Erasmus+	Project funded by: **Erasmus+ / Key Action 2 - Cooperation for innovation and the exchange of good practices, Strategic Partnerships for adult education (European Commission, EACEA)**

VeLoCiTy

Deliverable Number	**3**
Deliverable Title	**Technical report with the learning environment functionalities**
Intellectual Output Title	**VeLoCiTy virtual learning environment**
Activity description	**The purpose of this deliverable is to present the technical details of the 3D Virtual World learning environment used to provide the functionality and features required for the project needs**
Authors (per company, if more than one company provide it together)	**CTI**
Status (D: draft; RD: revised draft; F: final)	**F**
Date (versioning)	**30/11/2017 version 1.2 (Final Version)**

Document History

Authors	Reviewer	Date	File suffix [Version.Revision]	Approval
CTI		15/11/2017	1.0	
CTI	UCY	25/11/2017	1.1	Yes
CTI		30/11/2017	1.2	

Executive Summary

This report presents the technical details of the environment used to provide the functionality for the learners and educators using innovative ways of deploying immersive technology. This solution fulfilled the requirements to provide powerful learning experiences in the field of interview for geographically dispersed learners.

The report presents the technical details (development of the scenarios, scrips for selected NPC and items in the 3DVW) used to make the VeLoCiTy possible.

List of abbreviations

Abbreviation	Definition
3DVW	3D Virtual World
NPC	Non-Player Character

Table of contents

1. Background

The VeLoCiTy Project is a European Project financed by the European Commission under the Erasmus+ KA2. The goal of the VeLoCiTy project is to help those individuals seeking for a job by showing them the different types and theories of interviews among the European countries, using a 3D Virtual World (3DVW) to achieve this goal.

The goal of the VeLoCiTy project is to facilitate the search for job seekers in another country by making them familiar with the diversity in the theories and practices of the interviews, partially because of the culture and ethnic differences among the European countries. The above goal will be achieved through an innovative 3DVW learning environment that will include scenarios simulating the most representative interview processes. The 3DVW that will be developed should be able to fulfil several requirements to serve this purpose.

This report presents the details of the environment used to provide the functionality for the learners and educators using innovative ways of deploying immersive technology.

2. Using 3DVW for Educational Purposes

As already discussed in the deliverable IO3.A2.1.2, a virtual world is "*a synchronous, persistent network of people, represented as avatars, facilitated by networked computers*" (Bell, 2008). Using a 3DVW for educational purposes fits into the experiential learning principles that support pedagogies such as "learning by doing", which engage learners in critical thinking, problem solving and decision making, while Communal Constructivism as a potentially appropriate pedagogy for use in the 3DVW (Girvan & Savage, 2010). Virtual worlds are a valuable tool in modern education practices as well as providing socialisation, entertainment and a laboratory for collaborative work (Duncan, Miller, & Jiang, 2012). Therefore, the concept of using a 3DVW to educate learners about the principles of interviews provides a novel concept that will motivate learners. Having in mind that, as well as the fact that ICT technologies are important driving factor for the developments in education, this work aims additionally to empower the profile of the teaching professions in order to adopt novel approaches in teaching.

Many teachers find the idea of adding scenario-based learning to their teaching methods interesting, as it makes classroom experiences more appealing and highly engaging. Generally, scenario-based learning immerses the learners in real life or situational simulations or learning experiences that allow them to gather skills or information that they will recall for future use. The emergence of ICT has added another aspect that influences to a significant extent the way both educators (experts in the interviews) and participants view the learning process.

As the use of virtual worlds is becoming more popular, more and more learners get acquainted with virtual world environments and new rules for social interaction emerge (Petrakou, 2010). Virtual worlds have been used with success in education, for example Second Life in psychology (Baker, Wentz, & Woods, 2009), in medical and health education (Boulos, Hetherington, & Wheeler, 2007) and in computer science (Berns, Gonzalez-Pardo, & Camacho, 2013), but to the best of our

knowledge this is the first time that a 3DVW is going to be used for the specific purpose of this project which is the concept of interviews.

Our work in the VeLoCiTy project had taken into consideration recent reports on the usage of immersive virtual reality technology in education (Ott & Freina, 2015), user behaviours and motivations while they interact with other participants in virtual worlds (Lin & Wang, 2014), as well as how users represent ethnical diversities into virtual worlds (Lee, 2014) and differences in adult's behaviour into virtual worlds (Burnett & Merchant, 2014).

3. The concept behind the VeLoCiTy scenarios

The VeLoCiTy educational concept is to use an on-line three-dimensional virtual world learning environment where all users have the potential to experience several ‘real-life like’ scenarios, presented to them. By participating into these scenarios and navigating through the world, users feel immersed and experience real-life situations, get informed and educated about them and combine skills and knowledge to overcome them, without exposing themselves into real and stressful situations as it could be if they actually had participated in an interview. The main aims of this report are to: introduce this educational tool, present the scenarios that have been developed in this tool.

Many three-dimensional Virtual Worlds (3DVW) are available nowadays, with a lot of them tailored to specific user needs mostly for socialization and leisure activities. Some of these 3DVW are serving purposes such as commercial facilitation (i.e. sales, marketing, customer support, etc.) and education enhancement (i.e. training simulations, virtual laboratories). These 3DVW provide to their users a highly immersive three dimensional graphical and interactive on-line environments which can be either a replica of an existing physical place or an imaginary place, or even places that are impossible to visit in real life due to restrictions (i.e. cost, safety, etc.). Recently it is recognized that the special characteristics and the possibilities these 3DVW offer to the users make them a powerful technological tool towards enhancing the learning experience.

Research in learning shows that deep learning is an active and constructive process and it can be more effective when it comes through real-world learning styles; problem-based, cooperative, activity-led learning, etc., can boost student’s critical thinking skills. Moreover, learning emerges in the interaction between an active constructive learner and a supportive and simulating learning environment. In that sense, the 3DVW presented in this paper offers the appropriate infrastructure to provide learners with a full understanding of a situation using immersive experiences.

The development and usage of 3DVW environments for learning is an emerging field that challenges and enlarges the idea of learning environment. Using such environments learners could freely wander through the learning environment, explore it, obtain sense of purpose, act, make mistakes, collaborate and communicate with other learners. In a 3DVW the aim is to allow learners to feel immersed into the environment. Immersion is the feeling of “actually being there” and accompanied with the interaction with virtual objects can enhance learners’ interest and engagement to the learning tasks and help them to develop a stronger conceptual understanding, depending on the content (Xenos, Maratou, Ntokas, Mettouris, & Papadopoulos, 2017).

4. The VeLoCiTy learning environment scenarios

For the VeLoCiTy learning environment 7 educational scenarios have been developed and offered to the users, based on the educational concepts discussed in the previous sections. After completing a scenario, the learner receives a budge based on their performance (gold, silver & bronze). Hereinafter the scenarios are briefly presented:

4.1. Scenario 1: Preparation for an interview

The learner enters the 3DVW and after visiting the library to collect tips for the forthcoming interview is at ‘home’ where they can interact with various objects and communicate with the educator to help them prepare for their interview (scheduled for the next day).

For this scenario a number of digital objects have been modified to interact with both the learner and the educator, such as: Alarm clock, Chest of drawers, Mirror, Computer, CV, World Ball.

4.2. Scenario 2: Face to face interview

This scenario simulates a face-to-face interview, where the learner has to participate. For this scenario an NPC, the receptionist will also inform them that there are several job options available and that Learners have to select one offer, to read it (within 10’) and then to go to the interview room.

The NPC will say: “Good morning! Today there are many job openings available in the Interview Agency! First take those tips for the interview. Then, why don’t you take a look to see if any of the offers matches your abilities? Remember to make the best impression, but always being honest about you. You have 10 minutes to read the

offer and the tips. When you finish and have a clear idea about what this interview is about; it is time to go to the next room and face this interview! Good luck!"
Following this interaction, next to the NPC there is a PANEL with the offered jobs appeared by title. The learners will have to approach the panel, read the title of the job offers and click the one who prefer. Once the learners click the job offer they find them in their inventory under "Notecards" directory and then they will be able to participate in the face to face interview with an educator.

4.3. Scenario 3: Competence-based interview

This scenario simulates a competence-based interview, where the learner finds the scenario 3 building (either by walking or teleporting to it), receives the information notecard and then participates in the interview with an educator, based on the script on the notecard. The educator can take notes during the interview using the same notecard with the questions and the indicators. The notecard will be structured as follows:

- The question which the educator asks
- The evaluating indicator of the question
- The learner's response evaluation (left blank for the educators to fill)

4.4. Scenario 4: Panel interview

This scenario simulates a panel interview. In this scenario, an NPC is playing the role of the receptionist and the educators will be at the panel. Both learners and educators receive the information notecard and then participate in the panel interview based on the script on the notecard.

4.5. Scenario 5: Concrete questions in face to face interview

This scenario simulates a face to face interview and has the same NPC as scenario 2. In this case the learners receive their notecard and had time to prepare themselves. The NPC asks them to: “Good morning! Welcome to Interview Agency PHASE 2! First take those tips for the interview. Then have a look to our job offer and pick the one which matches your abilities. You have 20 minutes to read the offer and the tips. If you like, you can go to the cafeteria for the preparation. When you finish and have a clear idea about what this interview is about; it is time to go to the next room and face this interview! Good luck!”

Furthermore, during the interview some questions are visualized. The educator has a slides table next to his office with a blank slide as cover. He/she can click the slides to present these questions. Only the educator has access to click the slides and to reveal the questions.

4.6. Scenario 6: Group interview

This scenario simulates a group interview where learners participate jointly. This scenario uses a similar notecard and then participate in the group interview based on the script on the notecard. The educator can take notes during the interview using the same notecard with the questions and the indicators for each learner.

4.7. Scenario 7: Behavioural interview

This scenario simulates a behavioural interview where learners will have the opportunity to interact with NPC and information panels to learn about details of the firm they are applying at, before starting the interview. This scenario uses a similar notecard and then participate in the interview based on the script on the notecard.

The educator can take notes during the interview using the same notecard with the questions and the indicators for each learner.

5. Creation of NPC and objects in the 3DVW

For implementing the scenarios briefly described in the previous section, functionality was introduced to various object of the 3DVW, so to interact with learners and control participation. In the following script, is presented the swinging door of scenario 1 that allows only one learner to participate in the scenario each time:

```
// Swinging door LSL script #1
// Handles the touch event.
// Handles the collision event.
// Handles closing the door automatically via a timer event.
// Triggers sounds when the door opens or closes.

// Parameters you might want to change

float  delay = 3.0;                          // time to wait before
                                             // automatically closing door

float  direction = 1.0;                     // set to 1.0 or -1.0 to control
                                             // direction the door swings

float  volume = 0.5;                         // 0.0 is off, 1.0 is loudest

// Variables you will most likely leave the same

key    open_sound  = "cb340647-9680-dd5e-49c0-86edfa01b3ac";
key    close_sound = "e7ff1054-003d-d134-66be-207573f2b535";

// Processing for the script when it first starts up

default {
    // What we do when we first enter this state

    state_entry() {
        state closed;                          // Move to the open state
    }
}

// Processing for the script when it is in the closed state

state closed {
    // What we do when we first enter this state

    state_entry() {
        llTriggerSound(close_sound, volume); // Trigger the sound of the
door closing
        llSetRot(llEuler2Rot(<0, 0, direction * PI_BY_TWO>) * llGetRot());

    }
```

```
    // What we do when the door is clicked ("touched") with the mouse

    touch_start(integer total_number) {
        string name = llDetectedName(0);
        integer spaceIndex = llSubStringIndex(name, " ");
        string  firstName  = llGetSubString(name, 0, spaceIndex - 1);
        if(firstName != "Educator"){
            llSay(0, "Welcome to your home!  Please go to the library click
it and take your learner's tips. Then please go to your room!");
        } else {
            llSay(0,"Welcome to your friend's home!   Please go to the
living room and click the notecard in the coffee table and take your
evaluator's indicators");
        }
         state open;                          // Move to the open state
    }

    // What to do when something hits the door

    collision_start(integer total_number)
    {
        state open;                          // Move to the open state
    }

    // What to do when the timer goes off

    timer()
    {
        llSetTimerEvent(0.0);                // Set the timer to 0.0 to turn
it off
    }
}

// Processing for the script when it is in the open state

state open {
    // What we do when we first enter this state

    state_entry() {
        llTriggerSound(open_sound, volume);// Trigger the sound of the door
opening
        llSetRot(llEuler2Rot(<0, 0, -direction * PI_BY_TWO>) * llGetRot());

        llSetTimerEvent(delay);                      // Set the timer to
automatically close it
    }

    // What do do when pulling the door from Inventory if it was saved
while open

    on_rez(integer start_param) {
        state closed;
    }

    // What we do when the door is clicked ("touched") with the mouse
```

```
    touch_start(integer total_number) {
        state closed;                       // Move to the closed state
    }

    // What to do when something hits the door

    collision_start(integer total_number)
    {
        // Do nothing, the door is already open
    }

    // What to do when the timer goes off

    timer()
    {
        llSetTimerEvent(0.0);               // Set the timer to 0.0 to turn
it off
        state closed;                       // Move to the closed state
    }
}
```

The following script is for controlling the NPC Secretary that is used in scenario 3 (is similar to other scenarios, with different text):

```
key npc;
list lAvatars;
string secretaryChair;
integer AvatarPresent = FALSE;
string learnerTips = "learner_tips";

default
{
    state_entry()
    {
        llSay(0, "Script running");
    }

    touch_start(integer x)
    {
        integer n;
        list avatars = osGetAvatarList();
        integer exists =  llListFindList(avatars, "Jane Kein");
        if(exists>0){
            for(n=0;n<llGetListLength(avatars);n=n+3)
            {
                if(llList2String(avatars,n+2) == "Jane Kein"){
                    llOwnerSay("Attempting            to            remove
"+llList2String(avatars,n+2)+" with UUID "+llList2String(avatars,n+0));
                    osNpcRemove((key)llList2Key(avatars,n));
                }
            }
            //llResetScript();
```

```
        }
        string notecardName = "secretary1";
        npc = osNpcCreate("Jane", "Kein", llGetPos(), notecardName);
        llOwnerSay("Created npc from notecard " + notecardName);
        //llSensor("", NULL_KEY, AGENT, 3.0, PI);
        state hasNPC;
    }
}

   state hasNPC
    {
        state_entry()
        {
            llSensorRepeat("", "", AGENT, 5.0, PI, 1.0);
           // llSensor("", NULL_KEY, AGENT, 5, PI)
           secretaryChair = llGetKey();
           osNpcSit(npc,llGetKey(),OS_NPC_SIT_NOW);
        }

        sensor(integer N) {
            integer i;
            key avatarKey;
            /*if(N>0){
                AvatarPresent = TRUE;
            }*/

            //osNpcSit(npc,secretaryChair,0);
            for (i = 0 ; i < N; i ++) {
                avatarKey = llDetectedKey(i);

                if(llListFindList(lAvatars,[avatarKey]) < 0) {
                    lAvatars += avatarKey;
                    string  name       = llDetectedName(i);
                    integer spaceIndex = llSubStringIndex(name, " ");
                    string  firstName  = llGetSubString(name, 0, spaceIndex
- 1);

                    if(firstName != "Educator"){
                        llGiveInventory(avatarKey,learnerTips);

                        osNpcSay(npc, "Hello mr. "+ llDetectedName(i)+"! My
name is " + llKey2Name(npc)+" Please have a sit. While waiting you can read
these interview tips. The interview room is the one with the sign
'Interview office'.");
                    } else {
                        osNpcSay(npc, "Hello Educator! You can find the
interview question on your office");
                    }

                     if(llGetListLength(lAvatars) > 10) {
                         lAvatars = llDeleteSubList(lAvatars, 0,0);
                     }
                      //osNpcStand(npc);
                 }/* else {
                        osNpcSay(npc,     "Welcome     back     mr.     "+
llDetectedName(i));
```

```
                }*/
            }
             //llSetTimerEvent(3);
        }
        no_sensor()
        {
            //llSay(0, "Nobody is around.");
            //AvatarPresent = FALSE;
        }
        /*timer(){
            osNpcSit(npc,secretaryChair,0);
            llSetTimerEvent(0);
        }*/

    }
```

The following script is used for the presentations of the scenario 5:

```
integer slidecount = 0;
list inventory;

default {

    state_entry() {
    string  name;
    integer i;
    integer num = 4; // llGetInventoryNumber(INVENTORY_TEXTURE)
        for (i = 0; i < num; ++i) {
        name = llGetInventoryName(INVENTORY_TEXTURE, i);
        inventory += name;
        }
    }

    touch_start(integer num_detected) {
        key avatar = llDetectedKey(0);
        string name = llDetectedName(0);
        integer spaceIndex = llSubStringIndex(name, " ");
        string  firstName  = llGetSubString(name, 0, spaceIndex - 1);

        if(firstName == "Educator"){
             if(slidecount == (llGetListLength(inventory))) {
                slidecount = 0;
            }
            llSetTexture(llGetInventoryName(INVENTORY_TEXTURE,  slidecount),
ALL_SIDES);
            slidecount +=1;
        } else {
            llSay(0,"You are not allowed to view the interview questions");
        }
    }
}
```

The following script controls Adam, which is an NPC used in scenario 6:

```
key npc;
list lAvatars;
integer AvatarPresent = FALSE;
vector toucherpos;

default
{
    state_entry()
    {
        llSay(0, "Script running");
    }

    touch_start(integer x)
    {
        //osAgentSaveAppearance(llDetectedKey(0), "alex");
        integer n;
        list avatars = osGetAvatarList();
        integer exists =  llListFindList(avatars, "Adam Dexter");
        if(exists>0){
            for(n=0;n<llGetListLength(avatars);n=n+3)
            {
                if(llList2String(avatars,n+2) == "Adam Dexter"){
                    llOwnerSay("Attempting to remove
"+llList2String(avatars,n+2)+" with UUID "+llList2String(avatars,n+0));
                    osNpcRemove((key)llList2Key(avatars,n));
                }
            }
        }
        string notecardName = "adam";
        npc = osNpcCreate("Adam", "Dexter", llGetPos(), notecardName);
        llOwnerSay("Created npc from notecard " + notecardName);
        toucherpos = llDetectedPos(0);
        state hasNPC;
    }
}

   state hasNPC
    {
        state_entry()
        {
            osNpcMoveTo(npc, toucherpos+<0,0,-5>);
            llSensorRepeat("", "", AGENT, 5.0, PI, 1.0);
        }

        sensor(integer N) {
            integer i;
            key avatarKey;

            for (i = 0 ; i < N; i ++) {
                avatarKey = llDetectedKey(i);

                if(llListFindList(lAvatars,[avatarKey]) < 0) {
                    lAvatars += avatarKey;
                    string  name       = llDetectedName(i);
                    integer spaceIndex = llSubStringIndex(name, " ");
```

```
                    string  firstName  = llGetSubString(name, 0, spaceIndex
- 1);

                    if(firstName != "Educator"){
                        osNpcSay(npc, "Hello mr. "+ llDetectedName(i)+"!
This is the first step for scenario 6! In the stand next to me there is the
job offer for the scenario.  Select it and then go to the cafeteria to be
prepared for the interview. You have 20 minutes. Then, go to the building
labelled "Scenario 6 "and the secretary will give you more instructions for
the interview. Good Luck!");
                    } else {
                        osNpcSay(npc, "Hello mr. "+ llDetectedName(i)+"!
You have to go straight to Scenario 6 building!");
                    }

                     if(llGetListLength(lAvatars) > 10) {
                         lAvatars = llDeleteSubList(lAvatars, 0,0);
                     }
                }
            }
        }
        no_sensor()
        {

        }
    }
```

Similar scripts are used in various objects the learners and educators interact with during the various scenarios.

6. Technical requirements and final solution's technical details

Based on the report entitled "Study on 3D virtual world's platforms and technologies", the technical requirements of the environment were defined as:

1. To be cost-free and open-source, as stated in the project proposal.
2. To be able to allow the development of fully customisable and multiuser virtual worlds to simulate various scenarios.
3. To support straightforward server configuration and parameterisation to fully control the 3DVW and the usage rights at will.
4. To allow self-hosting in partners servers.
5. To be able to support multi-platform client software allowing for non-Windows users' participation.
6. To offer 3D graphics and human-like fully customisable avatars to support the issues of immersion and presence.
7. To offer a built-in 3D editor for the creation and editing of 3D virtual objects and landscapes that will be used in the scenarios.
8. To offer good real-time communication through text chat, instant messages and voice.

Focusing on these requirements, the VeLoCiTy solution offers all these features and managed to present these on the scenarios developed. In particular, it is based on OpenSimulator 3DVW platform, which is being developed in C# programming language and is released under a BSD License, making it both open source and commercially friendly to embed in products.

At the broadest architectural level, the main components of the VeLoCiTy system are three: a) the server (or "simulator"), b) the client (or "viewer") and c) the services. Of the three components in the OS system – clients, simulator and services - the most complex are the clients and simulators. The client is complex because of the work

involved in 3D graphical rendering of the virtual world. The simulator is complex because of its responsibility for maintaining the simulation itself. Services, on the other hand, are much simpler. They consist of little more than an interface that wraps around some means of persisting data (in a database, for example). They do not carry any state information between requests.

6.1. The VeLoCiTy Server

The VeLoCiTy server is responsible for the maintenance and update of the virtual world status, managing every user and/or object applied alteration to the virtual environment state. For this reason, it is also called "simulator". Changes can come from many sources such as a direct response to user actions (e.g. avatar push an object), from scripts acting on an object or from environmental effects (e.g. simulated gravity). All processes related to the physics in the environment and the script execution are being handled by the simulator. Additionally, it responds to all clients' requests and can support the simulation of one or more fixed virtual world areas the so called "Regions".

Currently the VeLoCiTy server is hosted at CTI premises, but the open architecture allows expandability to further “Regions”.

6.2. The VeLoCiTy clients

The client or "viewer" is the software responsible for the 3D graphical rendering of the avatars and the virtual world. Except during login, the client connects and interacts exclusively with the simulator(s) of the virtual world. A client connected to one Region of the virtual world is notified of simulation events occurring in neighbouring Regions, which allows the client to render a contiguous 3D environment that is not bounded by the resources of a single simulator.

The viewer acts mainly as an interface between the user and the simulator(s), offering tools for the client set up, landscape modification, avatar in-world customization, communication and movement, in-world 3D building, editing and scripting, file upload, social networking, etc. The quality of the graphics can vary a lot among different systems, depending mainly on the graphics adapter utilised.

Since OS does not come with a viewer in its distribution so in order to connect to the VeLoCiTy virtual world, a user needs to download and install locally one of the many compatible open-source viewers. The available viewers differ in certain characteristics and learners and educators are free to choose their preferred one.

6.3. Services

The backend of the system consists of the Services which provide the virtual world simulator(s) with the common resources requested. All the resources data elements are permanently stored in one or more databases under a unique ID which is called Universally Unique Identifier (UUID). In the classic architecture, access to these services always takes place via the simulator through well-known Uniform Resource Locators (URLs).

There are many Services offered by OS, but the most crucial for the client-server communication are the following:

- User Service: This persistently stores user data, such as names, passwords and biographies and offers authenticity mechanisms during the initial user login to the virtual world.
- Grid service: The grid service keeps information regarding the coordinates and the positioning of every region in the virtual world. Simulators of other regions request this information in order to find out about their neighbouring regions.

- Asset service: This service stores all media data persistently. Discrete items of media data are called "assets" in OS terminology. Types of asset include textures, scripts, sounds, object representations etc.
- Inventory service: Each avatar owns an inventory in the virtual world which is similar to a personal warehouse with items (objects, scripts, textures, etc.) from the virtual world. Every item in this inventory holds references to a specific asset. The inventory service persistently stores this information.

6.4. Avatars

The client program enables users to interact with each other through Avatars (human shaped computer figures). A single user account may have only one avatar at a time, although the appearance of this avatar can change between as many different forms as the user wishes. Avatar forms, like almost everything else, can be created by the user with the use of a high-level interface. The avatar can be dressed up as the user desires, interact with other objects and avatars and even animate some physical movements of a user's choice.

For the VeLoCiTy project avatars play a very important role to simulate the real-life situations that the learners will face.

6.5. Interactions

There are three fundamental ways an avatar can interact with the surroundings:

- Chat: Chatting is a tractable event that can be used to interact with other avatars or objects. Every avatar can chat in many different predefined channels in order to have simple interaction with objects such as answering a quiz question.
- Touch: An avatar can touch the objects around. An object may have a script to capture this touch event and thus trigger some appropriate action.

- Dialogs: The avatar can answer some system invoked dialogs so as to state the corresponding user's preference or answer a simple multiple-choice question.

Apart from these build-in features, voice is also available as a build-in feature. While VoIP functionality is not a built-in feature, there are several external modules that can enable users to have voice chat in-world. However not all of them are able to offer realistic voice representation (i.e. lip sync, 3D sound etc.) and they require some extra system configuration.

6.6. Scripts and NPC

An important feature of OS is the scripting which lies at the core of the platform's engine. OS script engine XEngine is responsible for the compilation of the scripts to .NETassembly before execution. OS currently supports LSL, OpenSim Scripting Language (OSSL) and C# scripts. These scripts can be written in-world and are embedded in the virtual objects, making them "alive", offering them capabilities like movement, communication, reaction to avatars' or other objects' actions, shape changing, etc.

If the developer needs more functionality than simple LSL or OSSL can offer, more complex forms of scripting, such as MRM scripting, are available. MRM is short for 'Mini Region Module' and is written in C# (like OpenSim itself), but unlike the currently implemented C# engine, contains a vastly different object-oriented API.

NPCs are programs that can be used to control in-world avatar in an automated way. NPCs are perceived as virtual clients, so they do not overload the server as if they were real users' clients. For this reason, they can be as many as the simulation/game designer needs. Such functionality can prove extremely helpful when trying to implement scenarios where the users have to interact with NPCs. Bots can also be

programmed to perform in-world maintenance or administration tasks. The OS platform offers numerous NPC-dedicated OSSL functions that can be applied to create scripts that control a Bot as desired.

To cover the VeLoCiTy requirements both the use of scripting language, as well as C# was used to create all the interaction features required for the scenarios.

Bibliography

Baker, S. C., Wentz, R. K., & Woods, M. M. (2009). Using virtual worlds in education: Second Life® as an educational tool. *Teaching of Psychology, 36*(1), 59-64.

Bell, M. W. (2008). Toward a definition of “virtual worlds”. *Journal For Virtual Worlds Research, 1*(1).

Berns, A., Gonzalez-Pardo, A., & Camacho, D. (2013). Game-like language learning in 3-D virtual environments. *Computers & Education, 60*(1), 210-220. doi:http://dx.doi.org/10.1016/j.compedu.2012.07.001

Boulos, M. N. K., Hetherington, L., & Wheeler, S. (2007). Second Life: an overview of the potential of 3-D virtual worlds in medical and health education. *Health Information & Libraries Journal, 24*(4), 233-245. doi:10.1111/j.1471-1842.2007.00733.x

Burnett, C., & Merchant, G. (2014). Points of view: Reconceptualising literacies through an exploration of adult and child interactions in a virtual world. *Journal of research in reading, 37*(1), 36-50.

Duncan, I., Miller, A., & Jiang, S. (2012). A taxonomy of virtual worlds usage in education. *British Journal of Educational Technology, 43*(6), 949-964. doi:10.1111/j.1467-8535.2011.01263.x

Girvan, C., & Savage, T. (2010). Identifying an appropriate pedagogy for virtual worlds: A Communal Constructivism case study. *Computers & Education, 55*(1), 342-349. doi:http://dx.doi.org/10.1016/j.compedu.2010.01.020

Lee, J.-E. R. (2014). Does virtual diversity matter?: Effects of avatar-based diversity representation on willingness to express offline racial identity and avatar customization. *Computers in Human Behavior, 36*, 190-197. doi:https://doi.org/10.1016/j.chb.2014.03.040

Lin, H., & Wang, H. (2014). Avatar creation in virtual worlds: Behaviors and motivations. *Computers in Human Behavior, 34*, 213-218. doi:https://doi.org/10.1016/j.chb.2013.10.005

Ott, M., & Freina, L. (2015). *A literature review on immersive virtual reality in education: state of the art and perspectives.* Paper presented at the Conference proceedings of» eLearning and Software for Education «(eLSE).

Petrakou, A. (2010). Interacting through avatars: Virtual worlds as a context for online education. *Computers & Education, 54*(4), 1020-1027. doi:http://dx.doi.org/10.1016/j.compedu.2009.10.007

Xenos, M., Maratou, V., Ntokas, I., Mettouris, C., & Papadopoulos, G. (2017). Game-based learning using a 3D virtual world in computer engineering education. Paper presented at the Global Engineering Education Conference (EDUCON).