the techniques how to design a 3D-space-realistic networked virtual environment and how to integrate the real campus to an educational virtual world."* (B. Chen et al., 2010)

According to B. Chen et al. (2010), the construction of VCUHK includes several steps:

- Integration of the hilly terrain into the VW.
- Modelling of the main buildings and processing of the photo-realistic textures.
- Construction of the landscape of the campus.
- Construction of the inner structure of the buildings and decoration.



(a) The building in real world **(b)** The building in virtual world

Figure 4.9: Building in VCUHK (B. Chen et al., 2010)

The area of the real campus is divided into 36 VW regions with a size of 256x256 meters. Over 100 buildings have been modelled in the VW. Figure 4.9 shows a building of the campus in real world and in the VW. (B. Chen et al., 2010)

4.8.1 Architecture

VCUHK is build on OS servers, the basic architecture of the server system is shown in figure 4.10. The Basic Universal Server Technology (BUST) is the core OS server. The BUST server and the User, Grid and Messaging (UGM) servers are build on Ubuntu Linux servers. The 36 regions are build on 6 Windows 2008 servers. (B. Chen et al., 2010)

Other applications like a visitor centre, meeting rooms or virtual classes within VCUHK are currently in planning stages. Virtual classes are an important facet of virtual campus applications. Both educators and students can be more creative in the learning environments. (B. Chen et al., 2010)

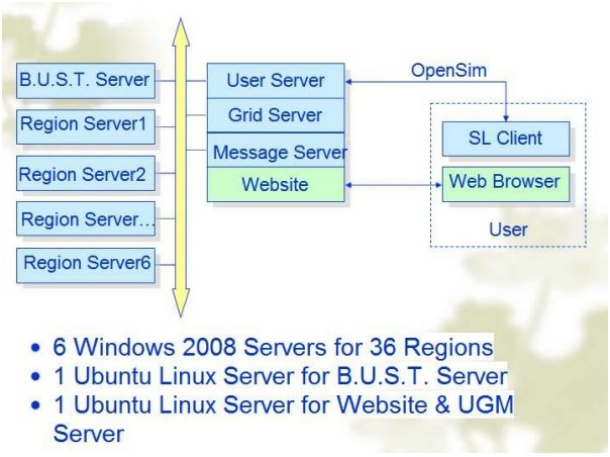


Figure 4.10: VCUHK - system architecture (B. Chen et al., 2010)

4.8.2 Findings

As identified by B. Chen et al. (2010), 3D modelling and texture processing needs a big amount of work. In VCUHK users can experience virtual education and other diverse campus activities. (B. Chen et al., 2010)

4.9 Best Digital Village

Best Digital Village (BDV) was build to support collaborative learning. 3D visualisation, multi-user, avatar tele-presence and other ICT mechanisms are supported. (Kuo & Lin, 2010)

(a) BDV 3D main window **(b)** BDV bird's-eye view

Figure 4.11: Best Digital Village - client interface view (Kuo & Lin, 2010)

> *"According to literature review, such 3D situated learning atmosphere increases learners' sense of immersion to get into the learning context, feeling the existence of the primary learning metaphor."* (Kuo & Lin, 2010, p. 5)

Figure 4.11a shows the client interface of BDV. The 3D main window shows a list of other online learners. Learners can retrieve the profile of other learners, which is believed to improve the quality of the learning information. A text messaging chat and a message broadcasting system are available to the learners. The arrow keys or the mouse can be used to navigate the avatar. Figure 4.11b contains a map of the learning space. (Kuo & Lin, 2010)

4.9.1 Architecture

The architecture of the framework is illustrated in figure 4.12. It is build on a three-tier architecture. Several databases with different purposes are created to support learning data storage and exchange. The application layer includes different types of information management modules and virtual buildings. The user can access the system by a web based user interface. (Kuo & Lin, 2010)

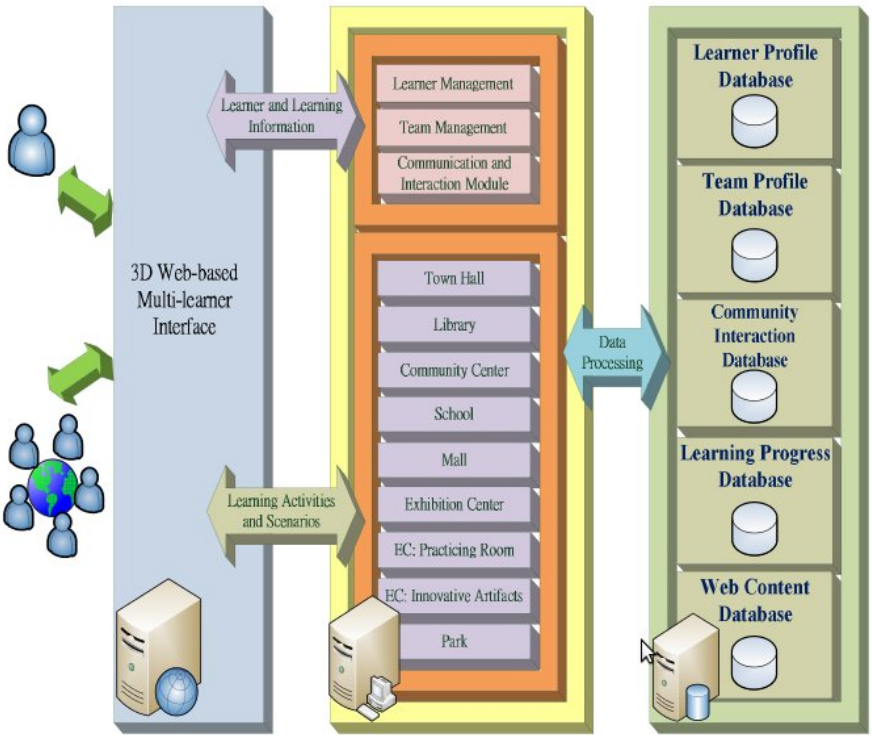


Figure 4.12: Best Digital Village - system architecture (Kuo & Lin, 2010)

4.9.2 Findings

According to Kuo and Lin (2010), the learning strategies of each learning scenario and task can be divided into three aspects of interfaces:

- *Learner - learning context*
  Using a 3D multi-learner virtual environment like BDV can provide the authenticity of situated learning, strengthen the human-machine interface and focus the learner on the learning activities.
- *Learner - learning activity*
  The learning activity is designed based on the learning content. Strategies of gaming and role-playing facilitate the learning engagement.
- *Learner - learner*
  The interaction and communication among learners are facilitated and supported more smoothly than in traditional Internet learning environments.

4.10 Medulla

Medulla is a framework to coordinate and support digital assets for learning, teaching and research within a VW. It has been created by the Federation of American Scientists (FAS) and provides several tools which are easy to use and services for identity management, team building, information sharing, project management, peer review, data versioning, data archiving, intellectual property management and learning management. (Fox et al., 2010, p. 87)

The goal of the Medulla project is to build themes which should allow teams to easily use VWs for education. All needed tools are currently available, the effort was to find tools and make them work together. FAS started the process of bringing such tools in an open source tool-set together. This tool-set, which is called Medulla, tries to provide a set of services which can be used and also not to limit on how the services are used and connected to each other. (Fox et al., 2010)

Fox et al. (2010) mention the following four tasks in order to set-up a new prototype:

1. Identifying the needed functionalities
2. Identifying existing software tools
3. Linking the found tools together
4. Find teams which use the tools to build prototype VWs

Figure 4.13 shows the Medulla algebra project which was designed to teach algebra 1 level mathematics. (Fox et al., 2010)

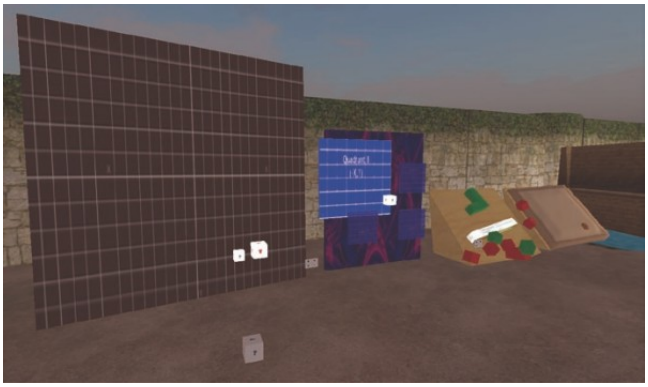


Figure 4.13: Medulla algebra project (Fox et al., 2010)

4.10.1 Findings

Many commercial tools already provide the functionality which is needed but many of them don‘t provide an API which would be needed to access the offered functionality. Another drawback is that the interfaces of the available tools are not consistent which makes it difficult to provide a common interface. (Fox et al., 2010)

First prototypes have been developed by the FAS and a team of volunteers. The prototypes are using the virtual 3D world SL. The materials which have been created during the construction of the prototypes are stored within Medulla and thus can be used within any other VW platform. (Fox et al., 2010)

4.11 The Web3D Project

The project described by Albion (2009) was funded by the Australian Learning and Teaching Council (ALTC). Its purpose is to explore the development of Low-Threshold Application (LTA) tools that can support content experts in the development of 3D virtual environments. Web3D is used to describe techniques that embed 3D content within web pages. The process could simplify the access to the material and improve the presentation as compared to more specialised 3D virtual environments. (Albion, 2009)

According to Albion (2009), the planned outcomes of the project are:

- Examples of Web3D applications for education.
- Web3D development tools and resources suitable for non-technical users.
- Guidelines for application of Web3D in education.
- An online community of practice for educators applying Web3D techniques.

3D models of objects such as the Interactive Skeleton, which presents a 3D model of a skeleton in a window within a web page, have been developed. The model can be rotated

and zoomed. Web3D allows the object to be viewed from different angles. Due to the fact that the objects are embedded in regular web pages, the material development and access is simplified. Additionally the 3D models can be embedded into Microsoft Office documents. (Albion, 2009)

To demonstrate the potential of Web3D applications for enhancing interactions between distant users, ALIVE Classmate has been developed. An avatar represents each participant. Voice over IP (VoIP) or text chat can be used to communicate. Screens can be installed to support slide shows, streaming video and a multi-user whiteboard. A YouTube video[1] shows an example of ALIVE Classmate. It is most suited to conventional pedagogies such as lectures. Though the environment can be configured to support different learning activities. (Albion, 2009)

The programmers developed a simplified editor for shaping a 3D virtual space. The growing library of objects lend hope that it is possible for non-technical academics to design and develop 3D spaces. (Albion, 2009)

4.11.1 Findings

Albion (2009) focused on how non-technical teachers and academics can create and use 3D content. The following findings have been stated in their work:

- The technical skills needed to use the available software tools for 3D object creation are beyond those possessed by most teaching academics.
- Academics will need to have access to collections of ready to use building blocks. The project has made a start by establishing the Web3D Exchange[2].
- Developing successful educational games requires bringing together expertise across different domains, for example content, learning design or game design.
- More challenges have been encountered than anticipated in making the creation of VWs achievable by non-technical users.

4.12 DWeb3D Toolkit

The purpose of the DWeb3D project was to develop a set of tools based on Extensible 3D (X3D) which should reduce the effort of developing 3D applications with X3D.

> *"X3D is a software standard for defining interactive web- and broadcast-based 3D content integrated with multimedia. X3D is intended for use on a variety of hardware devices and in a broad range of application areas such*

[1] http://www.youtube.com/watch?v=LV8G3e_Pk1Q
[2] http://web3dexchange.org

> *as engineering and scientific visualisation, multimedia presentations, entertainment and educational titles, web pages, and shared virtual worlds. X3D is also intended to be a universal interchange format for integrated 3D graphics and multimedia. X3D is the successor to the Virtual Reality Modelling Language (VRML), the original ISO standard for web-based 3D graphics (ISO/IEC 14772). X3D improves upon VRML with new features, advanced application programmer interfaces, additional data encoding formats, stricter conformance, and a componentized architecture that allows for a modular approach to supporting the standard."* (Web3D Consortium, 2008)

According to Quintella et al. (2010, p. 45), *"while X3D is an open standard, powerful, extensible and has a consortium, which maintains and supports it, it does not provide a very simple environment of quick development. This factor is probably a major cause of its low adoption rate"*.

Quintella et al. (2010) named a few features which applications in the real world would need and how they can be achieved with X3D:

- *Creating and manipulating a 3D scene*
  Achieved through the X3D definition.
- *Creating events that respond to user input*
  Achieved through specific structures and the possibility of using ECMAScript and extensions of the standard.
- *Loading and data persistence*
  Achieved through complex routines in ECMAScript and web services that have a session control state.
- *Interaction with web GUI*
  Can be made through ECMAScript calls, which requires knowledge of the internal structure of X3D.
- *Interaction between users accessing the same scene*
  Possible through complex solutions involving ECMAScript, code on the server and session control.
- *Integration with other applications*
  Can be done via web-services and ECMAScript.

In their work Quintella et al. (2010) developed tools for X3D and a demo application to show that X3D is flexible enough to support the features required by the developers. The work considered the topics collaboration between users, interaction with the web GUI, integration with other applications and data persistence. The goal of the developed toolkit, called DWeb3D, was to assist the development of collaborative applications using X3D. (Quintella et al., 2010)

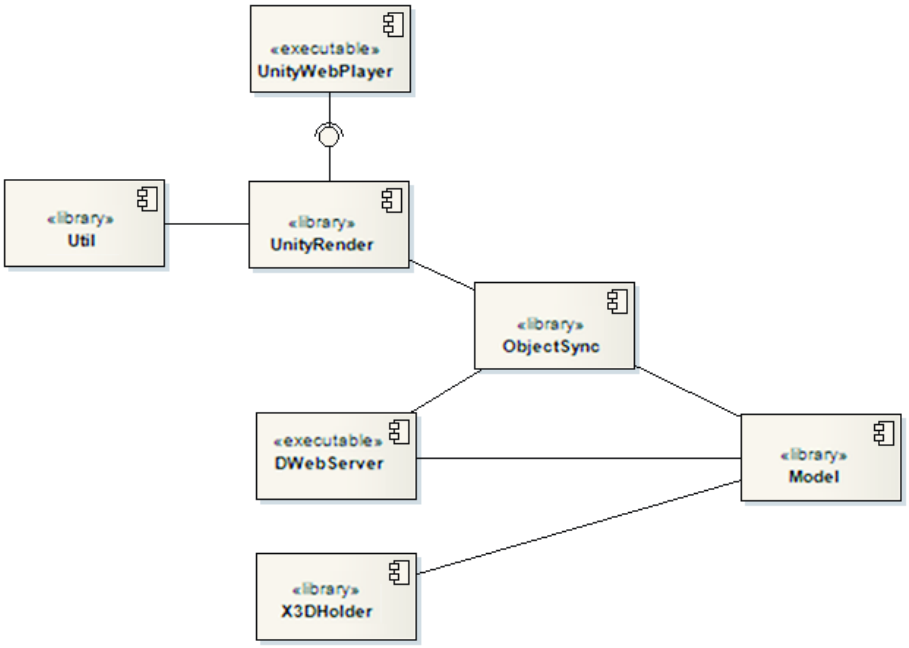


Figure 4.14: Components of DWeb3D (Quintella et al., 2010)

4.12.1 Architecture

The toolkit has been divided into modules organised by function. Figure 4.14 shows the components of the toolkit. Model contains the graph and classes that represent the structure of X3D and core classes for X3D file reading and writing. OjectSync is responsible for synchronisation of multiple objects across the network graph. DWebServer is the library which is creating a server. It forwards the synchronisation to the clients. UnityRender allows to use the X3D graph within the Unity3D[3] environment. Quintella et al. (2010, p. 47) introduced Unity3D as *"a graphics engine that aims to be modularised and extensible, supporting a wide range of formats for import"*. Util contains functions which are internally used by the toolkit. Finally the component Unity WebPlayer contains executable scripts to demonstrate the functionality. (Quintella et al., 2010)

The toolkit embeds low-level functions like socket creation or thread processing. Thus the developer does not have to deal with them any more. To synchronise any object, the object just has to implement an interface of the toolkit. To interact with other applications the X3D graph is converted into an Unity3D graph which allows to publish a web viewer. To interact with the Web GUI, Ajax3D has been developed which combines the traditional idea of Asynchronous JavaScript and XML (AJAX) and X3D. (Quintella et al., 2010)

[3]http://unity3d.com

4.12.2 Findings

Based on Quintella et al. (2010) the development of DWeb3D has uncovered the following findings:

- The world of X3D can be understood as several isolated 3D worlds. Thus providing communication between this worlds or the users is a challenging task and requires various complex solutions.
- X3D does not provide a simple environment for quick development.
- X3D does not deal very well with complex or large scenes.
- Application tests have shown that the toolkit can facilitate the process of developing applications that have the mentioned requirements.
- The use of DWeb3D reduces the amount of code needed to create 3D applications which allows the developers to focus on the goals of their applications.

4.13 Building 3D Worlds by Object Mapping

Moro, Mumolo, and Nolich (2010) present a method for automatically construction of V3DWs. The virtual objects correspond to objects detected in real environment. The operator takes a photo with a stereo camera of the real world environment. Regions of interest are extracted from the pictures, the content is classified and 3D virtual scenarios are reconstructed. (Moro et al., 2010)

> *"3D object mapping is the way to build a map of the environment which describes the shape and pose of the objects located in the environment."* (Moro et al., 2010, p. 31)

The key points for the presented technique are automatic detection of the regions of interest and object classification. Starting from the features obtained from a photograph, the objects in it are extracted. Then statistical models are trained for classification. For object characterisation the edges are used because they are more robust than other features. (Moro et al., 2010)

The image of an office environment contains items such as tables, chairs or persons. It is acquired with a stereo camera and an algorithm is used for building a 3D object map. This 3D object map is given as input to a 3D graphics package. The 3D objects are represented with corresponding graphical icons. The pose and the height of the objects are estimated. The icons are put in the VW with the estimated pose and height. (Moro et al., 2010)

Figure 4.15 shows the result of a case study which uses the algorithms described in the work of Moro et al. (2010). In the case study a picture of an office environment has been taken and the office environment in virtual 3D world has been generated. According to Moro et al. (2010, p. 35), *"the algorithm works sufficiently well in the simple environment"*. (Moro et al., 2010)

(a) Real world photograph **(b)** Generated virtual world objects

Figure 4.15: 3D virtual world building case study (Kuo & Lin, 2010)

4.14 Discussion

Previous research has shown that the construction of virtual 3D environments is a manual, time and cost-consuming task. Solution approaches have proven, that it is possible to generate those virtual environments automatically, which reduces the effort and costs needed to create the learning settings.

Four main requirements for virtual learning environments have been identified, they are adaptability, collaboration, appropriate environment design and reusability. Currently there are already plenty of tools, repositories and frameworks available but general standardised interfaces are missing.

Software tools for 3D creation are available but they require more technical skills as those possessed by most teaching academics. Tools and editors are needed which are easy to use and don't need much technological knowledge. Prototypes have already been developed which extract objects from photographs and generate them in a virtual world environment.

Tools like SLOODLE or SecondDMI enrich virtual worlds with learning management and communication functionalities. Further research like TwinSpace has shown the possibilities of the interconnection of virtual and real world learning environments. Prototypes have been implemented which enrich the virtual classroom with slide presenter, text messaging functionalities and other media.

Prototypes like the SL Room Manager focus on the automatic construction of learning settings in V3DWs. They have proven that it is possible to configure learning settings outside of the virtual world and to generate those settings automatically inside the world. StellarSim provides a configurable mechanism to create objects in OS. The approach has shown, that OS provides the flexibility for external management of in-world objects and settings.

The results have pointed out that although SL is a suitable learning environment for education in V3DWs, it has some drawbacks due to its commercial background. OS seems to be a reasonable alternative. The developer can work in standalone mode and has less restrictions as in SL. Due to the fact that the OS server is still in alpha status, it can not be used in productive systems yet. If the community manages to stabilise the server platform, it could become a major platform for education in VWs because of its open architecture.

The next chapter introduces the flexible learning setting management tool which has been developed in the context of this work. The application provides similar functionalities as the described SecondLife Room Manager. The proof of concept should show that it is possible to implement an even more flexible approach in OS because of its extensibility and the fact that the source code of the simulator is available and can be extended or even modified.

Chapter 5

Prototype

The initial goal of this work was to identify the capabilities of OpenSimulator in the context of learning in virtual worlds. Previous work at the Institute for Information Systems and Computer Media at the Technical University of Graz has shown the collaborative and flexible features of virtual worlds like Second Life. Due to the drawbacks of commercial virtual worlds (e.g. Second Life), like the high costs for land ownership or the limitations resulting from the fact that the source code is not available and can not be modified, the abilities of the open source platform should be investigated.

This chapter summarises the requirements of the implemented prototype, which have been identified in earlier work based on Second Life and an extensive review of the currently available literature. Further the design and implementation details of the prototype will be discussed in detail. Finally the experienced advantages and disadvantages of the open source approach are discussed.

The implemented prototype is a proof of concept and does not claim to be exhaustive or free of errors. Its primary goal is to analyse the competitiveness of the open source platform OpenSimulator in comparison to commercial approaches.

5.1 Requirements Analysis

The requirement analysis for the prototype has been split into two steps. In the first step the high level requirements have been identified by analysing recent literature and previously done work. Based on the results of the first step, the functional requirements for the prototype have been specified. The following sections describe the results of those two steps.

5.1.1 Identified High Level Requirements

Prior to this work a comprehensive review of the available literature has been done. Chapter 4 summarises the found results. Based on these findings several requirements have been identified which have to be fulfilled in order to provide a flexible collaborative virtual learning setting.

Low Cost Environment

A major disadvantage of commercial virtual world platforms are the costs for land ownership or parcel usage. Often these costs can not be paid by educational institutions which causes them to not provide virtual learning spaces. Open source alternatives or other free available platforms could lower the barriers to enter virtual learning environments.

As already mentioned the decision has been made to use OpenSim as the virtual world platform for the prototype. Chapter 3 depicts architecture, features and other capabilities of the open source platform.

Supported Learning Setting Creation

It has been found, that the creation of virtual world settings is a manual and time-consuming task. Teachers often do not have the technical skills to create the learning context in a way that it can effectively support the learning process. The manual construction of a learning setting does not seem to be cost and time-efficient. Automatic or supported creation of the educational environment in the virtual world could reduce the needed effort to create efficient learning settings. The setting creator should be able to focus on the learning needs rather than on the technical hurdles of creating objects in a virtual environment. Such a system or tool could also reduce the switching costs which would result from a change of the used virtual world platform.

In order to fulfil this requirement the prototype should provide a learning session configuration tool outside of the virtual world. It has been decided to create a web based tool.

Need for Collaboration

Another finding was, that collaboration can increase motivation and productivity and can reduce the feeling of isolation. In order to provide collaborative support within the learning setting, the web based tool should provide learning tools which allow the learners to interact and work together. Additional to static objects like chairs and tables, the setting creator should be able to choose from several different available collaborative tools and place them according to his needs in the learning room or space. Such a tool is for example a brain-storming board.

Adaptability

Learning settings must be flexible in order to support different learning processes. It turned out, that using the same predefined setting for different purposes is not effective enough. This means, that simply using pre-build learning rooms in virtual worlds for different teaching scenarios does not fulfil the individual needs of the learning process.

Thus the prototype should support the teacher to easily modify existing predefined or earlier used learning settings for new purposes. It must be possible to adjust the configuration of the learning session according to the needs of the learning process.

Session Management

To further support the teacher, it should be possible to save the state of a learning scenario. The configured setting (room, tables, chairs) as well as other objects like collaborative tools and their content should be saved by the prototype. It should be possible to create a learning setting which looks exactly like a previously saved learning setting. This functionality can be used for continuous learning sessions without much effort of recreation of the previously created content.

Basic Access Permission System

A basic authentication mechanism should be supported by the prototype. It should be possible to restrict access to a learning space or room to a specified group of persons (e.g. a learning group). Furthermore the tool should provide a mechanism to limit writing access to the learning tools.

5.1.2 Defined Functional Requirements

In the second step of the requirement engineering the following functional requirements have been specified:

- The learning setting configuration and management should be separated from the virtual world platform in order to be able to change the used virtual world.
- It should be possible to manage learning settings in different virtual worlds.
- Within a world it should be possible to create several rooms with different learning settings.
- A graphical room configuration editor should be available to create and mange the learning settings.
- Static room objects like chairs and tables should be available in the inventory of the editor.
- Collaborative learning tools should be available in the inventory of the editor.

- The tool should provide setting templates which can be extended for each specific learning process.
- The content of the collaborative tools should be saved automatically after each modification.
- It should be possible to continue learning sessions with the same content of the collaborative tools.
- A basic access permission system has to be available that allows to restrict access to the rooms and modifications of the collaborative tools content.

The goal of the prototype was to meet all those requirements and to show the potential of OpenSim in the context of flexible learning settings.

5.2 Design of the Prototype

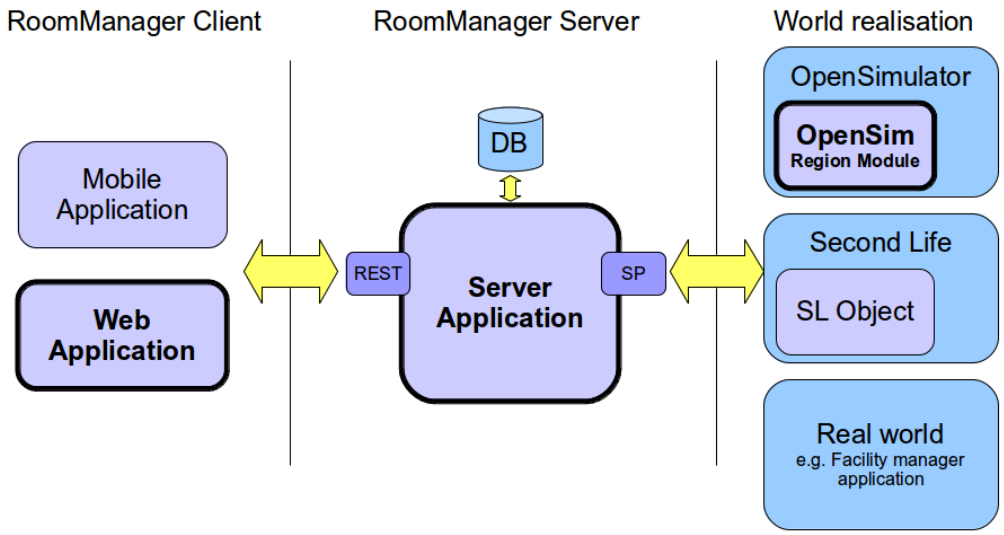


Figure 5.1: Basic prototype architecture

In order to meet all requirements which have been identified in the previous section, a flexible design was needed. Figure 5.1 shows the basic architecture of the prototype. All bold printed components are part of the proof of concept which has been implemented in the context of this master's thesis. All other components just show the possible flexibility of the chosen architecture (e.g. client applications for different platforms can be developed or world realisation components for different virtual world platforms can connect to the server). It would even be possible to connect a real world room management module to the *RoomManager Server* in order to connect the room management of virtual worlds and the real world.

The prototype consists basically of three components. The main component is the *RoomManager Server* which provides the other components with the needed data. The *RoomManager Client* provides the user interface to manage the configured worlds, rooms and learning settings. The third implemented component, the *OpenSim Region Module*, is responsible for realising the configured settings in an OpenSim region. The design of the three main components is described in detail in the following sections.

5.2.1 RoomManager Server

The main component of the prototype is the *RoomManager Server* application. It provides data to the client applications and the world realisation components. The server application stores all information of the prototype in a database. It provides a RESTful API to the client applications. The world realisation components can connect to a provided server socket port.

The functions of the component are:

- Stores and provides the configured worlds and rooms (size, position).
- Stores and manages the created learning settings (configuration of static objects and collaborative learning tools).
- Stores the content of previous learning sessions.
- Provides a basic assess permission system for the rooms and configured learning tools.
- Provides a room reservation mechanism which allows to assign learning settings to a room for a period of time.
- Provides a RESTful API to the client applications to access the data.
- Provides an API to the world realisation components. The communication to the world realisation components is bidirectional - the server sends room building requests to the components and receives modified learning setting objects in order to save the state of the objects.
- Schedules the room reservations and automatically creates the learning settings at the scheduled time in the configured world and room.
- Receives update notifications from the world realisation components each time an objects within a learning settings has been modified.

The following sections describe the managed object model and the provided APIs.

Managed Object Model

Figure 5.2 shows the managed entities and their relationships. Detailed information about each entity can be found in table A.1.

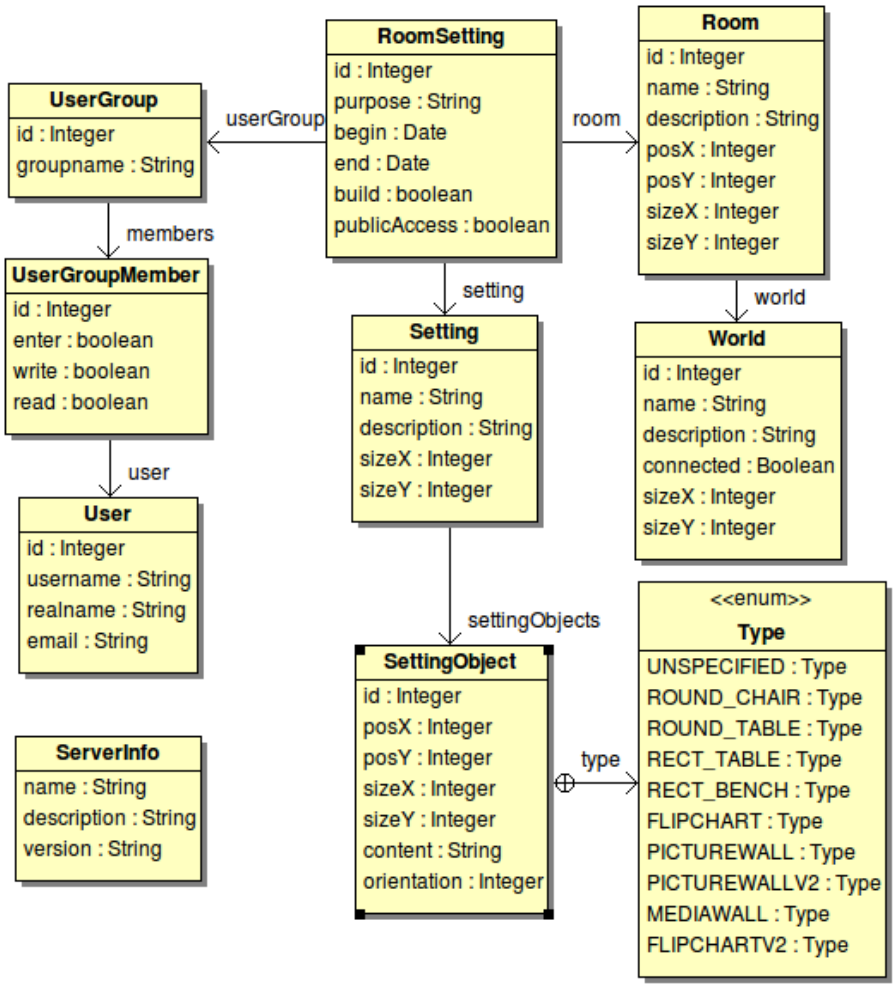


Figure 5.2: RoomManager Server - object model

Client Connections

The server provides an API, which is based on the REST architecture, to the client applications. Table A.4 describes all supported request URLs of the prototype. The payload of the requests and responses is transmitted as a JavaScript Object Notation (JSON) encoded string. According to json.org (2011), *"JSON (JavaScript Object Notation) is a lightweight datainterchange format. It is easy for humans to read and write. It is easy for machines to parse and generate [...] JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the Cfamily of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal datainterchange language."*

On each request the server returns a server response object. The server response object is encoded as JSON string and contains the elements described in table A.2. Due to the fact that the client just needs to be able to send Hypertext Markup Language (HTML) requests and to interpret JSON strings, it should be an easy task to access the server from

different platforms.

As shown in table A.4, the server provides an URL to retrieve information about the server. Assuming that the used server runs on the local machine and that the port number is 9002, the request *http://localhost:9002/serverinfo* returns the information shown in listing 5.1.

```
{
  serverInfo: {
    name: "RoomManager server",
    description: "The ultimate RoomManager server application",
    version: "0.1"
  },
  status: "SUCCESSFUL"
}
```

Listing 5.1: RESTful API - serverinfo

Listing 5.2 shows the result of the request *http://localhost:9002/usergroup/{id}* which retrieves the *UserGroup* element with the given id. The response contains a JSON encoded version of the *UserGroup* object which contains all assigned *User* objects and their permissions (entity *UserGroupMember*).

```
{
  serverInfo: {
    name: "RoomManager server",
    description: "The ultimate RoomManager server application",
    version: "0.1" },

  status: "SUCCESSFUL",

  userGroups:
  [
    { id: 1, groupname: "G1",
      members: [ { id: 2, enter: true, write: false, read: true,
                   user: { id: 4, username: "X Y", realname: "X Y", email: "" } },
                 { id: 1, enter: true, write: true, read: true,
                   user: { id: 1, username: "Test User", realname: "Test User", email:
                       "" } } ] },
    { id: 2, groupname: "G2", members: [ ] }
  ]
}
```

Listing 5.2: RESTful API - usergroup

As the examples shows, all objects are transferred between the *RoomManager Server* and a *RoomManager Client* by sending JSON encoded representations of the managed objects.

World Realisation Component Connection

As shown in figure 5.1, a server port is provided for World Realisation Components (WRCs) to connect. Several WRCs can connect to the server port. For each managed world, a connection to the server has to be established. The WRC has to establish the connection and send the name of the managed world in order to sign in at the server. The message format which is transferred between the RoomManager Server (RMS) and the

WRC is very simple. Each message contains a JSON encoded message object. The end of the message can be identified by a new-line character.

Table A.5 contains the elements of the message object. The table A.6 contains all message types which are send between the RMS and the WRCs. The first word of the action name indicates which component sends the message. Message actions starting with *CLIENT* are send by the WRC. All message actions which start with *SERVER* are send by the RMS component.

The WRC must not persist the room settings which have been sent by the RMS. The RMS assumes that the WRC does not store any information about previously received rooms or room settings when reconnecting to the server port. After a successful login, the RMS sends a message with the action *SERVER_BUILD_ROOM_REQUEST* for all room settings which should be available at this time. When the connection to the server breaks down, the WRC must delete all created objects and has to try to reconnect to the server. After establishing the connection again, the RMS will send all room settings again. Due to this mechanism it is not necessary to synchronise the RMS and WRCs in case of communication problems.

Listing 5.3 illustrates a not successful login attempt. Assuming that the world OSLS2 is not available at the RMS, the shown messages will be send.

```
World realisation component -> Room Manager Server:
{ action:"CLIENT_CONNECT",
  worldName:"OSLS2"
}

RoomManager Server -> world realisation component:
{ action:"SERVER_CONNECT_REQUEST",
  serverInfo:
  { name:"RoomManager server",
    description:"The ultimate RoomManager server application",
    version:"0.1"
  }
}
```

Listing 5.3: World realisation API - unsuccessful connection attempt

The content of a successful login attempt is shown in listing 5.4.

```
World realisation component -> Room Manager Server:
{ action:"CLIENT_CONNECT",
  worldName:"OSLS2"
}

RoomManager Server -> world realisation component:
{ action:"SERVER_CONNECT_SUCCESSFUL",
  serverInfo:
  {
    name:"RoomManager server",
    description:"The ultimate RoomManager server application",
    version:"0.1"
  }
}
```

Listing 5.4: World realisation API - successful connection attempt

The listing 5.5 contains the message which is send from the RMS to the WRC in order to build a room and the learning setting. The room contains a round chair, a rectangular table, a flip chart tool and a picture wall. The message contains all needed information to build the room. It contains information about the room (size and position), details about the setting objects (position, type, size and orientation) and information about the access permissions (user group).

```
RoomManager Server -> world realisation component:
{
  action:"SERVER_BUILD_ROOM_REQUEST", worldName:"OSLS1",
  serverInfo: { name:"RoomManager server", description:"The ...","version":"0.1" },
    roomSetting: { id:9,  purpose:"Test reservation", begin:"Jul 11, 2011 5:00:00 PM",
                   end:"Jul 12, 2011 6:00:00 PM", build:true,
      setting: { id:10, name:"New", sizeX:10000, sizeY:10000, type:"ROOM_SETTING",
        settingObjects: [
          { id:39, type:"ROUND_CHAIR",   posX:4150, posY:1175,
                   sizeX:500,  sizeY:500, orientation:0 },
          { id:40, type:"PICTUREWALLV2", posX:6550, posY:575,
                   sizeX:2500, sizeY:200, orientation:0 },
          { id:41, type:"FLIPCHARTV2",   posX:650 , posY:500,
                   sizeX:2500, sizeY:200, orientation:0 },
          { id:42, type:"RECT_TABLE",    posX:3650, posY:1950,
                   sizeX:2000, sizeY:750, orientation:0 }
        ] },
      room: { id:4, name:"R1", description:"", posX:1000, posY:1000,
                    sizeX:10000, sizeY:10000, wallType:"NONE",
        world:{ id:3, name:"OSLS1", description:"", connected:true,
                      sizeX:256000, sizeY:256000 } },
      userGroup: { id:1, groupname:"G1",
        members: [
          { id:1, enter:false, write:true, read:true,
            user: { id:1, username:"Test User", realname:"Test User", email:"" } },
          { id:2, enter:false, write:true, read:false,
            user: { id:4, username:"X Y", realname:"X Y", email:"" } } ] },
      publicAccess:true }
}
```

Listing 5.5: World realisation API - build room request

The last example, which can be found in listing 5.6, illustrates the message which is send from the WRC to the RMS in order to save the content of a collaborative learning tool.

```
World realisation component -> Room Manager Server:
{
  action: "CLIENT_UPDATE_SETTING_OBJECT",
  settingObject: { id:41, type:"FLIPCHARTV2", posX:650, posY:500, sizeX:2500, sizeY
      :200,
    content: "{
     \"pages\": [
       {\"Commands\":[
         \"MoveTo 272, 241; PenSize 3; PenColor Red; Rectangle 398, 330; \",
         \"FontSize 50; PenColor Black; MoveTo 195, 79; Text Hello Tool; \"] } ],
     \"x\":195,\"y\":129,\"xOld\":670,\"yOld\":571,\"autoLine\":false,
     \"helpActivated\":false,\"color\":\"Black\",\"fontSize\":50,\"pixelSize\":3,
     \"index\":0 }",
    orientation:0 }
}
```

Listing 5.6: World realisation API - update setting object

In this example the content of a flip chart tool has been modified and is send to the

server. The content of a flip chart tool is saved as a JSON encoded string which contains drawing commands. The flip chart contained one page with a red rectangle and the black text *Hello Tool*. The message contains just the setting object and the action. The new content of the object contains two drawing commands and some other settings which are managed by the flip chart tool implementation.

5.2.2 RoomManager Client - Web Application

The client application can be realised for several platforms. For the first proof of concept during this work, a web application has been chosen as the user interface. The user interface provides the functionality to manage all objects which are stored in the RMS application.

The functions of the component are:

- Manages the entities of the RMS application (worlds, rooms, users, ...) by accessing the RESTful interface of the RMS application.
- Provides a graphical room designer to create learning settings.
- Provides the functionality to manage learning setting templates. A preview list is available to simplify the selection of a suitable template.
- Provides a simple room management mechanism including room reservations.
- Supports the functionality to continue stored learning sessions.
- Provides a simple mechanism to manage the permission settings of a learning session.

The implemented proof of concept provides the basic functionality needed to illustrate the capabilities of configuring learning settings and managing virtual world rooms. It may not be used in a productive environment because the following aspects have not been considered:

- The client application does not provide a login system. Everyone who can access the client application is able to modify the content.
- Concurrent access and modification of the data has not been tested.
- Paging of the result sets has not been implemented. The prototype is not considered to be used with large datasets.

5.2.3 OpenSimulator Region Module

The WRC is responsible for realising the configured learning settings in the underlying virtual world platform. Within this work, an *OpenSim Region Module* has been developed. The capabilities of region modules are described in section 3.4.2 in detail.

The functions of the WRC application are:

- Provides a configuration mechanism to configure several managed regions within a simulator.
- Connects to the RMS application during start-up of the simulator. For each configured managed region, a separate connection has to be established.
- Reconnects to the RMS application after the connection has been closed.
- Receives commands from the RMS application by receiving messages as described earlier in section 5.2.1.
- Creates the virtual world objects which are needed to create the learning settings.
- Removes the created objects from the virtual world in order to remove a previously build learning setting.
- Sends update notifications to the server application whenever the content of a collaborative tool has been modified.
- The application does not persist the created objects. After losing the connection to the RMS application all created objects will be removed.

The region module expects that all configured regions are available within the simulator and that they have a flat ground at a level of 21 meters.

5.2.4 Component Interaction

Figure 5.3 illustrates the basic information flow between the three components. The RoomManager Client (RMC) application is responsible for providing a GUI in order to manage the stored information. For example whenever a learning session is configured, the component notifies the RMS component in order to save the data. The RMS manages all configured learning sessions.

Whenever a session reservation starts, the component sends a message to the OpenSimulator region module. The region module creates all objects in the virtual world. Those objects are walls, static objects like chairs or tables and interactive collaboration tools like flip charts.

The behaviour of the objects are implemented either directly within the region module or scripts are added to the objects. Tools which are able to save content modifications have to be implemented within the region module. Whenever the content of a tool changes, the region module sends a message with the new content of the tool to the RMS application.

The RMS component updates the content of the tool and whenever the session is continued, the latest state of the content will be used.

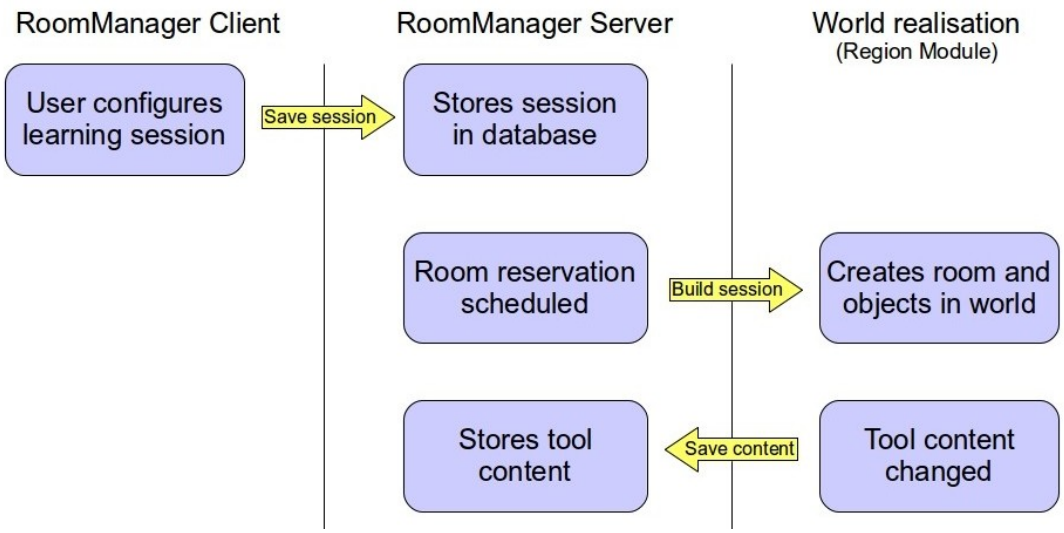


Figure 5.3: Prototype component interaction

5.3 Implementation Environment

This section describes the implementation details of the prototype. The prototype has been implemented in Ubuntu 10.10 *Maverick Meerkat*[1] (64-bit PC desktop).

As already mentioned three components have been implemented in the scope of this work. Please refer to figure 5.1 to get an overview about the basic architecture of the prototype. The server application and the web application have been implemented in Java. Due to the fact, that the OpenSim source is written in C#, the developed region module has been implemented in C#. The following sections describe the used Java and C# development environments.

The section contains a brief introduction of all used frameworks, tools and development Integrated Development Environments (IDEs). The prototype has not been tested in other operating systems.

5.3.1 Java Development Environment

The RMS and the RMC applications have been implemented in *Java SE*[2] (version 1.6.0_24). *Eclipse SDK*[3] (version 3.5.2) has been used as IDE. Various different existing Java frameworks have been used. The following list provides a short overview about the used frameworks and libraries:

[1] http://releases.ubuntu.com/maverick/
[2] http://www.oracle.com/technetwork/java/javase/downloads/index.html
[3] http://www.eclipse.org/

- *Hibernate*[4] (version 3.6.3)
  According to Hibernate (2011), *"Historically, Hibernate facilitated the storage and retrieval of Java domain objects via Object/Relational Mapping. Today, Hibernate is a collection of related projects enabling developers to utilize POJO-style domain models in their applications in ways extending well beyond Object/Relational Mapping"*. The framework is used by the RoomManager server to store all created and managed objects in a Database Management System (DBMS).
- *HSQLDB*[5] (version 2.1.0)
  As introduced by HyperSQL (2011), *"HSQLDB (HyperSQL DataBase) is the leading SQL relational database engine written in Java. It has a JDBC driver and supports nearly full ANSI-92 SQL (BNF format) plus many SQL:2008 enhancements. It offers a small, fast multi-threaded and transactional database engine which offers in-memory and disk-based tables and supports embedded and server modes. Additionally, it includes tools such as a command line SQL tool and GUI query tools"*. The database is used to store all created and managed objects.
- *Restlet*[6] (version 2.0.6)
  According to Restlet Wiki (2011), *"Restlet is a comprehensive yet lightweight RESTful web framework for Java that lets you embrace the architecture style of the Web (REST) and benefit from its simplicity and scalability"*. The framework is used for the communication between the RMS and the client applications. The server provides a RESTful API, which can be accessed by the clients.
- *Google Gson*[7] (version 1.7.1)
  Google (2011a) introduces Gson as, *"a Java library that can be used to convert Java Objects into their JSON representation. It can also be used to convert a JSON string to an equivalent Java object. Gson can work with arbitrary Java objects including pre-existing objects that you do not have source-code of"*. It is used to convert the managed Java objects into a JSON string representation in order to send it from the client to the server and vice versa.
- *Apache Log4J*[8] (version 1.2.16)
  The framework is used to print debug information to the console.
- *Apache Commons Configuration*[9] (version 1.6)
  As introduced by The Apache Software Foundation (2011), *"Commons Configuration provides a generic configuration interface which enables a Java application to read configuration data from a variety of sources"*. The interface is used to provide a property configuration to the RoomManager server application.

[4] http://hibernate.org/
[5] http://hsqldb.org/
[6] http://www.restlet.org/
[7] http://code.google.com/p/google-gson/
[8] http://logging.apache.org/log4j/1.2/index.html
[9] http://commons.apache.org/configuration/

- *GWT*[10] (version 2.3)
  According to Google (2011b), *"Google Web Toolkit (GWT) is a development toolkit for building and optimizing complex browser-based applications. Its goal is to enable productive development of high-performance web applications without the developer having to be an expert in browser quirks, XMLHttpRequest and JavaScript. GWT is used by many products at Google, including Google Wave and the new version of AdWords. It's open source, completely free, and used by thousands of developers around the world".*

- *gwt-dnd*[11] (version 3.1.0)
  To enable drag and drop in the GWT application, the GWT extension gwt-dnd has been used.

5.3.2 C# Development Environment

In order to provide a WRC for OpenSim, a OpenSim region module has been implemented. Details about region modules can be found in section 3.4.2. The application has been developed by using the *Mono*[12] development framework (version 2.6.7). According to Mono (2011), *"Mono is a software platform designed to allow developers to easily create cross platform applications. It is an open source implementation of Microsoft's .Net Framework based on the ECMA standards for C# and the Common Language Runtime".*

The source code of the application has been developed in *MonoDevelop*[13] (version 2.4). MonoDevelop (2011) introduces MonoDevelop as *"an IDE primarily designed for C# and other .NET languages. MonoDevelop enables developers to quickly write desktop and ASP.NET Web applications on Linux, Windows and Mac OSX. MonoDevelop makes it easy for developers to port .NET applications created with Visual Studio to Linux and to maintain a single code base for all platforms".*

The OpenSim source code version 0.7.1.1 has been used to develop the application. All used classes are either part of the mono development framework or available in the source code of the OpenSim project. No other third party libraries have been used.

5.4 Region Module Implementation Details

As already mentioned, a new OpenSimulator region module with the name *TURoomManagerModule* has been developed in order to create and mange the objects within the simulator. This section describes the implementation details of the new module.

[10] http://code.google.com/intl/de-AT/webtoolkit/
[11] http://code.google.com/p/gwt-dnd/
[12] http://mono-project.com/Main_Page
[13] http://monodevelop.com/

5.4.1 OpenSimulator Integration

The new module has been implemented as a non-shared region module. Details about OpenSimulator region modules can be found in section 3.4.2. A non-shared region module is instantiated for each available region within the simulator. An example configuration of the module can be found in section 5.7.3. Each region which should be able to contain learning rooms must be configured in this configuration file. The available rooms and their locations are flexible, rooms are created on demand. The region module does not know which rooms will be created by the RMS application.

The framework assigns a so-called *Scene* object to the region module. This object is used to access other region modules and the core functionalities of the simulator. For example new objects can be added to the region by using this object. The following list contains the used additional region modules:

- *DynamicTextureModule*
  This module is used to assign a texture dynamically to an object. This functionality is used for example by the flip chart tool to draw the content of the current page on the surface of the tool.
- *MessageTransferModule*
  Is used to send messages to the avatar. For example this is used to notify the user that he does not have the permissions to enter a room.
- *MoapModule*
  The media on a prim module is used to assign an URL to the media content of an 3D object.

5.4.2 Communication with the RoomManager Server

The region module connects to the server port provided by the RMS application in order to retrieve learning session information and to store the content of learning tools. The communication is implemented in the class *TURoomManagerConnector*. The host name and the port number of the RMS application must be configured in the configuration file of the module for each region. Thus it is possible to connect to different RMS applications. Each region can just connect to one server application. C# and Mono build-in functionalities are used to communicate with the server application and to serialise and deserialise the message content between string format and C# objects. The communication is bidirectional and a reconnect mechanism has been implemented.

5.4.3 Learning Session Creation

The region module retrieves all information that is necessary to create the learning session from the RMS application. The class *TURoomBuilder* is responsible for creating all needed objects. In the first step the walls, the roof, the floor, the sliding door and the room sign are

created. In OpenSimulator, each object is represented as an instance of *SceneObjectGroup*. Depending on the information contained in the received message, the size and the position of the room is calculated. The created objects also depend on the selected room type. Transparent and solid walls just differ in the texture of the walls.

In the next step all content objects of the learning session are created. A list of provided objects can be found in table 5.1. For each content type a C# class is available which is responsible for creating the object. For example the class *TUFlipChartV2Builder* creates the objects of the flip chart tool and adds them to the region by using the scene object which has been assigned to the region module. In order to add a new type of content object to the prototype, such a builder class must be added, which creates the needed simulator objects and handles the content of the tool.

The region module stores references to all created objects in order to remove them from the region whenever the learning session ends.

5.4.4 Learning Tool Interaction

Basically two possible ways of communication between 3D objects of the region and the avatar are available. The avatar can touch the surface of the object and messages can be send to the objects by using build-in communication channels. Both communication mechanisms are used within the application.

During the development of this prototype two different approaches to interact with the created objects have been implemented. The first approach was to add a LSL script to the object which stores the content and interacts with the avatar. In this case the builder class adds the script to the object by using the build-in functionalities of the assigned scene object. Listing A.1 illustrates an example of such a script. No solution has been found to send content changes to the region module and therefore the content of such tools is not stored on the server. For example the collaborative tool *Flip Chart* uses this approach.

Due to the requirement of the prototype to be able to save the content and to have the possibility to resume learning sessions, a second approach has been implemented which is used for example by the second version of the flip chart tool, *Flip Chart V2*. Instead of attaching scripts to the 3D objects, the region module directly communicates with the avatar and updates the content of the learning tools. The framework provides mechanisms to subscribe for region events. Such events are for example that an avatar touches the surface of a 3D object or that messages are send on a communication channel. By implementing the functionalities of the tools directly within the region module, all provided functionalities of the simulator can be used. The second generation of the tools automatically send the new content to the RMS application whenever the content has been modified by the interaction with the avatar.

5.5 RoomManager Client Usage

This section gives an overview about the functionalities of the client application. It describes the layout and the different functions in detail.

5.5.1 Basic Layout

Figure 5.4 shows the basic layout of the implemented web application. At the top of the page a header is printed. At the bottom of the page a status panel is available which shows error and status messages. On the left hand side the main menu is located. For each managed entity a button is available to access the content dialog of the entity. In the center of the page the content of the selected entity is displayed.

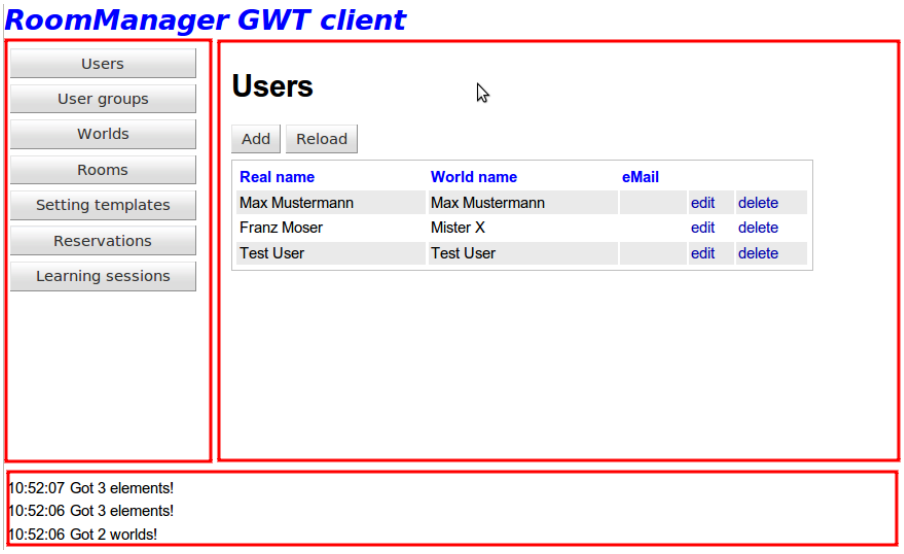


Figure 5.4: RoomManager Client - basic layout

At the top of the content section, the entity name is printed. In the next row a filter section is available which contains buttons to add a new element or to reload the content of the entity. Additional filter elements can be added. For all entities a table like view is available. Each available element of the entity is printed in one row of the table. At the end of the row, there are links available to modify the element. Such operations are for example delete, to delete the element, or edit, to change the content of an element.

Clicking on the button *Add* or on the link *edit* of an element opens a dialog window which is used to enter the values of the new element or of the element which should be modified. Figure 5.5 shows such a dialog window. The dialog contains a row for each field to edit. The first column contains the name of the field. The second column contains the widget (e.g. text box, combo box, check box, ...) which is used to edit the value of the field. In the last column an additional field prints information about which values are required and which values are allowed. At the bottom of the dialog the buttons *Save* and *Cancel* are

Figure 5.5: RoomManager Client - edit dialog

available. Clicking on *Cancel* closes the dialog without any modifications. Using the *Save* button causes the client to check the values of the widgets. For each not correct entered value, the text in the information column is displayed in red colour. Only if all fields are entered correctly, the client saves the entity by sending the data to the RMS application by using the provided API (see section 5.2.1). After closing the dialog window, the content table will be reloaded.

During any communication with the RMS application, a loading dialog will be shown, each dialog is modal. Figure 5.6 illustrates the loading dialog.

Figure 5.6: RoomManager Client - loading dialog

The connection to the RMS is stateless, each request causes a HTTP request to the RMS application. Whenever the client is not able to establish a connection to the server, a

dialog is shown to the user which can be seen in figure 5.7. This dialog can not be closed by the user. The application tries to reconnect to the server as long as the connection can't be established. The dialog will be closed automatically as soon as a connection can be established.

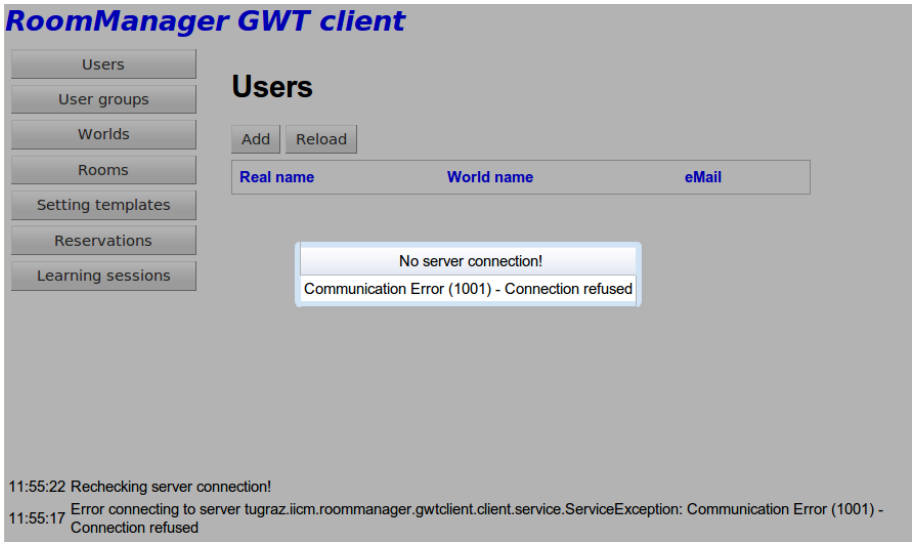


Figure 5.7: RoomManager Client - no connection dialog

5.5.2 Graphical Room Setting Editor

The RMC application provides a graphical setting editor which supports drag and drop to configure the objects within the settings. Figure 5.8 shows the basic layout of the drag and drop editor.

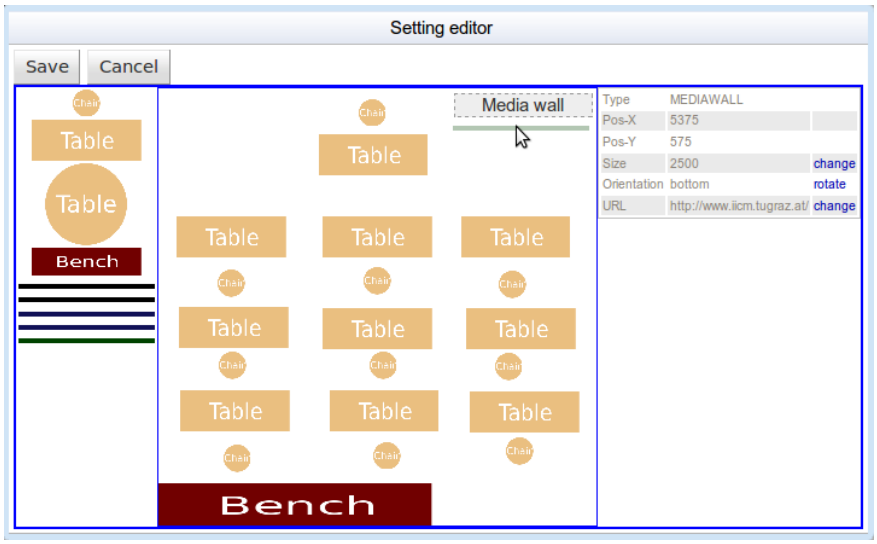


Figure 5.8: RoomManager Client - drag and drop setting editor

On top of the editor the buttons *Save* and *Cancel* are located. *Save* saves the configured setting by sending a request to the RMS and closes the dialog. *Cancel* just closes the dialog without applying any modifications. The main area of the editor is divided into three main columns. The first column is the inventory of the editor. It contains all implemented static objects and collaborative tools. Table 5.1 explains all implemented objects and tools in detail.

Tool	**Description**
Round chair	Simple round chair. The size and the position of the chair can be modified.
Rect. table	Simple rectangular table. The size and the position can be modified.
Round table	Simple round table. The size and the position can be modified.
Bench	Simple rectangular bench. The size and the position can be modified.
Flip Chart	The flip chart is a collaborative tool that allows all users, which have the needed permission, to modify its content. It is possible to draw lines, rectangles and ellipses as well as to add text on the displayed page. The flip chart provides several pages. It is possible to navigate to the first, next, previous and last page. The content of this tool is not stored on the server. Additionally to the size and the position, the orientation can be changed. It can be selected if the content has to be displayed at the bottom, at the left side, ad the top or at the right side of the tool.
Flip Chart V2	Additionally to the functionalities of the flip chart tool the content of this version will be saved on the server.
Picture Wall	The picture wall is also a collaborative tool. It can be used to view images or slide shows. Several images can be assigned in the graphical editor. Additionally to the size and position, the URLs of the images and the orientation can be specified. The index of the last shown picture will be saved on the server.
Picture Wall V2	Second version of the picture wall tool which uses a different in-world implementation.
Media Wall	This tool provides the functionality to load web pages. It can for example be used to edit documents like *Google docs*[14].

Table 5.1: RoomManager Client - implemented static objects and collaborative tools

[14]http://docs.google.com

The second column is the main editing area. The size of this area depends on the size of the setting. New tools can be added to the setting by dragging it from the left widget column to the main editor area. Several tools or static objects can be added to the main area. The position of each tool can be modified by dragging it again to a different position. In order to delete an object from the setting, the object has to be dropped into the left inventory area. The editor shows a tool-tip message with the type of the tool whenever the mouse is moved over a tool. It is possible to move several objects at the same time. In order to select multiple elements, the CTRL key has to be pressed during selection.

The right column shows information about the currently selected tool. The type, the position and the size of the object is displayed for all implemented tools. Further displayed information depends on the type of the selected tool (e.g. the media wall shows the set URL and provides a mechanism to change this URL). The position of the tool is updated each time the tool is moved to a different position.

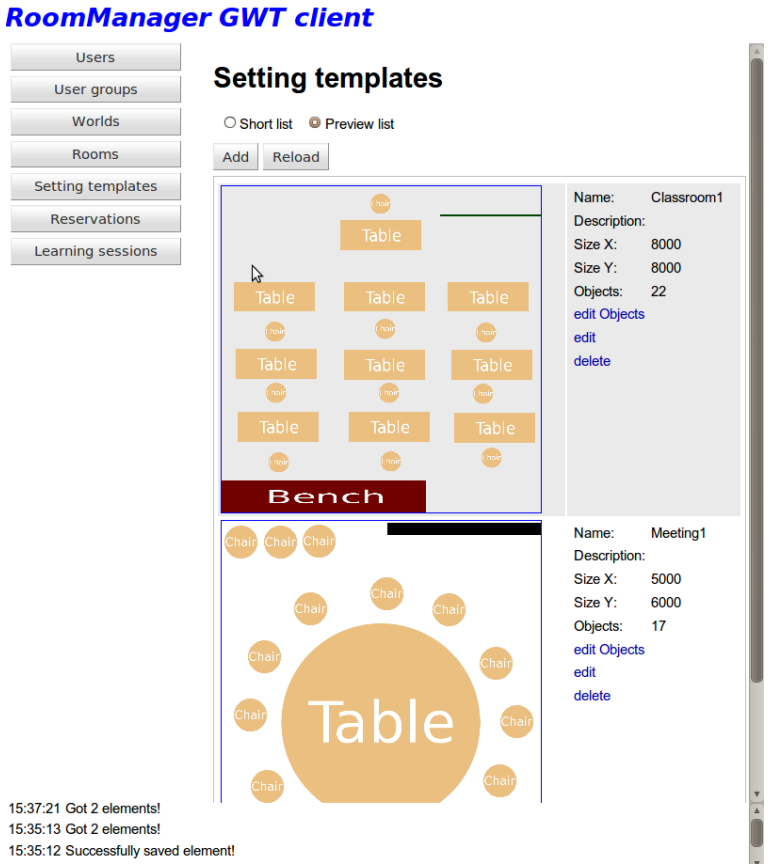


Figure 5.9: RoomManager Client - setting templates

5.5.3 Setting Templates

The application provides a template mechanism to mange learning settings which are common. Additionally to the list view, which shows a table with all available templates, the setting template screen provides a second view. This view is called *Preview list* and shows an image of all setting templates. Figure 5.9 shows such a preview list. The option link *edit Objects* opens the setting editor as shown in figure 5.8 in order to modify the learning setting.

5.5.4 Room Reservation

This section describes how a reservation for a room can be made by using the application. Before creating a room reservation, a world and a room for the reservation must be configured. In order to create a world, the world management screen has to be opened by clicking on the button *Worlds*. After loading all available worlds, the button *Add* can be clicked in order to configure a new world. A dialog opens in which the name and the size of the world must be entered. Figure 5.10a shows the values of the used example.

(a) 1. step: add a new world **(b)** 2. step: add a new room

Figure 5.10: RoomManager Client - add a new room

After clicking on the button *Save*, the world will be send to the RMS application and the list of worlds will be reloaded. In the next step a room will be added. By clicking on the option link *edit Rooms* the room management screen opens and the world *OSLS1* will be selected. By clicking on the button *Add* a new room can be configured. Figure 5.10b shows the dialog which is opened in order to edit the new room configuration. The world which contains the room needs to be selected. A room name and an optional description can be entered. One of three available wall types (solid, transparent, no wall) has to be selected. Finally the size and the position within the world must be entered in millimetres.

After saving the room configuration, the list of rooms will be reloaded. The option link *edit Reservations* can be used to access the reservation management screen. In this case the used room is preselected. A new reservation can now be added by clicking on the

Figure 5.11: RoomManager Client - add a new reservation

button Add. Figure 5.11 shows the opened reservation dialog. The chosen room has been preselected. The fields *User group* and *Allow public access* are used to manage the access permissions of the room which is described later in this chapter. A begin and an end time has to be entered, the end time must after the begin time. In the last section of the dialog the setting type must be selected. Either the user can choose from a list of available setting templates (all setting templates which are not bigger than the selected room) or he can start with a new empty setting. In the shown example the template with the name *Classroom1* has been used.

Figure 5.12: RoomManager Client - edit the learning setting

After clicking on the button *Next* the earlier described setting editor opens and the chosen

template can be modified. The modifications do not change the template. Before opening the setting editor, a copy of the template is created and used. The size of the setting will be set to the size of the room. If the room is bigger than the setting, then additional free space is available at the right side and the bottom of the editor. Figure 5.12 shows the final setting after moving the bench to the bottom of the room and adding of a round table in the lower right corner. By clicking on the button *Save* the new reservation will be stored at the RMS application.

5.5.5 Continuation of Learning Sessions

Another feature of the implemented RMC application is the possibility to continue learning sessions. It is possible to continue a finished learning session, copy a scheduled session or even to create a new reservation with the current state of an active session. The screen *Learning sessions* has to be used to access these functionalities. Figure 5.13 shows an example of the screen. It is possible to view all learning sessions or to select a world and a room in order to filter the list. The option link *show* opens a preview of the learning setting. This preview can be closed by clicking somewhere outside of the preview.

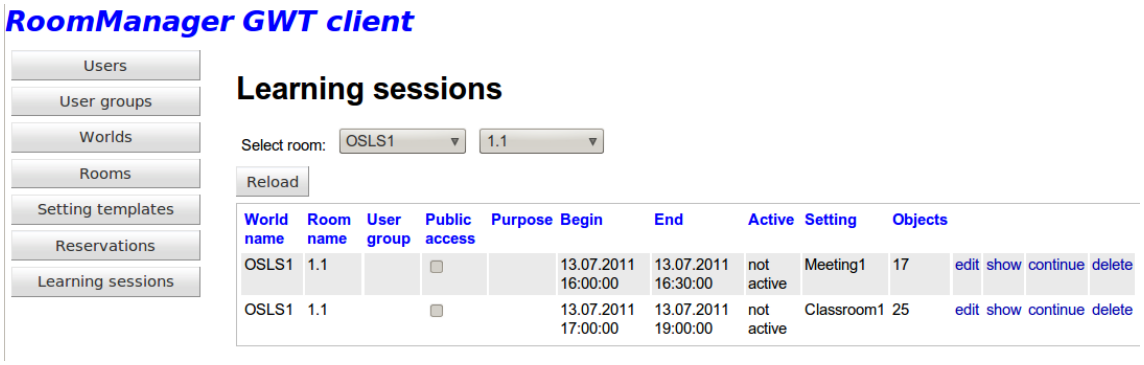


Figure 5.13: RoomManager Client - edit learning sessions

In order to continue or copy a session, the option link *continue* has to be used. A dialog will be opened to configure the copy of the learning session. The world and the room is already preselected. By clicking on the option link *Choose different room* it is possible to change the room in which the copied session should take place. Figure 5.14 shows an image of the dialog after clicking on this link. In this example, the room with the name *1.2* has been selected. Clicking on the button *Select* changes the room to the selected one. The other fields are the same as in the reservation editor. Its not possible to choose a setting template because the setting of the learning session which is copied will automatically be cloned and used for the new learning session. After clicking on the button *Next* the session editor will be opened in order to modify the setting. Modifying the setting does not change the setting of the learning session which will be continued. Figure 5.12 shows an image of the setting editor. After clicking on the button *Save* the new learning session will be send to the RMS application.

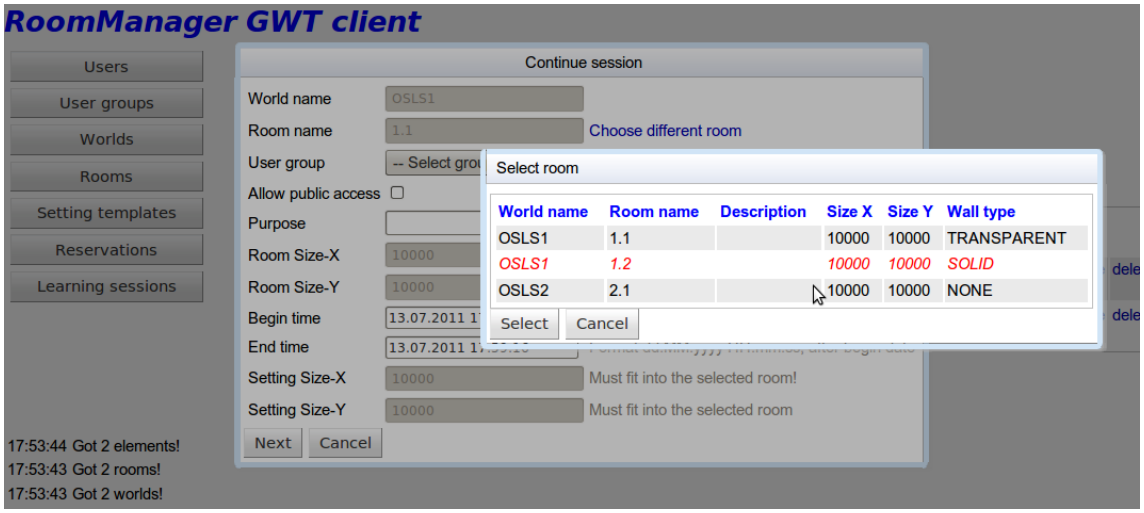


Figure 5.14: RoomManager Client - continue a learning session

In the Learning sessions screen it is also possible to change a setting, the option link *edit* has to be used.

5.5.6 Access Restriction

Figure 5.15: RoomManager Client - edit user groups

The implemented prototype offers a simplified permission management mechanism which basically provides the functionality to restrict the access to a learning session to a configured group of users. Further the possibility to change the content of collaborative tools can be restricted. The screen *Users* is used to add, modify and delete users. In order to grant permissions to a learning session to a user, the user must be added to a user group.

User groups can be modified in the screen *User groups*. Figure 5.15 shows the screen after clicking on the option link *editMembers* of the group with the name *G1*.

The option link opens a dialog in which the permissions *Enter*, *Read* and *Write* can be set for each user. The check box enter must be enabled if the user should be able to enter the room. The permission *Read* is currently not used but is supposed to grant the permission to see the content of the collaborative tools. The third permission *Write* allows the user to change the content of the collaborative tools. Users can be removed from the group by clicking on the option link *remove*. To add a user to the group, the user name has to be selected at the list box in the last row of the table. After selecting the user name the option link *add* must be used to add the user to the group. Clicking on the button *Save* sends all modifications to the RMS application and closes the dialog.

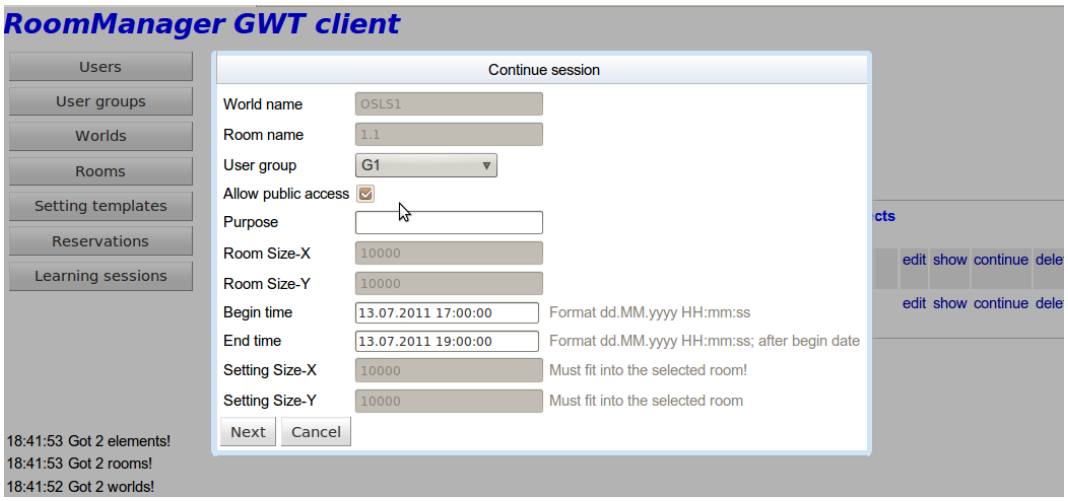


Figure 5.16: RoomManager Client - edit permission restrictions

Once the user group has been configured, it can be assigned to a learning session. Assigning a group to the session limits access to the room to the users which are members of the selected group and which have the permission to enter the room. This restriction just applies to rooms with walls, rooms that have no walls do not have an access restriction. Figure 5.16 shows the screen *Learning sessions* after clicking on the option link *edit* which is used to modify a learning session. The user group has been set to group *G1* and *Allow public access* has been enabled. This means that access to the room is not restricted but the content of the collaborative tool can just be modified by users which have the permission to write on the tools.

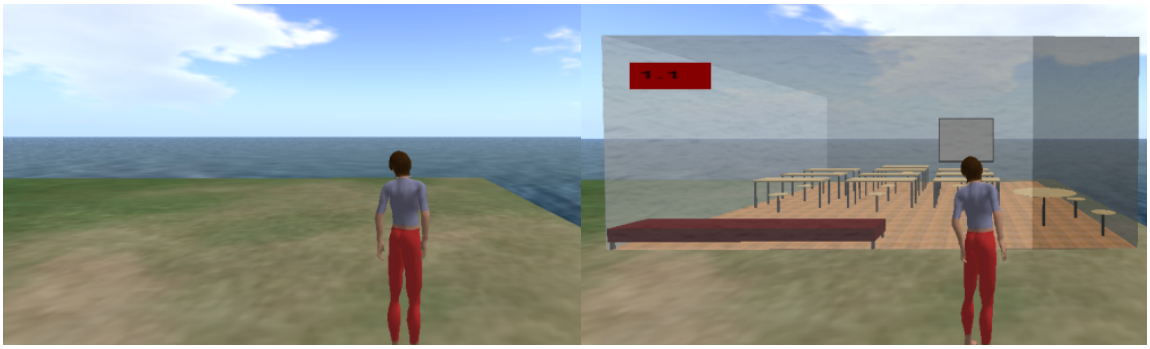
5.6 OpenSimulator Realisation Usage

The following sections describe the functionalities of the WRC in detail. All provided screen-shots are either taken with the *Second Life viewer* (version 2.7.4.235167) or the *Phoenix viewer* (version 1.5.2.1102). The SL viewer has the capability of using the col-

laborative media wall but does not show textures of the avatar. The Phoenix viewer does provide all available functionalities of the prototype except the collaborative media wall.

5.6.1 Build Learning Settings

As described earlier in section 5.2.1, the WRC receives a message with the action *SERVER_BUILD_ROOM_REQUEST* in order to build a room. The message contains information about the room, the access restriction and the setting objects. The region module automatically creates the necessary objects and scripts and adds them to the scene. Details about the different types of created objects are described in the following sections.



(a) Before building the room **(b)** After building the room

Figure 5.17: RoomManager World Realisation - room building

Figure 5.17a shows an image of the region before the RMS application sends a build room request message. The earlier shown classroom example has been used. After receiving the build room request, the region module creates all objects. The result can be seen in figure 5.17b. The picture shows a room which consists of the following objects:

- Transparent walls
- A sliding door at the right side
- A sign showing the room name at the left side
- Several different objects: chairs, tables, a bench and a media wall

5.6.2 Remove Learning Settings

When receiving a message with the action *SERVER_DESTROY_ROOM_REQUEST*, the application removes all created objects which belong to the given learning session. The region does again look like figure 5.17a after the built room has been destroyed.

5.6.3 Static Objects

When building a learning session the region module first of all creates the walls of the room. As described earlier there are three wall types available which are solid, transparent and no walls. If the room does not have walls, then just a small wall is created which contains the room sign. The other wall types differ only in their texture. Figure 5.18 shows all three types of rooms. Starting from right to left, the walls types are solid, transparent and no walls.

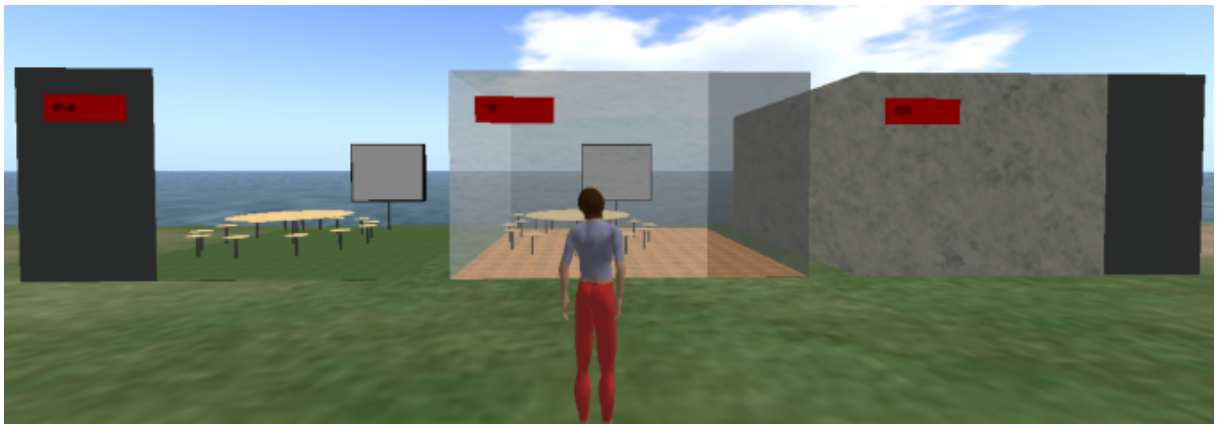


Figure 5.18: RoomManager World Realisation - wall types

In the next step, the sliding door is created. The door differs from a wall just by its texture and colour. The door can be opened and closed by touching it. The door opens by sliding to the left side. It will close automatically after a few seconds, if the avatar does not close it.

Finally all setting objects which have been configured will be created by the region module. All objects are made up of several primitive objects with different shapes, textures and colours. The static setting objects like chairs and tables do not contain any scripts or other interaction functionalities. The region module modifies the sit position of all objects so that the avatar sits in the middle of the objects. Figure 5.19 shows the avatar sitting on a round chair object.

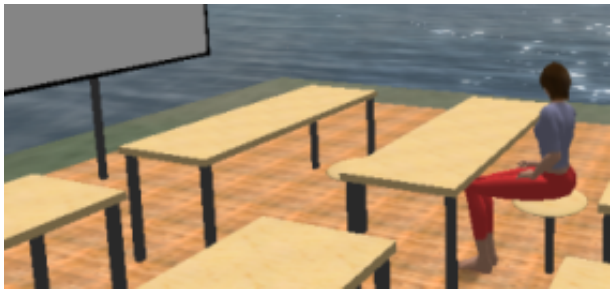


Figure 5.19: RoomManager World Realisation - sit position

5.6.4 Collaborative Flip Chart Tool

This section describes the collaborative tool *Flip Chart*. The tool can be used for collaborative interaction. It contains several pages and supports different types of drawing commands. Figure 5.20 contains an image of the flip chart tool. During this master's thesis two versions have been implemented.

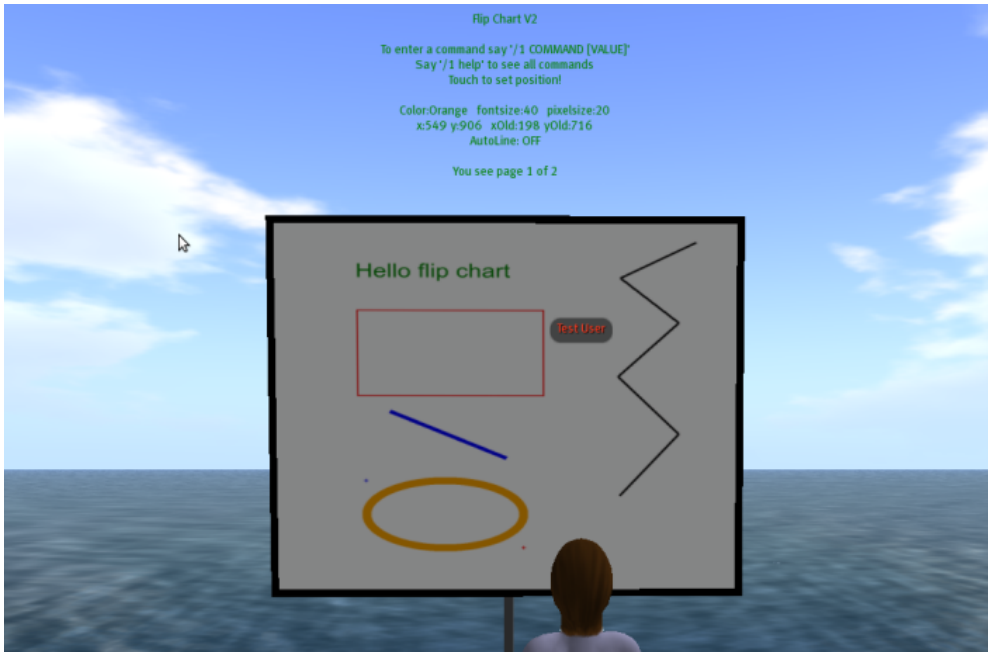


Figure 5.20: RoomManager World Realisation - flip chart

The functionalities of the first versions are:

- Changing the color, pen size and pixel size
- Adding text
- Drawing lines, rectangles, filled rectangles and ellipses
- Setting the position by touching the content of the wall
- Navigation to the first, next, previous and last page
- Limited to a maximum of 10 pages
- The first version does not store the content on the RMS application.
- Provides a continuous line drawing mode. Each touch on the wall connects the last position with the new position by a line.

Additionally to these functionalities, the second improved version, *Flip Chart V2*, supports the following features:

- The number of pages is not limited. Each request to access the next page when showing the last page adds a new page to the tool.

- The content of the tool will be send to the RMS application after each modification.
- The content can not be modified by avatars which do not have the permission to write on the tool.

There are two ways to communicate with the flip chart in order to change its content. The avatar can touch the content of the wall in order to set the cursor to a different position. The tool remembers the previous position, which is used by drawing functions. The current position can be identified by red crosslines and the previous position by blue crosslines. For all other interaction, messages must be send to the tool.

The tool shows usage information directly over the wall. This information contains the communication channel which has to be used in order to communicate with the tool. A command has to be send which has the format */CHANNEL COMMAND [VALUE]*. The different supported commands are described in table A.7. In order to prevent the user from typing long commands, a short version is available for each command except the *clearpage* command.

5.6.5 Collaborative Picture Wall Tool

This section describes the collaborative tool *Picture Wall*. The tool can be used to show images or slide shows to the audience. Several different images can be configured. During this master's thesis two versions have been implemented. In both versions it is possible to navigate to the first, next, previous and last configured picture. Figure 5.21 shows a picture wall which shows a screen shot of the web-page of the IICM.

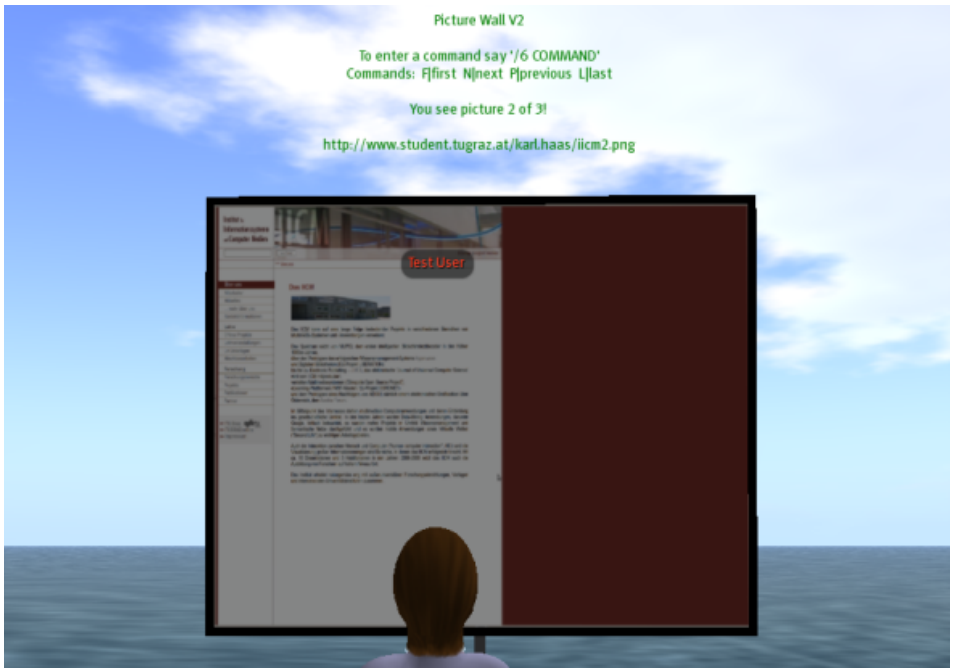


Figure 5.21: RoomManager World Realisation - picture wall

In order to navigate, there are two possible ways which are similar to those of the flip chart tool. By touching the content of the wall, the next image will be shown. If the picture wall already shows the last picture, then it will show the first picture again. Further it is possible to send messages to the wall. The wall shows the available commands directly above the wall. The available commands can be found in table A.8.

Both implemented versions save the index of the last viewed image on the server.

5.6.6 Collaborative Media Wall Tool

The third implemented collaborative tool is called *Media Wall*. It uses the functionality Media On A Prim (MOAP) of the new Second Life viewer, which is described in section 3.2.3. It can be used to show content which is available on the web. Figure 5.22 shows a media wall, which accesses the web-page of the IICM.

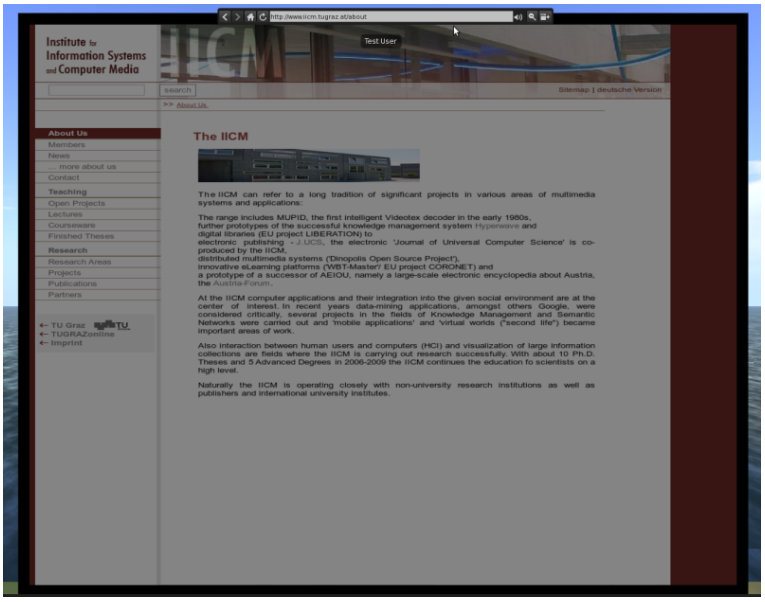


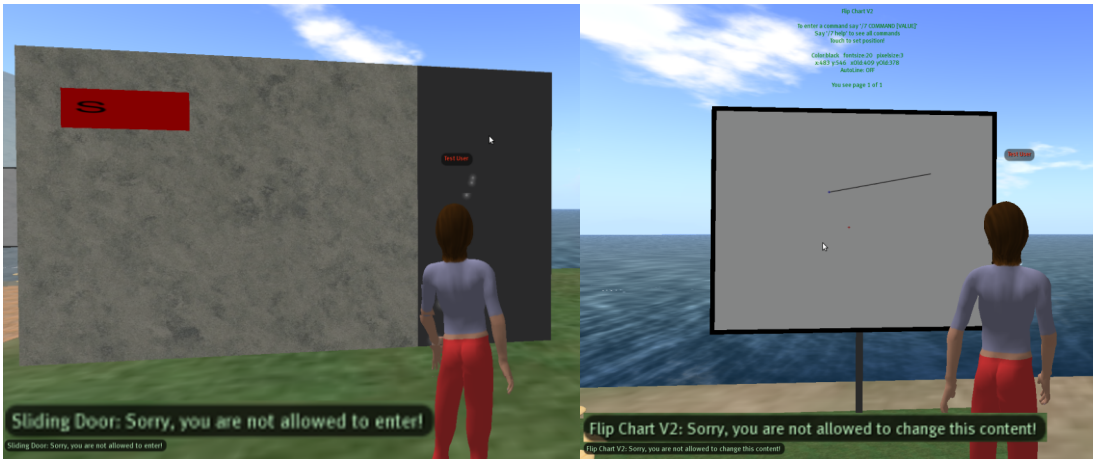
Figure 5.22: RoomManager World Realisation - media wall

At the address bar at the top of the wall it is possible to enter and access a different URL. Further it is possible to go to the last page, go to the previous page, reload the current page, change the volume, and to zoom in or out. The region module stores the last accessed URL by sending it to the RMS component.

5.6.7 Access Permissions

The implemented prototype provides two access restrictions. The sliding door which will be added to rooms with solid or transparent walls, can only be opened by users which have the permission to enter the learning session. If the learning session is not restricted to a user group, then each avatar can open the door. The door can be opened by touching

it. When opening, it slides to the left side and closes again after a few seconds by sliding back to its original position. The avatar can also close it by touching it again. If the user is not allowed to open the door, then the door will send a notification to the avatar which can be seen in figure 5.23a. The notification message can be seen at the lower left corner of the image.



(a) Sliding door access denied **(b)** Collaborative tool access denied

Figure 5.23: RoomManager World Realisation - access denied notification

The second implemented access restriction prohibits avatars from changing the content of the *Flip Chart V2* and the *Picture Wall V2* tools. If the learning session is assigned to a user group, then each user which should be able to change the content of those tools must have the permission to write. Users without permission trying to change the content of the tools receive a notification which can be seen in figure 5.23b.

Further access restrictions like preventing users from viewing the content of the tools have not been implemented in the scope of this work. Restricting reading access to the tool would be a challenging task. The content of the flip chart and the picture wall tool are created by assigning a new texture to the face of the object each time the content has been modified. The used version of the simulator did not offer the functionality to restrict the presentation of textures.

5.7 Configuration of the Prototype

This chapter describes the configuration capabilities of the implemented prototype. The following sections describe the configuration of the three implemented components.

5.7.1 RoomManager Server

The implemented server application provides a property configuration file with the name *roommanager.properties*. Listing 5.7 illustrates an example of the configuration.

```
# Log4J
log4j.rootCategory = INFO, S
log4j.appender.S = org.apache.log4j.ConsoleAppender
log4j.appender.S.layout = org.apache.log4j.PatternLayout
log4j.appender.S.layout.ConversionPattern = %d{dd/HH:mm:ss.SSS} | %-6.6t | %-5.5p |
    %-80m %C.%M#%L%n

# HSQLDB
hsqldb.port = 9001
hsqldb.dbname = roommanager
hsqldb.file = ./hsqldb_roommanager

# Hibernate
hibernate.dialect = org.hibernate.dialect.HSQLDialect
hibernate.connection.driver_class = org.hsqldb.jdbcDriver
hibernate.connection.url = jdbc:hsqldb:hsql://localhost:9001/roommanager
hibernate.connection.username = sa
hibernate.connection.password =
hibernate.show_sql = true
hibernate.format_sql = false
hibernate.hbm2ddl.auto = update
hibernate.current_session_context_class = thread

# Restlet
restlet.port = 9002

# World connector
worldconnector.port = 9003
```

Listing 5.7: Prototype configuration - RoomManager Server

The following sections describe the different areas of the configuration file.

Log4J

The first section describes the format of the logging information which are printed to standard out after starting the server application.

HSQLDB

The server application automatically starts the HSQLDB database server during start-up. The server port number, the database name and the location of the database files have to be configured.

Hibernate

The application uses the Java persistence framework Hibernate to store and access the data. This section configures the framework, the configured port number and database

name in the property *hibernate.connection.url* must match the properties *hsqldb.port* and *hsqldb.dbname* of the section HSQLDB.

Restlet

The property *restlet.port* contains the server port number to which the RMC applications have to connect. The server application will provide the RESTful API on this port.

World Connector

The property *worldconnector.port* contains the server port number to which the WRCs have to connect to.

5.7.2 RoomManager Client

The RMC application has been implemented as a web application. The application contains a file with the name *web.xml*. This file contains the configuration which will be loaded by the web application server (e.g. Apache Tomcat). Listing 5.8 shows the content of the provided web.xml file.

```
 <?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE web-app
    PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
    "http://java.sun.com/dtd/web-app_2_3.dtd">

<web-app>

  <!-- Servlets -->
  <servlet>
    <servlet-name>basicDataAccessService</servlet-name>
    <servlet-class>tugraz.iicm.roommanager.gwtclient.server.service.
        BasicDataAccessServiceImpl</servlet-class>
    <init-param>
      <param-name>server</param-name>
      <param-value>http://localhost:9002</param-value>
    </init-param>
  </servlet>

  <servlet-mapping>
    <servlet-name>basicDataAccessService</servlet-name>
    <url-pattern>/gwtclient/basicDataAccess</url-pattern>
  </servlet-mapping>

  <!-- Default page to serve -->
  <welcome-file-list>
    <welcome-file>gwtclient.html</welcome-file>
  </welcome-file-list>

</web-app>
```

Listing 5.8: Prototype configuration - RoomManager Client

The client application needs to know the Internet Protocol (IP) address and the port number of the server application. Both values are specified in the parameter with the

name *server*. This URL must match the IP address and the configured server port number (property *restlet.port*) of the RMS application.

5.7.3 OpenSimulator Region Module

The configuration file *TURoomManagerModule.ini* of the implemented region module contains the information about the OpenSim regions which have to be managed. An example configuration file which manages the two regions *OSLS1* and *OSLS2* contains the content shown in listing 5.9.

```
[TURoomManagerModule]

        Regions = "OSLS1,OSLS2"
        AdminUUID = "d43bf59a-83c4-42ed-837d-01a871a1cbb9"

        OSLS1.ServerHost = "127.0.0.1"
        OSLS1.ServerPort = "9003"

        OSLS2.ServerHost = "127.0.0.1"
        OSLS2.ServerPort = "9003"
```

Listing 5.9: Prototype configuration - OpenSimulator Region Module

The parameter *Regions* contains a comma-separated list of all managed regions. *AdminUUID* is the UUID of the user which will be used to create all objects. For each managed region the parameters *REGION_NAME.ServerHost* and *REGION_NAME.ServerPort* must be configured. *ServerHost* specifies the IP address of the RMS application and *ServerPort* specifies the port number on which the server provides the world communication interface (property *worldconnector.port*). Each region establishes a connection to the RMS application. Thus it would be possible to connect to different server applications within one simulator.

5.8 Lessions Learned

This section deals with the gain in experience in the area of creating flexible learning settings in the virtual 3D world OpenSimulator. The initial goal of the prototype, which has been developed during this master's thesis, was to identify the capabilities of the open source virtual world platform.

OpenSimulator

The following list describes the experience gained during the development of the OpenSim region module:

- The OpenSimulator community provides a Wiki system which contains information about development in OpenSimulator. Due to the alpha status of the project and

the fast changing source code, the information is outdated and thus not always very helpful. The documentation is by far not complete. Some documents in the wiki provide just a basic overview about the topics but the sometimes needed deeper look into the topic is often missing.

- There is already a growing community available. During the development several questions have been asked in the user mailing list of the project. The majority has been answered fast and in detail. A lot of information is spread over the Wiki, mailing lists and blogs.
- Creating simple objects and adding them to the region was an easy task because of the available documentation. Adding more complex objects and improving the functionality of the objects was sometimes a much more painful task. Many exceptions (e.g. NullPointerExceptions) have occurred because framework methods were called in the wrong sequence. Due to the missing documentation this could just be solved by analysing the source code of the framework.
- After overcoming some hurdles because of the missing documentation, adding different textures, colours and behaviour could be done with little effort. The best approach was to first select textures and colours with a client and then assign it by its UUID or colour value to the primitives within the region module.
- The simulator source code provides many powerful modules which can be accessed from your own developed region module. This features helped a lot when developing the region module. In order to find the right functionality it was often necessary to search through the existing source code of the project. Many example codes have been found in the existing source of the core project and have been used in a modified way.
- In order to implement a first version of the collaborative learning tools, the functions of LSL and OSSL have been used. The majority of the LSL functions are implemented in OpenSim and can be used exactly like in Second Life. Thus existing LSL scripts can be used in OpenSim with little effort. Heaps of information about writing scripts with LSL is available on the web. Due to the not yet complete documentation of OpenSim, the already available documentation for Second Life is a convenient way to start.

  Additionally the OSSL functions provide further functionalities offered by the simulator. For example the dynamic texture drawing functions have been used to draw elements and images as a texture on the created interactive walls. Once the basic concepts of scripting in the virtual world have been understood, it is an easy task to write scripts in OpenSim.

  Because of the fact, that all objects are created by a region module, the script also needs to be assigned by the region module to the created object. Adding the script to the object was a difficult task and was just solved by finding a similar solution on the web.
- In order to save the content of the collaborative tools and to manage the permissions dynamically, a second version of the tools has been implemented directly within the

region module without adding any scripts. As soon as it is figured out how to access other region modules and which functionalities they already offer, it is possible to use the powerful features of the simulator in your own written module. Implementing the functionality within the region module has the advantage, that all provided features of the programming language C# and all features of the simulator can be used.

- During the development several different viewer applications have been used. All viewers crashed from time to time and updates had to be made very frequently. The new Second Life viewer was used in order to use the MOAP functionality. The viewer was not able to download the clothing information from the server and because of that the avatar has not been displayed correctly.
- The simulator supports a media streaming functionality which can be used by all available viewers. This streaming functionality is limited to one media content per region. Due to this fact it could not be used by the prototype.
- The initial set-up and configuration of the simulator has been an easy task. Not much effort was needed to start the standalone version of the simulator because of the provided example configuration files. The prototype has just been tested in standalone mode, it has not been tested in an OpenSim grid.
- Restricted viewing access to the collaborative tools is not possible. No permission system for textures is available or has been found. Also a core developer replied on a question in the user mailing list that such a permission system is not available and would be a major change.
- Another problem concerning textures is the fact, that dynamically created textures are not shown correctly to users which are not logged in during the creation of the texture. Textures which were created before an avatar logged in, are not displayed to the user.
- New versions of the viewers are available very frequently. Due to the fact that the binary packages are quiet big (about 50 MB), keeping the viewer up to date can be an annoying task.
- During testing the prototype, the simulator crashed from time to time with a segmentation fault. Maybe those faults where caused by an incorrect use of the framework code, but either way, this should not cause the server to terminate exceptionally.

RoomManager Prototype

Some important characteristics and limitations of the provided proof of concept are discussed in the following points:

- Regions are limited to a maximum size of 256 meters in length and width. The prototype does not manage rooms which overlap the borders of a region, the RMC application has to take care that the room fits into the region. In order to manage

bigger rooms or rooms which overlap regions, the implemented region module should be changed to a shared region module. In this case the module would manage all configured regions and not just one. At the moment a new region module is instantiated for each configured region.

- Adding new learning tools to the application is an easy task. The tool must be added to the web application by adding an icon and a short implementation to manage the optional properties of the tool. The region module implementation depends on the functionality of the tool. A class must be added to the region module which is responsible for the creation of the 3D objects which represent the tool and the management of the content of the tool.
- The prototype is very flexible, the number of managed regions and rooms is not limited at all. A new managed world (eg. an OpenSim region) can be added by using the RMC web application during runtime of the application. The terrain level of the new OpenSim region must be set to a height of 21 meters.
- The approach of three separate components seems to be very flexible, the application is not limited to the use of OpenSimulator. Because of the fact, that the configuration and reservation mechanism is not part of the simulator, the virtual world platform could be exchanged with little effort.
- The region module does not monitor location changes of the managed tools and static objects. If for example an avatar moves a chair to a different position, this action is not recognised by the region module and the new location is not stored by the RMS application. If the learning session will be continued at any later time, the object will be placed at the original position. A possible solution would be to check the position of the tools periodically and to send the position to the RMS component after each location change.
- The access to the RMC web application is not secure at the moment, no authentication is necessary. Also the communication channels between the components are not encrypted. The application is a proof of concept and has not been tested with big amount of data.

Used Framework and Tools

The following list summarises the impression that the used frameworks and tools have made during the development of the prototype.

- The object relational mapping framework Hibernate has been used to store the created objects in a database. All needed functionalities worked well and without big effort. Due to the fact that the data model of the prototype is very simple and small amounts of data has been used, not much can be said about the performance of the framework.
- Using JSON as data transmission format and Gson to convert the objects to JSON strings and vice versa was very simple. As long as performance is not the main

concern (Gson uses reflection to serialise the data) and the objects do not have circular references this approach seems to be very suitable. Gson is not able to serialise objects with circular references. The build-in JSON capabilities of the Mono development framework also allow serialising and de-serialising the objects with very little effort.

- The Google Web Toolkit (GWT) framework provides powerful functionalities to develop web applications in an object oriented context. In some cases it has been difficult to find the reason why the application does not load and the browser stays empty. Adding add-ons to the used browser which shows Java script errors, helped to find the reason for such errors. Using and modifying a default GWT layout has been felt as not very intuitive and complicated.
- GWT does not support drag and drop out of the box. The add-on *gwt-dnd* has been used to provide the drag and drop editor. The extension is compact and easy to use. All of the needed drag and drop functionalities have been implemented by using this extension.
- The Restlet framework has been used to provide a RESTful API to the RMC applications and to access the world realisation API. The documentation of the framework was very limited but the available examples helped to get started. The needed functionalities have been offered by the framework and implementing the API has been found comfortable.
- The author felt that using MonoDevelop to implement the region module was sufficient but not very comfortable.

Chapter 6

Summary and Outlook

During this master's thesis a proof of concept has been developed which provides an easy to use learning setting configuration and management application. The work illustrates the capabilities of the open source virtual world platform OpenSimulator for automatic learning session creation. For educational institutions, OpenSimulator has some advantages compared to Second Life, which are mainly caused by its free and open source license. The high degree of extensibility of the core functionalities enables the developer to adjust the virtual worlds according to his needs.

By using the developed approach, teachers without deep technical knowledge can create learning scenarios in virtual spaces. A web application with a simple drag and drop editor is available to configure the learning settings. The application provides mechanisms to easily continue previous learning sessions by reconstructing the state of an old session. This feature reduces the effort needed to reconstruct the state of the session.

A detailed literature review has shown the importance of collaboration and interaction in E-Learning. Interactive tools have been implemented which offer the required support of collaboration in order to encourage the students to actively participate in the learning process.

The application has a very flexible architecture, the initial version contains three main components. Apart from the RoomManager Server component, which is the core component and stores all needed information, the RoomManager Client and the implemented OpenSimulator region module could be replaced or extended in order to support further platforms. Future extensions of the application could provide an user interface for mobile devices in order to change the learning configuration. Regions of other virtual world platforms could be managed by integrating the desired virtual world architecture by adding an additional world realisation component.

At the moment, the rooms can not overlap regions within the virtual world. By improving the implementation of the region module, it would be possible to manage very big rooms. Another drawback of the current implementation is the fact, that reading access of the content of the learning tools can not be restricted. Such limitations which are caused by

the virtual world platform could be avoided by using other platforms.

The learning experience could be further improved by adding more collaborative learning tools to the application. For example a video conference board could be added in order to connect a virtual meeting with a meeting in real world.

Appendix A

Implementation Details

A.1 RoomManager Server

A.1.1 Managed Objects

Object	Description
User	Contains the information of a managed user. The prototype relies on the property *username* which must contain the name of the user's avatar in OpenSim, the first and second name of the avatar separated by one space character.
UserGroup	Contains the information of a managed group of users. The property *groupname* identifies the user group in the client application. The group consists of a list of *UserGroupMember* entities which represent a user and its access permissions.
UserGroupMember	Membership information of a user within a group. Contains the permissions of an user within a specific group. The permission *enter* specifies if the user is allowed to enter the room, *write* must be set in order to allow the user to change the content of collaborative tools.
World	Identifies a managed word, which in OpenSim is a region. The region name in OpenSim must match the name of the world. *sizeX* and *sizeY* specify the size of the world in millimetres. The size of one region in OpenSim is 256 x 256 meters. In order to use the full size of the region, the sizes should be set to 256000. *connected* will be set to true by the server, if a world realisation component is connected, which manages the configured region or world.

Object	Description
Room	Describes one room within a world. *sizeX* and *sizeY* specify the size of the room in millimetres. *posX* and *poxY* specify the position of the room within the world.
Setting	A setting describes a set of objects (static objects and collaborative learning tools). *sizeX* and *sizeY* specify the size of the setting in millimetres. This entity describes the content of a learning session. It is independent of a room and can be used in every room which has at least the same size as the setting. Thus the prototype provides the flexibility to use a given setting in different rooms.
SettingObject	Represents one object within a setting. The size of an object, the position within the setting, the content and the orientation is stored. Further each Setting Object has a type. The type specifies which kind of object has to be created. Several different setting object types are delivered with the prototype. Static objects like chairs and tables and collaborative tools like an interactive flip chart or a picture wall are included. The enumeration *Type* contains all included types of setting objects.
RoomSetting	A *RoomSetting* can be seen as a room reservation or a learning session. The entity contains the information which *Setting* has to be created in which *Room* during a specified period of time. Further the access can be restricted to a group of users or public room access can be enabled by setting *publicAccess* to true.
ServerInfo	Describes the version of the server.

Table A.1: RoomManager Server - managed objects

A.1.2 Client API - Protocol Elements

Element	Description
serverInfo	Contains the server information as specified by the entity *ServerInfo*.
status	Indicates if a request has been processed successfully. The status *SUCCESSFUL* will be send if the request has been processed without errors. In case of an error an other value will be set. See table A.3 for all possible status values.

Element	Description
reason	In case of an error this element gives some detailed information.
rooms	Contains an array of rooms. If a list of rooms has been requested, the found rooms are stored in this array. If a single room has been requested or a room has been added or modified, then this array contains just one room object.
worlds	Contains the array of requested worlds or an array with just one world if the world has been added or modified.
settings	Contains the array of requested settings or an array with just one setting if the setting has been added or modified.
roomSettings	Contains the array of requested room settings or an array with just one room setting if the room setting has been added or modified.
users	Contains the array of requested users or an array with just one user if the user has been added or modified.
userGroups	Contains the array of requested user groups or an array with just one user group if the user group has been added or modified.

Table A.2: RoomManager Server - Client API - response elements

A.1.3 Client API - Response Status

Status	Description
SUCCESSFUL	The request has been processed without any errors.
GROUP_NAME_ALREADY_IN_USE	The name of the user group which should be added is already in use.
WORLD_NAME_ALREADY_IN_USE	The name of the world which should be added is already in use.
ROOM_NAME_ALREADY_IN_USE	The room name of the room which should be added is already in use.
ROOMS_OVERLAPPING	The room which should be added or modified overlaps an already existing room.

Status	Description
ROOM_OVERLAPS_WORLD	The room which should be added or modified overlaps the borders of the world.
ROOM_OCCUPIED	The room for which a reservation should be added is already occupied during the specified period of time.
INTERNAL_ERROR	Any other error occurred (e.g. an unhandled exception).

Table A.3: RoomManager Server - Client API - response status

A.1.4 Client API - Supported Requests

URL	Method	Description
/serverinfo	Get	Request to retrieve information about the RoomManager server. This message is used to check if a connection to the server can be established.
/worlds	Get	Returns a list of all currently configured worlds.
/world	Put	Adds the new world or saves the modified world. A JSON representation of the world entity must be send in the body of the request. The saved world will be send back in the response.
*/world/**id***	Get	Returns the world with the given id or an empty world array if no world with the given id exists.
*/world/**id***	Delete	Deletes the world with the given id. The response does not contain the deleted world.
*/world/**wid**/rooms*	Get	Returns all rooms of the world with the given world id. If the world does not exist or the world does not contain any rooms, then the array is empty.
/rooms	Get	Returns all available rooms.

URL	**Method**	**Description**
/room	Put	Adds a new room or saves a modified room. A JSON representation of the room must be send in the payload of the request. The saved room will be send back in the response. The server checks, if a room with the given room name already exists and returns the status *ROOM_NAME_ALREADY_IN_USE* if the name is already used. If the room overlaps the borders of the world, then the server returns *ROOM_OVERLAPS_WORLD*. Further the server checks, if the room overlaps an other existing room (*ROOMS_OVERLAPPING*).
*/room/**id***	Get	Sends the room with the given id in the response or an empty room array if no room with the given id exists.
*/room/**id***	Delete	Deletes the room with the given id, if it exists. The room will not send back in the response.
/settingtemplates	Get	Sets all setting templates in the response.
*/settingtemplates/**x**/**y***	Get	Sets all setting templates in the response which are not bigger then the given size specified by x and y.
/setting	Put	Adds a new setting or saves an existing setting. In order to save setting templates, this request has to be used. The saved setting will be send back in the response.
*/setting/**id***	Get	Returns the setting with the given id or an empty array if no setting with the given id exists.
*/setting/**id***	Delete	Deletes the setting with the given id. The setting will not be send in the response.
*/world/**wid**/reservations*	Get	Searches for all reservations of all rooms of the world with the given id. Returns an empty array if no reservations can be found.
*/room/**rid**/reservations*	Get	Searches for all reservations of the room with the given id. Returns an empty array if no reservations can be found.

URL	**Method**	**Description**
/reservations	Get	Searches for all reservations. Returns an empty array if no reservations can be found.
/reservation	Put	Adds a new reservation or saves a modified reservation. If another reservation during the set period of time exists, then the status *ROOM_OCCUPIED* will be send back.
*/reservation/**id***	Get	Returns the reservation with the given id or an empty array if no reservation with the given id can be found.
*/reservation/**id***	Delete	Deletes the reservation with the given id. The deleted item will not be returned.
*/world/**wid**/roomsettings*	Get	Searches for all room settings of all rooms of the world with the given id. Returns an empty array if no room settings can be found.
*/room/**rid**/roomsettings*	Get	Searches for all room settings of the room with the given id. Returns an empty array if no room settings can be found.
/roomsettings	Get	Returns an array of all room settings or an empty array if no room settings are available.
/roomsetting	Put	Adds a new room setting or saves a modified room setting. If another room setting during the set period of time exists, then the status *ROOM_OCCUPIED* will be send back.
*/roomsetting/**id***	Get	Returns the room setting with the given id or an empty array if no room setting with the given id can be found.
*/roomsetting/**id***	Delete	Deletes the room setting with the given id. The deleted item will not be returned.
/users	Get	Retrieves all available users or an empty array if no users are configured.
/user	Put	Adds a new user or saves a modified user. The saved user will be send in the response.
*/user/**id***	Get	Returns the user with the given id or an empty array if no user with the given id is available.

URL	Method	Description
*/user/**id***	Delete	Deletes the user with the given id. The deleted user will not be send in the response.
/usergroups	Get	Returns a list of all available user groups or an empty array if no groups are available.
/usergroup	Put	Adds a new group or saves a modified user group. The group will be send in the response.
*/usergroup/**id***	Get	Returns the user group with the given id or an empty array if no user group with the given id can be found.
*/usergroup/**id***	Delete	Deletes the user group with the given id. The deleted user group will not be send in the response.

Table A.4: RoomManager Server - Client API - URLs

A.1.5 World API - Message Elements

Element	Description
action	Describes the type of the message. All supported message actions are described in detail in table A.6
worldName	Contains the name of the managed world. Has to be send by the client when connecting to the server.
heartbeatInfo	Contains a heart beat information which will be send from the client to the server. The server replies the same heart beat info.
serverInfo	Contains information about the RoomManager Server.
roomSetting	Contains the room setting which should be build or destroyed. For an overview of the containing elements please refer to the entity *RoomSetting* in figure 5.2.
settingObject	Contains a *SettingObject* as described in table A.2. This element will be send from the client to the server when the content of a collaborative tool has been modified.

Table A.5: RoomManager Server - World API - message elements

A.1.6 World API - Action Values

Action	Description
CLIENT_CONNECT	The WRC sends a message with this action to the server in order to register itself for a given world. The element *worldName* needs to be send. If the world is available, the server answers with a *SERVER_CONNECT_SUCCESSFUL* message. If the world is not configured at the server, then the server answers with an *SERVER_CONNECT_REQUEST* message.
CLIENT_DISCONNECT	The WRC can send this message to tell the server that it does not manage the world any more.
CLIENT_HEARTBEAT	To check if the socket connection is still working, the WRC can send this message with any value in the element *heartbeatInfo*. The server just replies the content in the message *SERVER_HEARTBEAT_ACK* if it receives the message.
CLIENT_UPDATE_SETTING_OBJECT	If the content of a collaborative tool has been modified, the WRC has to send this message in order to save the status of the object. The message must contain the object in the element *settingObject*.
SERVER_HEARTBEAT_ACK	Answer message which is sent by the RoomManager Server after receiving a *CLIENT_HEARTBEAT* message.
SERVER_CONNECT_SUCCESSFUL	Answer message which is sent by the RoomManager Server after a successful login by a WRC.

Action	Description
SERVER_CONNECT_REQUEST	Answer message which is sent by the RoomManager Server after a not successful login by a WRC. This is the case, if the element *worldName* is not set in the login message or if a world with this name is not available on the server.
SERVER_SHUTDOWN	The *RoomManager Server* sends this message to all connected WRCs before closing the server socket.
SERVER_BUILD_ROOM_REQUEST	In order to build a room (and all its objects), the server sends this message to the WRC which manages the world of the room. All necessary information to build the room is available in the element *roomSetting*. It contains the room information and all setting objects.
SERVER_DESTROY_ROOM_REQUEST	In order to destroy a room, the server sends this message to the WRC.

Table A.6: RoomManager Server - World API - action values

A.2 OpenSimulator Region Module

A.2.1 Flip Chart Commands

Command	Short command	Description
next	N	Navigates to the next page and shows the content of the new page. The improved version V2 adds a new page to the tool, if the last page is displayed.
previous	P	Shows the previous page.
first	F	Shows the content of the first page.
last	L	Navigates to the last page.

Command	**Short command**	**Description**
delete	d	Deletes the last added command from the current page.
clearpage		Deletes the content of the current page. All drawing elements and texts will be removed.
color	c	Sets the colour for the next command. The name of the colour has to be set in the value of the command.
line	l	Connects the current position of the cursor and the previous position by a line. The pixel size and the set colour will be considered.
rectangle	r	Uses the current position of the cursor and the previous position to draw a rectangle. On position is the upper left corner and the other one is the lower right corner of the rectangle. The pixel size and the set colour are considered.
filledrectangle	f	Draws the same rectangle as the command *rectangle* does. The rectangle will be filled with the set color.
ellipse	e	Uses the current position of the cursor and the previous position to draw an ellipse. One position is the upper left corner and the other one is the lower right corner of the ellipse. The pixel size and the set colour are considered.
autoline	a	Activates or deactivates the continuous line drawing mode. If enabled, each touch on the wall connects the last two touched positions by a line. The colour and the pixel size are considered.
size	s	Sets the text size to the given value.
pixelsize	p	Sets the pixel size for drawing commands to the given value.
text	t	Adds a text to the wall. The font size and the colour is considered.
colorlist	C	Prints available colours above the wall.
help	h	Prints all available commands above the wall

Table A.7: RoomManager World Realisation - flip chart commands

A.2.2 Picture Wall Commands

Command	Short command	Description
next	N	Shows the next image. If already the last image is displayed, then the wall moves to the first picture.
previous	P	Shows the previous image. If already the first image is displayed, then the wall moves to the last picture.
first	F	Shows the first image.
last	L	Navigates to last image.

Table A.8: RoomManager World Realisation - picture wall commands

A.2.3 Picture Wall LSL Script

```
integer CHANNEL_LISTEN = 3;
list    pictures = [];
list    comments = [];
string  contentType = "image";
integer index = -1;
integer MIN_INDEX = 0;
integer MAX_INDEX = -1;

// Show the picture with the given index
showPicture(integer newIndex)
{
    if (newIndex > MAX_INDEX)
        return;
    if (newIndex < MIN_INDEX)
        return;
    if (newIndex == index)
        return;

    index = newIndex;
    osSetDynamicTextureURL("", contentType , llList2String(pictures, index) , "",
        5000);
    showStatus();
    llSay(CHANNEL_LISTEN, "" + index);
}

// Show the status text
showStatus()
{
    string status = "Picture Wall \n \n";
    if (MAX_INDEX == -1)
        status += "No pictures available!";
    else
    {
```

```
        status += "To enter a command say '/" + (string)CHANNEL_LISTEN + " COMMAND' \n
            ";
        status += "Commands:  F|first  N|next  P|previous  L|last \n \n";
        status += "You see picture " + (index + 1) + " of " + (MAX_INDEX + 1) + "\n \n
            ";
        status += llList2String(comments, index);
    }
    llSetText(status, <0.0, 0.5, 0.0>, 1.0);
}

// Left side of a seperator
string left(string src, string divider) {
    integer index = llSubStringIndex( src, divider );
    if(~index)
        return llDeleteSubString( src, index, -1);
    return src;
}

// Right side of a seperator
string right(string src, string divider) {
    integer index = llSubStringIndex( src, divider );
    if(~index)
        return llDeleteSubString( src, 0, index + llStringLength(divider) - 1);
    return src;
}

default
{
    state_entry()
    {
        llListen(CHANNEL_LISTEN, "", NULL_KEY, "");
        showPicture(0);          showStatus();
    }

    // The callback method which will be called when receiving a text
    // on the specified channel
    listen(integer channel, string name, key id, string message)
    {
        string command = left(message, " ");
        string value = right(message, " ");
        if (command == "next" || command == "N")
            showPicture(index + 1);
        else if (command == "previous" || command == "P")
            showPicture(index - 1);
        else if (command == "first" || command == "F")
            showPicture(0);
        else if (command == "last" || command == "L")
            showPicture(MAX_INDEX);
        else
        {
            integer newIndex = (integer) value;
            if (newIndex > 0)
                showPicture(newIndex - 1);
        }
    }

    touch_start(integer numDetected)
    {
        if (index == MAX_INDEX)
           showPicture(0);
        else
           showPicture(index + 1);
    }
}
```

Listing A.1: RoomManager World Realisation - picture wall script

Appendix B

Prototype Installation

This chapter describes how to install the prototype on an Ubuntu Linux system. The instruction describes all steps of the installation from downloading the operating system until starting and using the prototype.

B.1 Environment Set-up

This section describes how to install the operating system and the required additional software packages.

Operation System Installation

Download the Ubuntu 11.04 32 bit Linux distribution and follow the installation instruction. The download can be found at the Ubuntu Releases[1] web page. During this installation instruction the user *rm* with password *rm* has been used. The user can be added during the installation of the operating system.

C# Set-up

To compile the OpenSim region module, a C# compiler is needed. Further the mono development framework is needed to start the simulator. The command, which is shown in listing B.1, installs the .Net build tool nant and the complete mono development environment which contains the C# compiler.

```
sudo apt-get install nant mono-complete
```

Listing B.1: Prototype installation - setup Mono

[1] http://releases.ubuntu.com/

Java Set-up

In order to compile the RoomManager Server (RMS) and RoomManager Client (RMC) applications, Sun Java 6 SDK and the Apache build toolkit Ant are needed. To install the packages use the commands which are illustrated in listing B.2.

```
sudo add-apt-repository "deb http://archive.canonical.com/ lucid partner"
sudo apt-get update
sudo apt-get install ant sun-java6-jdk
sudo update-alternatives --set java /usr/lib/jvm/java-6-sun/jre/bin/java
```

Listing B.2: Prototype installation - setup Java

In order to verify that the correct version of Java is installed, use the commands shown in listing B.3.

```
rm@ubuntu:~$ java -version
java version "1.6.0_24"
Java(TM) SE Runtime Environment (build 1.6.0_24-b07)
Java HotSpot(TM) Client VM (build 19.1-b02, mixed mode, sharing)
```

Listing B.3: Prototype installation - verify Java version

Installation Directory

Listing B.4 creates a directory *prototype* in the home directory of the user *rm*. This directory will be the base directory for all further installations.

```
cd ~
mkdir prototype
cd prototype
```

Listing B.4: Prototype installation - create prototype directory

B.2 OpenSimulator Server Preparation

This section describes how to install and prepare the OpenSim simulator server.

Download and Extraction

During the development of the prototype, the OpenSimulator sever version 0.7.1.1 has been used. Download the file *opensim-0.7.1.1-bin.tar.gz* from the OpenSimulator wiki page *Download*[2] and copy it into the previously created directory prototype. Extract the containing binary distribution by using the command of listing B.5.

[2]http://opensimulator.org/wiki/Download

```
cd ~/prototype
tar -zxvpf opensim-0.7.1.1-bin.tar.gz
```

Listing B.5: Prototype installation - extract OpenSimulator server

Configuration and Start-up

After extracting the binary distribution, change to the bin directory of the simulator (listing B.6).

```
cd ~/prototype/opensim-0.7.1.1-bin/bin
```

Listing B.6: Prototype installation - OpenSimulator server binary directory

Before starting the server for the first time, the server needs to be configured. The binary distribution contains configuration templates which can be used without much modification. Listing B.7 illustrates which configuration templates are needed and how they have to be renamed.

```
cd config-include
cp StandaloneCommon.ini.example StandaloneCommon.ini
cp CenomeCache.ini.example CenomeCache.ini
cd ..
```

Listing B.7: Prototype installation - prepare OpenSimulator server

The prototype requires the support for OSSL functions. To enable the support set *AllowOSFunctions* in the file OpenSim.ini to true (line 551). This can be done by executing the commands contained in listing B.8.

```
cp OpenSim.ini OpenSim.ini.orig
cat OpenSim.ini.orig | sed s/"; AllowOSFunctions = false"/"AllowOSFunctions = true"/ >
    OpenSim.ini
```

Listing B.8: Prototype installation - enable OpenSimulator scripting extensions

Finally the simulator is ready to be started for the first time. The server is executed by using the Mono framework as shown in listing B.9.

```
mono OpenSim.exe
```

Listing B.9: Prototype installation - run OpenSimulator

Managed Regions Preparation

During the first start-up of the simulator, the server asks for some details for the initial user and region. Table B.1 shows which information has to be entered. An empty value

Prompt	**Value**
New region name []	OSLS1
Region UUID [51f90282-7689-4681-a3ae-f787c66af63b]	
Region Location [1000,1000]	
Internal IP address [0.0.0.0]	
Internal port [9000]:	
Allow alternate ports [False]	
External host name [SYSTEMIP]	
New estate name [My Estate]	
Estate owner first name [Test]	
Estate owner last name [User]	
Password	Test

Table B.1: OpenSimulator first execution settings

means that the default value is used. The region name *OSLS1* is used for the first region. A user with the user name *Test User* will be added.

After the first start-up the server starts the management console. The last few lines of a successful server start look like the content of listing B.10.

```
21:42:41 - [!]: STARTUP COMPLETE
Currently selected region is OSLS1
21:42:41 - [STARTUP]: Startup took 5m 35s
21:42:41 - [WATCHDOG]: Started tracking thread "Incoming Packets (OSLS1)" (ID 7)
21:42:41 - [WATCHDOG]: Started tracking thread "Heartbeat for region OSLS1" (ID 8)
21:42:43 - [REGION]: Enabling logins for OSLS1
21:42:43 - [GRID SERVICE]: region OSLS1 has 0 neighbours
21:42:43 - [INTERGRID]: Informing 0 neighbours that this region is up
Region (OSLS1) #
```

Listing B.10: Prototype installation - start-up OpenSimulator

In order to use the initially created user, the password of the user must be reset. Listing B.11 sets the password of the user *Test User* to *Test*.

```
reset user password Test User Test
```

Listing B.11: Prototype installation - enable OpenSimulator test user

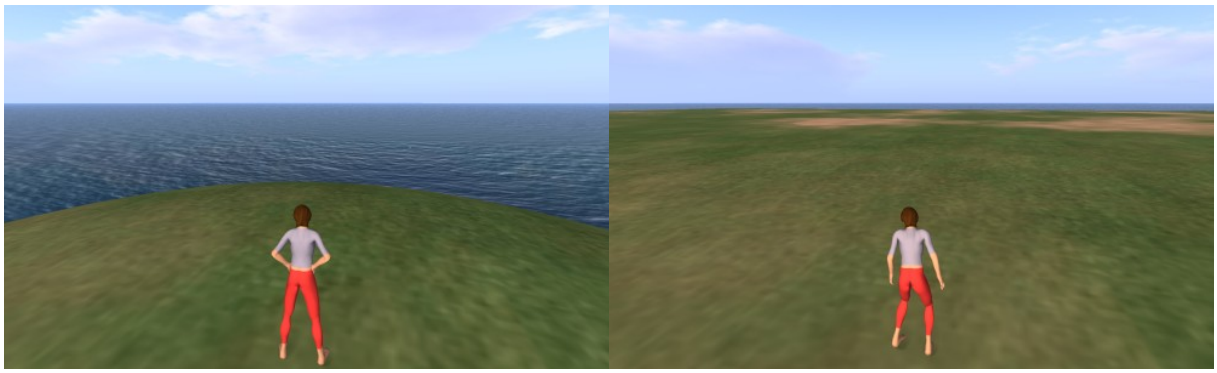
The simulator prints a confirmation that the password has been changed. It also prints the UUID of the user which will be needed later on. The printed information looks like

shown in listing B.12. The *principalID* of the user is generated and will differ from the shown one in most cases. Copy the ID for later reuse.

```
Region (OSLS1) # reset user password Test User Test
21:43:54 - [AUTHENTICATION DB]: Set password for principalID 6b987538-15a5-438f-8e47
    -01fd35294719
21:43:54 - [USER ACCOUNT SERVICE]: Password reset for user Test User
Region (OSLS1) #
```

Listing B.12: Prototype installation - reset OpenSimulator user password

Now it is already possible to connect to the server and view the empty region *OSLS1*. Detailed information about how to connect to the simulator can be found in section B.3. A newly created region is empty and contains just a small hill in the middle of the region. After connecting to the region, the viewer will show an image similar to figure B.1a.



(a) Region before lifting **(b)** Region after lifting

Figure B.1: Unmodified OpenSim region

The command shown in listing B.13 flattens the region and lifts the terrain to a height of 21 meters. The command can be entered at the management console of the server. After lifting the terrain, the viewer will simulate the region as shown in figure B.1b.

```
terrain fill 21
```

Listing B.13: Prototype installation - lift OpenSimulator terrain to 21 meters

To illustrate the capability of the prototype to manage several different worlds, a second region with the name *OSLS2* can be created. The commands illustrated in listing B.14 create a region and flatten the terrain at a height of 21 meters, which is required by the prototype. During the creation of the region, the location of the region and the internal port need to be specified. Both values must be unique. For the second region the location *1001,1000* and the internal port *9001* can be used.

```
create region OSLS2 Regions.ini
change region OSLS2
terrain fill 21
```

Listing B.14: Prototype installation - create second OpenSimulator region

The simulator stores the configuration of the regions in a file called *Regions.ini*. The location of the region and the port number can be modified in this file. After creating the two regions, the content of the file looks like shown in listing B.15.

```
rm@ubuntu:~/prototype/opensim-0.7.1.1-bin/bin$ cat ~/prototype/opensim-0.7.1.1-bin/bin
    /Regions/Regions.ini
[OSLS1]
RegionUUID = 6fdbc67c-4151-41a0-94cd-5061c12f3854
Location = 1000,1000
InternalAddress = 0.0.0.0
InternalPort = 9000
AllowAlternatePorts = False
ExternalHostName = SYSTEMIP
[OSLS2]
RegionUUID = 1809ae06-e451-4e9c-84e7-cf10f4f9c0cf
Location = 1001,1000
InternalAddress = 0.0.0.0
InternalPort = 9001
AllowAlternatePorts = False
ExternalHostName = SYSTEMIP
rm@ubuntu:~/prototype/opensim-0.7.1.1-bin/bin$
```

Listing B.15: Prototype installation - OpenSimulator region configuration

B.3 Viewer Installation

This section describes how to install the latest Second Life and Phoenix viewers. Two different client applications have been used because of the differing functionalities of the viewers. The official Second Life viewer provides the media on a prim functionality which is described in section 3.2.3. On the other hand the client was not able to download clothing information of the avatar which was the main reason to use an additional viewer during development and testing.

B.3.1 Second Life Viewer

To install the latest version of the Second Life viewer, download the viewer from the Second Life download page[3] and copy it into the previously created directory prototype. During the development of the prototype, the file SecondLife-i686-2.7.5.235722.tar.bz2 has been downloaded. The command in listing B.16 can be used to extract the downloaded viewer.

```
tar -jxvpf SecondLife-i686-2.7.5.235722.tar.bz2
```

Listing B.16: Prototype installation - extract Second Life client

After successful extraction, the viewer can already be used. Make sure that the simulator is running when trying to connect. In order to connect to a simulator which is running on the local machine, the connection URI shown in listing B.17 can be used.

[3] http://secondlife.com/support/downloads/

```
cd SecondLife-i686-2.7.5.235722/
./secondlife --loginuri http://127.0.0.1:9000
```

Listing B.17: Prototype installation - start Second Life client

Use the user name 'Test User' and password Test to connect to the virtual world.

B.3.2 Phoenix Viewer

To install the latest version of the Phoenix viewer, download the viewer from the project's home-page[4] and copy it into the previously created directory prototype. In this introduction the version PhoenixViewer-i686-1.5.2.1102.tar.bz2 is used. Listing B.18 illustrates how to extract the downloaded viewer.

```
tar -jxvpf PhoenixViewer-i686-1.5.2.1102.tar.bz2
```

Listing B.18: Prototype installation - extract Phoenix viewer

After successful extraction, the viewer can be used without any further preparation or configuration. The commands which are needed to start the viewer are shown in listing B.19.

```
cd PhoenixViewer-i686-1.5.2.1102
./snowglobe
```

Listing B.19: Prototype installation - start Phoenix viewer

In order to connect to the simulator enter *Test* in the field *First Name*, *User* in the field *Last Name* and *Test* in the field *Password*. Select *local* at the select box called *Grid* and click on *Log in*.

B.4 Prototype Installation and Execution

After preparing the simulator and installing the development environment, the prototype can be installed. The following sections describe how to install the three components of the prototype.

Prototype Extraction

Copy the file *OpenSimRoomManager.zip*, which contains the prototype, to the created directory with the name prototype. The content of the archive can be extracted by using the command unzip as shown in listing B.20.

[4] http://www.phoenixviewer.com/

```
unzip OpenSimRoomManager.zip
```

Listing B.20: Prototype installation - extract the prototype

After the described steps, the content of the directory prototype should contain the elements which are listed in listing B.21.

```
rm@ubuntu:~/prototype$ ls -l
total 32
drwxr-xr-x  7 rm rm 4096 2011-07-20 02:37 GWTClient
drwxr-xr-x  5 rm rm 4096 2011-07-20 02:38 OpenSim
drwxr-xr-x  8 rm rm 4096 2011-05-27 15:45 opensim-0.7.1.1-bin
drwxr-xr-x 11 rm rm 4096 2011-07-20 02:19 PhoenixViewer-i686-1.5.2.1102
-rw-r--r--  1 rm rm 2240 2011-06-30 06:51 README.txt
drwxr-xr-x 10 rm rm 4096 2011-07-19 23:23 SecondLife-i686-2.7.5.235722
drwxr-xr-x  6 rm rm 4096 2011-07-20 02:37 Server
drwxr-xr-x  5 rm rm 4096 2011-06-30 05:16 ThirdParty
```

Listing B.21: Prototype installation - content of the prototype archive

RoomManager Server Start-up

In order to compile and start the server, change to the directory *Server* and start the application by using the command *ant* (refer to listing B.22). If no error messages occurs, then the server is ready to accept connections from client applications and world realisation components.

```
cd Server
ant
```

Listing B.22: Prototype installation - start the RoomManager Server

RoomManager GWT Client Installation

To install the client, change to the directory *GWTClient* within the directory prototype and execute the command *ant install*. This command installs the application as a web application to the Apache Tomcat which is delivered as part of the prototype in the directory *ThirdParty*. Listing B.23 illustrates how to install the web application.

```
cd ~/prototype/GWTClient
ant install
```

Listing B.23: Prototype installation - install the RoomManager Client application

In order to access the application, the Apache Tomcat must be started. The commands to start the web server can be found in listing B.24. Afterwards it is possible to access the web application by using a web browser. Use the URI *http://127.0.0.1:8080/gwtclient/* in order to access the application.

```
cd ~/prototype/ThirdParty/apache-tomcat-6.0.32/bin/
./startup.sh
```

Listing B.24: Prototype installation - start the web application

OpenSim Region Module Installation

In order to install the developed region module, the OpenSimulator server must be stopped. Use the command *shutdown* at the management console of the simulator. The installation directory of the simulator must be configured in order to compile and install the region module. The property *opensimbase.dir* in the file *OpenSim.build* must be set to the installation directory ([your base directory]/prototype/opensim-0.7.1.1-bin/). Listing B.25 illustrates how to install the region module. Afterwards the server can be started again, as described in section B.2.

```
cd ~/prototype/OpenSim/
gedit OpenSim.build #edit the property and save and close the file
nant install
```

Listing B.25: Prototype installation - install the region module

After starting the simulator, error messages will be printed to the console because of the fact that the regions are not configured on the RMS. Now the prototype can be used. Configure the worlds OSLS1 and OSLS2 and manage the rooms within the worlds. Please refer to section 5.2.2 for further information about how to use the application.

Appendix C

CD Content

The developed prototype and additional information are available on a CD-ROM which is attached to this work. The CD has the following content:

- The used version of OpenSimulator
- The source code of the RoomManager Server application
- The source code of the RoomManager Client web application
- The source code of the OpenSimulator region module
- Screen-shots and videos of the prototype
- The used version of the Google Web Toolkit
- The used version of Apache Tomcat

References

ActiveWorlds. (2011). *Overview of Activeworlds.* Retrieved 27.12.2011, from http://www.activeworlds.com/overview.asp

Albion, P. (2009). Mere mortals creating worlds: low threshold 3d virtual environments for learning. *The International Journal of Learning*, *16*(6). Available from http://eprints.usq.edu.au/5803/2/Albion_IJL_2009_PV.pdf

Balasundaram, S. R. (2011). Securing tests in e-learning environment. In *Proceedings of the 2011 international conference on communication, computing & security* (pp. 624–627). New York, NY, USA: ACM. Available from http://doi.acm.org/10.1145/1947940.1948070

Bardzell, J., Bardzell, S., & Nardi, B. (2011). World of warcraft as a global artifact. In *Part 2 - proceedings of the 2011 annual conference extended abstracts on human factors in computing systems* (pp. 169–172). New York, NY, USA: ACM. Available from http://doi.acm.org/10.1145/1979482.1979485

Bartle, R. A. (2010). From muds to mmorpgs: The history of virtual worlds. In J. Hunsinger, L. Klastrup, & M. Allen (Eds.), *International handbook of internet research* (p. 23-39). Springer Netherlands. Available from http://dx.doi.org/10.1007/978-1-4020-9789-8_2

Blascovich, J., Loomis, J., Beall, A. C., Swinth, K. R., Hoyt, C. L., & Bailenson, J. N. (2002). Immersive virtual environment technology as a methodological tool for social psychology. *Psychological Inquiry*, *13*(2), 103-124. Available from http://www.informaworld.com/index/785830928.pdf

Blizzard Entertainment. (2011a). *Contest screenshots.* Retrieved 27.12.2011, from us.battle.net/wow/en/media/screenshots/contests

Blizzard Entertainment. (2011b). *What is World of Warcraft.* Retrieved 27.12.2011, from http://us.battle.net/wow/en/game/guide/

Bouras, C., Guannaka, E., & Tsiatsos, T. (2008). Exploiting virtual environments to support collaborative e-learning communities. *International Journal of Web-Based Learning and Teaching Technologies*, *3*(2), 1-22.

Bouras, C., & Tsiatsos, T. (2006, June). Educational virtual environments: design rationale and architecture. *Multimedia Tools Appl.*, *29*, 153–173. Available from http://portal.acm.org/citation.cfm?id=1152787.1152812

Burkle, M., & Kinshuk. (2009, 9). Learning in virtual worlds: The challenges and opportunities. In *Cyberworlds, 2009. cw '09. international conference on* (p. 320 -327).

Caeiro-Rodríguez, M., Llamas-Nistal, M., & Anido-Rifón, L. (2005). Towards a bench-

mark for the evaluation of ld expressiveness and suitability. *Journal of Interactive Media in Education*, *2005*(1). Available from http://jime.open.ac.uk/article/2005-4/268

Callaghan, M., McCusker, K., Lopez Losada, J., Harkin, J., & Wilson, S. (2009a, 8). Integrating virtual worlds & virtual learning environments for online education. In *Games innovations conference, 2009. ice-gic 2009. international ieee consumer electronics society's* (p. 54 -63).

Callaghan, M., McCusker, K., Lopez Losada, J., Harkin, J., & Wilson, S. (2009b, 12). Teaching engineering education using virtual worlds and virtual learning environments. In *Advances in computing, control, telecommunication technologies, 2009. act '09. international conference on* (p. 295 -299).

Chang, V., & Gütl, C. (2008). *Ecosystem concept and models to support e-learning 2.0.*

Chen, B., Huang, F., & Lin, H. (2009, 8). Using virtual world technology to construct immersive 3d virtual university. In *Geoinformatics, 2009 17th international conference on* (p. 1 -5).

Chen, B., Huang, F., Lin, H., & Hu, M. (2010, 10). Vcuhk: Integrating the real into a 3d campus in networked virtual worlds. In *Cyberworlds (cw), 2010 international conference on* (p. 302 -308).

Chen, S.-J., Hsu, C.-L., & Caropreso, E. (2009, March). Designing e-learning 2.0 environment for cross-cultural collaborative learning. In I. Gibson, R. Weber, K. McFerrin, R. Carlsen, & D. A. Willis (Eds.), *Proceedings of society for information technology & teacher education international conference 2009* (pp. 2257–2263). Charleston, SC, USA: AACE. Available from http://www.editlib.org/p/30961

Childers, B. (2009, March). Run your own virtual reality with opensim. *Linux J.*, *2009*. Available from http://portal.acm.org/citation.cfm?id=1513992.1513998

Costa, C. J., & Aparicio, M. (2011). Analysis of e-learning processes. In *Proceedings of the 2011 workshop on open source and design of communication* (pp. 37–40). New York, NY, USA: ACM. Available from http://doi.acm.org/10.1145/2016716.2016726

Dafli, E., Vegoudakis, K., Pappas, C., & Bamidis, P. (2009, 11). Re-use and exchange of an opensim platform based learning environment among different medical specialties for clinical scenarios. In *Information technology and applications in biomedicine, 2009. itab 2009. 9th international conference on* (p. 1 -5).

De Lucia, A., Francese, R., Passero, I., & Tortora, G. (2009, August). Development and evaluation of a system enhancing second life to support synchronous role-based collaborative learning. *Software-Practice & Experience*, *39*, 1025–1054. Available from http://dx.doi.org/10.1002/spe.v39:12

De Troyer, O., Kleinermann, F., Pellens, B., & Bille, W. (2007). Conceptual modeling for virtual reality. In *Tutorials, posters, panels and industrial contributions at the 26th international conference on conceptual modeling - volume 83* (pp. 3–18). Darlinghurst, Australia, Australia: Australian Computer Society, Inc. Available from http://portal.acm.org/citation.cfm?id=1386957.1386959

Dohi, H., & Ishizuka, M. (2009). A life-like agent interface system with second life avatars on the opensimulator server. In *Proceedings of the 3d international conference on*

online communities and social computing: Held as part of hci international 2009 (pp. 182–190). Berlin, Heidelberg: Springer-Verlag. Available from http://dx.doi.org/10.1007/978-3-642-02774-1_20

Farley, H. (2009, June). Reusable learning designs and second life: Issues and strategies. In G. Siemens & C. Fulford (Eds.), *Proceedings of world conference on educational multimedia, hypermedia and telecommunications 2009* (pp. 4047–4052). Honolulu, HI, USA: AACE. Available from http://www.editlib.org/p/32065

Ferretti, S., Mirri, S., Muratori, L. A., Roccetti, M., & Salomoni, P. (2008). E-learning 2.0: you are we-lcome! In *Proceedings of the 2008 international cross-disciplinary conference on web accessibility (w4a)* (pp. 116–125). New York, NY, USA: ACM. Available from http://doi.acm.org/10.1145/1368044.1368070

Fox, M. R., Kelly, H., & Patil, S. (2010). Medulla: A cyberinfrastructure-enabled framework for research, teaching, and learning with virtual worlds. In W. S. Bainbridge (Ed.), *Online worlds: Convergence of the real and the virtual* (p. 87-100). Springer London. Available from http://dx.doi.org/10.1007/978-1-84882-825-4_7

Freudenthaler, S. (2011). *Flexible learning settings in second life.* Unpublished master's thesis, Graz University of Technology.

Gerbaud, S., Gouranton, V., & Arnaldi, B. (2009). Adaptation in collaborative virtual environments for training. In M. Chang, R. Kuo, G.-D. Kinshuk Chen, & M. Hirose (Eds.), *Learning by playing. game-based education system design and development* (Vol. 5670, p. 316-327). Springer Berlin / Heidelberg. Available from http://dx.doi.org/10.1007/978-3-642-03364-3_40

Gerbaud, S., Mollet, N., Ganier, F., Arnaldi, B., & Tisseau, J. (2008). Gvt: a platform to create virtual environments for procedural training. In *Virtual reality conference, 2008. vr '08. ieee* (p. 225 -232).

Google. (2011a). *google-gson - A Java library to convert JSON to Java objects and vice-versa.* Retrieved 2.5.2011, from http://code.google.com/p/google-gson/

Google. (2011b). *Google Web Toolkit Overview.* Retrieved 2.5.2011, from http://code.google.com/intl/de-AT/webtoolkit/overview.html

Gütl, C. (2011). Techniques for fostering collaboration in online learning communities: Theoretical and practical perspectives. In D. P. Francesca Pozzi (Ed.), (p. 278-299). Hershey: IGI Global.

Gütl, C., Chang, V., & Freudenthaler, S. (Eds.). (2010, 9). *How to support more flexible learning settings in second life.* Hasselt, Belgium.

Henckel, A., & Lopes, C. V. (2010). Stellarsim: A plug-in architecture for scientific visualizations in virtual worlds. In O. Akan et al. (Eds.), *Facets of virtual environments* (Vol. 33, p. 106-120). Springer Berlin Heidelberg. Available from http://dx.doi.org/10.1007/978-3-642-11743-5_9

Hibernate. (2011). *Relational Persistence for Java and .NET.* Retrieved 2.5.2011, from http://hibernate.org/

Hornik, S., Johnson, R. D., & Wu, Y. (2007). When technology does not support learning: Conflicts between epistemological beliefs and technology support in virtual learning environments. *JOEUC*, *19*(2), 23-46.

HyperSQL. (2011). *HSQLDB - 100% Java Database.* Retrieved 2.5.2011, from http://

hsqldb.org/

Jacobson, M. J., Kim, B., Miao, C., Shen, Z., & Chavez, M. (2010). Design perspectives for learning in virtual worlds. In M. J. Jacobson & P. Reimann (Eds.), *Designs for learning environments of the future* (p. 111-141). Springer US. Available from http://dx.doi.org/10.1007/978-0-387-88279-6_5

Jones, D. (1996). Computing by distance education: problems and solutions. In *Proceedings of the 1st conference on integrating technology into computer science education* (pp. 139–146). New York, NY, USA: ACM. Available from http://doi.acm.org/10.1145/237466.237616

json.org. (2011). *Introducing JSON.* Retrieved 8.7.2011, from http://www.json.org/

Justin Clark-Casey. (2008). *OpenSim Tech Basics : My God, it's full of modules!* Retrieved 19.4.2011, from http://justincc.org/blog/2008/05/08/opensim-tech-basics-my-god-its-full-of-modules/

Konstantinidis, A., Tsiatsos, T., & Pomportsis, A. (2009). Collaborative virtual learning environments: design and evaluation. *Multimedia Tools and Applications*, *44*, 279-304. Available from http://dx.doi.org/10.1007/s11042-009-0289-5 (10.1007/s11042-009-0289-5)

Kopeinik, S. (2010). *Virtual 3d worlds for blended and distance learning in higher education.* Unpublished master's thesis, Graz University of Technology.

Kuo, M.-S., & Lin, C.-S. (2010, 4). Virtual parabola festival: The platform design and learning strategies for virtual learning community of practice. In *Digital game and intelligent toy enhanced learning (digitel), 2010 third ieee international conference on* (p. 3 -9).

Land, R., & Bayne, S. (2006). Issues in cyberspace education. In M. Savin-Baden & K. Wilkie (Eds.), *Problem-based learning online* (pp. 14–23+). Open University Press.

Layman, L., Cornwell, T., & Williams, L. (2006). Personality types, learning styles, and an agile approach to software engineering education. In *Proceedings of the 37th sigcse technical symposium on computer science education* (pp. 428–432). New York, NY, USA: ACM. Available from http://doi.acm.org/10.1145/1121341.1121474

Leidl, M., & Rößling, G. (2007, June). How will future learning work in the third dimension? *SIGCSE Bull.*, *39*, 329–329. Available from http://doi.acm.org/10.1145/1269900.1268897

LibOpenMetaverse Developer Wiki. (2011). *Introduction.* Retrieved 28.12.2011, from http://lib.openmetaverse.org/wiki/Main_Page

Mehlenbacher, B., Holstein, K., Gordon, B., & Khammar, K. (2010). Reviewing the research on distance education and e-learning. In *Proceedings of the 28th acm international conference on design of communication* (pp. 237–242). New York, NY, USA: ACM. Available from http://doi.acm.org/10.1145/1878450.1878490

Mono. (2011). *What is Mono.* Retrieved 18.7.2011, from http://mono-project.com/What_is_Mono

Mono Project. (2011). *Mono.Addins.* Retrieved 22.4.2011, from http://www.mono-project.com/Mono.Addins

MonoDevelop. (2011). *MonoDevelop.* Retrieved 18.4.2011, from http://monodevelop

.com/

Moore, M. G. (1993). Theory of transactional distance. In D. Keegan (Ed.), *Theoretical principles of distance education* (p. 22-38). London: Routledge. Available from http://www.uni-oldenburg.de/zef/cde/support/readings/moore93.pdf

Moro, A., Mumolo, E., & Nolich, M. (2010). Building virtual worlds by 3d object mapping. In *Proceedings of the 2010 acm workshop on surreal media and virtual cloning* (pp. 31–36). New York, NY, USA: ACM. Available from http://doi.acm.org/10.1145/1878083.1878092

Nardi, B., & Harris, J. (2006). Strangers and friends: collaborative play in world of warcraft. In *Proceedings of the 2006 20th anniversary conference on computer supported cooperative work* (pp. 149–158). New York, NY, USA: ACM. Available from http://doi.acm.org/10.1145/1180875.1180898

Open Wonderland. (2011a). *About Open Wonderland.* Retrieved 27.12.2011, from http://openwonderland.org/about/about-project-wonderland

Open Wonderland. (2011b). *Features.* Retrieved 27.12.2011, from http://openwonderland.org/about/features

OpenMetaverse Foundation. (2011a). *LibOpenMetaverse.* Retrieved 21.4.2011, from http://www.openmetaverse.org/projects/libopenmetaverse

OpenMetaverse Foundation. (2011b). *Welcome.* Retrieved 21.4.2011, from http://www.openmetaverse.org/

OpenSimulator Wiki. (2011a). *Configuration.* Retrieved 18.4.2011, from http://opensimulator.org/wiki/Configuration

OpenSimulator Wiki. (2011b). *Connecting.* Retrieved 28.12.2011, from http://opensimulator.org/wiki/Connecting

OpenSimulator Wiki. (2011c). *FAQ - About OpenSim.* Retrieved 28.12.2011, from http://opensimulator.org/wiki/FAQ

OpenSimulator Wiki. (2011d). *How to create a dynamic plugin.* Retrieved 28.12.2011, from http://opensimulator.org/wiki/How_to_create_a_dynamic_plugin

OpenSimulator Wiki. (2011e). *Introduction and Definitions.* Retrieved 18.4.2011, from http://opensimulator.org/wiki/OpenSim:Introduction_and_Definitions

OpenSimulator Wiki. (2011f). *Main page.* Retrieved 22.12.2011, from http://opensimulator.org/wiki/Main_Page

OpenSimulator Wiki. (2011g). *OsGetAvatarList.* Retrieved 28.12.2011, from http://opensimulator.org/wiki/OsGetAvatarList

OpenSimulator Wiki. (2011h). *Permissions.* Retrieved 24.6.2011, from http://opensimulator.org/wiki/OpenSim:Permissions

OpenSimulator Wiki. (2011i). *Region Modules.* Retrieved 19.4.2011, from http://opensimulator.org/wiki/RegionModules

OpenSimulator Wiki. (2011j). *REST.* Retrieved 21.4.2011, from http://opensimulator.org/wiki/REST

OpenSimulator Wiki. (2011k). *Scripting Documentation.* Retrieved 8.6.2011, from http://opensimulator.org/wiki/Scripting_Documentation

OpenSimulator Wiki. (2011l). *Streaming Media in OpenSim.* Retrieved 14.6.2011, from http://opensimulator.org/wiki/Streaming_Media_in_OpenSim

Pape, D., Anstey, J., Dolinsky, M., & Dambik, E. J. (2003). Ygdrasil–a framework for composing shared virtual worlds. *Future Generation Computer Systems*, *19*(6), 1041–1049. Available from http://www.sciencedirect.com/science/article/B6V06-48WJM7N-G/2/c19fa713672536c53939788f659ebd1e (3rd biennial International Grid applications-driven testbed event, Amsterdam, The Netherlands, 23-26 September 2002)

Pollini, A., Giusti, L., & Napoletano, L. (2011). Emerging informal learning 2.0 practices: a preliminary exploration. In *Proceedings of the 9th acm sigchi italian chapter international conference on computer-human interaction: Facing complexity* (pp. 71–75). New York, NY, USA: ACM. Available from http://doi.acm.org/10.1145/2037296.2037316

Quintella, F., Soares, L. P., & Raposo, A. B. (2010). Dweb3d: a toolkit for developing x3d applications in a simplified environment. In *Web3d'10* (p. 45-54).

Reilly, D. F., Rouzati, H., Wu, A., Hwang, J. Y., Brudvik, J., & Edwards, W. K. (2010). Twinspace: an infrastructure for cross-reality team spaces. In *Proceedings of the 23nd annual acm symposium on user interface software and technology* (pp. 119–128). New York, NY, USA: ACM. Available from http://doi.acm.org/10.1145/1866029.1866050

Restlet Wiki. (2011). *Welcome to the Restlet Framework!* Retrieved 2.5.2011, from http://wiki.restlet.org/docs_2.0/13-restlet/21-restlet.html

Schmidt, M., Galyen, K., Laffey, J., Ding, N., & Wang, X. (2010, December). Leveraging open source software and design based research principles for development of a 3d virtual learning environment. *SIGCAS Comput. Soc.*, *40*, 45–53. Available from http://doi.acm.org/10.1145/1929609.1929614

Scopes, L. J. M. (2009). *Learning archetypes as tools of cybergogy for a 3d educational landscape: a structure for eteaching in second life.* Unpublished master's thesis, University of Southampton, School of Education.

Second Life. (2011). *Destination guide.* Retrieved 28.3.2011, from http://secondlife.com/destination/nus

Second Life Community. (2011). *Shared media.* Retrieved 22.7.2011, from http://community.secondlife.com/t5/English-Knowledge-Base/Shared-Media/ta-p/700145

Second Life Wiki. (2011a). *Building tools.* Retrieved 25.7.2011, from http://wiki.secondlife.com/wiki/Building_Tools

Second Life Wiki. (2011b). *Getting started with lsl.* Retrieved 13.6.2011, from http://wiki.secondlife.com/wiki/Getting_started_with_LSL

Second Life Wiki. (2011c). *Shared media.* Retrieved 22.7.2011, from http://wiki.secondlife.com/wiki/Shared_Media

SLOODLE Blog. (2011). *About.* Retrieved 28.12.2011, from http://www.sloodle.org/blog/?page_id=2

Snowdon, D., Churchill, E. F., & Munro, A. J. (2001). Collaborative virtual environments: Digital spaces and places for cscw: An introduction. *Collaborative Virtual Environments Digital Places and Spaces for Interaction London SpringerVerlag*, 3-17. Available from http://elizabethchurchill.com/professional/pubs/Papers/

IntrotoCVEs.pdf

Tate, A., Chen-Burger, Y.-H., Dalton, J., Potter, S., Richardson, D., Stader, J., et al. (2010, July). I-room: A virtual space for intelligent interaction. *IEEE Intelligent Systems*, *25*, 62–71. Available from http://dx.doi.org/10.1109/MIS.2010.5

Tatum, M. (2000, May). Active worlds. *SIGGRAPH Comput. Graph.*, *34*, 56–57. Available from http://doi.acm.org/10.1145/351440.351582

The Apache Software Foundation. (2011). *Commons configuration - Intro.* Retrieved 2.5.2011, from http://commons.apache.org/configuration/

Thomas, D., & Brown, J. S. (2009). Why virtual worlds can matter. *International Journal of Learning and Media*, *1*(1), 37-49. Available from http://www.mitpressjournals.org/doi/abs/10.1162/ijlm.2009.0008

Varvello, M., Ferrari, S., Biersack, E., & Diot, C. (2011, February). Exploring second life. *IEEE/ACM Trans. Netw.*, *19*, 80–91. Available from http://dx.doi.org/10.1109/TNET.2010.2060351

Vilela, A., Cardoso, M., Martins, D., Santos, A., Moreira, L., Paredes, H., et al. (2010). Privacy challenges and methods for virtual classrooms in second life grid and opensimulator. In *Proceedings of the 2010 second international conference on games and virtual worlds for serious applications* (pp. 167–174). Washington, DC, USA: IEEE Computer Society. Available from http://dx.doi.org/10.1109/VS-GAMES.2010.30

Vrellis, I., Papachristos, N., Bellou, J., Avouris, N., & Mikropoulos, T. (2010, 7). Designing a collaborative learning activity in second life - an exploratory study in physics. In *Advanced learning technologies (icalt), 2010 ieee 10th international conference on* (p. 210 -214).

Vygotsky, L. S. (1986). *Thought and Language*. MIT Press. Available from http://mitpress.mit.edu/catalog/item/default.asp?ttype=2&tid=7480

Wains, S. I., & Mahmood, W. (2008). Integrating m-learning with e-learning. In *Proceedings of the 9th acm sigite conference on information technology education* (pp. 31–38). New York, NY, USA: ACM. Available from http://doi.acm.org/10.1145/1414558.1414568

Web3D Consortium. (2008). *Extensible 3D (X3D).* Retrieved 28.3.2011, from http://www.web3d.org/x3d/specifications/ISO-IEC-19775-1.2-X3D-AbstractSpecification/

Woo, Y., & Reeves, T. C. (2007). Meaningful interaction in web-based learning: A social constructivist interpretation. *The Internet and Higher Education*, *10*(1), 15-25. Available from http://www.sciencedirect.com/science/article/B6W4X-4MWH3FH-2/2/f71595b05f9bc3d86cc45f457d881c8b (Special Section of the AERA Education and World Wide Web special Interest Group (EdWeb/SIG))

Yordanova, K. (2007). Mobile learning and integration of advanced technologies in education. In *Proceedings of the 2007 international conference on computer systems and technologies* (pp. 92:1–92:6). New York, NY, USA: ACM. Available from http://doi.acm.org/10.1145/1330598.1330695

Zender, R., Dressler, E., Lucke, U., & Tavangarian, D. (2009, 3). Pervasive media and messaging services for immersive learning experiences. In *Pervasive computing and*

communications, 2009. percom 2009. ieee international conference on (p. 1 -6).