UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

PROYECTO FIN DE CARRERA

Ingeniería de Telecomunicación

Implantación de una plataforma para desarrollar mundos virtuales para la educación

Diego Aguilera Gréjon

Junio 2017

Implantación de una plataforma para desarrollar Mundos Virtuales para la educación.

Autor: Diego Aguilera Gréjon

Tutor: Xavier Alamán

Escuela Politécnica Superior

Universidad Autónoma de Madrid

Junio 2017

Resumen

Este proyecto está dedicado a la implantación de una plataforma de mundo virtual que permita a sus usuarios utilizarla para dar apoyo a sus clases/talleres de índole educacional a distintos niveles, desde colegios e institutos a centros de educación especial. En particular, se ha trabajado estrechamente con el Colegio de Educación Especial "Santa Teresa de Jesús", en Granada, en la creación de un museo virtual donde los visitantes puedan ver las obras creadas por los alumnos de este centro.

Para llevar a cabo la implantación se ha realizado en primera instancia una investigación sobre la capacidad necesaria de las máquinas que albergarán los servidores del mundo virtual. Seguidamente se han instalado y configurado los programas necesarios para acondicionar el mundo virtual de acuerdo a las necesidades requeridas por sus usuarios de mayor nivel, los profesores. Además, se ha añadido una interfaz web enlazada con las bases de datos del mundo virtual para permitir a los profesores el control de manera externa de ciertos parámetros que puedan serles necesarios; se han creado Pendrives USB con capacidad de ejecutar un programa cliente para conectarse al mundo virtual. El proyecto tiene también la intención de servir como manual de usuario para el mantenimiento y futuras mejoras de la plataforma.

Palabras clave

Metaverso, Gamificación, Game Based Learning, Sandbox, Out of the box, Feedback, Script, Backend

Abstract

This Project is dedicated to the establishment of a virtual world platform that allows its users to use it as a support tool for their lessons and school workshops at different levels of education, from elementary schools and high schools to special education centers. Particularly, there has been a joint effort with the Special Education School “Santa Teresa de Jesús”, in Granada, with the creation of a virtual museum where its visitors can see the artwork of the students of the institute.

To accomplish this establishment, we have done, in the first place, an investigation of the requirements on the machines that will host the servers of the virtual world. Next, the necessary programs have been installed and configured to set up the virtual world according to the needs required by the higher level users, the teachers. Furthermore, a web interface linked with the data base of the virtual world has been added to allow the teachers an external way to control certain parameters; USB keys have been created with the capacity to run a client program to connect to the virtual world. The project also intends to serve as a manual for the maintenance and future improvements.

Key Words

Metaverse, Gamification, Game Based Learning, Sandbox, Out of the box, Feedback, Script, Backend

Agradecimientos

A mis padres, por el apoyo recibido a lo largo de estos años y su paciencia para que haya conseguido terminar mi Carrera, sin vosotros no habría sido posible, gran parte de este logro también es vuestro.

A Xavier Alamán, mi tutor de este proyecto, por su labor inestimable como tutor, que me ha guiado en todo momento en los objetivos y necesidades de este Proyecto de Fin de Carrera.

A mis compañeros de la Escuela, en especial a aquellos con los que he compartido tantos buenos momentos en las escaleras y los bancos, porque apoyarnos los unos en los otros en los malos momentos nos ha ayudado a seguir adelante para dejarlos atrás y juntos llegar a los buenos momentos y disfrutarlos.

A mi grupo de amigos los Titos, que siempre me han ofrecido sus ánimos en mis peores momentos y me han servido de vía de escape.

Índice de contenidos

Índice de ilustraciones

Índice de tablas

Glosario

TIC Tecnologías de la Información y la Comunicación

3D Tres Dimensiones

HTTP Hypertext Transfer Protocol

UDP User Datagram Protocol

GVSP GigE Vision Stream Protocol

CPU Central Processing Unit

RAM Random Access Memory

IRC Internet Relay Chat

IP Internet Protocol

FSM Finite State Machine

VOIP Voice Over IP

1. Introducción

1.1 Motivación

Hoy en día vivimos en un Mundo donde cada día las Tecnologías de la Información y la Comunicación (TIC) tienen más presencia y peso en todos los aspectos de la vida de las personas en casi todos los rincones del planeta. Uno de esos ámbitos y uno de los más importantes es la educación, que no ha quedado ni mucho menos apartado de la influencia de las TIC [1].

Son por tanto muy comunes los Sistemas de Gestión de Aprendizaje tales como "Moodle" para gestionar los recursos necesarios al aprendizaje de una materia, monitorizar el proceso de aprendizaje y evaluar los conocimientos de los alumnos. Pero existen más tecnologías y herramientas que están siendo introducidas poco a poco en el ámbito educativo con el fin de mejorar y facilitar el aprendizaje a los estudiantes. Una de ellas son los mundos virtuales, en donde las personas físicas se encarnan en avatares de un mundo virtual que simula un entorno adaptado que reúne las características deseadas por aquellos usuarios que van a habitarlo [2].

En efecto, una de las mayores ventajas de los mundos virtuales es la posibilidad de crear un entorno enfocado a sus usuarios que les permita realizar tareas y actividades que serían difíciles de realizar en el mundo real debido a limitaciones económicas, horarias, de localización o de seguridad [3]. Existen además numerosos estudios que aseguran que el uso de mundos virtuales en el ámbito educativo sirve de catalizador para el desarrollo de habilidades sociales, comunicativas, de trabajo en grupo, creativas, etc [4] [5].

De ahí surge la motivación de este proyecto, que consiste en dotar a los centros participantes, de una herramienta innovadora en la que apoyarse para impartir la materia de sus asignaturas tanto dentro como fuera de las aulas.

1.2 Objetivos

El objetivo de este Proyecto de Fin de Carrera es realizar la implantación de una plataforma para desarrollar mundos virtuales en Opensim.

El proyecto consta de tres facetas. La primera es la obtención de requisitos técnicos, instalación del hardware, parametrización del sistema, y puesta en marcha y mantenimiento automatizado de un servidor de mundos virtuales, empleando la tecnología abierta Open Simulator.

La segunda faceta se centrará en realizar mejoras sobre el código de la versión "Out of the box" del Simulador de código abierto *Opensimulator* dirigidas a satisfacer las necesidades tanto de los profesores para preparar el entorno que deseen como de los estudiantes al a hora de aventurarse en él. Por último, se realizarán pruebas de estrés para comparar las estimaciones iniciales sobre las capacidades del sistema y las reales.

La tercera faceta consiste en el desarrollo directamente desde el interior del mundo virtual que permita la realización de un museo virtual conjuntamente con un centro de educación especial en el que sus estudiantes puedan exhibir sus creaciones artísticas para que visitantes del centro tengan una forma distinta de acercarse a estos jóvenes.

1.3 Metodología y plan de trabajo

El proyecto se dividirá en las siguientes fases:

a) Compra de los servidores previo estudio y estimación sobre las necesidades de Hardware para dar el servicio integral de la aplicación a las instituciones a las que ésta se dirija.

b) Instalación y puesta en funcionamiento de los servidores: Sistema Operativo, Antivirus, Firewall, Bases de datos, etc. así como la automatización del mantenimiento.

c) Consenso con las instituciones a las que va dirigida la aplicación sobre las necesidades a cubrir.

d) Programación de la aplicación dentro del mundo virtual e integración de distintos servicios para la aplicación, tales como mensajería offline, inserción de contenidos multimedia y una herramienta web de administración de usuarios.

e) Tests de estrés para determinar las capacidades reales de servicio de la aplicación del servidor.

f) Documento de proyecto fin de carrera realizado a partir las anotaciones llevadas a cabo durante la realización del proyecto.

1.4 Estructura del documento

Como desenlace del presente capítulo, vamos a describir la estructura que se va a seguir a lo largo del documento. La memoria está organizada en los siguientes capítulos:

- En el capítulo 2, "Estado del arte", realizamos un breve repaso a la evolución de las TIC a lo largo del siglo XX y analizaremos su inclusión en la educación y los resultados que esto ha tenido.

- En el capítulo 3, "Opensimulator", se analizarán las características de la plataforma elegida para crear nuestro entorno virtual dedicado a la educación, Opensimulator. Profundizaremos en sus virtudes y desventajas.

- En el capítulo 4, "Implementación", hablaremos del proceso completo que se ha llevado a cabo para poner el sistema en funcionamiento: desde el dimensionamiento y estimación de las capacidades del hardware, pasando por las etapas de configuración del simulador y los componentes anexos a él, y terminando por un test de estabilidad y estrés del sistema implantado para evaluar sus límites.

- En el capítulo 5, "Desarrollo de scripts para museo virtual", se detallarán los scripts desarrollados para el museo virtual del Colegio de Educación Especial "Santa Teresa de Jesús" mediante el uso del lenguaje de scripting LSL (Linden Scripting Language).

- En el capítulo 6, “Manual técnico de configuración de Opensim”, se detallan los pasos a seguir para la instalación y mantenimiento del sistema tal y cómo se encuentra en la fecha de redacción de esta memoria.

- En el capítulo 7, “Resultados”, expondremos algunos de los casos concretos de utilización del mundo virtual implantado así como otros datos relevantes.

- Finalmente, en el capítulo 8, “Conclusiones y trabajo futuro”, se plantean las conclusiones obtenidas de los resultados del proyecto y se proponen nuevas líneas de implementación para el sistema que aumenten su valor añadido y cubran necesidades de carácter menos urgente que no se han llegado a abordar en este proyecto pero que aumenten el potencial del mundo virtual.

2. Estado del arte

2.1 Introducción

En este capítulo se revisa el estado del arte en el área del empleo de los mundos virtuales como herramienta para la docencia. El objetivo de este capítulo es conocer las tecnologías recientes que se han introducido en la pedagogía para ser usadas por los docentes como herramientas de apoyo para la docencia.

Empezaremos introduciendo el contexto del uso de mundos virtuales en la educación. Seguidamente expondremos algunos estudios realizados sobre el impacto del uso de los mundos virtuales en la educación, tanto de los beneficios aportados como de los posibles problemas. Por último, nos centraremos específicamente en la plataforma Opensim de creación de mundos virtuales.

2.2 Mundos virtuales en la educación

2.2.1 El uso de los mundos virtuales como herramienta aplicada a la educación

La evolución de las TIC desde los comienzos de la era de las computadoras, a mediados de los cincuenta, hasta nuestros días es sin duda una de las más relevantes en la historia moderna. La infinita variedad de aplicaciones que tienen las han hecho una herramienta tan versátil que en pocas décadas han cambiado muchos conceptos y prácticas socio-culturales a distintos niveles: político, económico, cultural, y por supuesto educacional. En las décadas siguientes, tras conquistar el mundo de la investigación y el mundo laboral, las TIC pasaron a conquistar los hogares, implementando lo que hoy conocemos como la **Sociedad de la Información.**

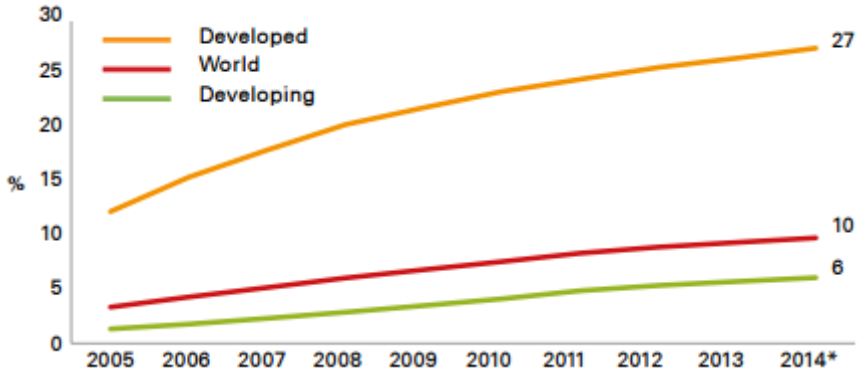

Ilustración 1: Evolución del porcentaje de hogares con Internet de banda ancha (Fuente: ITU-ICT Facts and Figures 2014)

A pesar de que, como se ve en la gráfica anterior, los ordenadores y el acceso a internet no entraron en los hogares hasta finales del siglo XX, el empleo de las TIC en la educación ya era una realidad a principios de los años sesenta. Ya entonces se atisbaban las realidades que conocemos hoy en día: un mundo altamente automatizado. Dado que la educación se basa en formar a las personas para valerse en el mundo en el que viven, se llegó a la conclusión de que parecía lógica la inclusión de material tecnológico en la enseñanza [6]. Por otra parte, algunos investigadores detractores del uso de ordenadores por estudiantes de edades demasiado tempranas argumentaban que podían provocar que se desarrollasen inadecuadamente, alejándose de las características humanas de ética y moralidad y asemejándose demasiado a la frialdad de una máquina, carente de éstas, volviéndose los menores personas demasiado matemáticas y lógicas [7].

Dentro de los distintos enfoques de uso que se han ido dando a las TIC en la educación, uno de que se popularizó hace ya casi dos décadas fue el uso de entornos virtuales 3D, comúnmente llamados mundos virtuales. Los mundos virtuales son espacios tridimensionales en donde las personas pueden conectarse con avatares con los que pueden interactuar libremente con los avatares de otras personas y los objetos que haya en dicho mundo virtual. Esta característica es la que motiva su uso en la educación, ya que está directamente relacionado con la principal corriente de metodología pedagógica actualmente más propagada en el mundo, el **constructivismo**, cuyas figuras claves fueron Jean Piaget y Lev Vygotsky. El método constructivista sostiene que los conocimientos no pueden ser transmitidos, sino que deben ser construidos por el propio

individuo [8]. El individuo pasa de ser un sujeto pasivo que recibe información, a ser un sujeto activo que, mediante actividades significativas, se convierte a sí mismo en su propia fuente de herramientas cognitivas que le permitan seguir aprendiendo en el futuro de manera independiente; en otras palabras, aprender a aprender.

Para profundizar más en los resultados de utilizar los mundos virtuales en la educación haremos un repaso de algunos estudios realizados sobre el uso de mundos virtuales en la educación, analizando las conclusiones, tanto positivas como negativas, obtenidas por parte de los investigadores.

2.2.2 Análisis de estudios sobre las ventajas de la utilización de mundos virtuales en la educación

Como se ha expuesto anteriormente existen numerosos aspectos a tener en cuenta sobre el uso de los mundos virtuales en la educación. En esta sección vamos a cubrir algunos de los más relevantes.

La primera cuestión es por qué usar los mundos virtuales en la educación. A menudo se asocian los mundos virtuales con los videojuegos, como vía de entretenimiento en donde una persona asume el rol de su alter-ego en un mundo ficticio. Estos entornos carecen de las "molestias" que sí existen en el mundo real, y las personas por tanto hacen uso de ellos como vía de escape. Entonces, ¿por qué queremos crear mundos virtuales que tengan el mayor parecido a la realidad posible?

Uno de los usos más evidente es la posibilidad de replicar actividades en el mundo virtual para el entrenamiento específico de habilidades, procedimientos, o maniobras sin riesgo alguno. En el caso del uso de los mundos virtuales para entrenamientos médicos [9] o militares [10] adquiere total sentido que los entornos sean lo más parecido a la realidad. La posibilidad de repetir una operación quirúrgica o militar permite no poner en riesgo vidas humanas en el entrenamiento de personas que en el momento de llevar a cabo su desempeño en el mundo real sí vayan a ponerse a sí mismos o a otros en peligro físico.

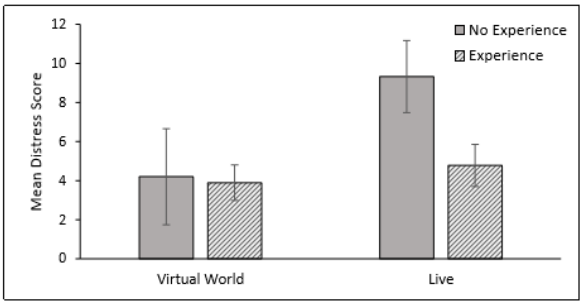

Figure 1. Mean Post-Training Distress Scores per Condition Based on Building Clearing Training Experience

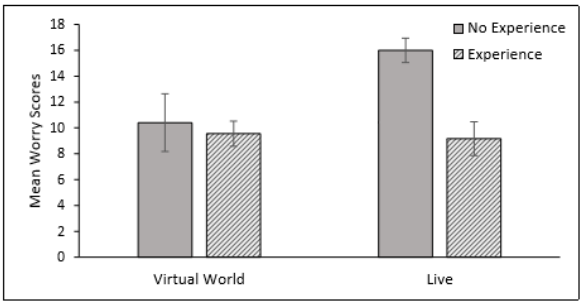

Figure 2. Mean Post-Training Worry Scores per Condition Based on Building Clearing Training Experience

Ilustración 2: Comparativa de soldados en un entrenamiento con y sin experiencia previa en simulador virtual [10]

La clara ventaja que ofrece un entrenamiento de una posible situación de combate primero en un entorno virtual y posteriormente en el mundo real queda reflejada en el anterior gráfico. Este tipo de entrenamientos son metódicos y repetitivos en el marco militar y médico tienen total sentido, y es en estas disciplinas en las que se hizo uso de los mundos virtuales en primer lugar.

En el caso del uso de los mundos virtuales en etapas de la educación más elementales, no ha sido hasta recientemente que se ha empezado a popularizar su uso. Al tratarse de niveles de enseñanza que se apoyan en la metodología constructivista anteriormente mencionada, necesitamos justificar de forma diferente el uso de los mundos virtuales. Para alumnos de primaria o primeros cursos de secundaria, la posibilidad para un alumno de aventurarse a la realización de prácticas ya sean plausibles, irrealizables o inusuales, no tiene consecuencias ni riesgos, por lo que potencia su interés por la búsqueda de identidad personal y fomenta el aprendizaje experiencial, la colaboración y comunicación con otros estudiantes [11]. Esto encaja totalmente con la idea del aprendizaje constructivista.

Por otro lado, los mundos virtuales también pueden no ajustarse totalmente a la realidad y ser ésto una característica positiva. El poder encarnarse en un avatar que tenga las mismas características que todos los demás en el mundo virtual, lo cual no es un reflejo fiel de la realidad, puede suponer una ventaja en determinados usos de mundos virtuales. Estudios del empleo de SecondLife en personas con desorden del espectro autista han reportado resultados positivos al carecer éste de barreras sociales y ofrecer a los usuarios situarse en un entorno de mayor posición de control y confianza para socializar [12]. Parece claro que todos los niveles de enseñanza parecen tener algún tipo de justificación de peso para integrar los mundos virtuales como herramienta de apoyo, recalibrando cada uno su fidelidad con la realidad en los ejemplos mencionados en función de sus necesidades.

El empleo no sólo de mundos virtuales sino de otras tecnologías en ámbitos diversos al juego entra dentro del contexto global que recoge el término gamificación. La gamificación es un término acuñado en los años 2008 pero que no empezó a coger forma hasta los años 2010 y 2011, y se define como "el uso de mecánicas de juego en entornos ajenos al juego" [13].

Del concepto de gamificación se desarrolla a día de hoy el Game Based Learning [14]. Los mundos virtuales empleados en la educación son sin duda uno de los mayores representantes del concepto Game Based Learning ya que poseen características clásicas de los videojuegos, tales como la posibilidad de otorgar puntos, crear clasificaciones, y comparativas, niveles de juego, así como posteriormente dar feedback al usuario [15]. La posibilidad de mejorar sus puntuaciones y escalar en el ranking de puntuaciones en el desempeño de una tarea dentro del mundo virtual es una motivación para el alumno. El promover en primera instancia la diversión, la cooperación y la socialización en un entorno virtual consigue mejorar el interesar y motivar al alumno [16]. El aprendizaje se adquiere como subproducto que ocurre al realizar aquellas tareas que le suponen entretenidas al alumno. La siguiente ilustración propone un modelo de flujo para el modelo de aprendizaje basado en videojuegos

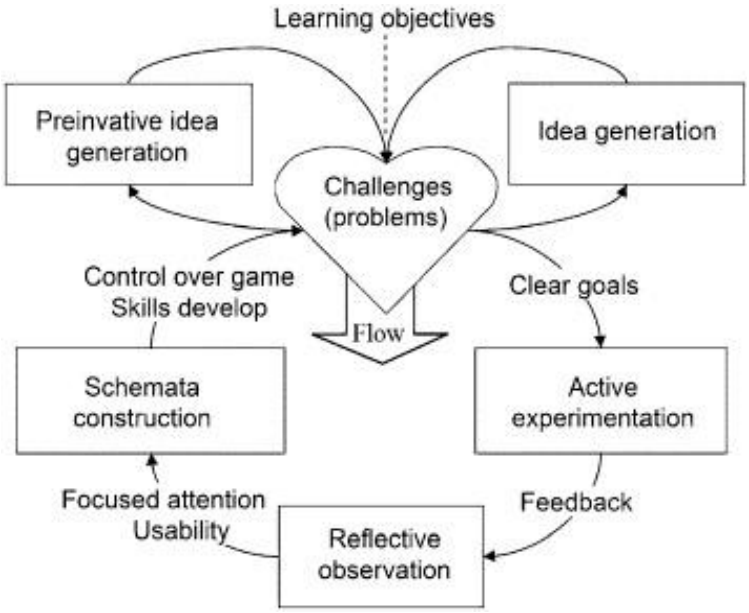

Ilustración 3: Experiential Gaming Model [15]

La clave parece estar en conseguir encontrar el equilibrio entre el entretenimiento y el cumplimiento del temario a impartir. Si se pone demasiado énfasis en la educación como principal objetivo la motivación y la involucración pueden decrecer (la paradoja del brócoli recubierto de chocolate); si se pone insuficiente énfasis no se alcanzarían los objetivos curriculares mínimos fijados.

Además de la problemática arriba mencionada, existen todavía gran cantidad de retos a la hora de integrar los mundos virtuales en la educación de manera efectiva [17] como el cumplimiento de las promesas que parece ofrecer dicha integración [18], los costes logísticos, económicos y técnicos [19], o incluso definir con claridad cuál es la finalidad de las aplicaciones concretas.

2.2.3 Obstáculos al uso de mundos virtuales en la educación

Implantar y utilizar de manera eficaz un mundo virtual no es ni mucho menos una tarea trivial ya que para ello hace falta la conjunción de muchos requisitos y el cumplimiento de normas y criterios mínimos. En esta sección vamos a tratar de realizar una

panorámica de todos los puntos principales a tener en cuenta a la hora de implantar un mundo virtual orientado a la educación de alumnos de primaria y secundaria.

El equilibrio al que nos referíamos anteriormente es una de las problemáticas fundamentales hoy en día a la hora de integrar el uso de mundos virtuales en la educación. El empleo de mundos virtuales de baja personalización puede estar limitado y ser insuficiente para ser incluido de manera efectiva como herramienta auxiliar en la educación [20].

Por otra parte, el empleo de mundos virtuales como SecondLife y Opensim, que ofrecen total libertad de creación de contenidos, puede a su vez dificultar la creación de dicho mundo virtual. Demasiada libertad provoca incertidumbre sobre cómo acotar el contenido para que se ajuste a los objetivos y necesidades del material educativo que se desea impartir. Además, se añade también la necesidad para el profesorado de tener unos ciertos conocimientos técnicos de programación y/o de sistemas no triviales para poder configurar dichos entornos virtuales. Los alumnos a su vez necesitan también de unos mínimos conocimientos sobre el uso de los periféricos para manejarse en el entorno virtual, así como de una tutela de uso del mundo virtual.

Los contenidos anteriormente mencionados causan otra de las problemáticas. Algunos mundos virtuales como SecondLife tienen un claro enfoque hacia los adultos, y en paralelismo con el ideal de "vía de escape" pueden tener contenidos o temáticas inadecuados para niños (pornografía, violencia, drogas, etc.) [21]. Es por tanto necesario que los profesores se aseguren a la hora de hacer uso de un mundo virtual de que los contenidos del lugar donde van a realizarse las actividades sean adecuados.

En la línea del contenido adecuado para la edad de los alumnos, decidir sobre qué plataforma es la más adecuada no es tarea fácil ya que existen muchas (SecondLife, Opensim, Active Worlds, Unity, Minecraft, High Fidelity, etc.) y todas tienen aspectos positivos y negativos. Es relevante mencionar que la elección de una u otra plataforma conlleva distintos costes económicos de implantación, configuración y mantenimiento. Estos costes económicos, ya sean mediante contratación de servicios externos, o alojados por el propio centro, suelen ser a menudo bastante altos, por lo que es necesario que para ser implantados vengan apoyados y avalados por nuevos proyectos de política educativa [22], lo cual no siempre es el caso.

2.2.4 Conclusiones

Cómo hemos podido observar, existen problemáticas sin resolver a la hora de usar los mundos virtuales como herramienta de apoyo a la enseñanza, ya sea para alumnos de edades elementales o de alumnos adultos que sigan entrenamientos concretos. En este último caso, para la realización de actividades de repetición para su perfeccionamiento, parece que el empleo de los mundos virtuales se ajusta perfectamente.

En el caso del empleo de los mundos virtuales para estudiantes en edades elementales y de educación secundaria existen riesgos en caso de mal uso de la plataforma, pudiendo desviarse los resultados de la intención inicial, debido a la libertad que ofrecen los mundos virtuales. Sin embargo, la proliferación en la última década de iniciativas del uso de los mundos virtuales indica que el profesorado está realmente consiguiendo resultados positivos, no sólo a la hora de transmitir conceptos curriculares obligatorios en la enseñanza básica de los estudiantes, sino también motivando a los alumnos y fomentando la creatividad, la colaboración, la comunicación y el aprendizaje auto-estimulado, todas ellas prácticas alineadas con las metodologías pedagógicas modernas.

2.3 Opensim y otros Mundos Virtuales y Sandboxes

2.3.1 Introducción

Opensim es una de las plataformas de creación de mundos virtuales más potentes para su uso en la educación. Al tratarse de un gemelo de SecondLife, el entorno virtual más popular en el mundo, pero de código libre, nos permite una total libertad de creación de contenidos y de gestión de un mundo virtual. En este proyecto se eligió como plataforma Opensim por su flexibilidad y la capacidad de reúso del material que se construya. Sin embargo, se tuvieron en cuenta otras plataformas que a continuación analizaremos. Más detalles sobre las características de Opensim serán dados en el capítulo 3 de esta Memoria Proyecto de Fin de Carrera.

2.3.2 SecondLife

SecondLife es el mundo virtual del cuál nació Opensim (Más detalles en la sección 3.2 de esta Memoria). Multitud de iniciativas de uso como herramienta principal para usos educativos nacieron en SecondLife tras su lanzamiento oficial en 2003. Sin embargo, el principal problema es que se trata de una plataforma de pago, en la que el alojamiento de nuestro mundo virtual, las regiones, los objetos y cada avatar son de pago, lo cual aumenta muchísimo los costes del proyecto. Además, el control sobre los contenidos que puede ver el menor, o sobre las personas con las que puede interaccionar es muy relativo.

2.3.2 Minecraft

Minecraft es un videojuego de tipo sandbox basado en la construcción utilizando bloques. Existe una versión orientada a su uso como herramienta educativa llamada "Minecraft Education", la cual cuenta con una extensa comunidad de desarrolladores que comparten material para las clase. Al tratarse de una versión lanzada en 2016 a raíz del éxito cosechado por el videojuego desde 2011, estamos ante una opción que ofrece plenas garantías en cuanto a estabilidad y fiabilidad en su uso. El desarrollo durante años de los propios jugadores, que con su ingenio han llegado a crear maravillas, tales como calculadoras modernas funcionales, monumentos reales e incluso ciudades, además del propio desarrollo de la compañía, con constantes actualizaciones y mejoras, hace de Minecraft una gran alternativa. Sin embargo, el elevado coste de uso a largo plazo -5 dólares al año por usuario- lo hace una alternativa cara con respecto a Opensim, teniendo en cuanta que a fecha de escritura de esta Memoria de Proyecto de Fin de Carrera el Metaverso realizado consta de más de 400 usuarios de los cuales más de 200 son activos mensualmente. Además, la flexibilidad de Minecraft es muy inferior a la ofrecida por otras alternativas.

2.3.3 Unity

Unity es el motor de videojuego multiplataforma más extendido actualmente. Cómo su nombre indica no es un mundo virtual en sí mismo cómo lo pueden ser Opensim, SecondLife o Minecraft con sus similitudes o diferencias. Al tratarse de un motor de

videojuego es una herramienta en la que desarrollar por completo y desde el principio la idea que tengamos. En nuestro caso, a pesar de ofrecernos la posibilidad de crear un mundo virtual perfectamente acorde a nuestras necesidades, se desmarca de la idea principal de este proyecto, que es implantar configurar y mantener un sistema que aloje un mundo virtual para los colegios asociados al Proyecto, y no desarrollar dicho mundo virtual.

2.3.4 High Fidelity

High Fidelity es probablemente la alternativa con mayor potencial de uso para la educación en los próximos años y actualmente está todavía en fase inicial de implementación. La compañía, fundada por el mismo fundador de Linden Labs, la cual desarrolló SecondLife, ofrece una plataforma donde sus usuarios pueden crear mundos virtuales. De código abierto, dotada de unos gráficos modernos y compatibilidad nativa con interfaces físicas cómo Occulus Rift, High Fidelity es la alternativa futura de mayor potencial.

2.4 Conclusiones

A lo largo de éste capítulo hemos abordado las características en general de los mundos virtuales y las ventajas y desventajas que ofrecen en los distintos usos más comunes en la actualidad. Dentro de la gran variedad de plataformas para el desarrollo de mundos virtuales, Opensim es a día de hoy el que mejor se ajusta para la realización de este Proyecto de Fin de Carrera. La posibilidad de configurar el mundo virtual con distintas opciones según las necesidades del proyecto a todos los niveles posibles, su reducido coste de implantación, su gran escalabilidad, además del vasto conocimiento previo sobre la plataforma de los profesores asociados a este proyecto que lo emplearán en sus clases, nos hizo decantarnos por Opensim cómo la plataforma más adecuada para este Proyecto.

3. Opensimulator

3.1 Introducción

En este capítulo haremos un análisis de la plataforma Opensimulator. Para ello comenzaremos con un breve repaso sobre su historia y sus orígenes. A continuación, expondremos algunas de las características principales que posee Opensim dentro del mundo virtual. Seguidamente entraremos más en detalle técnico, exponiendo primero los distintos modos de configuración de Opensim y después los tipos de comunicaciones y protocolos que conectan los clientes y los distintos servidores entre sí.

3.2 Orígenes y evolución de Opensimulator

Opensimulator, u Opensim (no confundir con el proyecto de mismo nombre ligado a la simulación biomecánica) es el nombre de un proyecto que nació en 2007 como plataforma para la creación de mundos virtuales. Opensimulator es a fecha de redacción de este documento la principal alternativa a SecondLife, entorno propiedad de Linden Labs, creado en 2003 para ofrecer a desarrolladores y usuarios por igual un entorno similar pero totalmente abierto y gratuito.

En 2003 ya existían y habían existido varios proyectos de creación de entornos virtuales de código libre que habían fracasado por la enorme tarea que suponía desarrollar todas las comunicaciones cliente-servidor. Sin embargo, la publicación en 2007 del código del cliente de SecondLife por parte de Linden Labs dio a los desarrolladores originales de Opensim el impulso inicial que necesitaban para comenzar a desarrollar una nueva plataforma para la creación de entornos virtuales. Sabían del potencial que un mundo virtual de código libre podía tener para multitudes de aplicaciones. Algunos de los desarrolladores del proyecto por entonces trabajaban en IBM en la investigación de oportunidades en lo referente a internet en entornos 3D, y rápidamente se hicieron eco de un nuevo proyecto llamado Opensim. En los meses posteriores fueron interesándose cada vez más desarrolladores, creándose ya los primeros *Grids* donde crear y compartir

contenido, algunos de los cuales como OSGrid o Kitely siguen existiendo hoy en día con miles de usuarios activos al mes.

Como ya se ha comentado anteriormente lo que hace tan potente a Opensimulator es que sea una plataforma de código abierto, ya que de esta forma cualquier persona puede aplicar mejoras al código existente que contribuyan a mejorar la plataforma. De la misma forma, también hay desarrolladores que han aumentado el valor de Opensim creando programas o funcionalidades que se asocian a Opensim de manera externa, tales como interfaces web, mensajería asíncrona, etc. Algunas de éstas serán añadidas a nuestro sistema, tal como se especifica en la sección 4.3

3.3 Características básicas principales

Opensimulator es una plataforma multiusuario de código abierto que permite la creación de mundos virtuales de acuerdo a las necesidades del desarrollador o de los usuarios. El mundo virtual es habitado por avatares, que es el nombre que adquieren los usuarios al conectarse al mundo virtual. Los avatares pueblan cuadrículas de terreno dentro del mundo virtual conocidas como regiones o islas. Dichas regiones son creadas por el administrador del metaverso y son visitables por avatares que, en función de la configuración y restricciones aplicadas por el dueño del metaverso, pueden crear contenido mediante la creación de objetos y scripts programables, o en entornos más restrictivos pueden hacer uso del entorno y las actividades propuestas por el propietario del mundo virtual. Para poder acceder al mundo virtual se necesita un visor dedicado especialmente a ello (para este proyecto usaremos el visor **Singularity**) así como los datos de acceso al mundo virtual, es decir la IP y puerto del servidor que aloja la plataforma, y la dupla de credenciales usuario/contraseña para poder iniciar sesión en el mundo virtual con nuestro avatar.

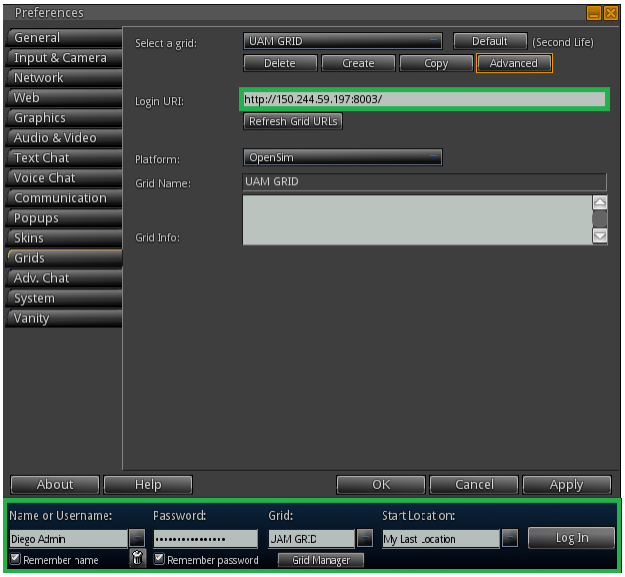

Ilustración 4: Datos necesarios para conectarse a UAM GRID

Estos avatares tienen total capacidad de caracterización pudiendo cambiar su apariencia y sexo para hacerlos únicos y así obtener una mayor sensación de inmersividad en el mundo virtual.

3.3.1 Avatares

Los avatares en Opensim son las representaciones virtuales del alter ego de la persona detrás del teclado y el ratón que los maneja. En la actualidad la gran mayoría de mundos virtuales, casi todos contenidos dentro de la categoría de videojuego, permiten algún tipo de personalización de la entidad que se va a manejar dentro del mundo virtual, tenga ésta forma humanoide o no (coches, aviones, etc.), pero casi siempre con ciertas restricciones. Opensim permite la modificación de todos los aspectos físicos del avatar: sexo, altura, rasgos faciales y físicos, ropa, etc.

3.3.2 Inventario

El inventario es un elemento crucial de Opensim. En él se encuentran todos los objetos que posee el avatar. En el inventario se encuentran distintos tipos de objetos, como los objetos que confieren su aspecto físico al avatar (Body parts) y los objetos de tipo ropa, que se pueden emplear para cambiar la apariencia del avatar. Además, en el inventario se pueden guardar objetos, scripts, calling cards, y otros recursos.

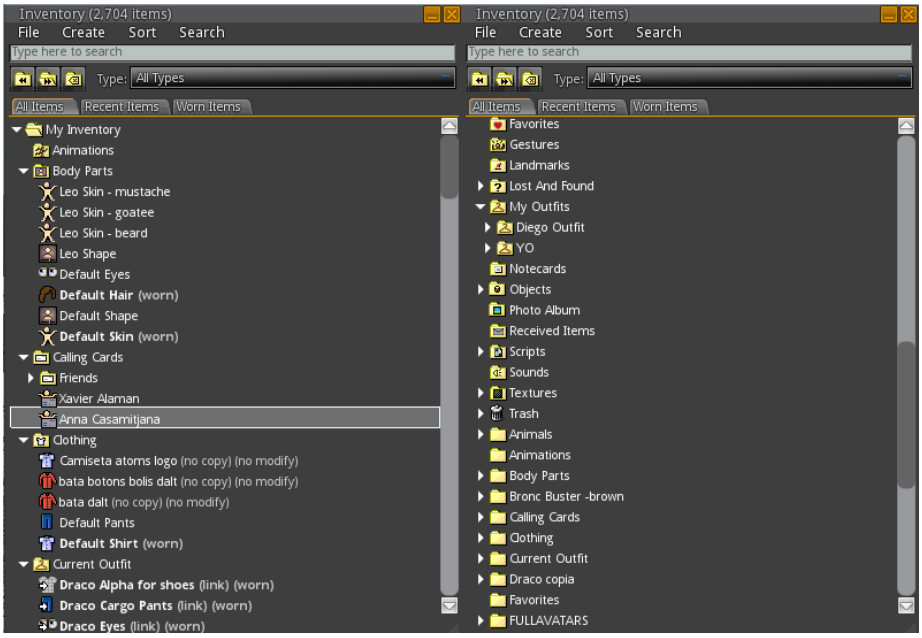

Ilustración 5: Inventario

3.3.3 Prims

Los Prims son los objetos básicos que se crean en Opensim. Un prim es un objeto que parte de una forma geométrica básica, y que se puede modificar de tamaño, forma, textura, posición etc.

Ilustración 6: Creación de un prim

La agrupación de distintos prims es lo que nos permite la creación de objetos complejos que sería imposible construir con un sólo prim, tales como edificios, máquinas, vegetación, etc.

3.3.4 Scripts

Los scripts son la herramienta que nos proporciona el simulador para "dar vida" a nuestro mundo virtual. Los scripts realizan acciones sobre los propios objetos en los que se alojan, en los objetos a su alrededor, y pueden incluso interactuar con los avatares cercanos. Como en cualquier lenguaje de programación avanzado las posibilidades que ofrece son casi infinitas, por lo que los scripts son el motor que mueve la máquina de la actividad, o evento que se quiera organizar en una región. En este proyecto se realizó una serie de scripts los cuales se detallan en la sección 5 de esta memoria.

3.3.5 Comunicaciones

Los distintos tipos de estándares y protocolos de comunicación empleados por Opensim para las comunicaciones Cliente-Servidor y Servidor-Servidor son de sobra conocidos en las comunicaciones por internet. Como explicaremos en la sección 3.4.2, el modo Grid es el empleado en este proyecto, en el que existen tres máquinas comunicándose entre sí e intercambiando información: el cliente (el visor Singularity), el servidor del simulador de regiones y el servidor de usuarios. Los protocolos empleados para estas comunicaciones son **HTTP** a nivel de aplicación y **UDP** a nivel de transporte, además de otros protocolos empleados puntualmente como **GVSP** o **MPEG**. En la siguiente figura podemos ver el diagrama básico de comunicaciones.

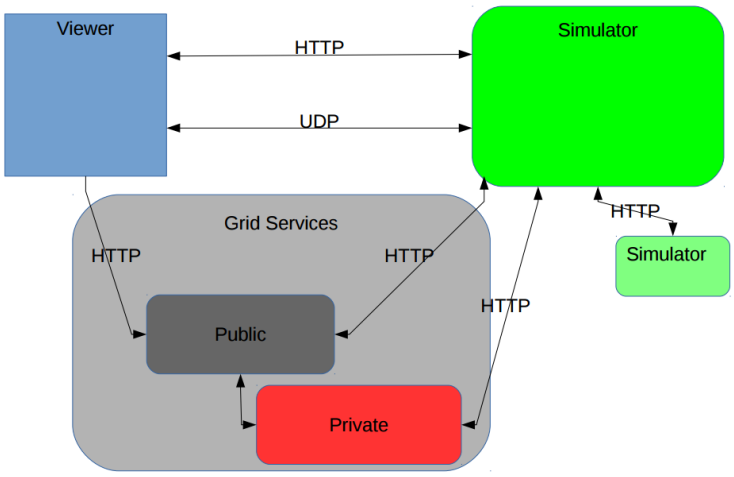

Ilustración 7: Comunicaciones Cliente-Servidor (Fuente: Opensimulator.org)

3.4 Modos de Configuración

3.4.1 Standalone

El modo Standalone es el modo más sencillo de configuración de Opensim y por tanto el más recomendado para aquellos que por primera vez quieran montar un mundo virtual. En la siguiente figura podemos ver la estructura lógica de la configuración en modo Standalone.

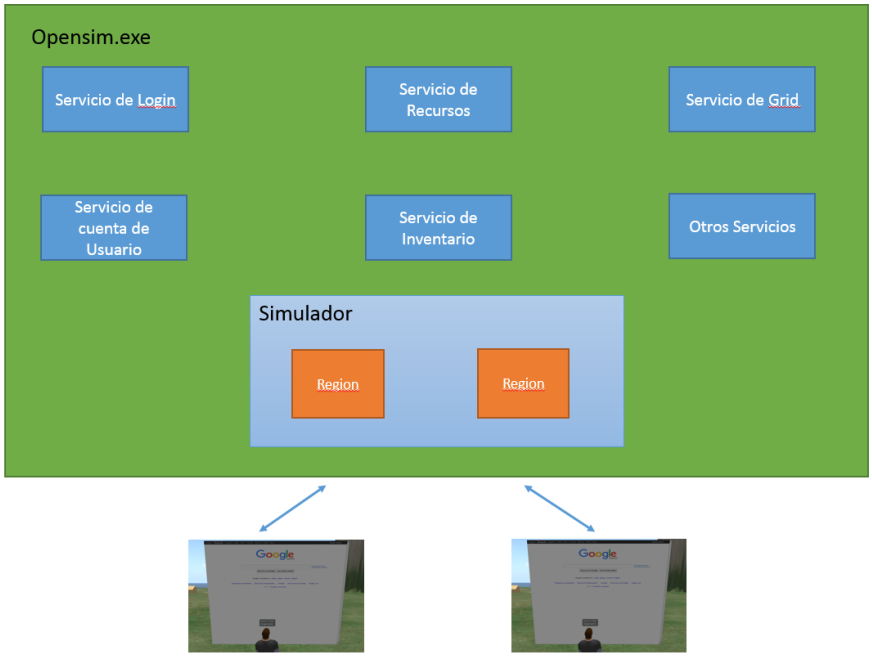

Ilustración 8: Modo Standalone

Como podemos observar, en este modo todos los servicios y el simulador funcionan dentro de un mismo proceso al ejecutar Opensim.exe. La ventaja evidente de este modo es su simplicidad y para entornos de tamaño reducido es la alternativa más adecuada y aconsejable. Sin embargo, esta misma característica es a su vez una desventaja con respecto al modo Grid, el cual detallamos en la siguiente sección, ya que en caso de crecer el Metaverso que hayamos construido nos veríamos limitados a un número máximo de regiones, avatares, prims, scripts, etc. que puede soportar el servidor. Una vez alcanzado el límite, para poder ampliar nuestro mundo virtual deberíamos ejecutar un nuevo proceso OpenSim.exe, que a su vez contendría nuevas regiones y nuevos datos que no estarían conectados de ninguna manera con nuestro primer proceso. Estaríamos por tanto creando dos mundos virtuales completamente separados sin ningún tipo de comunicación ni relación entre sí.

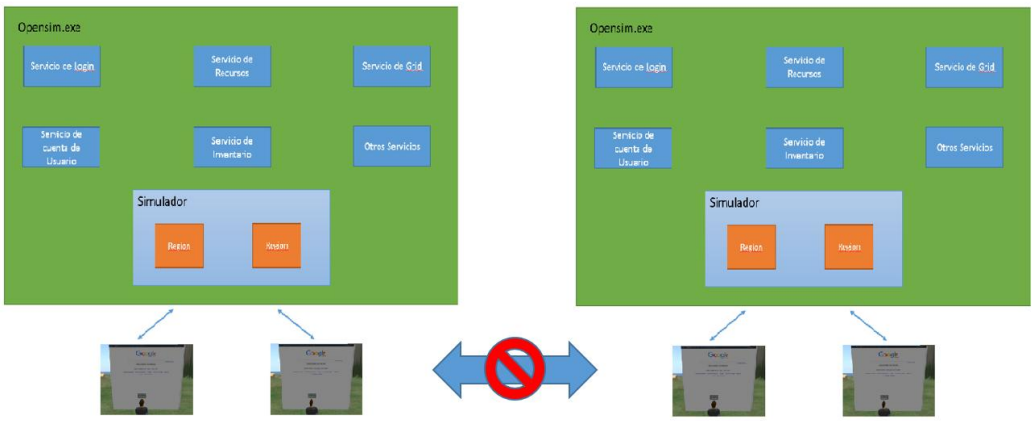

Ilustración 9: Dos Standalone no comunicados

3.4.2 Modo de configuración en Grid

Debido a las limitaciones del modo Standalone anteriormente vistas, se va a explicar las ventajas del modo Grid con respecto al modo Standalone. El modo Grid es un modo avanzado y complejo de configurar pero que por otra parte proporciona mayor flexibilidad al sistema para adaptarse a cambios en las necesidades que surjan debido al crecimiento de nuestro Metaverso.

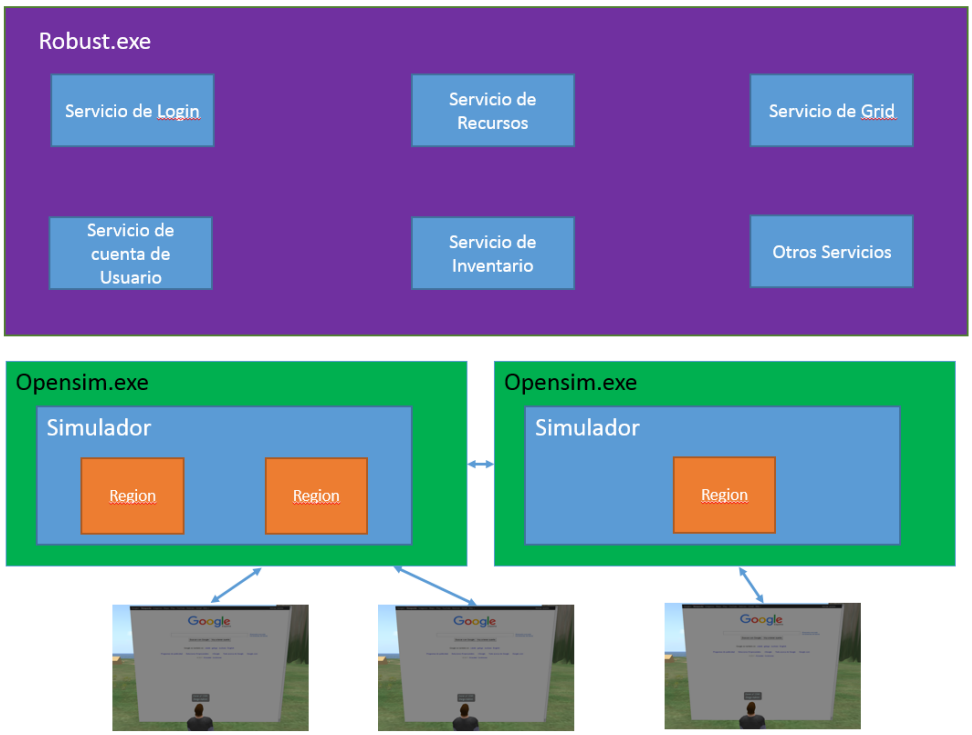

Ilustración 10: Modo Grid

En la anterior figura podemos observar notables diferencias con respecto al modo Standalone. Al configurar nuestro Metaverso en modo Grid podemos ejecutar varios procesos Opensim.exe conectados al proceso Robust.exe, el cual se encarga de gestionar los servicios de Backend transparentes al usuario.

La flexibilidad a la que nos referíamos anteriormente queda en evidencia al poder aumentar o reducir la cantidad de simuladores y de regiones conectados al proceso Robust.exe, pudiendo escalarse el tamaño de nuestro Grid en función del uso que se le dé a cada Simulador y sus regiones. Aumentar el número de simuladores no perjudica al usuario como lo hace el caso extremo revisado en la anterior sección. Un avatar puede pasar de una región ubicada en un simulador a otra región en otro simulador completamente distinto, ya que realmente forman parte del mismo Metaverso al compartir los servicios de datos que provee el proceso Robust.exe, al cual todos los simuladores y por tanto sus regiones están conectados. Esta característica también es completamente transparente para el usuario.

Por todas las razones argumentadas es por lo que se ha decidido implantar el mundo virtual con un modo Grid en detrimento del modo Standalone, mucho más susceptible de ser insuficiente y poco práctico a medio-largo plazo. Existe un tercer modo de configuración de mundo virtual, HyperGrid, el cual explicaremos a continuación, que ofrece unas posibilidades que para los requisitos y el uso de este proyecto son indeseadas.

3.4.3 Modo HyperGrid

Además de los modos de configuración anteriormente vistos, existe el modo HyperGrid, el cual no es un modo de configuración como tal como el Grid o el Standalone. HyperGrid es una opción que nos ofrece Opensim de conectar nuestro mundo virtual con otros. Esta conexión puede ser completa o parcial, pudiendo conectar sólo las regiones que se elijan a otros mundos virtuales, restringiendo el acceso de los visitantes externos a ciertas regiones. Existen distintas topologías para diferenciar el uso que hagamos de HyperGrid. Las siguientes imágenes representan algunos de los conceptos de uso más relevantes en forma de topologías que se le puede dar al modo HyperGrid.

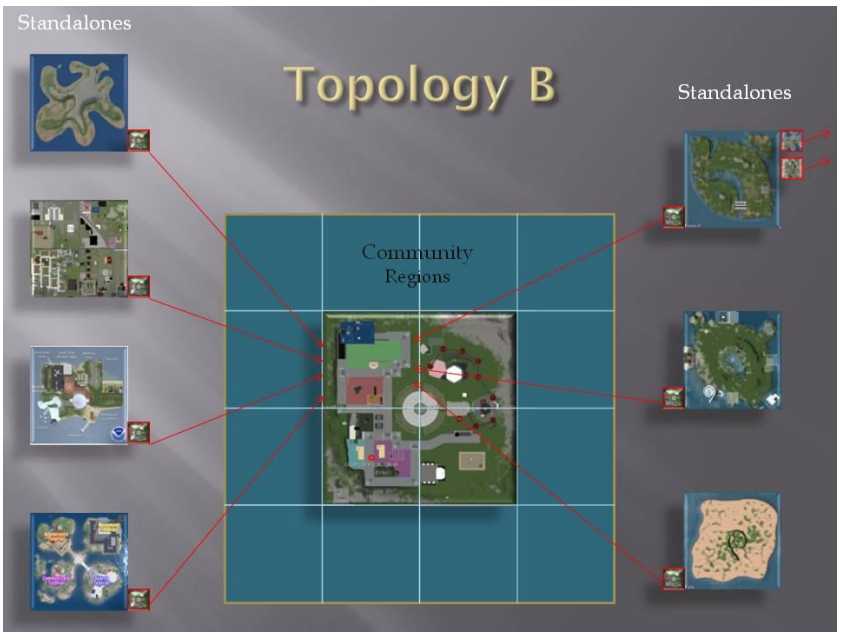

Ilustración 11: Topología Comunidad HyperGrid (Fuente: Opensimulator.org)

La topología B es un tipo de topología en donde existen una serie de simuladores en modo Standalone, cada uno con una o varias regiones. Como se ha comentado anteriormente, todos estos simuladores Standalone son independientes entre sí, cada uno es un Metaverso propio. Sin embargo, al conectar todos estos Metaversos a un mismo HyperGrid, todos los usuarios de cada simulador individual pueden encontrarse en este espacio común. Esta topología es útil para conectar con personas con intereses comunes para realizar actividades conjuntas. En el caso de este Proyecto de Final de Carrera, este tipo de topología tendría cabida, al poder separar cada instituto en simuladores independientes, pero dado que todos se encuentran en un proyecto común se decidió emplear el modo Grid obviando el modo HyperGrid.

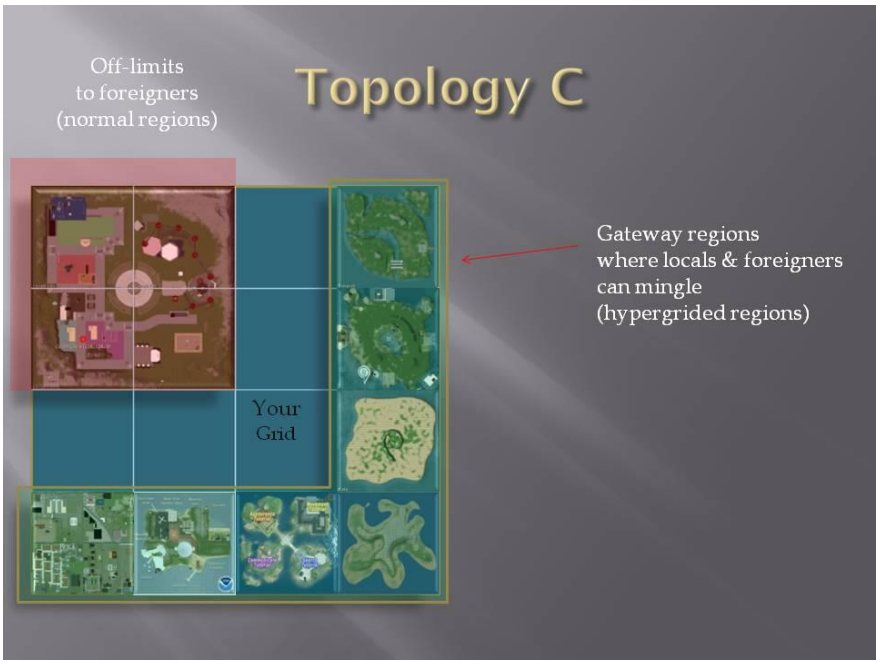

Ilustración 12: Topología HyperGrid con regiones con acceso restringido (Fuente: Opensimulator.org)

La topología C representa un Grid que sólo está en parte conectado a otros Metaversos, separando (no forzosamente geográficamente como en la imagen) las regiones visitables por usuarios externos a nuestro simulador de aquellas que queremos mantener privadas. Esta topología tiene funciones principalmente comerciales, en las que un proveedor de contenidos de Opensim puede abrir sus puertas a determinadas regiones de su Grid para que un usuario de otro mundo virtual pueda visitarlas sin necesidad de formar parte del

mundo virtual que está visitando. En caso de querer visitar aquellas regiones a las que no tenga acceso, deberá registrarse como miembro en dicho mundo virtual. Pasaría entonces a tener en este caso dos avatares totalmente diferenciados. Uno perteneciente al mundo virtual de origen, con acceso a su propio simulador y a las regiones de la imagen a través de HyperGrid. El otro avatar pertenecería al otro mundo virtual, pudiendo no sólo visitar estas regiones públicas sino también las privadas que no están conectadas al exterior mediante HyperGrid.

4. Implantación

4.1 Introducción

En este capítulo abordaremos los pasos realizados en la implantación y configuración del sistema completo y de las mejoras añadidas para aumentar la funcionalidad y usabilidad del metaverso.

Comenzaremos con el dimensionamiento de los servidores a utilizar basándonos en las previsiones iniciales de usuarios al inicio de este proyecto, así como de las previsiones de uso del Grid en un futuro próximo. Para ello nos apoyaremos en las experiencias de uso obtenidas por otros dueños y usuarios de mundos virtuales, así como de la información ofrecida por los paquetes de servicios de proveedores de alojamiento de Opensimulator, que habitualmente detallan las características ofrecidas en función del servicio contratado.

A continuación, detallaremos la secuencia de configuraciones efectuadas, desde el arranque del sistema con la configuración del simulador *out of the box* en modo *Standalone* hasta el estado final con el modo *Grid* y todos los añadidos. Primero empezaremos por aquellas características esenciales y de primera prioridad; las siguientes características añadidas son aquellas que, aunque no son imprescindibles, resultan de gran utilidad y comodidad tanto para los alumnos y profesores, como para los administradores.

Por último, analizaremos las capacidades reales de nuestro mundo virtual, realizando pruebas de estrés que nos permitan evaluar la escalabilidad de nuestro sistema actual en los términos de Opensim, es decir regiones, objetos, avatares y scripts que podrá llegar a soportar.

4.2 Dimensionamiento de servidores.

En esta sección analizaremos las necesidades de hardware de nuestro sistema para poder soportar desde un primer momento una gran cantidad de carga pero que a su vez soporte

el crecimiento que paulatinamente irá viviendo nuestro mundo virtual, añadiendo regiones, creando mas prims, ejecutando más scripts, y conectando más avatares

4.2.1 Estimaciones de uso preliminares

Para la estimación de las capacidades de los servidores a adquirir, existen distintas variables a tener en cuenta que afectan al rendimiento de los servidores, por lo que necesitamos implantar nuestro mundo virtual en unas máquinas que nos permitan cubrir las necesidades tanto a corto plazo, y ofrezcan margen para soportar el crecimiento y expansión. Las variables más relevantes son las siguientes:

- Número de avatares conectados concurrentemente.

- Número de regiones.

- Número de prims por región.

- Número de scripts ejecutándose y su complejidad computacional.

En este proyecto se ha partido con la idea de que los servidores que alojan el mundo virtual que se ha implantado para este proyecto van a estar destinados en una primera fase a tres clases distintas de tres centros educativos, dos de ellos de enseñanza secundaria y uno de educación para personas con necesidades especiales. Habitualmente, los centros de enseñanza secundaria tienen entre 25 y 30 alumnos por clase/profesor; por otra parte, los centros de educación especial tienen un máximo de 10 alumnos por clase/profesor en función de las necesidades. Por tanto, podemos hacer la estimación de que en el peor de los casos pueden llegar a conectarse al simulador **entre 70 y 75 personas** de manera concurrente si sumamos alumnos y profesores.

En cuanto al número de regiones, hemos estimado que cada centro necesitará de varias regiones, ya sea para el desarrollo de las actividades propuestas, entornos de pruebas, regiones de prueba Sandbox o regiones “Almacén” de recursos como prims o scripts. Aproximadamente prevemos que cada centro necesite de unas 6 u 8 regiones para sus actividades, y para regiones de aprovisionamiento de objetos y scripts pondremos a disposición en torno a 10 regiones. Esto hace un total aproximado de **28 a 34 regiones**.

La tercera y cuarta variables son complementarias y no tendría sentido dimensionar una sin la otra. Los prims son esencialmente objetos de formas geométricas que pueden unirse a otros para formar estructuras y objetos complejos. Si los propietarios no quieren que sus usuarios se encuentren con una región yerma, deberán incluir una gran cantidad de objetos para amueblar y darle a la región un aspecto atractivo y acorde a la temática. Pero un entorno además de tener sentido y ser visualmente atrayente debe ser funcional para lo cual es necesario agregar scripts que le "den vida". Para crear una región con una temática o un objetivo concreto se ha estimado que harán falta miles de prims y una fracción de ellos deberá tener scripts. Si contamos con que cada región necesita de tal cantidad de prims entonces necesitamos que nuestra máquina sea capaz de manejar unas cantidades que sobrepasen los **cientos de miles de prims**.

Habiendo analizado los parámetros aproximados que va a tener nuestro Metaverso, vamos a proceder a explorar en la siguiente sección las ofertas ofrecidas por proveedores privados que ofrecen servicios de alojamiento y/o configuración de mundos virtuales, para así poder hacernos una idea más ajustada del hardware necesario para montar nuestros servidores.

4.2.2 Proveedores privados

Una de las opciones más sencillas para desarrolladores de entornos virtuales que quieran disponer de un mundo virtual propio, ya sea para uso privado o para enlazarlo con algún Grid público, como InWorldz o Kitely, es la contratación de un servicio de alojamiento y configuración personalizada del mundo virtual. Existen multitud de compañías que ofrecen estos servicios para aquellas personas o instituciones que tengan la necesidad de tener su propio Metaverso, pero carezcan de los conocimientos o del tiempo que requiere poner a punto Opensim**.** En el caso de este Proyecto de Final de Carrera esto no es una posibilidad contemplada, pero estos proveedores ofrecen los servidores con unas características de hardware a distintos precios. Para estimar las prestaciones que deberán ofrecernos las máquinas que van a alojar nuestro mundo virtual vamos a apoyarnos en las características que ofrecen a sus clientes. Dentro de las ofertas ofrecidas, los proveedores limitan el número de avatares y de prims por región en función del tipo de hardware contratado, por lo que podemos emplear dichos valores para orientarnos sobre las características que necesitaran nuestros servidores.

Para ello hemos realizado una tabla comparativa entre distintos proveedores:

Empresa de hosting	Procesador	#cores	RAM (GB)	#avatares /región	#prims/región	#Regiones o simuladores
TomaHost-Región única	i7-4770 Haswell	4	-	80	25.000	-
TomaHost-Servidor dedicado	i7-4770 Haswell	4	32	-	-	-
CloudServe VPS	-	8	16	-	-	64 (regiones/simulador)
Hispalab	-	-	-	40	100.000	1 región
Kitely	Intel Xeon E5-2670	8	4	80	120.000	16 regiones

Tabla 1: Comparativa de proveedores de alojamiento privados

Como podemos observar, las ofertas de servicios de alojamiento varían de unos proveedores a otros. El problema principal es que la mayoría de compañías que ofrecen servicios apenas informan sobre el tipo de hardware en el que va a ejecutarse la solución que contratemos; solamente detallan la cantidad de regiones, prims, scripts, avatares que soportan. Además, cada compañía ofrece tipos de servicios distintos, orientándose unas a la cantidad de prims que una región puede soportar y otras a la cantidad de visitantes que puede tener cada región. Esto se debe a que las regiones pueden tener distintos usos y por tanto unas necesitarán mayor cantidad de prims y otras mayor espacio para visitantes.

El análisis de los servicios que ofertan proveedores especializados en Opensim es por tanto incompleto y no ofrece una respuesta definida, por lo que vamos a apoyarnos también en experiencias de usuarios de la red que hayan creado sus propios mundos virtuales mediante Opensim.

4.2.3 Experiencias de otros usuarios y dueños de Metaversos

En la práctica es imposible saber con exactitud qué requisitos mínimos son necesarios, ya que Opensim no es un videojuego al uso con escenarios fijos en cuanto a número de objetos y un soporte máximo de usuarios por región/instancia en ejecución/servidor. Al poder crear los usuarios el propio contenido y permitirse la libre circulación de avatares, las cargas en ciertas regiones o servidores puede variar en cualquier momento.

Cómo hemos visto los propios proveedores no parecen tener tampoco los mismos criterios sobre requisitos mínimos en función de avatares, scripts, regiones o prims. Para poder decidir mejor las características del hardware que vamos a necesitar se abrió un hilo de consulta en el foro de OSGRID, uno de los Grids más populares donde la gente conecta sus mundos virtuales para que otros los visiten. A continuación, exponemos las respuestas obtenidas.

- "50 users will require quite a beefy machine. Osgrid will run of as little as a Raspberry Pi, but to host 50 concurrent users i'd look at at a quad core or better, with at least 16GB of RAM. The more RAM and bandwith you feed it, the better it will perform."

- "You dont need a whole lot of diskspace, you're unlikely to use more than 2 GB, even if you make extremely large builds with loads of scripts. Raw CPU power, RAM and Bandwith. The more you have, the better it will work."

- "I would recommend 16 gigs at least, and depending on what you find a 6 or 8 core would not hurt either."

- "As a "rule for bare minimum", stick to 1 CPU and 1 GB of RAM per region (this should give you "normal" performance throughout your regions). Obviously all depends on the total amount of objects, scripts and concurrent visiting users."

Estas respuestas son muy similares a las obtenidas en algunas sesiones de chat IRC -realizadas durante el estudio de requisitos de hardware para este Proyecto de Final de carrera- de Opensim donde muchos expertos se conectan para informar de errores o dar soporte a otros usuarios. Así mismo, existe una sección en la Wiki del proyecto de Opensim que expone lo siguiente:

"As a rule of thumb, a region with lots of avatars, 15000 or more prims and 2000 scripts may require 1G of memory. So a simulator with 4 such regions may require 4G. One could use less memory if not all regions will be occupied with avatars simultaneously, or where there are fewer scripts, for instance."

Por tanto, parece que nuestras primeras estimaciones con respecto a la necesidad de un hardware de altas prestaciones son correctas, siendo recomendados más de 4 núcleos para la CPU y al menos 16GB de RAM. El ancho de banda también parece ser importante, pero dado que las máquinas se instalarán en la sala de servidores de la Escuela Politécnica Superior con conexiones de 10GigabitEth, no deberíamos tener problemas en este apartado. Otro aspecto mencionado es el del tamaño de los discos duros, los cuales no parece ser necesario que deban tener mucha capacidad, lo cual encaja con el análisis de proveedores que no mencionan esta característica en ningún momento. Sin embargo, montaremos discos duros adicionales para poder almacenar Backups de regiones, inventarios e imágenes del Sistema Operativo.

4.2.4 Conclusiones y elección del hardware y sus características

A pesar de la escasa información obtenida sobre las características del hardware mínimas o recomendadas para montar una plataforma Opensim que funcione de manera fluida, necesitamos elegir qué tipo de hardware vamos a necesitar para poder realizar este proyecto. Como concluimos anteriormente, necesitaremos unas máquinas que desde un primer momento puedan soportar unas 75 personas conectadas concurrentemente, más de 30 regiones, cientos de miles de prims y cientos de scripts. Además, deben tener capacidad suficiente para poder soportar ampliaciones del Metaverso en caso de que nuevos centros o clases de los centros ya incluidos en el proyecto se unan al sistema, lo que conllevaría un incremento en todos los parámetros (regiones, avatares, scripts, etc.).

Parece lógico por tanto pensar que, de acuerdo a lo visto en la tabla de comparativa de proveedores privados y las opiniones vertidas por usuarios experimentados, no va a ser suficiente con equipos con unas características similares a las de uno de uso doméstico de buen rendimiento. Necesitamos máquinas de uso profesional de gran rendimiento, con un procesador potente y una gran cantidad de memoria RAM. Teniendo en cuenta estas cuestiones, se decidió adquirir dos servidores para configurar nuestro sistema en modo Grid. Uno para el proceso Robust.exe que gestionará todo el Grid, y otro para el

proceso de simulación Opensim.exe, ambos con mismas características, pero con la diferencia de que el servidor para el proceso Robust.exe monta dos procesadores. Esto se hace en previsión de un futuro escalado del sistema, ya que como se ha explicado en la sección 3.4.2 sólo se puede disponer de un único proceso Robust para controlar todos los simuladores que se quieran añadir al Grid. Las características técnicas de los servidores son las siguientes:

- Servidor SuperServer SYS-5018R-M en rack

- Procesador Haswell 8C E5-2630V3 2.4G 20M 8GT/s QPI

- Memoria 32GB DDR4-2133 2Rx4 LP ECC REG RoHs

- Intel S3510 80GB, SATA 6Gb/s, MLC 2.5" 7.0mm, 16nm 0.3DWPD

- 2 HD Toshiba 3.5" 1TB SATA 6Gb/s 7.2K RPM 64M 512N

- Kit para SSD

Una vez escogido el hardware que usaremos para crear nuestro mundo virtual, vamos a pasar a explicar los pasos realizados en la configuración del sistema, desde la instalación de Opensim en su versión “Out of the box” hasta la versión final de configuración con todas las funcionalidades añadidas que fueron requeridas por los usuarios.

4.3 Configuración del sistema

En esta sección explicaremos los pasos seguidos en toda la configuración de nuestro mundo virtual en Opensim. Empezaremos con la instalación de los servidores en la sala de servidores de la Escuela Politécnica de la Universidad Autónoma de Madrid. A continuación, detallaremos la configuración de los archivos necesarios para arrancar Opensim en modo Grid de acuerdo con las necesidades de los distintos proyectos. Por último, recopilaremos todas las mejoras efectuadas al sistema respecto a las características que ofrece la versión Opensim “Out of the box”.

4.3.1 Configuraciones iniciales esenciales

Opensimulator ofrece una gran variedad de opciones que no vienen configuradas por defecto configuradas y que para poder emplearlas requiere de modificaciones en archivos de configuración, bases de datos, etc. A continuación, vamos a detallar algunas de las mejoras aportadas al mundo virtual que hemos creado, que han sido peticiones expresas de profesores para poder disponer de más y mejores herramientas para aumentar el potencial de Opensim. Así mismo, se han realizado configuraciones y añadido características transparentes a los usuarios para garantizar un correcto funcionamiento en todo momento del mundo virtual.

4.3.1.1 Avatares modificables

En primer lugar, se recibieron peticiones que mejorasen aspectos relacionados con la inmersividad en el mundo virtual, como es la posibilidad de editar el avatar de cada usuario. Opensim no permite por defecto la modificación del género de los avatares creados, que tienen la apariencia por defecto del avatar coloquialmente llamado “Ruth”, que como su nombre indica tiene género femenino. Este avatar por defecto tiene bloqueada la posibilidad de que con un simple click del ratón podamos desde el visor Singularity cambiar el género a masculino.

Ilustración 13: Avatar por defecto "Ruth"

Para poder entonces cambiar el género de un avatar de femenino a masculino hay que crear una nueva forma del cuerpo, “Shape”, y entonces usarla en el avatar y modificarla a partir de ahí. Se decidió que esto complicaba mucho la customización del avatar para

usuarios que se introdujeran por primera vez en Opensim, por lo que se decidió buscar una alternativa.

La alternativa elegida fue hacer uso de una de las funcionalidades que aporta la plataforma web WIFI, que a la hora de crear un avatar nos permite preseleccionar el género, cosa que no es posible cuando se crea un avatar desde la terminal de Robust. Esto nos permite que el usuario parta desde el inicio con un avatar que corresponda con su género real y por tanto se identifique rápidamente con él o ella. A partir de ahí el usuario puede en pocos minutos cambiar la apariencia de su avatar mediante los deslizadores que podemos ver en la anterior ilustración.

4.3.1.2 Grupos de usuarios

Otra de las necesidades primarias fue la posibilidad de agrupar a avatares en grupos. Los grupos en Opensim son una de las formas que los usuarios tienen de pertenecer a diferentes grupos de interés común con otros usuarios. En el caso de este proyecto la finalidad principal de los grupos es la de agrupar a los alumnos en clases, donde los alumnos pueden comunicarse con toda el aula incluido el profesor, por lo que se convierte en el canal y el lugar de comunicaciones común de toda la clase, como en la vida real sería el aula.

Además, el profesor puede utilizar el grupo para mandar mensajes de aviso, “Notices”, que les llegarán a todos los usuarios del grupo que estén conectados, y a aquellos que no estén conectados una vez se conecten. De esta manera puede enviar recordatorios o tareas a todos los alumnos sin necesidad de ir uno por uno.

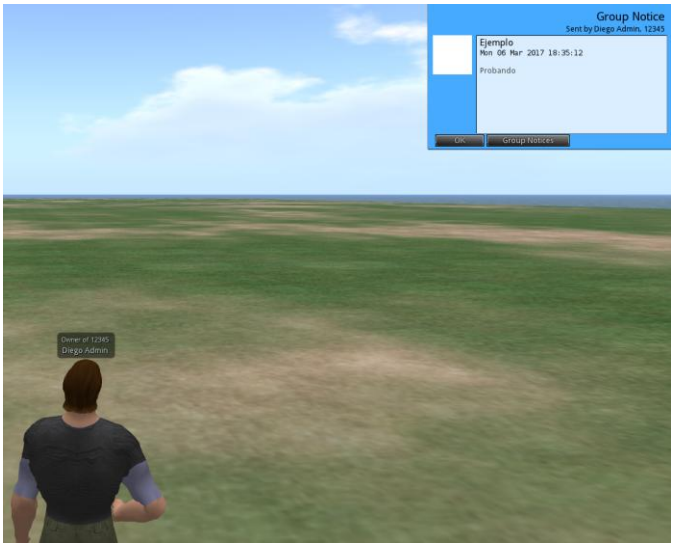

Ilustración 13: Aviso de Grupo

La creación de grupos en Opensim no viene configurada por defecto por lo que es necesario editar adecuadamente en el archivo Opensim.ini. y Robuts.ini, y además es necesario configurar unas claves de lectura y escritura en Opensim.ini que se correspondan con las establecidas en Backend con el archivo config.php. (C:\wamp\www\XmlRpcGroupsServer en el servidor Robust)

4.3.1.3 Mensajería asíncrona

La mensajería asíncrona ofrece la posibilidad de mandar un mensaje a un usuario que no está conectado, y que éste lo reciba la siguiente vez que se conecte. Aunque esta característica es similar a la del envío de mensajes a todo un grupo de usuarios anteriormente comentada, ofrece la posibilidad de mandar mensajes individuales, lo cual puede ser de particular utilidad.

La mensajería asíncrona no es una característica de Opensim "Out of the box", y a diferencia de los grupos, no es configurable en Opensim mediante configuraciones de archivos ".ini". Para poder activar la mensajería offline es necesario activar el módulo correspondiente en el archivo "Opensim.ini" y activar el servicio de mensajería offline en el archivo "Robust.ini". Además, se debe crear una tabla en la base de datos del simulador que tendremos que conectar con dos archivos que deberemos de crear en nuestro servidor de red Apache. Estos archivos serán los encargados de guardar en la

tabla los mensajes almacenados y de recuperarlos para que el simulador los envíe cuando se conecte el destinatario.

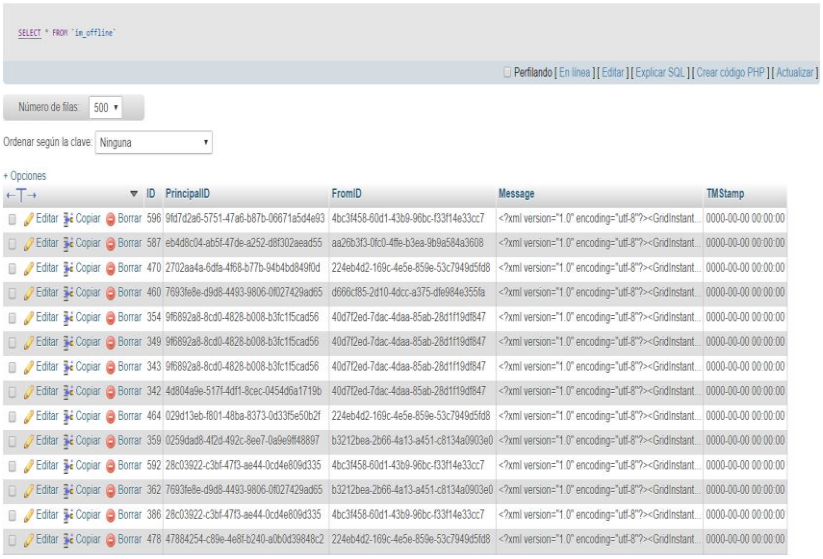

Ilustración 14: Tabla IM_Offline

4.3.1.4 Inserción de Streaming en Prims

La inserción de contenidos Web en Opensim es otra característica útil dentro de los mundos virtuales, ya que nos permite visitar páginas Web sin necesidad de tener que abrir el navegador. El problema es que cada avatar visualizará por separado el Streaming de vídeo como si lo viera en el navegador corriente, por lo que si lo pausa lo pausa sólo para sí mismo.

Esta característica viene por defecto desactivada en Opensim, ya que exige muchos recursos para el visor. Para activarla sólo es necesario habilitar la opción en el fichero Openmsim.ini.

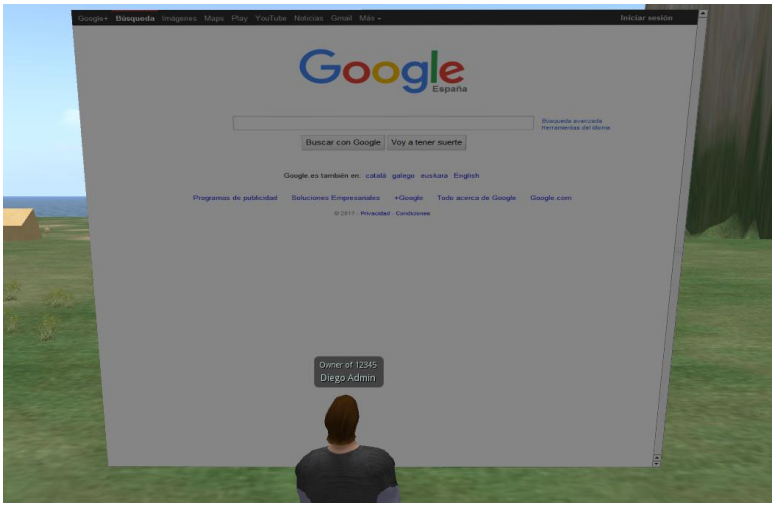

Ilustración 15: Web en un Prim

4.3.1.5 Suministro de contenidos prefabricados

Otra de las necesidades de los distintos centros es la de contar dentro del propio mundo virtual con una buena cantidad de objetos con los que empezar a construir sus entornos. Opensim es una plataforma en la que, aunque todavía se encuentra en fase Beta, los desarrolladores de contenidos llevan años creando contenido. Existen numerosas webs que ofrecen gratuitamente la descarga de objetos o incluso de regiones con gran cantidad de objetos. Estas regiones podemos cargarlas directamente a nuestro mundo virtual mediante la consola del simulador Opensim.exe. Una vez cargadas en nuestro mundo virtual los usuarios pueden visitarlas y crear copias de los objetos que se encuentran en estas regiones para después usarlos como punto de partida y modificarlos a su gusto o directamente colocarlos en su región.

4.3.2 Valores añadidos

4.3.2.1 Sim-On-A-Stick

La posibilidad de conectarse a Opensim sin necesidad de tener que conocer la dirección donde está alojado el mundo virtual, ni la dupla usuario/contraseña es también una funcionalidad a tener en cuenta. En particular, el Colegio de Educación Especial "Santa Teresa de Jesús", en Granada, que ha creado un museo virtual, requería de una forma de conexión al museo de una forma sencilla para cualquier usuario. Se propuso la posibilidad de ofrecer una versión de Singularity instalada en una llave USB que con sólo conectarla a un ordenador personal se conectara automáticamente al mundo virtual.

Desde la versión de Singularity 1.8.6. (Anterior a la 1.8.7empleada en este proyecto) se soporta la portabilidad del visor, lo cual permite instalarlo en una llave USB.

La instalación y configuración es sencilla, ya que sólo es necesario modificar algunos archivos:

- Agregar los argumentos " --settings settings_singularity.xml --channel "Singularity" --portable --login usuario apellido contraseña" al ejecutable.

- Cambiar Default_grids.xml, agregando los datos de conexión a nuestro mundo virtual como son la URI y el nombre del Grid

Una vez realizados estos cambios podemos ejecutar Singularity Viewer y éste se conectará directamente al mundo virtual "Grid UAM" con el usuario elegido en la región donde se encuentra el museo virtual y podrán visitarlo.

4.3.2.2 Interfaz Web

Uno de los añadidos que se decidió incluir para el proyecto fue una interfaz Web para la gestión del mundo virtual creado. La estructura de la página Web forma parte del proyecto Diva Canto, el cual ofrece una versión de Opensim modificada con algunos añadidos. Entre algunas de sus características la Web nos permite:

- Creación de usuarios de manera libre sin tener que pasar por el administrador principal y que éste los tenga que crear a través de la consola. Sólo es necesario que un usuario con poderes de administrador (profesor) se conecte a la web con las mismas credenciales de acceso que al mundo virtual y podrá crear los nuevos usuarios. Adicionalmente, cómo se ha comentado anteriormente, se crean los usuarios a través de la Web como medida para solucionar el problema de cambio de Género del avatar Ruth que trae Opensim por defecto, el cual no permite cambiarlo.

- Actualización de datos de los usuarios: Los usuarios pueden conectarse a la Web y cambiar su contraseña y su correo electrónico.

- Gestión de inventario: Nos permite reordenar nuestro inventario, crear nuevas carpetas o sub-carpetas, y eliminar objetos.

- Habilitar o deshabilitar login: Esta funcionalidad permite a un profesor poder habilitar o bloquear a un usuario el acceso al mundo virtual.

- Control de creación de cuentas: La creación usuarios está abierta para cualquiera que visite la web. Pero, es necesario que un usuario administrador active la cuenta para que pueda conectarse al mundo virtual. Dado que se trata de un mundo virtual enmarcado en un proyecto con unos centros determinados y con un contenido controlado y adecuado para el uso que se le va a dar, tener una web pública con registros abiertos puede suponer un problema, por lo que se decidió establecer un cierto control.

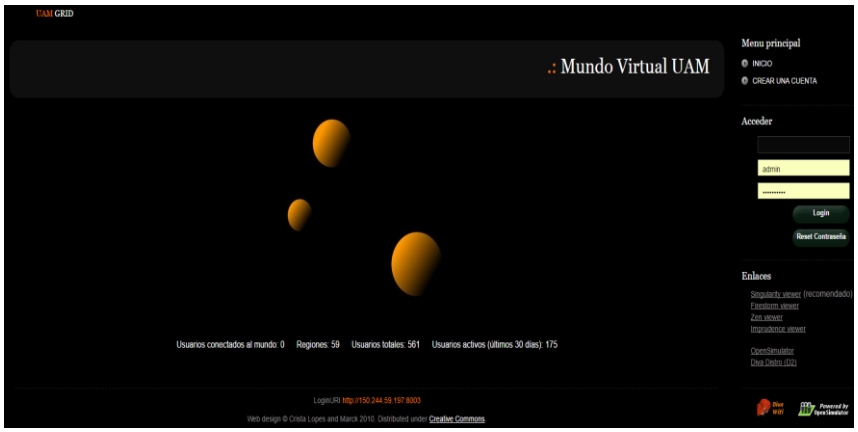

Ilustración 16: Web UAM GRID

Además, la web tiene en la pantalla de inicio enlaces de descarga al visor Singularity, la IP y el Puerto de acceso al mundo virtual, y algunos datos del Grid como el número total de usuarios, regiones y avatares conectados actualmente.

4.3.3 Mantenimiento

Además de la configuración que es visible para el usuario se ha procedido a automatizar ciertos procesos de mantenimiento transparentes al usuario para asegurar que una caída del sistema no perjudique al usuario con la pérdida de información vital. Los mantenimientos realizados incluyen:

- Backups diarios de las bases de datos.

- Backups diarios de todas las regiones y su contenido. Esto se configura activando el módulo AutoBackupModule en el archivo de configuración Opensim.ini. Este Módulo realizará una copia de todos los recursos de una región -prims, texturas, scripts, etc.- y

las almacena en archivos de tipo ".oar". Pese a que los Backups de las bases de datos ya realizan la copia de todos los datos que existen en nuestro mundo virtual, es útil realizar los Backups específicos de las regiones por separado, ya que estos archivos pueden después cargarse directamente al mundo virtual a través de la consola de Opensim.exe restaurándose la región a su estado en el momento de la copia.

- Backups diarios de las carpetas de Opensim y Wamp en los servidores Robust y Opensim. Esto nos permite tener backups de la configuración de nuestro mundo virtual. Si se aplican actualizaciones o se incorporan mejoras futuras al mundo virtual, podremos utilizar estos archivos para añadir otro simulador al Grid si se quiere distribuir la carga, o si se quiere crear un nuevo Grid.

4.4 Pruebas de estrés

La realización de pruebas de estrés es algo de vital importancia para poder determinar dentro de lo posibleácuales son los límites del sistema que hemos configurado e implantado. Como prueba de estrés no existe ninguna mejor que una real, esto es, comprobar el funcionamiento del sistema con un elevado número de avatares conectados a distintas regiones realizando actividades tales como desplazarse, crear objetos, activar scripts, navegar por páginas web incrustadas en prims, etc. El problema es que esto en la práctica es irrealizable ya que las distintas clases y centros que hacen uso de Opensim en el momento de redacción de esta memoria de Final de Proyecto de Carrera lo hacen en horarios distintos.

Existe una alternativa, que es el empleo de la aplicación PcampBot. Esta aplicación nos permite la creación de bots, que se conectarán a nuestro mundo virtual y adoptarán el comportamiento que decidamos, que puede variar entre que no se mueva a que se desplace constantemente por la región tocando objetos y texturas. El primero parece más destinado a comprobar la carga que genera una simple conexión al servidor, mientras que el segundo busca comprobar la carga máxima que un avatar puede generar sobre el servidor.

A pesar de que disponemos de esta herramienta, cómo se ha comentado anteriormente, no se puede comparar una gran cantidad de conexiones reales, desde distintas localizaciones, a distintas regiones y con comportamientos totalmente aleatorios y

humanos, con la emulación de conexiones desde una misma máquina y con comportamientos programados. Por tanto, los resultados obtenidos al finalizar las pruebas de estrés deberán ser valorados en su contexto, teniendo en cuenta, como se ha comentado anteriormente, que una conexión real sin duda añadirá una carga mayor sobre el sistema que una conexión simulada mediante la herramienta PcampBot.

4.4.1 Datos iniciales

En el momento de redacción de esta memoria el mundo virtual está compuesto por 59 regiones con más de 200.000 prims y más de 400 avatares creados. Sin ningún avatar conectado las máquinas tienen el siguiente uso de memoria y CPU:

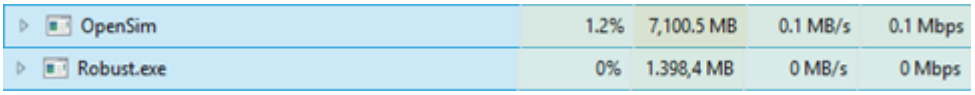

Ilustración 17: Carga de CPU y uso de RAM para los servidores de Opensim y Robust en reposo

Además del proceso Opensim el sistema está usando aproximadamente 6GB adicionales en distintos procesos como el antivirus, el servidor web Apache, el gesto de bases de datos MySql y distintos procesos asociados al Sistema Operativo, por lo que podríamos disponer de aproximadamente 23GB de memoria RAM para dedicárselos al simulador. En el caso del proceso Robust el sistema está usando 4GB en procesos similares a los anteriores mencionados. Estos datos corroboran la idea básica de la configuración del sistema en modo Grid, en donde los simuladores acumulan los datos de las regiones y objetos, ejecutan los scripts, y lo más importante, ejecuta el mundo virtual al completo (en nuestro caso, de haber más simuladores con las regiones repartidas, la carga se repartiría); mientras que Robust se encarga de gestionar el Grid y proveer de datos a los simuladores. En el caso de este proyecto, al encontrarse todavía en una fase temprana de despliegue y adhesión de participantes, no es necesario aún añadir simuladores adicionales (cómo se comentó en el apartado 3.4.2).

En esta prueba se ha realizado un test de conexión de 25 bots a una sóla región, simulando las condiciones de una clase típica de alumnos de instituto. Los mismos tests se han realizado conectando 25 bots a dos regiones, a tres y finalmente cuatro regiones distintas. De este modo podemos comprobar la evolución de la carga del simulador y de Robust.

4.4.2 Datos Finales

25 usuarios distribuidos en 1 Región:

50 usuario distribuidos en 2 Regiones:

75 usuarios distribuidos en 3 Regiones:

100 usuarios distribuidos en 4 Regiones:

Como podemos observar el incremento de usuarios progresivamente en distintas regiones revela varios datos relevantes:

- La carga sobre la CPU en el simulador incrementa gradualmente con el número de usuarios, así como la cantidad de memoria RAM que consume el proceso Opensim.exe. Esto es lógico debido a que los bots están configurados en un modo en el que se mueven constantemente por la región, interactuando con los objetos que encuentran a su paso, cuya información se encuentra en la base de datos asociada al Simulador, por lo que no fuerzan al proceso Robust a proveer de datos al Simulador.

- La máquina ejecutando el proceso Robust.exe, encargado de gestionar el Grid, no ve incrementada su carga sobre la CPU ni la memoria RAM consumida. Esto es debido a que los bots son avatares con inventarios vacíos y las texturas de su

aspecto son por defecto de Opensim, por lo que las llamadas a la base de datos para seguidamente proveer de datos al simulador son mínimas o casi inexistentes.

- Podemos concluir a raíz de estos datos que el sistema está perfectamente capacitado para soportar la conexión de decenas, sino de cientos de avatares reales de manera concurrente a lo largo de las distintas regiones existentes en el Metaverso a día de escritura de este proyecto.

5. Desarrollo de scripts para Museo Virtual

5.1 Linden Scripting Language

El Linden Scripting Language -LSL- es un lenguaje de programación de scripting, fuertemente tipado, basado en Java y C, desarrollado por Linden Labs específicamente para SecondLife y que también se emplea en Opensim. LSL es un lenguaje de programación orientado a eventos, es decir un tipo de máquina de estados finitos, y posee una amplia librería con más de 300 funciones predefinidas que el usuario puede ampliar definiendo sus propias funciones para usos complejos.

Los eventos se recogen dentro de uno o varios estados los cuales a su vez poseen una o varias funciones que se ejecutarán en función del evento que ocurra y del estado en el que se encuentre el script en el momento que ocurra ese evento. Entonces, de la misma forma que para un autómata finito (FSM), la ejecución de la función puede hacer que el estado cambie, haciendo que para el próximo evento se ejecute otra función distinta. Los scripts se programan y se colocan dentro de prims y recogen datos del sistema tales como tiempo, movimiento, chat, colisiones, etc. En el diagrama inferior podemos ver cómo se estructura un script.

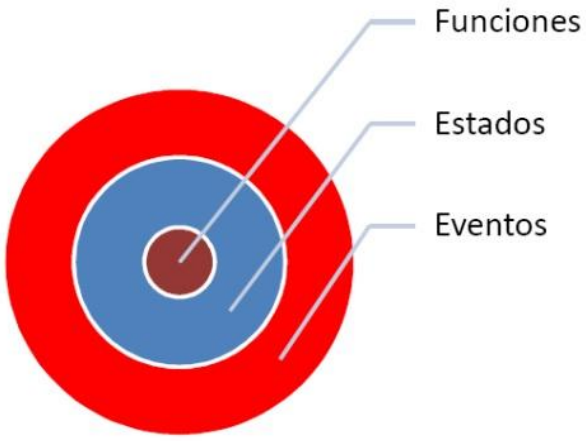

Ilustración 18: Diagrama de FSM de LSL

5.2 Scripts desarrollados

En colaboración con el Colegio de Educación Especial "Santa Teresa de Jesús", se ha desarrollado una batería de scripts para la región creada como museo virtual para que los estudiantes de dicho centro puedan exponer sus obras artísticas. Estos scripts están diseñados para sorprender al visitante y hacer que la visita no sea como a un museo convencional donde las obras se exponen en cuadros. Los scripts hacen que las obras cobren vida con movimientos, apareciendo y desapareciendo, construyéndose a sí mismos e incluso proponiendo mini juegos a los visitantes. De esta forma se pretende captar la atención del visitante y hacerle sentir que este nuevo modelo de museo no sólo se diferencia del clásico porque se visite a distancia y de forma virtual, sino que además ofrece añadidos que serían imposibles de tener en un museo corriente. Estos añadidos son los scripts incorporados a los cuadros, que harán a los cuadros cobrar vida, ya sea con movimientos, apareciendo o desapareciendo frente al visitante, e incluso interactuando con él.

5.2.1 Rotación de imágenes en un prim

Este script nos permite colocar una serie de imágenes en un prim, en este caso de la forma de un cuadro de pintura, y rotar la imagen mostrada en el prim periódicamente. Se puede elegir que las imágenes cambien aleatoriamente, asegurándonos de no repetir la misma imagen dos veces seguidas, o de cambiar las imágenes en orden. Se ha añadido un reset en forma de click al prim que resetea el script.

```
// Variables globales modificables
integer side = ALL_SIDES;//Lado a cambiar, # de lado, o ALL_SIDES para
cambiar todos los lados
float frequency = 2.0;//Cada cuantos segundos cambiar la textura. No
menos de 0.5


integer random = FALSE;//Mostrar las texturas aleatoriamente o en
orden
integer duplicatecheck = TRUE;//si random es true, esto comprobará si
la próxima selección es una textura distinta


////////////////////////////////////////////////////////
//Variables globales no modificables
```

```
integer numberoftextures;//número de texturas en el inventario
integer currenttexture;//número de inventario de la actual textura

changetexture()
{

llSetTexture(llGetInventoryName(INVENTORY_TEXTURE,currenttexture),side
);//Establece el nombre de la textura en uso
}

default
{
    on_rez(integer start_param)// Reset con Rez.
    {
        llResetScript();
    }
    state_entry()
    {
        llOwnerSay("Script rotador de imagenes");
        numberoftextures =
llGetInventoryNumber(INVENTORY_TEXTURE);//numero de texturas en el
inventario
        if(numberoftextures <= 0)//sin texturas
            llOwnerSay("No hay texturas en mi inventario. Por favor
añade texturas.");
        else if(numberoftextures == 1)//Solo 1 textura
            llOwnerSay("Sólo hay una textura en mi inventario, por
favor añade más para poder empezar a rotarlas.");
        else//más de 1 textura
        {
            llOwnerSay("Hay " + (string)numberoftextures + " imagenes
que se cambiarán cada"
                + (string)frequency + " segundos.");
            llSetTimerEvent(frequency);
        }
    }
    timer()
    {
        if(random)//muestra imagenes aleatorias
        {
            integer randomtexture;
            if(duplicatecheck)//nos aseguramos que no se repita la
imagen
            {
                do
                {
                    randomtexture= llRound(llFrand(numberoftextures -
1));
                    //llOwnerSay("r" + (string)randomtexture);//debug
```

```
                }while(randomtexture == currenttexture);//nos
aseguramos que la siguiente textura aleatoria no es la misma que la
actual
            }
            else//comprobación de duplicado
                randomtexture = llRound(llFrand(numberoftextures -
1));//generación aleatoria de número de textura
            currenttexture = randomtexture;//fijamos la textura actual
a la seleccionada aleatoriamente
            changetexture();//cambia la textura
            //llOwnerSay("c" + (string)currenttexture);//debug
        }
        else//no random, rotacion en orden numerico
        {
            ++currenttexture;
            if(currenttexture == numberoftextures)//si #textura actual
= numero de texturas, reseteamos el contador
                currenttexture = 0;
            changetexture();//cambia la textura
            //llOwnerSay("c" + (string)currenttexture);//debug
        }

    }
    changed(integer change)
    {
        if(change & CHANGED_INVENTORY)
        {
            llOwnerSay("Cambio en el inventario detectado.");
            numberoftextures =
llGetInventoryNumber(INVENTORY_TEXTURE);
            llOwnerSay("Hay " + (string)numberoftextures + " imagenes
que se cambiarán cada"
                + (string)frequency + " segundos.");
        }
    }
}
```

5.2.2 Prim que aparece/desaparece

Este script es uno de los scripts básicos con intención de introducir al visitante en la interacción. El script hará que el prim aparezca o desaparezca si en un radio alrededor suyo se escribe una palabra clave en el chat.

```
integer cloaked = TRUE;
default
{
    state_entry() {
        llSensorRepeat("", NULL_KEY, AGENT, 20, PI, 0.1); //Por orden
elegimos el radio, el ángulo siendo PI el máximo y el refresco de
escucha.
        llListen(0,"",NULL_KEY,"");//Canal a escuchar, 0 = canal
público
    }
    //Establecemos el mensaje a escuchar, en este caso "hola". Si
escuchamos el mensaje y el prim está oculto (cloaked), entonces
aparece.
    listen( integer channel, string name, key id, string message ){
        if( message == "hola" && cloaked == FALSE){
            llSetStatus(STATUS_PHANTOM, TRUE);
            llSetAlpha(1,ALL_SIDES);
         cloaked = FALSE;
        }
    } //Si el sensor deja de detector a una persona en su proximidad,
pasamos el valor cloaked a false, para que según se escuche la palabra
clave el prim se descubra.
    sensor(integer total_number) {
        if (cloaked == TRUE) {
            cloaked = FALSE;
        }
    }
        //Si el sensor deja de detector personas a su alrededor,
automáticamente desaparece el prim.
    no_sensor(){
        if(cloaked == FALSE){
            cloaked = TRUE;
            llSetStatus(STATUS_PHANTOM, FALSE);
            llSetAlpha(0,ALL_SIDES);
        }
    }
}
```

5.2.3 Prim que aparece mediante un acertijo

Este script continúa con la idea del script anterior. Una vez el visitante ha comprendido el funcionamiento básico para hacer aparecer y desaparecer el prim del script anterior, puede pasar a interactuar con este script. Mediante un juego de preguntas y respuestas

este script aparece ante el visitante o se reinicia en caso de errar la respuesta. Se puede aumentar la complejidad anidando preguntas y respuestas sucesivas al gusto del desarrollador con el uso de sucesivas sentencias "if" como se puede observar en el script.

```
integer cloaked = TRUE;
integer Start=FALSE;
default
{
    state_entry() {
        llSensorRepeat("", NULL_KEY, AGENT, 10, PI, 0.1); //Por orden
elegimos el radio, el ángulo siendo PI el máximo y el refresco de
escucha.
        llListen(0,"",NULL_KEY,"");
    }
    listen( integer channel, string name, key id, string message ){

        if(((message == "hola")||(message == "Hola")) && (cloaked ==
FALSE)){
            llWhisper(0, "Hola, un poco de mates, cuanto es 2x7?");
            Start = TRUE;
        }
        if(((message == "14") && (Start == TRUE)) && (cloaked ==
FALSE)){
            llWhisper(0, "Correcto");
            //llSetStatus(STATUS_PHANTOM, TRUE);
            llSetAlpha(1,ALL_SIDES);
            cloaked = FALSE;
            Start = FALSE;
        }
    }//Si el sensor deja de detector a una persona en su proximidad,
pasamos el valor cloaked a false, para que según se escuche la palabra
clave el prim se descubra.
    sensor(integer total_number) {
        if (cloaked == TRUE) {
            cloaked = FALSE;
        }
    }
        //Si el sensor deja de detector personas a su alrededor,
automáticamente desaparece el prim.
    no_sensor(){
        if(cloaked == FALSE){
            cloaked = TRUE;
            //llSetStatus(STATUS_PHANTOM, FALSE);
            llSetAlpha(0,ALL_SIDES);
        }
    }}
```

5.2.4 Prim que se eleva por proximidad

Este script se coloca en un prim específico en forma de cuadro en un tablón anclado al suelo. En un primer momento el prim está oculto bajo tierra/matorrales/agua, y al detectar la presencia de un avatar en sus proximidades se eleva desde su base realizando un arco de 90 grados hasta quedar erguido.

Ilustración 14: Elevación del cuadro por proximidad

```
integer open = FALSE;
default
{
    state_entry() {
        llSensorRepeat("", NULL_KEY, AGENT, 5, PI, 1);
    }
    sensor(integer total_number) {
        if (open == FALSE) {
            open = TRUE;
            llSetRot(llEuler2Rot(<0, PI_BY_TWO, 0>) * llGetRot());
        }
    }
    no_sensor(){
        if(open== TRUE){
            open = FALSE;
            llSetRot(llEuler2Rot(<0, -PI_BY_TWO, 0>) * llGetRot());
        }
    }
}
```

5.2.5 Prim que gira en círculo

El siguiente script es un script que hace rotar al prim alrededor de un punto geográfico fijado. Funciona calculando en intervalos de tiempo la siguiente posición del prim con

respecto a la actual. En caso de fijarse un intervalo bajo se puede conseguir la impresión de un movimiento fluido y continuo en vez de discreto.

```
vector root;
float ellipse_x;
float ellipse_y;
float angle = 0;
string axis_name = "Hub";
default {

    on_rez( integer Parameter ) { llResetScript(); }

    state_entry() {

        root = llGetPos;
        ellipse_x = 3.0;
        ellipse_y = 2.0;

        llSetTimerEvent( 0.1 );
        llSensor( axis_name, NULL_KEY, PASSIVE | ACTIVE, 20.0, PI );
    }

    timer() {
        angle += 5.0 * DEG_TO_RAD; if( angle >= TWO_PI ) angle -=
TWO_PI;
        float x = ellipse_x * llCos(angle);
        float y = ellipse_y * llSin(angle);
        llSetPrimitiveParams( [ PRIM_POSITION, root + <x, y, 0>,
PRIM_ROTATION, llEuler2Rot( <0.0, 0.0, angle> ) ] );
    }

    sensor(integer num_detected)
    {
        root = llDetectedPos(0);
    }
    no_sensor()
    {
        llWhisper(0, axis_name + " not found.");
        llSensorRemove();
    }
}
```

5.2.6 Prim que gira sobre sí mismo

A diferencia del script anterior, el cual se puede seguir con la mirada girando la cámara, este script gira sobre su propio eje vertical, por lo que la imagen no se puede visualizar a menos que el script se detenga. Para detener la rotación es necesario estar a una

distancia máxima del cuadro y clickar sobre él, de lo contrario no se detendrá. En caso de detenerse el cuadro, la rotación no se reanudará hasta que no detecte ningún avatar en el radio fijado.

```
// CASOS:
// 1. Mientras se esté dentro de la distancia máxima, si se toca el
cuadro cuando está girando este se detiene.
// Una vez detenido el cuadro no vuelve a girar hasta que no se
detecta a nadie a la distancia máxima de detección, se  toque el
// cuadro o no.
// 2. Si se está a una distancia mayor a la máxima, no importa que
toquemos el cuadro que este no se detendrá.
// 3. Si no se detecta a nadie en la proximidad el cuadro comienza a
girar de nuevo.
integer Switch = FALSE; //TRUE = ENCENDIDO (GIRAR), FALSE = APAGADO
(NO GIRAR).
integer Proximity = FALSE;
default
{
    state_entry(){
        llTargetOmega(<0,0,1>,PI,1.0); // Comienza a girar
        llSensorRepeat("", NULL_KEY, AGENT, 10, PI, 0.1);
    }
    touch_start(integer total_number){

        if(Proximity == TRUE && Switch == FALSE){
            llTargetOmega(<0,0,1>,0,1.0);
            Switch = TRUE;
            llWhisper(0,"Switch = FALSE. STOP SPIN");
            //llSay(0,llDetectedName(0) + " is facing " +
(string)llDetectedRot(0) + ". (That is, " +
(string)(llRot2Euler(llDetectedRot(0))*RAD_TO_DEG) + ".");

        }
        else if (Proximity == TRUE && Switch == TRUE){
            llTargetOmega(<0,0,1>,PI,1.0);
            Switch = FALSE;
            llWhisper(0,"Switch = TRUE. START SPIN");
        }
    }
    sensor(integer total_number){
        Proximity = TRUE;
    }

    no_sensor(){
        Proximity = FALSE;
        Switch = FALSE;
        llTargetOmega(<0,0,1>,PI,1.0);
    }
```

```
}
```

5.2.7 Script de Reconstrucción de Imagen

Este Script necesita de una serie de prerrequisitos para poder ser puesto en funcionamiento. En primer lugar es necesario recortar la imagen que se quiere mostrar en cuatro imágenes del mismo tamaño. A continuación se coloca cada imagen recortada en cada prim, y se colocan en la forma mostrada a la derecha en la imagen siguiente. Una vez hecho, se debe modificar el script a continuación en las líneas especificadas para cada número de la imagen recortada, siendo sólo necesario comentar o descomentar la línea adecuada. La idea es colocar cada trozo del cuadro en un prim separado con el script. Una vez se haga click en uno de los cuatro cuadros, éste se moverá desde el suelo en frente del avatar. Para reconstruir el cuadro se debe hacer click en todos los cuadros dispersados por el suelo. A continuación se presentan dos imágenes para ilustrar el efecto deseado.

Ilustración 15: Script antes y después de hacer click sobre cada imagen

```
integer Switch = FALSE;
integer Proximity = FALSE;

    default { state_entry(){
    llSensorRepeat("", NULL_KEY, AGENT, 10, PI, 0.1);
    }
```

```
touch_start(integer total_number){
    if(Proximity == TRUE && Switch == FALSE){
        Switch = TRUE;
                  // IMAGEN 1
                  llSetPos(llGetPos() + <1,2,-2.5>);
                  //IMAGEN 2
                  llSetPos(llGetPos() + <-4,3,-2.5>);
                  //IMAGEN 3
                  llSetPos(llGetPos() + <2,-1,-1.5>);
                  //IMAGEN 4
                  llSetPos(llGetPos() + <1,-3,-1.5>);
        llSetRot(llEuler2Rot(<PI_BY_TWO, 0, 0>) * llGetRot());
    }

    else if (Proximity == TRUE && Switch == TRUE){
        Switch = FALSE;
                  // IMAGEN 1
                  llSetPos(llGetPos() + <-1,-2,2.5>);
                  //IMAGEN 2
                  llSetPos(llGetPos() + <4,-3,2.5>);
                  //IMAGEN 3
                  llSetPos(llGetPos() + <-2,1,1.5>);
                  //IMAGEN 4
                  llSetPos(llGetPos() + <-1,3,1.5>);
        llSetRot(llEuler2Rot(<-PI_BY_TWO, 0, 0>) * llGetRot());
        }
    }

sensor(integer total_number){
    Proximity = TRUE;
}
no_sensor(){

    Proximity = FALSE;
    }
}
```

6. Resultados

6.1 Introducción

Uno de los resultados más relevantes obtenidos tras la realización de este proyecto es la acogida que ha tenido. En este capítulo se tratará de condensar los resultados obtenidos en los centros adheridos al proyecto. Expondremos dos casos concretos de uso a día de redacción de esta Memoria de Proyecto de Fin de Carrera, que han sido documentados en sendos artículos [23][24], y los resultados numéricos de uso del mundo virtual.

6.2 Casos de uso

6.2.1 Herramienta de refuerzo para estudiantes con trastornos de aprendizaje

Una de las prácticas de uso del mundo virtual es su utilización para clases especiales para alumnos con trastornos de aprendizaje. En este caso, el Institut Narcís Oller, situado en Tarragona, organizó unas clases de refuerzo para estudiantes con déficit de atención, hiperactividad, etc. Estos estudiantes emplearon el mundo virtual para recrear conceptos curriculares relativos a la geografía humana, tales como entornos agrarios, granjas, etc que más adelante fueron empleados por sus compañeros sin dificultades, por lo que ambos grupos de alumnos se beneficiaron, resultando en una experiencia positiva para ambas partes.

Ilustración 16: Recreación de entornos agrarios ([24])

6.2.2 Integración de alumnos con desórdenes cognitivos

Otro de los usos más destacables quizá sea el llevado a cabo por el Colegio de Educación Especial "Santa Teresa de Jesús", en Granada, donde los alumnos de este centro realizan obras de arte como parte de su educación especial. El empleo de Opensim aportó una nueva experiencia para estos alumnos, que realizaron actividades adaptadas a su nivel de discapacidad, consiguiéndose resultados muy positivos según las entrevistas semi-estructuradas llevadas a cabo a profesores del centro [23].

Ilustración 17: Museo 3D Asprogrades (Fuente: [23])

Además, mediante el uso del mundo virtual los alumnos exponen sus obras en el Museo 3D Asprogrades, para que visitantes externos puedan verlas, consiguiendo así tanto mejorar su visibilidad como una contribución en la financiación del centro. Este formato de visita virtual es posible gracias a las llaves USB preconfiguradas para acceder fácilmente al mundo virtual [4.3.2.1].

6.3 Datos globales

El éxito de la implantación del mundo virtual se ve reflejado en una serie de valores relevantes. En el momento de redacción de este documento, el mundo virtual cuenta con 64 islas (o regiones) con una población de 440 alumnos dirigidos por 28 profesores de 10 centros distintos.

En cuanto a la percepción de los usuarios las valoraciones son altamente positivas. Se realizaron una serie de entrevistas semi-estructuradas con algunos de los profesores que utilizan el mundo virtual en sus clases los cuales ensalzaron las cualidades de Opensim

como herramienta alternativa en sus talleres, para alumnos de edades y niveles distintos, obteniendo los resultados y objetivos fijados.

7. Manual técnico de configuración de Opensim

Este Manual tiene como finalidad explicar todos los pasos necesarios para la instalación y configuración de Opensimulator en modo Grid con las mismas características y mejoras introducidas en este Proyecto Final de Carrera. En este manual se empleará un sólo servidor para mayor sencillez, ya que escalar el sistema a varios servidores es trivial una vez se ha realizado la instalación en una misma máquina. Para la ejecución de este manual necesitaremos:

- Programa WAMP, el cual nos proporciona el gestor phpMyAdmin para nuestras bases de datos MySQL, y un servidor web HTTP, Apache. Podemos descargarlo del siguiente enlace http://www.wampserver.com/en/.

- Versión 0.8.2.1 de Opensimulator, empleada en este Proyecto Final de Carrera que podemos descargar del siguiente enlace http://opensimulator.org/wiki/Download.

Una vez descargados Opensimulator y WAMP, procederemos a descomprimir el primero e instalar el segundo en los directorios que elijamos. A continuación, vamos a detallar los pasos a realizar en las configuraciones específicas de cada instancia a ejecutar, es decir “Opensim.exe” y “Robust.exe”, para enlazar el simulador con Robust, así como de las configuraciones relacionadas con las bases de datos y el servidor web que nos proporcionan WAMP.

Para configurar un Grid con las mismas características que el que se ha implantado en este proyecto, sólo hay que copiar los archivos proporcionados y modificar las líneas que se detallarán en las siguientes secciones, que son principalmente datos para habiltiar las comunicaciones cliente servidor y servidor servidor, así como comunicaciones entre front end y back end. El resto de configuraciones son mejoras a la versión de Opensim “Out of the box”, y si se desea mantenerlas no es necesario modificarlas. En caso de querer desactivar alguna de las funcionalidades sólo es necesario comentar las líneas correspondientes.

Para simplificar la lectura se ha decidido asignar el término "Robust" al servidor encargado de ejecutar el proceso Robust.exe el cual se encarga de la gestión de los simuladores conectados a él. En este proyecto se ha empleado solamente un simulador para todas las regiones y la descripción de los pasos a seguir para configurar el simulador es idéntica a si se quiere conectar múltiples simuladores a un mismo Grid.

7.1 Configuración del simulador

Para configurar el simulador en una máquina tendremos que modificar los archivos Opensim.ini, GridCommon.ini y Regions.ini.

7.1.1 Opensim.ini

Esta sección detalla las líneas modificadas del archivo Opensim.ini así como en qué sección se encuentran dichas líneas.

```
[Const]
BaseURL = http://XXX.XXX.XXX.XXX
PublicPort = "8002"
PrivatePort = "8003"
```

Este es la configuración básica para conectar nuestro simulador al Grid que gestionará Robust. La IP será la de la máquina donde se esté ejecutando la instancia "Robust.exe". En el caso de esta memoria al ser la misma máquina podemos dejar el atributo con valor "http://127.0.0.1" o con valor "localhost", que es lo mismo. Los puertos pueden mantenerse por defecto ya que son los empleados de manera global por la comunidad Opensim.

```
[Startup]
crash_dir = "crashes"
```

```
[Map]
GenerateMaptiles = true
MapImageModule = "MapImageModule"
MaptileRefresh = 0
TexturePrims = true
```

Esto nos permitirá crear un mapa visual con la geografía de nuestro mundo virtual.

```
[Permissions]
permissionmodules = DefaultPermissionsModule
serverside_object_permissions = true
allow_grid_gods = true

region_owner_is_god = true
region_manager_is_god = true
parcel_owner_is_god = true
simple_build_permissions = true
```

Esta configuración nos asegura que los propietarios de las regiones tengan absoluto control sobre ellas para poder garantizar así que el contenido esté totalmente controlado.

```
[Chat]
whisper_distance = 10
say_distance = 20
shout_distance = 100
```

Ajuste de parámetros del chat para garantizar un uso realista del mismo.

```
[Messaging]
OfflineMessageModule = "Offline Message Module V2"
StorageProvider = Opensim.Data.MySQL.dll
MuteListModule = MuteListModule
MuteListURL =  http://XXX.XXX.XXX.XXX:8003/Mute.php
ForwardOfflineGroupMessages = true
```

Esta sección es una de las necesarias para poder activar posteriormente la mensajería asíncrona. El parámetro MuteListUrl debe ser descomentado para que funcione la mensajería asíncrona, aunque no exista el archivo Mute.php en la carpeta del servidor Web de la máquina.

```
[Groups]
Enabled = true
LevelGroupCreate = 0
Module = "Groups Module V2"
StorageProvider = OpenSim.Data.MySQL.dll
ServicesConnectorModule = "Groups Remote Service Connector"
LocalService = remote
GroupsServerURI = "${Const|BaseURL}:${Const|PrivatePort}"
MessagingEnabled = true
```

```
MessagingModule = "Groups Messaging Module V2"
NoticesEnabled = true
MessageOnlineUsersOnly = true
DebugEnabled = false
XmlRpcServiceReadKey    = 1234
XmlRpcServiceWriteKey   = 1234
```

Esta sección activa en el simulador la posibilidad de que los avatares creen grupos que comparten chat, archivos, etc.

```
[AutoBackupModule]
AutoBackupModuleEnabled = true
AutoBackup = true
AutoBackupInterval = 1440
AutoBackupBusyCheck = True
AutoBackupSkipAssets = False
AutoBackupKeepFilesForDays = 300
AutoBackupDir = "D:\BackupsOARs"
```

Estas líneas activan el módulo que trae Opensim para realizar Backups de las regiones, en este caso en un intervalo de 24 horas en el directorio especificado en la última línea.

```
[MediaOnAPrim]
Enabled = true;
```

Esto nos permite insertar contenido Web y/o vídeos en prims

```
[Terrain]
InitialTerrain = "pinhead-island flat"
```

Modifica la región inicial, que por defecto es una pequeña isla rodeada de agua y la sustituye por una región llana en su totalidad y sin agua.

```
[UserProfiles]
ProfileServiceURL = ${Const|BaseURL}:${Const|PublicPort}

[Architecture]
Include-Architecture = "config-include/Grid.ini"
```

Esta línea es la que configura el simulador y activa el tipo de arquitectura que se quiere ejecutar.

El resto de configuraciones se mantendrán por defecto comentadas/desactivadas.

7.1.2 GridCommon.ini

```
[DatabaseService]
```

```
StorageProvider = "OpenSim.Data.MySQL.dll"
ConnectionString="DataSource=”http://;Database=;User ID=;Password=;Old Guids=true;Convert Zero Datetime=True;Allow Zero Datetime=True;”
```

Deberemos escribir la IP donde se aloja nuestra base de datos del simulador, en nuestro caso en la misma máquina “localhost”; el nombre de la base de datos del simulador y la dupla usuario/contraseña de acceso a la base de datos. Las demás líneas deben dejarse por defecto ya que son las que conectan al Simulador con el Grid mediante los valores constantes `"${Const|BaseURL}:${Const|PublicPort}"` los cuales se leen del archivo Opensim.ini.

7.1.3 Regions.ini

Este archivo es el que se encarga de poblar de regiones el simulador, a continuación exponemos un ejemplo de cómo crear una región para el simulador:

```
[Region#1]
RegionUUID = 72919c78-64bf-4625-93b6-346cb874a14c
Location = 9999,9999
SizeX = 256
SizeY = 256
SizeZ = 256
InternalAddress = 0.0.0.0
InternalPort = 9999
AllowAlternatePorts = False
ExternalHostName = SYSTEMIP
lastmap_refresh="0"
```

Para cada región tendremos un UUID único, así como una localización geográfica en el Grid única. El tamaño máximo es de 256 unidades. El resto de parámetros no son relevantes y pueden ser dejados por defecto.

Como se explica en la sección 3.4.2 de este documento, cada región que creemos en este archivo pertenecerá a este simulador, pero en caso de existir varios simuladores en el mismo Grid cada región será visitable por los avatares del Grid, ya que los simuladores se encuentran en el mismo Grid

7.1.4 Mensajería asíncrona

Para poder habilitar la mensajería asíncrona necesitamos realizar pasos adicionales aparte de las líneas de código modificadas en el archivo “Opensim.ini”. Primero debemos crear dos archivos en la carpeta “wamp\www” donde hemos instalado WAMP en nuestra máquina de simulador. Estos archivos son “mysql.php” y “offline.php.

Ambos tienen el código disponible en el siguiente enlace: http://opensimulator.org/wiki/Offline_Messaging. Tan sólo deberemos modificar las primeras líneas del archivo "Offline.php" con los datos de conexión a MySQL y el nombre de la base de datos.

```
$dbName = "mysql";
$dbHost = "localhost";
$dbUser = "Usuario";
$dbPassword ="Password";
```

Para asegurar que no existan fallos al utilizarse este módulo de mensajería asíncrona, es preferible crear el archivo ".htaccess", cuyo código encontramos en el link anterior, en la misma carpeta que los anteriores archivos php descritos. Además, hay que activar el módulo mod_rewrite de Apache. Para ello navegamos a la carpeta de instalación de Wamp y abrimos el archivo httpd.conf que encontraremos en la ruta "bin\apache\apache2.4.9\conf" y eliminamos la almohadilla al principio de la línea para descomentar y así activar el módulo. Es necesario reiniciar el servidor Apache para que se registre este cambio.

Por último, deberemos crear en nuestra base de datos la tabla donde se almacenarán los mensajes que los usuarios envíen a otros usuarios no conectados. Para ello sólo tenemos que ejecutar las siguientes líneas de código sql desde la interfaz web que tenemos accesible via navegador web gracias a phpmyadmin incluido en WAMP:

```
CREATE TABLE IF NOT EXISTS `Offline_IM` (
 `uuid` varchar(36) NOT NULL,
 `message` text NOT NULL,
  KEY `uuid` (`uuid`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
```

7.2 Configuración de Robust

7.2.1 Robust.ini

Este archivo es el que se encarga de la configuración de todo el Grid y los servicios que va a proveer a los simuladores que estén conectados al Grid. A continuación se detallan las líneas que hay que modificar en el archivo Robust.ini:

```
[Const]
BaseURL = "http://XXX.XXX.XXX.XXX"
PublicPort = "8002"
PrivatePort = "8003"
```

Aquí establecemos unos parámetros que servirán de constantes para este archivo, teniendo así la comodidad de modificarlos una vez y hacerse efectivos los cambios en el resto del docmuento, ya que todos los parámetros que necesiten de la URL o alguno de los dos puertos vendrán bajo el formato “Const | BaseUrl” y sucesivos.

```
[WifiService]
GridName = "NOMBRE DE NUESTRO GRID"
LoginURL = "http://XXX.XXX.XXX.XXX:PUERTO"
WebAddress = "http://XXX.XXX.XXX.XXX:PUERTO"
AdminFirst = "ADMIN_NOMBRE"
AdminLast = "ADMIN_APELLIDO"
AdminEmail = "CORREO@DOMINIO.COM"

AccountConfirmationRequired = true
StatisticsUpdateInterval = 60
StatisticsActiveUsersPeriod = 30
AvatarAccount_Female = "Female Avatar"
AvatarAccount Male = "Male Avatar"
AvatarAccount_Neutral = "Neutral Avatar"
AvatarPreselection = "Neutral"
```

Esta sección cubre configuraciones relativas a la interfaz Web, WIFI. Debemos darle un nombre que será el que será visible en nuestra Web, así como las direcciones IP para registrarse y para visualizar la web (en este caso la misma). Es necesario establecer un admin y un correo electrónico.

Las siguientes configuraciones son para que el avatar que un usuario cree a través de la Web reciba activación por parte de un administrador para poder logearse al mundo virtual. Podemos además asignar avatares que hayamos creado previamente cómo avatares con un aspecto por defecto. Es necesario crear estos avatares con los mismo nombres para que los nuevos usuarios a partir de ese momento tengan como apariencia inicial el aspecto/género que hayamos asignado a cada uno de los avatares Female/Male/Neutral.

```
[ServiceList]
WifiServerConnector = "8003/Diva.Wifi.dll:WifiServerConnector"
AssetServiceConnector                                          =
"${Const|PrivatePort}/OpenSim.Server.Handlers.dll:AssetServiceConnecto
r"
InventoryInConnector                                           =
"${Const|PrivatePort}/OpenSim.Server.Handlers.dll:XInventoryInConnecto
r"
GridServiceConnector                                           =
"${Const|PrivatePort}/OpenSim.Server.Handlers.dll:GridServiceConnector
"
```

```
GridInfoServerInConnector                                              =
"${Const|PublicPort}/OpenSim.Server.Handlers.dll:GridInfoServerInConne
ctor"
AuthenticationServiceConnector                                         =
"${Const|PrivatePort}/OpenSim.Server.Handlers.dll:AuthenticationServic
eConnector"
OpenIdServerConnector                                                  =
"${Const|PublicPort}/OpenSim.Server.Handlers.dll:OpenIdServerConnector
"
AvatarServiceConnector                                                 =
"${Const|PrivatePort}/OpenSim.Server.Handlers.dll:AvatarServiceConnect
or"
LLLoginServiceInConnector                                              =
"${Const|PublicPort}/OpenSim.Server.Handlers.dll:LLLoginServiceInConne
ctor"
PresenceServiceConnector                                               =
"${Const|PrivatePort}/OpenSim.Server.Handlers.dll:PresenceServiceConne
ctor"
UserAccountServiceConnector                                            =
"${Const|PrivatePort}/OpenSim.Server.Handlers.dll:UserAccountServiceCo
nnector"
GridUserServiceConnector                                               =
"${Const|PrivatePort}/OpenSim.Server.Handlers.dll:GridUserServiceConne
ctor"
AgentPreferencesServiceConnector                                       =
"${Const|PrivatePort}/OpenSim.Server.Handlers.dll:AgentPreferencesServ
iceConnector"
FriendsServiceConnector                                                =
"${Const|PrivatePort}/OpenSim.Server.Handlers.dll:FriendsServiceConnec
tor"
MapAddServiceConnector                                                 =
"${Const|PrivatePort}/OpenSim.Server.Handlers.dll:MapAddServiceConnect
or"
MapGetServiceConnector                                                 =
"${Const|PublicPort}/OpenSim.Server.Handlers.dll:MapGetServiceConnecto
r"
OfflineIMServiceConnector                                              =
"${Const|PrivatePort}/OpenSim.Addons.OfflineIM.dll:OfflineIMServiceRob
ustConnector"
GroupsServiceConnector                                                 =
"${Const|PrivatePort}/OpenSim.Addons.Groups.dll:GroupsServiceRobustCon
nector"
UserProfilesServiceConnector                                           =
"${Const|PublicPort}/OpenSim.Server.Handlers.dll:UserProfilesConnector
"
```

Estos son los servicios que proveerá Robust a los simuladores conectados al Grid.

```
[DatabaseService]
StorageProvider = "Diva.Data.MySQL.dll"
ConnectionString                          =                          "Data
Source=http://XXX.XXX.XXX.XXX;Database=NOMBRE_DB_ROBUST;User        ID=
USUARIO_MYSQL;Password=PASSWORD-MYSQL;Old    Guids=true;Convert    Zero
Datetime=True;Allow Zero Datetime=True;"
```

El parámetro Storage Provider debe cambiarse a “Opensim.Data.MySQL,dll” si no se va a emplear la interfaz Web. El parámetro ConnectionString contiene los datos para conectarse a MySQL a la base de datos donde el proceso Robust almacenará y extraerá

datos para proporcionárselos a los simuladores. En nuestro caso la base de datos está alojada en la misma máquina, pero en caso de alojarse en otra máquina habría que escribir la IP correspondiente.

```
[LoginService]
WelcomeMessage = "Hola, Avatar!"
DSTZone = "local"
```

En esta sección las únicas líneas que debemos modificar son el mensaje de bienvenida cada vez que los usuarios se conectan, así como el huso horario para ajustarse al de nuestro servidor.

```
[Groups]
MaxAgentGroups = 50
```

Esta pequeña sección especifica el número máximo de grupos a los que puede unirse un avatar.

```
[GridInfoService]
login = ${Const|BaseURL}:${Const|PublicPort}/
gridname = "Nombre de nuestro Grid"
gridnick = "Nick de nuestro Grid"
```

Aquí se especifican los datos de login a nuestro Grid, con la URL y el puerto público.

El resto de líneas de este archivo Robust.ini se dejarán por defecto.

7.3 Interfaz Web “WIFI”

Descargamos la Distribución correspondiente a nuestra versión de Opensim, en este caso la 0.8.2.1: http://metaverseink.com/cgi-bin/link_counter.php?url=http://metaverseink.com/download/diva-r08210.zip.

Descomprimimos el archivo y copiamos el contenido de la carpeta “bin” del archivo que acabamos de descomprimir y lo copiamos dentro de la carpeta “bin” de nuestro Opensim. A continuación copiamos la carpeta “WifiPages” y la pegamos en la carpeta de Opensim al mismo nivel que la carpeta bin de nuestro Opensim. También debemos modificar el archivo Robust.ini lo cual está especificado en la sección 6.2.1.

Para modificar el idioma de la web, debemos copiar los archivos en la carpeta “es” dentro de la carpeta “WifiPages” que anteriormente copiamos junto al directorio bin de nuestro Opensim. Nos pedirá sobreescribir los archivos y así tendremos archivos traducidos.

7.4 Reglas FireWall y puertos

Para que nuestro mundo virtual funcione es necesario realizar una serie de configuraciones en el Firewall de las máquinas donde se han instalado Robust y el/los simulador/es. Deberemos comprobar que los puertos 80 y 8002 (privado) están abiertos en el servidor de Robust y que los puertos 80, 8003 y los puertos únicos asignados a cada región de un simulador están abiertos en cada simulador.

7.5 Bases de datos

Para poder hacer funcionar nuestro mundo virtual debemos crear las bases de datos manualmente en cada máquina. Para ello debemos crear en la máquina de Robust y en la máquina del simulador la base de datos cuyo nombre coincida con la que pusimos en el parámetro “ConnectionString” de cada archivo de configuración “Robust.ini” y “Opensim.ini”. Una vez ejecutemos Robust y el simulador se crearán las tablas necesarias dentro de las bases de datos.

7.6 Backups

Para la realización de backups de las bases de datos se ha procedido a configurar la ejecución diaria de una script de tipo Batch. Para ello se crearon en cada sistema una regla de ejecución del script diariamente. Para ello abrimos el panel de control -> Sistema y seguridad -> Tareas programadas. Creamos una nueva tarea que ejecutará el script “Backup.bat” a una hora determinada. A continuación tenemos el código del script:

```
:: ------------------ Date and Time Modifier -----------------------

@echo off

for /f "tokens=1-8 delims=.:/-, " %%i in ('echo exit^|cmd /q /k"prompt
$D $T"') do (
   for /f "tokens=2-4 delims=/-,() skip=1" %%a in ('echo.^|date') do (
set dow=%%i
set mm=%%j
set dd=%%k
set yy=%%l
set hh=%%m
set min=%%n
set sec=%%o
set hsec=%%p
)
)

if %hh%==0 set hh=00
if %hh%==1 set hh=01
if %hh%==2 set hh=02
if %hh%==3 set hh=03
if %hh%==4 set hh=04
if %hh%==5 set hh=05
if %hh%==6 set hh=06
if %hh%==7 set hh=07
if %hh%==8 set hh=08
if %hh%==9 set hh=09

set timeStamp=%yy%%mm%%dd% _%hh%-%min%-%sec%

:: --------- END TIME STAMP DIAGNOSTICS ----------------------
cd C:\wamp\bin\mysql\mysql5.6.17\bin
 mysqldump.exe     --all-databases     --result-file="D:\Data     Base
Backups\databases.%timeStamp%.sql"    --user="Usuario_Base_Datos"    --
password=Password_Base_Datos
```

8. Conclusiones y trabajo futuro

El objetivo principal de este proyecto ha sido la implantación de la plataforma Opensim para la creación de un mundo virtual orientado a la educación. Para lograr este objetivo, se llevó a cabo un estudio previo del Estado del Arte y las líneas de investigación relevantes para poder decidir qué plataforma es la más adecuada.

Durante la implantación de la plataforma Opensim se ha aprendido principalmente las fases en las que se divide un proyecto de sistemas, desde su implantación, pasando por el desarrollo, hasta el mantenimiento del mismo. Así mismo, los aspectos interdisciplinares asociados han añadido conocimientos adicionales y una motivación extra al proponer una línea de trabajo más transversal.

El resultado final del proyecto es enteramente satisfactorio, habiéndose conseguido la implantación del mundo virtual ajustado a las necesidades del proyecto, el cual a día de hoy ha sido empleado por más de 400 personas, y que a raíz de las conclusiones extraídas tras las pruebas de estrés, está capacitado para ser aún más extenso y poblado.

En cuanto a las líneas de trabajo adicionales futuras, se propone integrar mejoras nativas existentes tales como VoIP para aumentar la inmersividad o un sistema que permita la creación de parcelas privadas para cada avatar dentro de las regiones que pertenezcan a su clase. Además, se propone la migración al sistema operativo Linux que sirve cómo entorno de desarrollo nativo de módulos externos, llamados addins, los cuales añadan posibilidades a Opensim aún no existentes en las versiones actuales, cómo es el caso de la interfaz Web integrada en este proyecto.

Referencias

[1] Mikropoulos, T. A., & Natsis, A. (2011). Educational virtual environments: A ten-year review of empirical research (1999–2009). Computers & Education, 56(3), 769-780. http://alexcazeli.pbworks.com/w/file/fetch/48700146/Educational%20virtual%20environments%20-%20A%20ten-year%20review%20of%20empirical%20research%20(1999%E2%80%932009).pdf

[2] Mateu, J., Lasala, M. J., & Alamán, X. (2013). Education for Inclusion Using Virtual Worlds: An Experience Using OpenSim. K-12 Education: Concepts, Methodologies, Tools, and Applications: Concepts, Methodologies, Tools, and Applications, 111. http://www.irma-international.org/viewtitle/71871/

[3] Dickey, M. D. (2003). Teaching in 3D: Pedagogical affordances and constraints of 3D virtual worlds for synchronous distance learning. Distance education, 24(1), 105-121. http://www.peakwriting.com/nu/readings/dickey.pdf

[4] Dede, C. (1995). The evolution of constructivist learning environments: Immersion in distributed, virtual worlds. Educational technology, 35(5), 46-52. http://vcell.ndsu.nodak.edu/~ganesh/seminar/1995_Dede_The%20Evolution%20of%20Contructivist%20Learning%20Environments%20Immersion%20in%20Distributed,%20Virtual%20Worlds.pdf

[5] Ciaramitaro, B. (Ed.). (2010). Virtual Worlds and E-Commerce: Technologies and Applications for Building Customer Relationships: Technologies and Applications for Building Customer Relationships. IGI Global.

[6] Keenan, T. A. (1964). Computers and education. Communications of the ACM, 7(4), 205-209.

[7] Setzer, V. W., & Monke, L. (2001). Challenging the Applications: An alternative view on why, when and how computers should be used in education. education and technology: critical and reflective practices, 141-172.

[8] Vygotsky, L. S. (1980). Mind in society: The development of higher psychological processes. Harvard university press.

[9] Ghanbarzadeh, R., Ghapanchi, A. H., Blumenstein, M., & Talaei-Khoei, A. (2014). A decade of research on the use of three-dimensional virtual worlds in health care: a systematic literature review. Journal of medical Internet research, 16(2), e47.

https://www.researchgate.net/profile/Reza_Ghanbarzadeh2/publication/260253713_A_Decade_of_Research_on_the_Use_of_Three-Dimensional_Virtual_Worlds_in_Health_Care_A_Systematic_Literature_Review/links/0c9605362e0432b301000000.pdf

[10] Lackey, S., Salcedo, J., Matthews, G., & Maxwell, D. (2014). Virtual world room clearing: a study in training effectiveness. In Interservice/Industry Training, Simulation, and Education Conference. Orlando. https://www.researchgate.net/profile/Douglas_Maxwell4/publication/292984866_Virtual_World_Room_Clearing_A_Study_in_Training_Effectiveness/links/56b4000c08ae1f8aa45359ec.pdf

[11] Dawley, L., & Dede, C. (2014). Situated learning in virtual worlds and immersive simulations. In Handbook of research on educational communications and technology (pp. 723-734). Springer New York. https://pdfs.semanticscholar.org/4cb8/4034eff141cc02675b902b3e9a2de19d0218.pdf

[12] Stendal, K., & Balandin, S. (2015). Virtual worlds for people with autism spectrum disorder: a case study in Second Life. Disability and rehabilitation, 37(17), 1591-1598. http://s3.amazonaws.com/academia.edu.documents/41881766/Virtual_worlds_for_people_with_autism_sp20160202-20413-jbnxp3.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1496927383&Signature=ZnYH2UpTi1sFSOeaADU8%2FU2Bprw%3D&response-content-disposition=inline%3B%20filename%3DVirtual_worlds_for_people_with_autism_sp.pdf

[13] Deterding, S., Dixon, D., Khaled, R., & Nacke, L. (2011, September). From game design elements to gamefulness: defining gamification. In Proceedings of the 15th international academic MindTrek conference: Envisioning future media environments (pp. 9-15). ACM. http://s3.amazonaws.com/academia.edu.documents/30609294/MindTrek_Gamification_PrinterReady_110806_SDE_accepted_LEN_changes_1.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1496927544&Signature=iKOalfv1et2xoCvyRW3aoaBtses%3D&response-content-

disposition=inline%3B%20filename%3DFrom_game_design_elements_to_gamefulness.pdf

[14] Prensky, M., & Prensky, M. (2007). Digital game-based learning (Vol. 1). St. Paul, MN: Paragon house. http://www.savie.ca/SAGE/Articles/Prensky-Marc-2005%20.pdf

[15] Kiili, K. (2005). Digital game-based learning: Towards an experiential gaming model. The Internet and higher education, 8(1), 13-24. http://www.savie.ca/sage/articles/940_300027-kiili-2005.pdf

[16] Dickey, M. D. (2005). Three-dimensional virtual worlds and distance learning: two case studies of Active Worlds as a medium for distance education. British journal of educational technology, 36(3), 439-451. https://www.researchgate.net/profile/Michele_Dickey/publication/227495238_Three-dimensional_virtual_worlds_and_distance_learning_Two_case_studies_of_Active_Worlds_as_a_medium_for_distance_education/links/0c9605355968bd7337000000.pdf

[17] Pelgrum, W. J. (2001). Obstacles to the integration of ICT in education: results from a worldwide educational assessment. Computers & education, 37(2), 163-178. http://www.academia.edu/download/27368066/article5_pelgrum.pdf

[18] Tinio, V. L. (2003). ICT in Education. http://unpan1.un.org/intradoc/groups/public/documents/unpan/unpan037270.pdf

[19] Gerard, R. W. (1967). Shaping the mind: Computers in education. Applied Science and Technological Progress: a report to the Committee on Science and Astronautics, 207-228.

[20] Castellanos Monsalve, Y., & Salazar Velandia, J. V. (2016). El Videojuego Minecraft para potenciar el trabajo colaborativo en el aula de clase. http://reposital.cuaed.unam.mx:8080/jspui/bitstream/123456789/4825/1/VE16.698.pdf

[21] Dickey, M. D. (2011). The pragmatics of virtual worlds for K-12 educators: Investigating the affordances and constraints of ActiveWorlds and SecondLife with K-12 in-service teachers. Educational technology research and development, 59(1), 1-20.

[22] Moreira, M. A. (2010). El proceso de integración y uso pedagógico de las TIC en los centros educativos. Un estudio de casos1 The process of integration and the pedagogical use of ICT in schools. Revista de educación, 352, 77-97. https://www.researchgate.net/profile/Manuel_Area/publication/44204811_El_proceso_de_integracion_y_uso_pedagogico_de_las_TIC_en_los_centros_educativos_Un_estudio_de_casos/links/0c9605189251dafe4c000000.pdf

[23] Lasala, M. J., Alamán, X., & Gea, M. (2016). A Proposal for Using Virtual Worlds for the Integration. In Ubiquitous Computing and Ambient Intelligence: 10th International Conference, UCAmI 2016, San Bartolomé de Tirajana, Gran Canaria, Spain, November 29–December 2, 2016, Proceedings, Part I 10 (pp. 430-436). Springer International Publishing.

[24]] Lasala, M. J., Jiménez, J.M., Alamán, X., & Gea, M. (Accepted). Living in Virtual and Real Worlds: Didactic Experiences for Inclusive Education through Virtual Worlds. Sensors (Accepted).

Anexo A: Pliego de Condiciones

El proyecto presenta por un lado la implantación de un mundo virtual como plataforma orientada a la educación de centros de enseñanza obligatoria y centros de educación especial. Además, se ha desarrollado dentro del mundo virtual una batería de scripts para el Museo Virtual de la asociación “Asprogrades” en colaboración con el Colegio de Educación Especial "Santa Teresa de Jesús", en Granada.

Por ello, la memoria en un primer lugar plantea las investigaciones que existen sobre el uso de los mundos virtuales, en particular Opensim, como herramienta de apoyo a la enseñanza y plantea una nueva vía a las ya existentes.

A.1 Entregables

- Memoria del proyecto. Se entrega una versión original y dos copias de la misma, encuadernadas de forma normalizada. Cesión a la Universidad Autónoma de Madrid, de forma gratuita y no exclusiva, por el máximo plazo legal y con ámbito universal, para que pueda ser utilizada de forma libre y gratuita por todos los usuarios del repositorio, los derechos de reproducción, distribución y de comunicación pública, tal y como queda descrito en la Ley de Propiedad Intelectual.

- Código. Por un lado, el código necesario para poner en funcionamiento el mundo virtual y todas las funcionalidades añadidas. Por otro lado, el código de los scripts desarrollados para el Museo Virtual de la asociación Asprogrades.

A.2 Condiciones de desarrollo – Recursos hardware

- Equipos de desarrollo: Dos servidores X con procesador Intel® Xeon™ E5-2630@2.4 GHz (2 procesadores), 32GB de memoria RAM, dos discos duros de 1TB, un disco duro SSD de 256GB. Utilizado para la implantación del mundo virtual y su configuración y pruebas.

- Equipo de desarrollo de la documentación: Equipo con procesador lntel® Core™ 2 Quad Q8300@2.5GHz, 4GB de memoria RAM, disco duro de 500GB. Utilizado para el desarrollo de scripts para el museo [Asprogrades].

A.3 Condiciones de desarrollo – Recursos Software

- Sistema operativo Windows 8.1 Pro utilizado en los equipos donde se ha implantado el mundo virtual, así como en el equipo de desarrollo de scripts.

- Microsoft Office Word 2013 utilizado para la redacción de la documentación.

Anexo B: Presupuesto del proyecto

1) Ejecución Material

- Compra de Servidores (Software incluido)..... ..... 2.000 €
- Material de oficina..... 200 €
- Total de ejecución material..... 2.200 €

2) Gastos generales

- 16 % sobre Ejecución Material..... 352 €

3) Beneficio Industrial

- 6 % sobre Ejecución Material..... 132 €

4) Honorarios Proyecto

- 640 horas a 15 € / hora..... 9600 €

5) Material fungible

- Gastos de impresión..... 60 €
- Encuadernación..... 200 €

6) Subtotal del presupuesto

- Subtotal Presupuesto..... 12060 €

7) I.V.A. aplicable

- 21% Subtotal Presupuesto ..... 2532.6 €

8) Total presupuesto

- Total Presupuesto..... 14593,6 €

Madrid, Junio de 2017

El Ingeniero Jefe de Proyecto

Fdo.: Diego Aguilera Gréjon

Ingeniero de Telecomunicación