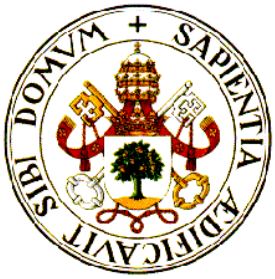

Universidad de Valladolid

E.T.S. Ingeniería Informática

TRABAJO FIN DE GRADO

Grado en Ingeniería informática

Despliegue automático de diseños de aprendizaje en mundos virtuales 3D realistas basados en tecnología Google

Autor:

Mario Marinero Domingo

Tutores:

Alejandra Martínez Monés

Juan Alberto Muñoz Cristóbal

Índice

Capítulo 1

Introducción y objetivos

1 Introducción y objetivos

En esta sección se hace una pequeña introducción a las áreas de estudio relacionadas con el TFG seguido de los objetivos propuestos.

1.1 Introducción

El aprendizaje mejorado por tecnología (en inglés *Technology-Enhanced Learning* - TEL) se está aplicando en múltiples plataformas como pueden ser: la Web, con herramientas como *Moodle*[1]; entornos de realidad aumentada que combinan información del mundo real y virtual o mundos virtuales 3D que se desarrollan en un ambiente completamente virtual. (Muñoz-Cristóbal, Prieto-Santos et al. 2012)
Los mundos virtuales 3D están constituidos por personas interconectadas a través de una red de computadoras. Las características básicas de estos mundos incluyen: la persistencia entre sesiones, la representación de los actores por medio de avatares, la comunicación de forma síncrona y un entorno 3D donde aparece el contenido y se representan los avatares. (Bell 2008)

Un mundo virtual 3D puede estar basado en un entorno ficticio o en una representación del mundo real. En este último caso existen tecnologías como Google Earth que no son un mundo virtual 3D debido a carecer de herramientas de comunicación y avatares. Sin embargo, sí que cumplen el resto de requisitos y permiten ser mejoradas por desarrolladores independientes, constituyen lo que llamaremos de aquí en adelante *mundos virtuales 3D realistas*.

Volviendo a las tecnologías aplicadas al aprendizaje, existe una plataforma llamada GLUE!-PS[2] (Prieto, Asensio-Pérez et al. 2011) creada por el Grupo de Sistema inteligentes y cooperativos de la Universidad de Valladolid [3]. Esta plataforma permite desplegar cursos especificados formalmente mediante herramientas de autoría de diseño de aprendizaje (e.g., WebCollage[4]) en una serie de entornos de aprendizaje entre los que destacan los entornos de aprendizaje virtual (VLE, de su traducción al inglés) (ej. Moodle, Sakai[5]…) pero que pueden ser también plataformas web como Wikis o herramientas de realidad aumentada.

Así, podrían usarse también mundos virtuales 3D realistas como entornos de aprendizaje, integrándolos en GLUE!-PS, de manera que diseños de aprendizaje formalizados con herramientas de autoría, puedan desplegarse automáticamente, además de en VLEs, plataformas Web y herramientas de realidad aumentada, en mundos virtuales 3D realistas, como Google Earth.

[1] https://moodle.org (último acceso: 01/09/2013)
[2] http://www.gsic.uva.es/glueps/(último acceso: 01/09/2013)
[3] http://www.gsic.uva.es (último acceso: 01/09/2013)
[4] http://pandora.tel.uva.es/wic2/ (último acceso: 01/09/2013)
[5] http://www.sakaiproject.org (último acceso: 01/09/2013)

1.2 Objetivos

Los objetivos aparecen desglosados en un objetivo principal de alto nivel que se descompone en varios subobjetivos.

1.2.1 Objetivo principal:

Desplegar cursos en mundos virtuales 3D realistas mediante un adaptador entre GLUE!-PS y una plataforma de mundos virtuales 3D realistas concreta.

1.2.2 Descomposición del objetivo principal:

1. Estudiar varias tecnologías de mundos virtuales 3D realistas y analizarlas en función de sus posibilidades adaptación a las tecnologías aplicadas al aprendizaje y su integración con GLUE!-PS.
2. Seleccionar una de las tecnologías seleccionadas e implementar el adaptador de GLUE!-PS que permita el despliegue de cursos.
3. Desarrollar herramientas que conformen la interfaz de usuario y complementen las propias del mundo virtual 3D realista seleccionado.

1.3 Metodología de desarrollo

Para el desarrollo de este TFG se ha optado por seguir el Proceso Unificado para la Educación (UPEDU del inglés *Unified Process for EDUcation*). UPEDU es una adaptación de *Rational Unified Proccess* (RUP). Debido a esto ambos procesos guardan similitudes importantes:

- Ambos están basados en el proceso en espiral de desarrollo del software.
- Tienen 4 fases: inicio, elaboración, construcción y transición.
- Los conceptos básicos como actividad, artefactos y gran parte de la terminología se mantienen

Las dos razones fundamentales para la selección de UPEDU son la familiaridad que tengo con el proceso y la simplificación respecto a RUP que lo convierte en más apropiado para realizar un desarrollo individual.

El desarrollo del proyecto se guiará por un plan de desarrollo de software que determinará el número de iteraciones que tiene cada fase así como los artefactos que se obtendrán al final de cada iteración y las fecha estimadas de comienzo y fin de cada fase.

UPEDU como RUP es un proceso guiado por riesgos, lo que significa que hay que mantener una lista de riesgos a lo largo del proyecto que deben ser controlados con el fin de evitar sus consecuencias negativas.

1.4 Estructura de la memoria

Capítulo 1 *"Introducción y objetivos"*: proporciona el marco en el que se desarrolla el TFG incluyendo como punto fundamental los objetivos del mismo.

Capítulo 2 *"Planificación"*: describe la planificación realizada para el desarrollo del TFG así como su evolución a lo largo del mismo.

Capítulo 3 *"Selección de tecnologías"*: En este capítulo se exploran las diferentes tecnologías disponibles para la creación de mundos virtuales 3D realistas.

Capítulo 4 *"Análisis"*: En este capítulo se desarrolla el análisis de las herramientas a desarrollar, análisis del sistema existente (GLUE!-PS requisitos, casos de uso y modelo de dominio

Capítulo 5 *"Diseño"*, En este capítulo se detalla el diseño de las herramientas de software a desarrollar: arquitectura de software, realización de casos de uso y diagrama de clases.

Capítulo 6 *"Implementación"*: En este capítulo se describe la implementación de la herramienta a partir del diseño desarrollado en el capítulo anterior.

Capítulo 7 *"Validación y pruebas"*: Relación de pruebas realizadas sobre la herramienta final y sus resultados.

El Capítulo 8 *"Conclusiones"*: Revisión de los resultados obtenidos en el TFG y posibilidades de ampliación en el futuro.

El Capítulo 9 *"Referencias bibliográficas"*: Relación de las fuentes bibliográficas consultadas.

Anexos
Anexo 1 *"Manuales":* Manuales de usuario e instalación de la herramienta desarrollada.

Capítulo 2

Planificación

2 Planificación

2.1 Planificación de las fases

En este apartado se procede a la división de TFG en fases.

Las fases dividen el proyecto en unidades amplías que serán planificadas con más detalles a medida que avance el proyecto.

En esta división se establecen estimaciones temporales en términos de horas/hombre (dado el carácter unipersonal del trabajo se puede resumir simplemente en horas) y se añade una calendarización del desarrollo del TFG.

FECHA DE COMIENZO DEL TFG: *17/05/13*

FECHA DE FIN DEL TFG: *15/08/13*

2.1.1 Descripción de los paquetes de trabajo

Investigación Inicial
Fecha de inicio: 17/05/13
Fecha de fin: 25/06/13
Este paquete se compone principalmente de una primera introducción en el campo de las TIC, mundos virtuales 3D y GLUE!-PS

Una primera fase incluye la lectura de la documentación y recursos básicos proporcionados por los tutores del TFG. A partir de esta documentación se obtiene una visión general del problema a tratar y se efectúa una profundización en los aspectos concretos que se van a abordar, especialmente de cara a la fijación de objetivos y adquisición de documentación sobre las tecnologías que se van a emplear.

En una segunda parte se utiliza esta información para el comienzo del análisis donde se fija finalmente el software que se va a desarrollar y se plasma la planificación de la siguiente iteración y una descripción técnica del trabajo a realizar.

Elaboración
Fecha de inicio: 26/06/13
Fecha de fin: 12/07/13
El paquete de elaboración comenzará una vez se haya finalizado la investigación inicial y este claro la planificación del TFG.

Este paquete contendrá la fase de análisis para las aplicaciones software que serán desarrolladas, esto incluye el adaptador para GLUE!-PS así como las aplicaciones e interfaces que se decidan construir alrededor de las aplicaciones de mapas seleccionadas

Este paquete también abarca el diseño hasta un 70% de los casos de uso de las aplicaciones software que van a ser desarrolladas.

Al final de este paquete deberá obtenerse un prototipo que permita demostrar la funcionalidad básica de GLUE!-PS a través del adaptador y un prototipo de las aplicaciones adicionales que se desarrollaran.

Construcción
Fecha de inicio: 12/07/13
Fecha de fin: 30/07/13
El paquete de construcción estará compuesto fundamentalmente por las actividades de implementación de las aplicaciones software diseñadas en la fase anterior y la finalización del diseño.

El objetivo es obtener una versión beta del sistema completo. Para ello debe completarse el adaptador para GLUE!-PS También es necesario desarrollar al menos un conjunto de herramientas que permitan el despliegue de un curso con la funcionalidad requerida.

El final de esta fase se centrará en las pruebas del sistema y la corrección de errores.

Transición
Fecha de inicio: 30/07/13
Fecha de fin: 15/08/13
En esta última fase se finalizará la documentación del sistema con los manuales de instalación y usuario.

Finalmente se realizará el despliegue del sistema que permitirá, tras someterlo a una última fase de pruebas, emplearlo en situaciones reales que podrán ser simuladas con cursos específicamente diseñados.

Con todas las demás actividades terminadas se finalizará la memoria del TFG y se preparará la presentación para la defensa.

Tabla de Hitos

Paquete de trabajo	Hitos	Estimación temporal	Fecha
Investigación Inicial	Introducción y análisis técnico.	45 h	26/06/13
Elaboración	Documento de diseño. Prototipo herramientas de mapas.	80 h	12/07/13
Construcción	Versión beta del sistema completo.	90h	30/07/13
Transición	Versión final de sistema. Memoria TFG.	75h	15/08/13

Tabla 1 Hitos del proyecto

Paquete de trabajo	Artefactos
Investigación Inicial	Introducción. Planificación del proyecto.
Elaboración	Documento de análisis. Documento de diseño. Prototipo adaptador GLUE!-PS Prototipo herramientas para mundos virtuales 3D Plan de construcción.
Construcción	Herramientas aplicación de mapas. Adaptador GLUE!-PS Manual de usuario. Sistema Beta. Plan de transición.
Transición	Manual de instalación. Sistema final. Resultados pruebas y simulación. Memoria del TFG.

Tabla 2 Artefactos del sistema

Diagrama de fases

En este de fases se pueden apreciar las estimaciones de inicio y final de cada fase. La fase inicial aparece partida al producirse una parada del trabajo durante varios días.

	Nombre de tarea	Duration	Start	Finish	Predece
1	Investigación inicial	15.5 days	Fri 17/05/13	Tue 25/06/13	
2	Introducción e análisis técnico	0 days	Tue 25/06/13	Tue 25/06/13	1
3	Elaboración	12 days	Wed 26/06/13	Thu 11/07/13	1
4	Documento de diseño	0 days	Thu 11/07/13	Thu 11/07/13	3
5	Construcción	13 days	Fri 12/07/13	Tue 30/07/13	4
6	Versión beta	0 days	Tue 30/07/13	Tue 30/07/13	5
7	Transición	12 days	Wed 31/07/13	Thu 15/08/13	6
8	Versión final	0 days	Thu 15/08/13	Thu 15/08/13	7

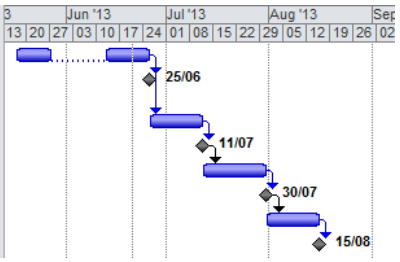

Ilustración 1 Diagrama de fases

2.2 Gestión de riesgos

A continuación se listan los riesgos identificados durante la planificación del desarrollo de software. Se han seleccionado y ordenado los riesgos de acuerdo a la probabilidad de que incidan en el proyecto y la gravedad de la incidencia. Se ha tratado de mantener una lista de tamaño razonable que permita priorizar los riesgos identificados sin consumir una cantidad excesiva de recursos en gestionar riesgos de menor entidad.

2.2.1 Medidas

Magnitud: Del 1 al 10, siendo el 1 el menor riesgo y 10 el mayor. La magnitud se mide en función de la probabilidad y el impacto del riesgo sobre el proyecto.

Descripción: Breve descripción del riesgo.

Impacto:

- *Crítico:* afecta a la funcionalidad básica del proyecto y la duración.
- *Alto:* Afecta a los stakeholders y funcionalidad importante.
- *Medio:* Dependiendo de las circunstancias el riesgo tendrá un plan de mitigación asociado o se asumirá su impacto en el proyecto.
- *Bajo:* Se evitaran rápidamente si es posible o se asumirán como una contingencia en caso contrario.

Indicador: Métrica a seguir para detectar y seguir el impacto del riesgo.

Mitigación/contingencia: Plan para mitigar el impacto del riesgo o evitarlo.

2.2.2 Listado de riesgos

2.2.2.1 Disponibilidad temporal

Magnitud: 8

Descripción: Durante el desarrollo del proyecto puede haber variaciones en la disponibilidad temporal del alumno y de los tutores y esto afectar a las reuniones y entregas.

Impacto: Alto

Indicador: Tiempo dedicado al proyecto y fidelidad con la que se sigue la planificación

Mitigación/contingencia: Planificación teniendo en cuenta los posibles imprevistos y reservando tiempo antes de la fecha de entrega.

2.2.2.2 Requisitos no identificados

Magnitud: 8

Descripción: Detectar un requisito a implementar en una fase avanzada del proyecto, esto podría impedir el avance del proyecto e incluso rehacer partes del mismo.

Impacto: Alto, más allá de la fase de elaboración crítico.

Indicador: Ambigüedades e incertidumbre a la hora de tomar decisiones durante la elaboración y construcción del proyecto.

Mitigación/contingencia: Comunicación fluida con los tutores

2.2.2.3 Desconocimiento de la tecnología empleada

Magnitud: 7

Descripción: Parte de la tecnología empleada en este proyecto tanto de GLUE!-PS como de Google no había sido empleada previamente por el autor.

Impacto: Medio

Indicador: Problemas para avanzar en la implementación e incertidumbre sobre las posibilidades para resolver incidencias.

Mitigación/contingencia: Comunicación con los tutores en lo que toca a GLUE!-PS y búsqueda de documentación apropiada para el resto de tecnologías.

2.2.2.4 Variabilidad de la tecnología empleada

Magnitud: 6

Descripción: Las tecnologías de mapas empleadas están mejorando continuamente y por tanto cambian rápidamente, esto puede afectar si se dan cambios durante el desarrollo del proyecto y si la documentación no es actualizada correctamente.

Impacto: Alto

Indicador: Aparición de nuevas versiones de las APIs utilizadas y las modificaciones a las mismas que aparecen en los listados de cambios publicados.

Mitigación/contingencia: Revisión de la documentación periódicamente para detectar los cambios antes de que sean activados.

2.2.2.5 Integración en otro proyecto desarrollado activamente

Magnitud: 5

Descripción: GLUE!-PS es una plataforma que está siendo desarrollada, ampliada y mejorada en varios aspectos, aunque el desarrollo del proyecto se lleva a cabo sobre una versión hay que tener en cuenta que partes de la plataforma han sido modificadas recientemente y lo serán en el futuro por lo que se tratarán de minimizar las dependencias e incompatibilidades.

Impacto: Medio

Indicador: Problemas y bugs detectados durante la implementación en GLUE!-PS

Mitigación/contingencia: Notificar los posibles problemas y usar las interfaces de la plataforma apropiadamente y sin dependencias en detalles de implementación. Otra posibilidad es utilizar una versión razonablemente estable de la plataforma y no integrar los cambios que se vayan produciendo en GLUE!-PS

2.2.2.6 Seguimiento desordenado del plan de desarrollo de software

Magnitud: 4

Descripción: Un defecto común en otros desarrollos realizados es pasar a la fase de construcción del proyecto obviando partes de las tareas de análisis y diseño que son completadas a posteriori. Esto impacta negativamente en la calidad del proyecto y hace aumentar la magnitud de varios de los riesgos anteriores.

Impacto: Medio

Indicador: Desajustes en el seguimiento del plan de proyecto, implementación finalizada para partes del proyecto sin la fase de diseño completa.

Mitigación/contingencia: Seguir escrupulosamente el plan de desarrollo y realizar la implementación y diseño en las iteraciones apropiadas.

Capítulo 3

Selección de tecnologías de mundos virtuales 3D realistas

3 Selección de tecnologías de mundos virtuales 3D realistas

En esta sección se analizan 4 tecnologías basadas en mapas y modelos 3D que permiten ser navegadas de forma interactiva y por tanto proporcionan algunos de los elementos básicos para construir un mundo virtual 3D y particularmente se adapten a las necesidades de nuestra aplicación.

3.1 Criterios de selección

Los criterios de selección para las tecnologías que se han seleccionado son los siguientes.

Funcionalidad esencial: La tecnología seleccionada debe soportar: Elementos visuales básicos (líneas, iconos…) y contenido textual. Es esencial la posibilidad de posicionar estos elementos con libertad sobre el entorno 3D.

Programabilidad: Este criterio evalúa si se ha proporcionado una API para poder crear añadir elementos dinámicos a la aplicación así como personalizar la experiencia de usuario más allá de las funcionalidades soportadas por la tecnología por defecto y que por tanto sea susceptible de integrarse con GLUE!-PS.

Documentación: La correcta documentación de la tecnología y especialmente de su API. Es importante que se incluyan ejemplos del funcionamiento y posibilidades de la tecnología así como una referencia detallada de cada función disponible.

Estabilidad y mantenimiento de la tecnología: Se valora el tiempo que ha estado disponible la herramienta, las revisiones que se han hecho de la misma y las garantías ofrecidas de que el servicio seguirá funcionando en el futuro.

Número de plataformas soportadas: Para ser accesible a un número mayor de usuarios así como evitar la necesidad de un equipamiento específico durante el desarrollo se valora las plataformas en las que cada tecnología puede ser utilizada. Contar con una versión web es especialmente deseable al permitir su utilización desde cualquier navegador compatible.

3.2 Tecnologías seleccionadas

Google Earth[6]: Es un visor de mapas en 3D. Nació como una aplicación de escritorio debido a sus requisitos gráficos, existe un *plug-in* para navegadores que permite manejar su funcionamiento

[6] https://developers.google.com/earth/ (último acceso: 01/09/2013)

mediante Javascript. Google Earth usa imágenes por satélite para generar un modelo del globo terráqueo. Google Earth se centra en la experiencia visual y la exploración en lugar de la gestión de rutas o la búsqueda de localizaciones. Es posible introducir dentro de la aplicación modelos 3D de edificios y monumentos y de hecho existen un gran número de esos modelos creados por los usuarios de la aplicación. Recientemente se ha añadido modelos 3D generados mediante técnicas automatizadas a partir de imágenes aéreas de una lista limitada de ciudades[7]. Google Earth también cuenta con una API pública así como el concepto de capas que permite añadir y superponer información en la aplicación. Para introducir está información se puede emplear el formato estándar KML.

Existen ejemplos de aplicaciones 3D dinámicas que permiten manipular con libertad tanto la cámara como los modelos cargados en el mapa.

Google Street View[8]: Este servicio nació posteriormente a Google Earth y se integra en tanto dentro de Google Earth como de Google Maps como un modo adicional que cuenta con sus controles específicos. Aunque forme parte de otras herramientas Google Street View permite recorrer localizaciones próximas desde su propia interfaz y podría utilizarse como un entorno para el despliegue de cursos diferenciado. Google Street View se compone de imágenes en 360 grados tomadas desde vehículos especialmente equipados. Por su capacidad de inmersión y representación de localizaciones con imágenes de gran resolución, desde el punto de vista de una persona, Google Street View ofrece una posibilidad única de ser integrada en un entorno educativo virtual.

Apple Maps[9]: En junio de 2012 Apple lanzo una aplicación de mapas para sus dispositivos móviles. Hasta ese momento la aplicación de mapas para IOS, el sistema operativo móvil de Apple, utilizaba los datos de Google Maps. La función principal de la aplicación es la búsqueda de localizaciones y rutas

Entre las características de la aplicación se encuentra una vista 3D y la posibilidad de crear aplicaciones que incrusten una vista de los mapas. Además ofrece una API para modificar los mapas incrustados en la aplicación y añadir elementos visuales lo que permite su adaptación a las necesidades específicas del desarrollador.

Nokia Here[10]: La aplicación de mapas de Nokia ha cambiado de nombre varias veces, siendo anteriormente Ovi Maps y Nokia Maps. Hasta 2011 la solución de Nokia estaba orientada a proporcionar servicios de localización y búsqueda para sus teléfonos inteligentes. En 2011 sin embargo lanzó una versión beta con la representación 3D de varias ciudades y a partir de ese momento han introducido una vista similar a Google Street View y ha implementado todo ello con tecnologías web estándar.

Nokia Here también está disponible para plataformas móviles y sus recursos son utilizados por otras plataformas de mapas como Bing Maps y Yahoo! Maps.

[7] http://support.google.com/earth/bin/answer.py?hl=en&answer=2661942 (último acceso: 01/09/2013)
[8] https://developers.google.com/maps/documentation/streetview/ (último acceso: 01/09/2013)
[9] http://www.apple.com/ios/maps/(último acceso: 01/09/2013)
[10] http://developer.here.com/(último acceso: 01/09/2013)

Como en los casos anteriores se ofrece una API para desarrolladores que permiten al desarrollador sobreimpresionar información y soporta el formato KML, la vista 3D sin embargo no cuenta con documentación específica y los ejemplos se limitan a utilidades 2D.

3.3 Tabla de resultados

Aplicación	Funcionalidad esencial	Programabilidad	Documentación	Estabilidad y mantenimiento	plataformas soportadas
Google Earth	Completa. Soporte para modelos 3D personalizados.	Completa	Completa. Versionada Ejemplos avanzados en 3D	12 años de antigüedad. Última versión: 7.1.1.1888 (Julio 12, 2013) Actualizado a la versión 7.x en todas las plataformas	Windows, OSX, Linux, Android, IOS, Web mediante plug-in
Google Street View	Parcial: La libertad de movimientos en el entorno 3D está limitada por los trazados fotografiados	Completa	Completa. Integrada en la documentación de Google Maps. Ejemplos básicos.	6 años de antigüedad. Última versión: Julio 9, 2013 Revisiones frecuentes del contenido con nuevas imágenes	Web, versiones de escritorio de Google Earth. Mediante Google Maps en Android e IOS.
Apple Maps	Parcial: La vista 3D no soporta libertad de movimientos, ni elementos visuales personalizados.	Parcial: La vista 3D no es programable	Completa. Guiada por ejemplos	1 año de antigüedad. Última versión: Mayo 2, 2013 Actualizaciones frecuentes y corrección de errores.	IOS, anunciado para OSX.
Nokia Here	Completa.	Completa: Para mapas 3D Web en fase beta. Difiere ligeramente entre plataformas.	Completa: Ejemplos avanzados e interactivos sin 3D	6 años de antigüedad. Última versión: Se actualizan partes de la API independientemente. La funcionalidad 3D en versión Web sigue en beta.	Windows Phone, Android, IOS, otras plataformas móviles y Web

Tabla 3 Resultados comparativa de herramientas de geoposicionamiento

3.4 Conclusiones de la selección

Atendiendo a la tabla presentada los mejores resultados en todos los apartados evaluados los presenta Google Earth. Cuenta con toda la funcionalidad esencial y tanto la interfaz de programación como la documentación son de gran calidad.

Nokia Here proporciona también una plataforma muy completa pero con la desventaja de que la funcionalidad 3D que tienen un gran peso para nuestros objetivos sigue estando en fase beta. En un futuro y si cumple los pronósticos de la propia Nokia, Here podría convertirse en una solución equiparable a la de Google.

Google Street View obtiene unos buenos resultados que se ven limitados por la naturaleza de la aplicación que al estar basada en imágenes obtenidas durante trayectorias fijas no ofrecen un entorno 3D completo.

Apple Maps obtiene los peores resultados debido fundamentalmente a la falta de soporte en más plataformas y a limitaciones respecto a lo que es posible realizar con la vista 3D lo cual implica una pobre adaptación a nuestro caso de uso.

En vista de estos resultados en este TFG se va a optar por emplear Google Earth como plataforma de desarrollo del adaptador GLUE!-PS aunque Google Street View también dispondrá de un adaptador y una funcionalidad al menos parcial debido a que existen similitudes en la API que simplifican el trabajo a realizar.

Capítulo 4

Análisis

4 Análisis

4.1 Sistemas existentes

Existen dos sistemas principales sobre los que se desarrollara este proyecto: la plataforma GLUE!-PS y Google Earth.

La plataforma GLUE!-PS que permite el despliegue de diseños educativos en varios entornos de aprendizaje y proporciona el servidor y las interfaces de usuario en las que el profesor puede importar un diseño educativo generado con una herramienta de autoría y modificar algunos parámetros del diseño. Este proyecto utilizará las interfaces ya disponibles en GLUE!-PS de forma que no haya que desarrollar una aplicación de diseño educativo ni una interfaz de usuario en esa parte del sistema (Prieto, Asensio-Pérez et al. 2011).

Google Earth es el sistema con el que los estudiantes más entran en contacto. El globo terráqueo 3D navegable al que se puede acceder públicamente a través de la aplicación de escritorio es la interfaz principal de la aplicación pero una parte importante del sistema consiste en modificar programáticamente esta interfaz para proporcionar el contenido añadido por el profesor y dotar a la aplicación de los elementos requeridos para que los estudiantes puedan navegar de forma sencilla y acceder a los recursos.

El resto de la sección se dedica a explicar en más profundidad los sistema GLUE!-PS y Google Earth.

4.1.1 GLUE!-PS

La plataforma GLUE!-PS (*Group Learning Unifed Environment - Pedagogical Scripting* – Entorno unificado de aprendizaje en grupo – Guiado pedagógico) es una solución al problema de desplegar diseños educativos en entornos de aprendizaje. En vez de desarrollar una herramienta de diseño y un entorno de aprendizaje GLUE!-PS proporciona adaptadores para una gran variedad de herramientas ya disponibles como WebCollage, Moodle, navegadores de realidad aumentada... Esto permite que la barrera de entrada para los distintos usuarios sea mucho menor al poder utilizar sistemas ampliamente usados.

La arquitectura de GLUE!-PS está orientada a servicios y sigue a alto nivel el patrón adaptador. Se puede ver un esquema simple de esta arquitectura en la ilustración 2. A grandes rasgos hay dos grupos de adaptadores, por un lado los enfocados a herramientas de diseño de aprendizaje y por otro los enfocados a los entornos de aprendizaje. En este proyecto nos centraremos fundamentalmente en los segundos.

Entre los adaptadores para entornos de aprendizaje existe un subgrupo de especial interés que son los adaptadores para aplicaciones de realidad aumentada. Estos adaptadores ya tienen implementado geoposicionamiento y aunque el entorno es notablemente diferente al de Google Earth la funcionalidad e interfaces empleadas son muy similares.

Existen adaptadores de realidad aumentada para Layar, Junaio y Mixare. De los 3 el más simple es el adaptador para Mixare al carecer de funcionalidades más complejas que no son de interés fuera de la realidad aumentada. Por tanto el adaptador de Mixare será la referencia a partir de la cual desarrollar el adaptador para Google Earth.

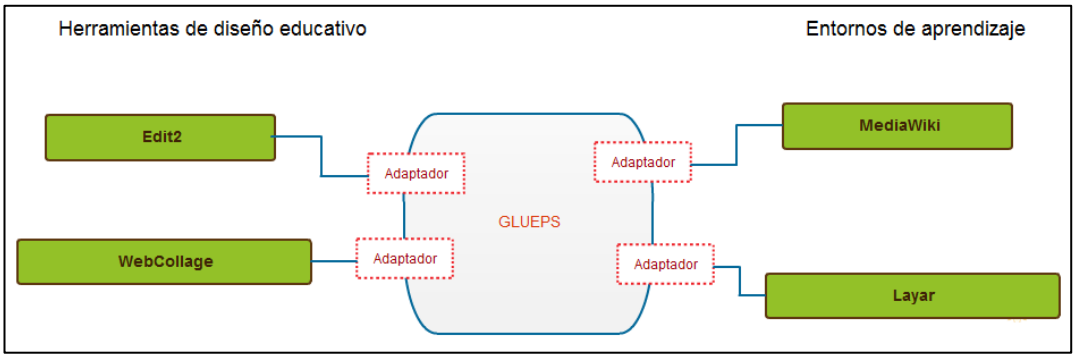

Ilustración 2 Diagrama GLUE!-PS

Un concepto fundamental de GLUE!-PS es el modelo de datos. Este modelo es el que hace posible que una gran variedad de herramientas se integren en una sola plataforma y compartan una gran parte de la funcionalidad. El modelo de datos de GLUE!-PS define una serie de conceptos del mundo del diseño educativo y de los entornos de aprendizaje que engloban las características más importantes de ambos.

4.1.2 Tecnologías Google

4.1.2.1 Google Maps API

Google Maps API es la interfaz de programación más importante y estable de las ofrecidas por Google. Esto se debe al gran número de clientes que usan la interfaz. La versión actual de la API es la 3.x

La API contiene cuatro secciones de especial interés a la hora de construir una aplicación sobre ella: eventos, controles, capas y *overlays.*

Eventos: Permite registrar funciones que se dispararan cuando el usuario realice determinadas acciones sobre el mapa. Los eventos son también un sistema de comunicación de sucesos en la aplicación y proporcionan una abstracción que permite añadir funcionalidad a la aplicación sin necesidad de modificar las funciones internas de la librería.

Controles: Especialmente la parte referida a controles personalizados, modificando los controles por defecto y añadiendo controles propios se puede construir una interfaz de usuario rica que permita realizar acciones no contempladas en la API. Para crear estos controles personalizados se puede utilizar CSS y HTML por lo que existe una gran flexibilidad y se manejan mediante eventos como los mencionados arriba.

Capas: El concepto de capas se asocia aquí a la inclusión de dos tecnologías principalmente sobre los mapas. Por un lado KML que se analizará junto a Google Earth al ser un formato de

información geográfica especialmente ligado a esa aplicación. Y por otro lado Fusion Tables que es una tecnología experimental de tablas. La posibilidad de usar KML tanto en Google Maps como en Google Earth presenta una ventaja de cara a la posibilidad de extender la aplicación y de utilizar una tecnología con amplio soporte, sin embargo, en este caso hay que analizar la implementación concreta para evitar posibles limitaciones al integrar esta tecnología.

Overlays: Esta parte de la API se refiere a la inclusión de gran variedad de elementos gráficos en los mapas. Los *overlays* van asociados a una posición en el mapa y pueden ser formas geométricas, marcadores, iconos, cuadros de texto… También soportan una cantidad limitada de animaciones. Estas herramientas son útiles a la hora de construir elementos de la interfaz y sobre todo para proporcionar información textual o gráfica al usuario.

4.1.2.2 Google Street View

Street View ofrece imágenes en 360 grados a lo largo de cualquier ruta transitable de una ciudad. El resultado es un 'mundo' navegable por las calles que han sido recorridas y fotografiadas y en las que en cada punto donde se ha realizado una fotografía se puede contemplar la escena desde todos los angulos.

Para este trabajo sería necesario situar elementos y controles personalizados sobre la interfaz por defecto de la aplicación. Para conseguir esto se deberán emplear un subconjunto de lo descrito para Google Maps ya que Street View soporta menos funcionalidades.

Para ofrecer una interfaz más rica es posible utilizar contenedores HTML independientes para Street View de forma que se visualice de forma independiente el mapa y Street View e incluso que se mantenga sincronizada la posición del usuario.

Street View también soporta *overlays* pero están limitados a marcadores, iconos y ventanas de información por lo que no se pueden emplear polígonos y otras herramientas de dibujo, sin embargo, a partir de los iconos y sobre todo las ventanas de información es posible utilizar imágenes y HTML que proporcionen tanto información como controles.

Ilustración 3 Icono en Street View

Otro servicio proporcionado por la API posibilita acceder a las imágenes correspondientes a una localización de forma programática y en ficheros estándar, sin estar incrustadas en la interfaz. Esto permitiría construir una interfaz de usuario alternativa con manipulando los ficheros de imagen

obtenidos pero tiene el inconveniente de perder los controles de desplazamiento del mapa, una solución mixta consistiría en usar la interfaz de Street View en el uso normal de la aplicación pero pasar al servicio de imágenes cuando se deba presentar información de forma más flexible o se requieran controles que no puedan ser incrustados en la vista normal.

Finalmente existe la posibilidad de usar una fuente de imágenes propia lo que posibilita crear entornos que no son accesibles en Street View pero que es pueden ser de interés al desarrollar un curso. En todo caso, la posibilidad de usar panorámicas personalizadas conlleva un trabajo adicional para obtener imágenes e integrarlas en la interfaz no contemplado en este trabajo.

4.1.2.3 Google Earth

La API de Google Earth se compone de varias interfaces y funciones expuestas en javascript mientras que Google Earth se ejecuta en su propio *plug-in* de modo similar a como lo hace Adobe Flash Player. La API de Google Earth está muy ligada al formato KML existiendo la posibilidad de usar ficheros XML o usar funciones Javascript con el mismo nombre y funcionalidad que las instrucciones descritas para el formato KML.

4.1.2.3.1 KML

Se trata de un lenguaje de marcado basado en XML y orientado a la información geográfica. El lenguaje permite asociar a posiciones geográficas diversos elementos. El formato es un estándar internacional desde 2008[11]

Para editar ficheros KML se puede usar directamente la aplicación de escritorio de Google Earth, pero algunas de las funciones deben ser modificadas con un editor de texto sobre el fichero XML.

Al igual que con el API de mapas, en un fichero KML se pueden especificar *overlays* e incrustar de esta forma imágenes, polígonos, trayectorias… Es posible incluir overlays asociados a una posición de la pantalla en vez de a un punto geográfico.

Finalmente una característica fundamental de KML es el soporte de modelos 3D que son mostrados sobre el mapa y se han usado principalmente para la reconstrucción digital de ciudades a partir de modelos de sus edificios. Para crear estos modelos se usa el formato COLLADA, también estándar. Esto permite introducir modelos arbitrarios que pueden representar elementos fijos o animados dentro del mapa. Evidentemente el grado de integración en el mapa que se logra es muy superior al obtenido mostrando cuadros de texto o limitándonos a imágenes planas.

La API de Google Earth también soporta eventos con una sintaxis idéntica a la del API de Google Maps pero con una funcionalidad que difiere al tratarse de dos aplicaciones distintas. En este caso además esta interfaz es esencial ya que al tratarse de un *plug-in* no existe la posibilidad de analizar y modificar el comportamiento de la aplicación directamente como en el caso de una aplicación puramente en JavaScript/HTML, la funcionalidad está estrictamente limitada por la API. De todos

[11] http://www.opengeospatial.org/pressroom/pressreleases/857 (último acceso: 01/09/2013)

modos y como se puede ver en distintas demostraciones[12] esto no limita una interacción rica entre el código JavaScript y el propio *plug-in* siendo posible implementar controles, animaciones…

Un elemento que cobra importancia en esta API es el control de la cámara. Al tratarse de un entorno completamente 3D la libertad de movimientos es superior a la vista para Google Maps y Street View.

Google proporciona una serie de demostraciones de las funciones del API incluyendo aplicaciones finalizadas que la usan extensivamente.

4.2 Sistema propuesto

En esta sección nos centramos ya en el sistema que se va a desarrollar y se describe el proceso de análisis del mismo. Para una mejor puesta en situación, especialmente en lo referente a GLUE!-PS, se ha incluido un escenario que describe el uso habitual de la plataforma.

4.2.1 Especificación de Requisitos

4.2.1.1 Requisitos funcionales

1. El sistema consistirá en un servicio web siguiendo el patrón del resto de adaptadores de GLUE!-PS.
2. El sistema deberá mostrar los puntos de interés geolocalizados introducidos en los diseños educativos desplegados.
3. El sistema deberá mostrar los puntos en la posición correcta de acuerdo a las coordenadas de latitud y longitud
4. El sistema deberá mostrar la información adicional que acompañe a los puntos de interés en forma de recursos como enlaces, documentos o modelos 3D.
5. El sistema deberá renderizar los modelos 3D siempre que la tecnología de visualización empleada en ese momento lo soporte.
6. El sistema deberá contar con un sistema de acceso de usuarios que permita identificarlos y ligarlos a un despliegue concreto.
7. El sistema deberá permitir la navegación entre los puntos de interés utilizando para ello las herramientas de navegación disponibles en el plug-in de Google Earth.
8. El sistema deberá monitorizar las acciones de los usuarios utilizando para ello las interfaces apropiadas de GLUE!-PS
9. El sistema deberá permitir la recuperación de puntos de interés de forma dinámica en función de la posición del usuario sobre el mapa.
10. El sistema deberá proporcionar una interfaz donde se incrustará el plug-in de Google Earth y que ofrezca información complementaria, enlaces de interés y acceso a la ayuda.
11. El sistema empleará siempre que sea posible las estructuras de datos ya presentes en GLUE!-PS.

[12] https://developers.google.com/earth/documentation/demogallery (último acceso: 01/09/2013)

4.2.1.2 Requisitos no funcionales

4.2.1.2.1 Usabilidad

1. El acceso a la aplicación se realizará mediante un navegador web compatible con HTML, CSS, Javascript y el plug-in de Google Earth.
2. La navegación por los mapas será la proporcionada por Google Earth.

4.2.1.2.2 Confiabilidad

3. La aplicación almacenará toda la información en la base de datos siendo este el único elemento necesario para poder recuperar el estado de la aplicación en caso de fallo.
4. Los usuarios tendrán permisos de acceso a la aplicación de acuerdo a su rol e identificación y no será posible que hagan uso de funciones reservadas a otros usuarios.

4.2.1.2.3 Rendimiento

5. La aplicación se dimensionará para un número de usuarios del orden de las decenas de usuarios simultáneos.
6. Los requisitos de almacenamiento para los despliegues almacenados en la base de datos serán del orden de los MB para el sistema básico y aumentaran linealmente con el número de despliegues.

4.2.1.2.4 Mantenibilidad

7. El sistema se desarrollará con el lenguaje de programación Java en el caso del servidor (GLUE!-PS usa Java) y Javascript en el cliente.
8. El servicio web será de tipo REST al igual que el resto de adaptadores de GLUE!-PS
9. El código fuente de la aplicación se mantendrá en un repositorio de Subversion.
10. El utilizará software de libre acceso siempre que sea posible y se evitará software que pueda afectar a las condiciones de distribución de GLUE!-PS
11. Se evitará siempre que sea posible la modificación de las estructuras de datos presentes en GLUE!-PS con el fin de evitar introducir dependencias innecesarias y la modificación del resto de adaptadores.

4.2.1.2.5 Documentación:

12. Se deberá generar un manual de usuario.
13. Se deberá generar un manual de instalación.
14. Se deberá realizar un análisis y seguimiento de riesgos

4.2.1.2.6 Entrega:

15. El sistema y su documentación deberá estar completado antes del día 6 de Septiembre de 2013

4.2.2 Escenario Inicial

Para mejorar la compresión de la plataforma GLUE!-PS en su conjunto se presenta un supuesto en el que se describirá el uso tanto de los sistemas existentes como de la aplicación a desarrollar.

Un profesor de la Universidad de Valladolid quiere realizar un curso con un grupo de alumnos. Para ello va a utilizar una herramienta de diseño educativo (WebCollage) y posteriormente usar algunos de los entornos educativos disponibles en GLUE!-PS para desplegar el curso. Todas las herramientas empleadas son aplicaciones web.

Para comenzar accede a WebCollage que es una herramienta de autoría y que le permite definir elementos del curso como los participantes, actividades que se van a realizar, organización en grupos…

En este caso va a realizar un sencillo diseño con 10 alumnos que realizan dos actividades de forma grupal. La interfaz de WebCollage es muy visual resultando un diagrama de lo que se va a realizar.

Una vez está satisfecho con el resultado accede a GLUE!-PS donde puede cargar el diseño educativo realizado. GLUE!-PS cuenta con una interfaz menos visual y está orientado a determinar cómo se desplegará el curso además de permitir modificaciones simples del diseño realizado. En este caso el profesor va a añadir un documento de Google docs y un modelo 3D, ambos geolocalizados. Para introducir estos elementos la interfaz proporciona diálogos para cargar el modelo 3D y un mapa donde se puede seleccionar la posición exacta donde se visualizará el elemento.

Una vez ha completado la configuración del despliegue lo confirma y GLUE!-PS se encarga de generar los recursos seleccionados.

En este punto GLUE!-PS soporta navegadores de realidad aumentada que permitirían acudir a los puntos geolocalizados por el profesor pero por razones geográficas, meteorológicas o de tiempo sería deseable que los alumnos pudiesen realizar las actividades desde el aula sin perder del todo el componente geográfico y la idea de desplazarse entre los distintos puntos de interés.

Por tanto gracias al sistema que se desarrollará en este proyecto esto será posible. Una vez uno de los usuarios accede al sistema y se identifica puede navegar por el mapa tridimensional y un icono le indica donde están situados los puntos de interés. Al hacer click sobre los mismos puede acceder al documento de Google Docs en un caso y visualizar el modelo 3D integrado dentro del propio mapa.

4.2.3 Modelo de casos de uso

Los siguientes casos de uso modelan el comportamiento del sistema en los procesos más importantes que deben realizar los usuarios.

4.2.3.1 Descripción de los actores

4.2.3.1.1 Profesor

El profesor está familiarizado con el resto del sistema GLUE!-PS y la actividad que se va llevar a cabo en el sistema. Su función es la supervisión de los cursos que están siendo completados por los alumnos. Además puede guiar y orientar a los alumnos a la hora de completar las distintas actividades previstas.

4.2.3.1.2 Alumno

El alumno utiliza el sistema para explorar los contenidos de la actividad propuesta por el profesor. Su objetivo es navegar por el sistema descubriendo y utilizando los distintos contenidos geolocalizados.

4.2.3.2 Diagrama de casos de uso

El diagrama de casos de uso ejemplifica los actores presentes en el sistema y los comportamientos que pueden llevar a cabo

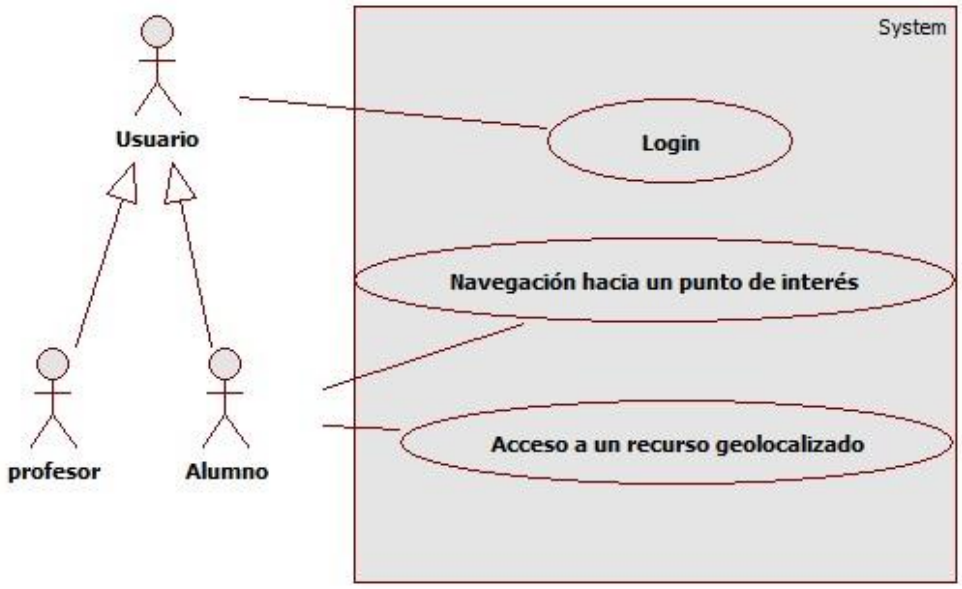

Ilustración 4 Diagrama de casos de uso

4.2.3.3 Caso de uso 1: Login

1. **Descripción**
   El caso de uso describe como el profesor o el alumno accede al sistema y se identifica en el mismo.

2. **Actores**
   Usuario

3. **Precondiciones**
   El sistema está activo

4. **Secuencia normal**
   1. El caso de uso comienza cuando el usuario accede a la página de inicio del sistema
   2. El sistema generará una página de inicio donde se podrán introducir los datos de acceso.
   3. El usuario introduce sus datos de acceso
   4. El sistema comprueba los datos de acceso y procede a mostrar la actividad que está siendo realizada e indicar al usuario que está correctamente identificado.
   5. El caso de uso finaliza.

5. **Excepciones**
   Datos de acceso erróneos
      1. En el paso 4 el sistema detecta que los datos introducidos no corresponden a ningún usuario del sistema. El caso de uso finaliza devolviendo al usuario a la página de inicio.

6. **Postcondiciones**
   El usuario está identificado con los datos proporcionados al sistema.

4.2.3.4 Caso de uso 2: Navegación hacia un punto de interés

1. **Descripción**
   El caso de uso describe como el alumno navega por el mapa hasta alcanzar un punto de interés introducido por el profesor en GLUE!-PS

2. **Actores**
   Alumno

3. **Precondiciones**
   El alumno se encuentra identificado en el sistema

4. **Secuencia normal**
   1. El caso de uso comienza cuando el sistema genera un modelo del globo terrestre navegable y con los puntos de interés del curso.
   2. El alumno puede usar comandos para avanzar en diferentes direcciones con el objetivo de llegar a un punto de interés y contemplar el entorno.
   3. El sistema modifica apropiadamente la vista en función de los comandos del alumno simulando el movimiento esperado.
   4. El alumno se acerca finalmente al punto de interés que quería alcanzar.
   5. El sistema muestra el punto de interés en forma de icono o modelo 3D.

5. **Excepciones**
   Cerrar
   Entre el paso 1 y 5 el alumno cierra la página y el caso de uso finaliza

6. **Postcondiciones**
   El alumno se encuentra en las cercanías de un punto de interés.

4.2.3.5 Caso de uso 3: Acceso a un recurso geolocalizado

1. **Descripción**
   El caso de uso describe como el alumno accede a un recurso una vez que ha localizado un punto de interés.

2. **Actores**
   Alumno

3. **Precondiciones**
   El alumno está identificado y tiene visualizado un punto de interés

4. **Secuencia normal**
   1. El caso de uso comienza cuando el sistema está mostrando un punto de interés en pantalla.
   2. El alumno selecciona el punto de interés pulsando sobre la representación del punto de interés.
   3. El sistema muestra una pantalla con información referente al recurso contenido y un enlace al mismo.
   4. El alumno pulsa el enlace.
   5. El sistema despliega el recurso sin cerrar la página actual y por tanto permitiendo al alumno volver al mismo punto cuando abandone el recurso.
   6. El caso de uso finaliza con el alumno visualizando el recurso.

5. **Excepciones**
   cancelar
   En el paso 4 el alumno puede cerrar el cuadro de información sin acceder al recurso y el caso de uso finaliza.

6. **Postcondiciones**
   El alumno está visualizando el recurso que ha seleccionado.

4.2.4 Diagramas de clases

El diagrama de clases de análisis del sistema descrito proporciona un modelo de alto nivel de los distintos conceptos involucrados y como pueden colaborar para llevar a cabo los casos de uso descritos en la sección anterior. Este diagrama se ha elaborado con clases frontera, control y entidad.

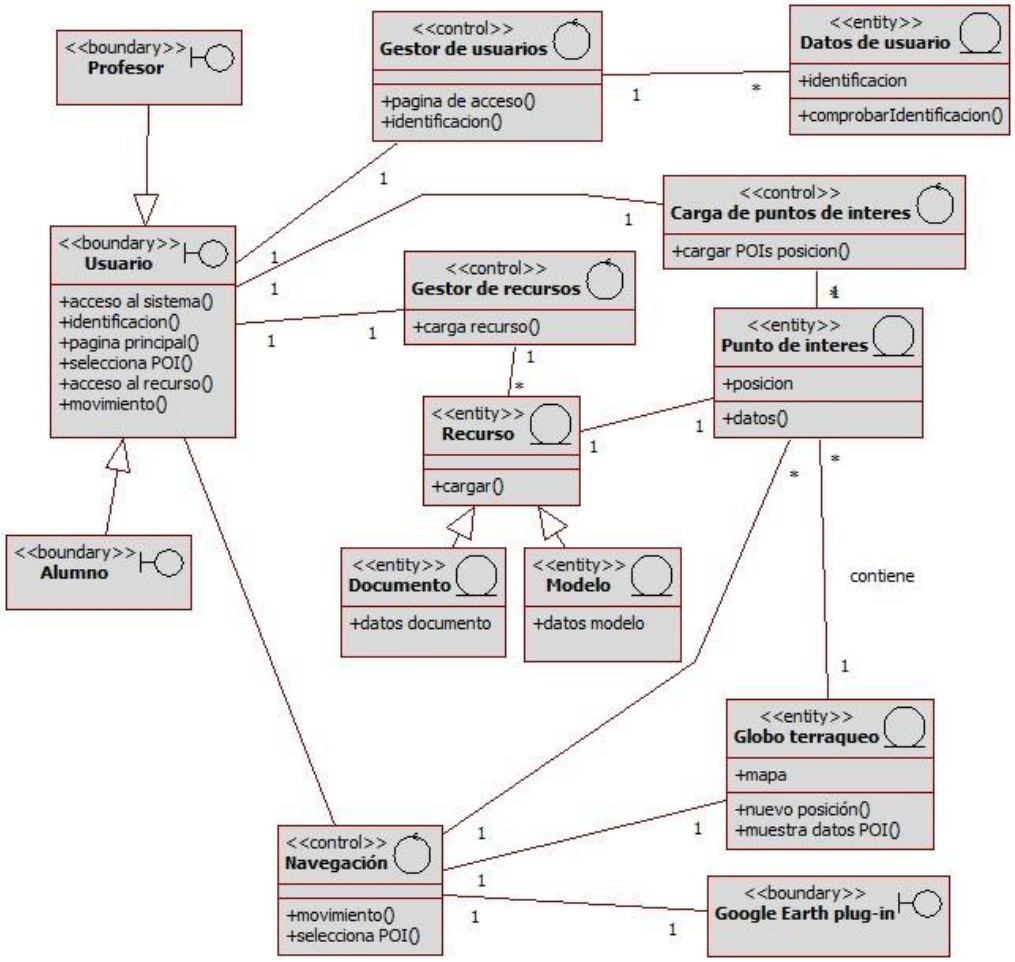

Ilustración 5 Diagrama de clases de análisis

4.2.5 Diagramas de secuencia

Los siguientes diagramas se corresponden a los casos de uso descritos y se apoyan en el diagrama de clases presentado en la sección anterior.

4.2.5.1 Caso de uso 1: Login

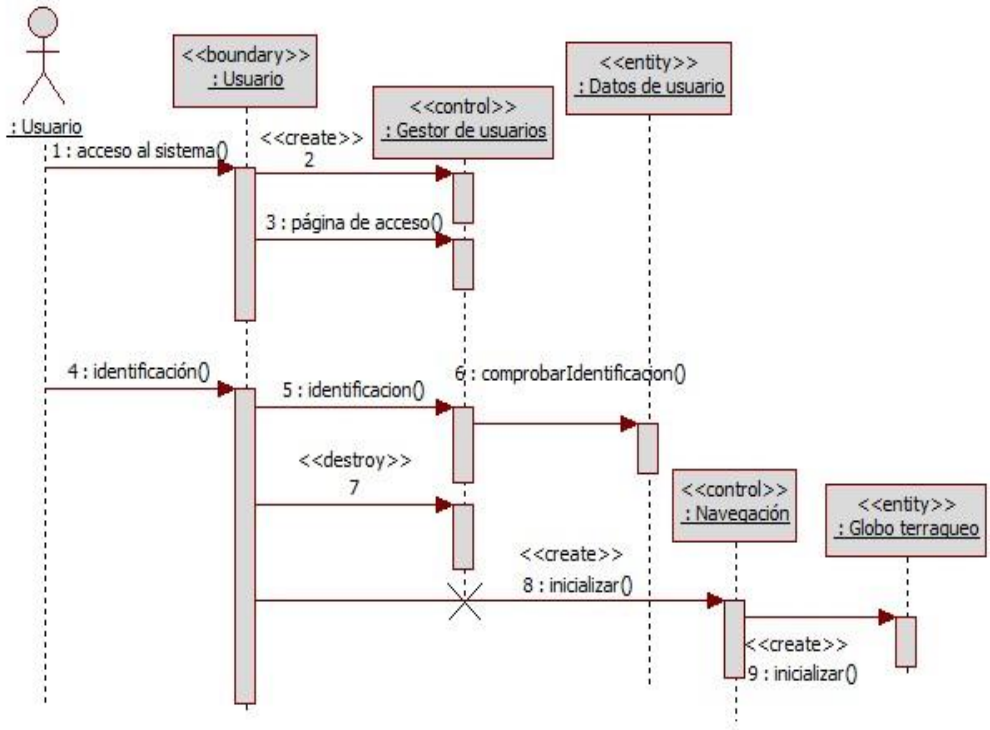

Ilustración 6 diagrama de secuencia caso de uso 1

4.2.5.2 Caso de uso 2: Navegación por un punto de interés

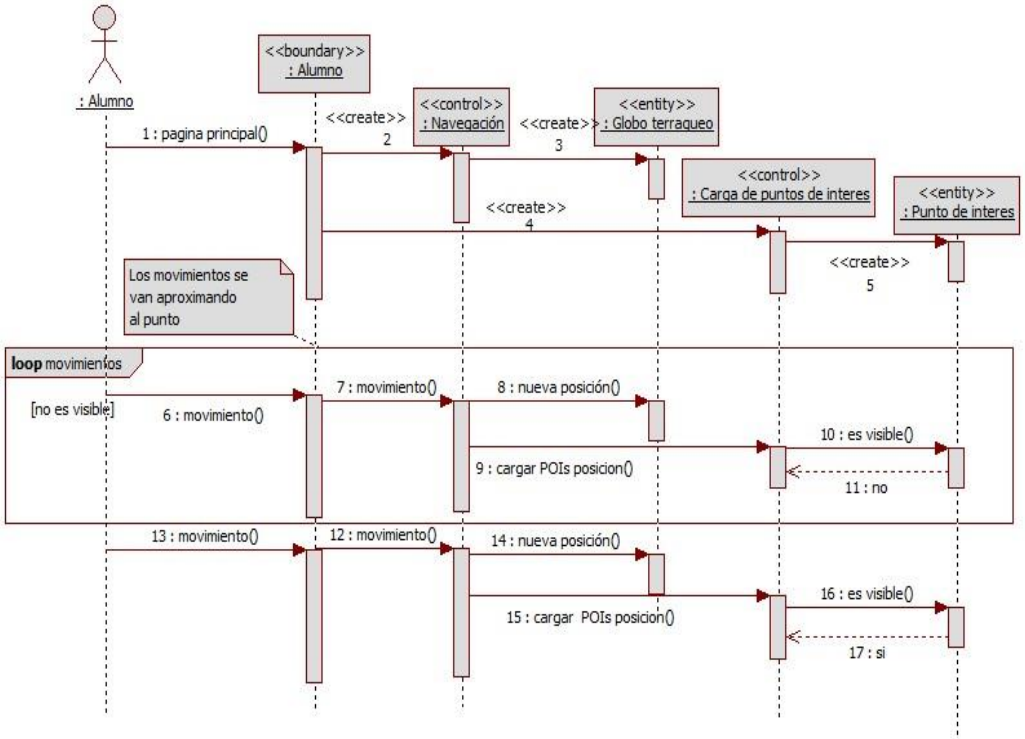

Ilustración 7 diagrama de secuencia caso de uso 2

4.2.5.3 Caso de uso 3: Acceso a un recurso geolocalizado

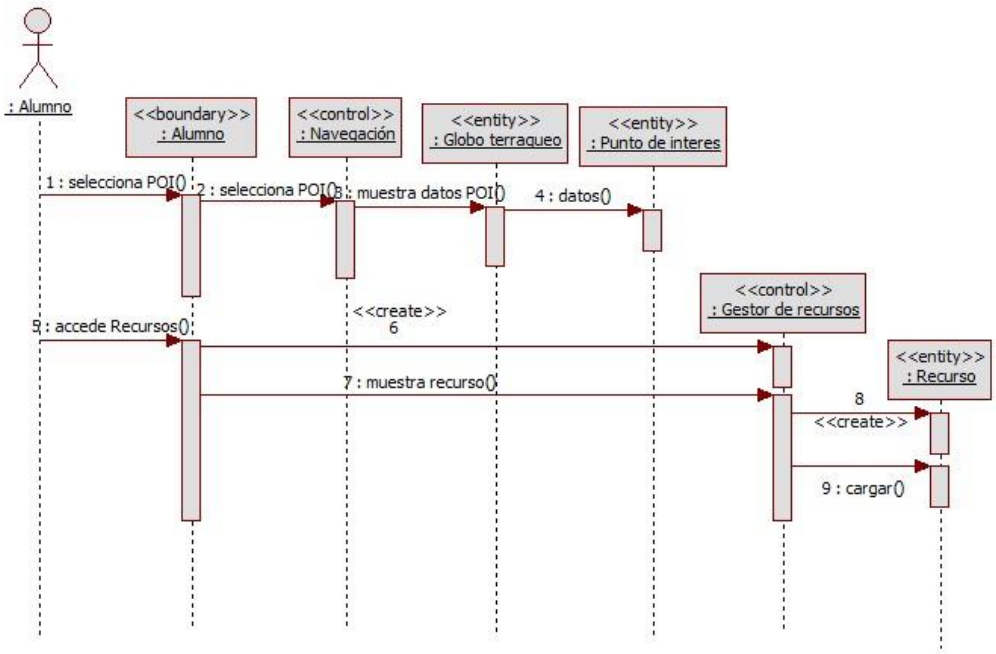

Ilustración 8 diagrama de secuencia caso de uso 3

Capítulo 5

Diseño

5 Diseño

5.1 Arquitectura propuesta

5.1.1 Visión global del sistema

El sistema se encuentra integrado en la arquitectura de GLUE!-PS que utiliza servicios web de tipo REST (*Representational state transfer* - Transferencia de Estado Representacional) que detallará en la sección de modelo arquitectónico. Estos servicios web permiten obtener los recursos necesarios y mantener la comunicación con el servidor.

La otra parte fundamental del sistema es el programa Javascript que interactúa con el plug-in de Google Earth en el cliente y se encarga de realizar las distintas peticiones al servicio Web.

5.1.2 Tecnologías propuestas

5.1.2.1 Jetty[13]

Jetty es un servidor y cliente HTTP y un contenedor de servlets Java basado en tecnología Java. Fue creado en 1995 y actualmente se encuentra en su versión 9.0.4. Desde 2009 el proyecto está hospedado en la Fundación Eclipse.

La antigüedad del proyecto no impide que soporte tecnologías de reciente creación como el protocolo SPDY[14] que proporciona una funcionalidad similar a HTTPS y Websockets que permiten una comunicación fluida entre navegador y servidor por medio de sockets.

Jetty es un servidor con un consumo de recursos reducido teniendo en cuenta que está programado completamente en Java lo que podría afectar negativamente a su rendimiento. Además es usando en proyectos como Google App Engine[15] y Apache Hadoop[16] que requieren una gran escalabilidad y un buen rendimiento.

Algo a considerar es que Jetty es un servidor multi-hilo y hay que prestar atención a cuestiones de concurrencia, no obstante el acceso a base de datos y en otras tareas habituales esto es resuelto por las librerías utilizadas sin necesidad de tomar medidas especiales.

El servidor cuenta con una buena integración en el IDE Eclipse lo que permite depurar las aplicaciones de forma cómoda y modificar el código sin reiniciar el servidor lo cual es de especial ayuda durante el desarrollo de aplicaciones.

[13] http://www.eclipse.org/jetty/ (último acceso: 01/09/2013)
[14] http://www.chromium.org/spdy (último acceso: 01/09/2013)
[15] https://appengine.google.com/ (último acceso: 01/09/2013)
[16] http://hadoop.apache.org/ (último acceso: 01/09/2013)

5.1.2.2 Restlet[17]

Se trata de un *framework* para construir APIs REST escrito en Java. Restlet soporta los conceptos REST presentados en la sección 5.3 y simplifica ajustarse al estándar cubriendo buena parte de los requisitos.

Al igual que Jetty estar basado en Java permite que se pueda instalar en varias plataformas sin cambios significativos en su funcionamiento.

El *framework* proporciona además una capa de abstracción sobre el servidor que permite desplegar aplicaciones creadas para Restlet en cualquiera de las plataformas soportadas: JavaEE, Google App Engine…

Las dos características más importantes de Restlet desde el punto de vista del diseño son posiblemente los *Routers* y los *Resources*.

- Los *Routers* permiten asociar URIs y recursos y por tanto determinan qué clase se responsabiliza de servir cada una de las peticiones. Para esto tienen una sintaxis basada en cadenas que permite asociar patrones con variables a recursos determinados. Los routers pueden ser encadenados de forma jerárquica para organizar los servicios en grupos y evitar que tengan demasiadas responsabilidades.
- Los *Resources* son clases que implementan un servicio y como su nombre indican devuelven recursos en terminología REST. Mediante anotaciones Java se identifican los métodos que responden a cada uno de los métodos HTTP, normalmente GET y POST.

La arquitectura del sistema una vez contemplado Jetty y Restlet es vista en la siguiente figura. Como se puede ver se corresponde con un patrón capas.

[17] http://restlet.org/ (último acceso: 01/09/2013)

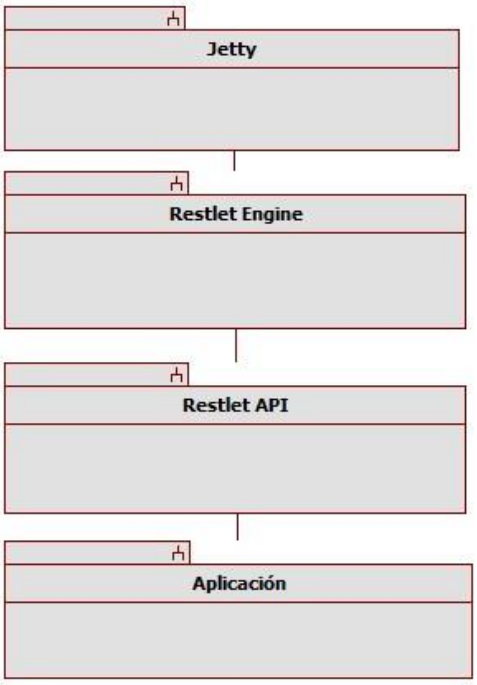

Ilustración 9 Arquitectura de las tecnologías del servidor

5.1.2.3 JQuery[18]

Una de las librerías Javascript más populares creada por John Resig. Sus principales ventajas son una gran simplificación de tareas complejas y habituales en el campo de las llamadas Ajax y también en la manipulación del documento HTML.

Adicionalmente usar los métodos proporcionados por la librería garantiza que el código será compatible con los navegadores Web más populares y a menudo abstrae las diferencias que estos presentan entre sí.

JQuery soporta los navegadores más populares incluyendo: Internet Explorer 6.x+, Firefox, Opera, Safari y Google Chrome. En la versión 2.x el soporte de Internet Explorer se limita a la versión 9 o superior.

Desde el punto de vista del diseño la estructura de los programas que hacen uso de JQuery destacan por el empleo de selectores que permiten invocar métodos sobre colecciones de elementos como si fueran uno solo y también por el uso de *callbacks*. Estos *callbacks* son funciones usadas como parámetros que permiten introducir funcionalidad que será ejecutada cuando se dispare un evento determinado.

[18] http://jquery.com/ (último acceso: 01/09/2013)

5.1.2.4 Java Persistency API[19]

La API de persistencia Java (JPA) es una tecnología desarrollada en el contexto de Java EE (Enterprise Edition) con el propósito de dotar a Java de una tecnología de mapeado objeto-relacional estándar y orientada al entorno empresarial. Posteriormente JPA pasó a formar parte también de Java SE (Standard Edition) y por tanto disponible en un mayor número de instalaciones Java.

Como se ha señalado JPA hace un mapeado objeto-relacional, estas tecnologías de las que existen otros ejemplos como Hibernate[20], tratan de simplificar el pasó de la programación orientada a objetos a la programación de base de datos relacionales. Para ello JPA dispone de unos objetos ligeros denominados Entidades a los que dota de persistencia y son cuyos datos son finalmente almacenados en las filas de una tabla de la base de datos. Existen métodos para recuperar entidades desde la base de datos, modificarlas y eliminarlas. Para realizar consultas JPA proporciona un lenguaje específico: Java Persistence Query Language (JPQL) similar a SQL pero operando sobre entidades.

La configuración de JPA se puede realizar mediante anotaciones Java en los propios objetos o mediante ficheros XML que realizan el mapeado. En cualquiera de los casos existen herramientas que automatizan el proceso en ambos sentidos, es decir, si se tienen las entidades se pueden generar las tablas correspondientes y si se parte del esquema de la base de datos se pueden generar entidades a partir de los datos del esquema.

5.1.2.5 Twitter Bootstrap[21]

Bootstrap es un conjunto de herramientas orientadas a la apariencia de las páginas web. La premisa de este *framework* es agilizar el desarrollo de páginas con una apariencia agradable y proporcionar funcionalidades de carácter estético a los distintos elementos HTML.

El *framework* ha sido desarrollado por la compañía detrás de la red social Twitter y su aspecto por defecto recuerda al diseño de esta red social. Desarrollado internamente inicialmente fue publicado con una licencia de código abierto en 2011.

Bootstrap soporta un conjunto limitado de elementos de HTML 5 y CSS 3 pero a cambio es compatible con los navegadores más populares incluyendo versiones móviles y de escritorio y ofrece un aspecto consistente entre ellos solucionando un problema fundamental del diseño web debido a ligeras variaciones en la implementación de los estándares por los distintos navegadores.

El *framework* está compuesto fundamentalmente de hojas de estilo programadas en LESS[22] un lenguaje para hojas de estilos que incluye mecanismos como la herencia y las variables que permiten un desarrollo modular y fácilmente modificable en contraposición a las hojas de estilo CSS que crecen de tamaño rápidamente y son problemáticas a la hora de mantenerlas cuando

[19] http://www.oracle.com/technetwork/java/javaee/tech/persistence-jsp-140049.html (último acceso: 01/09/2013)
[20] http://www.hibernate.org/ (último acceso: 01/09/2013)
[21] http://getbootstrap.com/2.3.2/index.html (último acceso: 01/09/2013)
[22] http://lesscss.org/ (último acceso: 01/09/2013)

alcanzan un cierto tamaño. Las hojas de estilo LESS son posteriormente compiladas a CSS para que puedan ser interpretadas por el navegador.

Bootstrap proporciona a grandes rasgos dos conjuntos de funcionalidades:

- Un sistema de 'rejilla' mediante clases que permite posicionar los elementos HTML en columnas y manejar los espacios de forma sencilla con la desventaja de perder la precisión y el control que ofrece usar CSS para fijar las propiedades de cada elemento.
- Estilos atractivos para elementos HTML y controles básicos. Con estos elementos se logra un aspecto de la aplicación web consistente y atractivo y en el caso de los controles añade código Javascript para lograr funcionalidad no estándar pero muy común como pueden ser menús desplegables o mensajes de alerta que pueden ser cerrados, navegación por pestañas, paginación...

5.2 Topología del sistema

En el siguiente diagrama se puede ver la estructura del sistema. Los nodos de servidor, servidor de recursos y servidor de base de datos se pueden unificar en uno solo que contenga los 3 elementos. El servidor CDN de Google almacena recursos Javascript como librerías y permite optimizar la velocidad de carga de la Web y uso de la caché del navegador al proporcionar un origen unificado para la librería que pueden compartir múltiples sitios Web.

El diagrama muestra un sistema cliente servidor en el que existen dos servidores: GluePSManager (Componente controlador de GLUE!-PS) y el servidor Google que realizan la mayor parte del trabajo respecto a la comunicación con el cliente y otros dos servidores: GlueletManager (Componente controlador de GLUE!) y CDN que hospedan recursos que pueden ser descargados, GlueletManager además realiza redirecciones a otros servidores externos al sistema.

El servidor de base de datos utiliza el protocolo TCP/IP para comunicarse con el servidor.

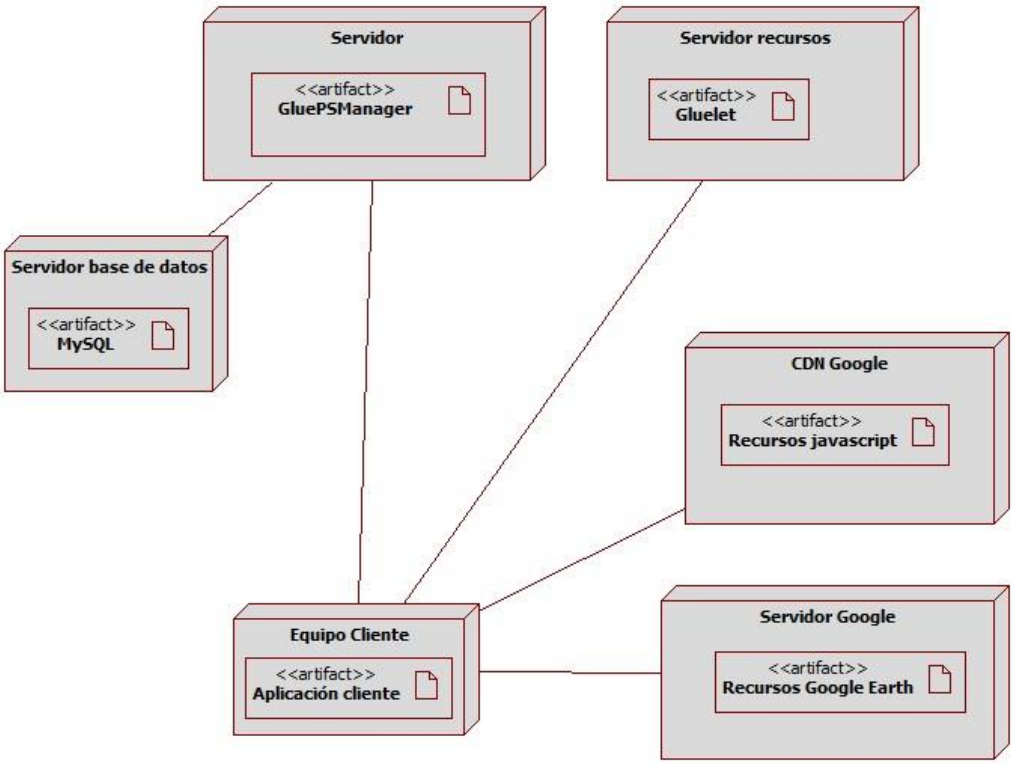

Ilustración 2 Diagrama de despliegue

5.3 Modelo arquitectónico

El patrón arquitectónico fundamental del sistema es SOA[23] (*Service Oriented Architecture* – Arquitectura orientada a servicios).

La idea central de SOA es la orientación a servicios de forma que la funcionalidad del sistema se puede ver como una colección de servicios prestados.

Los servicios en SOA son representaciones lógicas auto-contenidas de funciones y actividades que se pueden realizar repetidamente. Los servicios funcionan como una caja negra y ofrecen una interfaz que permite realizar llamadas y obtener respuestas, internamente el servicio puede estar implementado de diversas formas incluyendo la composición de otros servicios. El conjunto de servicios que conforman un sistema deben contar con pocas dependencias entre sí.

La razón por la que se ha decidido usar este patrón es que la plataforma GLUE!-PS usa este mismo patrón y por tanto con el fin de reutilizar los recursos de la plataforma y conseguir una buena integración con la misma, la mejor opción era no introducir una arquitectura diferente.

Es importante destacar que la implementación de SOA utilizada en el sistema emplea REST desarrollado por el W3C TAG (*Technical Architecture Group* – Grupo técnico de arquitecturas) y en especial por Roy Fielding [24] en paralelo al estándar HTTP 1.1. Una arquitectura REST cuenta con las siguientes restricciones.

- Modelo cliente-servidor: Cada uno tiene sus responsabilidades y se ajusta especialmente a su utilización en aplicaciones web.
- Sin estado: El servidor no guarda ninguna información de sesión y por tanto del estado del cliente, cada petición es independiente y auto-contenida.
- Caché: Los servicios deben especificar si pueden ser cacheados con el fin de mejorar la eficiencia del sistema.
- Capas: Muy relacionado con la caché consiste en que el cliente no sabe si está conectado al servidor final a uno intermedio que contiene una copia de los recursos solicitados.
- Interfaz uniforme: Un concepto importante en REST es el recurso. Los recursos tienen un identificador, normalmente una URI, y se trabaja con ellos mediante representaciones de los mismos en un formato determinado. Las representaciones de los recursos contienen suficiente información para manipular los mismos en el servidor, además estás representaciones incluyen metadatos que indican al cliente como deben ser procesados. Finalmente se emplean hipermedios en el sentido de que para navegar por recursos REST y descubrir otros nuevos se emplean fundamentalmente hiperenlaces.

Tanto la arquitectura SOA como REST imponen por tanto una serie de limitaciones a la estructura del sistema y se tratará durante el diseño y la implementación de cumplirlas. No obstante ya sea por limitaciones tecnológicas o simplicidad, en puntos concretos se podrían flexibilizar estas limitaciones.

[23] http://www.opengroup.org/soa/source-book/soa/soa.htm (último acceso: 30/08/2013)
[24] http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm
(último acceso: 30/08/2013)

5.4 Casos de uso de diseño

5.4.1 Caso de uso 1: Login

1. **Descripción**
   El caso de uso describe como el profesor o el alumno accede al sistema, se identifica en el mismo y selecciona el curso que va a realizar.

2. **Actores**
   Usuario

3. **Precondiciones**
   El sistema está activo

4. **Secuencia normal**
   1. El caso de uso comienza cuando el usuario accede a la página de inicio del sistema
   2. El sistema generará una página de inicio, sin el plug-in de Google Earth activado donde se podrán usuario y contraseña.
   3. El usuario introduce sus datos de acceso
   4. El sistema comprueba los datos de acceso, registra que se ha producido el acceso y recupera un listado de los deploys en los que participa ese usuario en ese momento, muestra el listado al usuario para que seleccione el deploy apropiado.
   5. El usuario selecciona el deploy que vaya a explorar.
   6. El sistema carga el plug-in de Google Earth y los puntos de interés asociados a ese deploy y usuario.
   7. El caso de uso finaliza.

5. **Excepciones**
   Datos de acceso erróneos
      1. En el paso 4 el sistema detecta que los datos introducidos no corresponden a ningún usuario del sistema. El caso de uso finaliza devolviendo al usuario a la página de inicio y mostrando un mensaje indicando el error que se ha producido.

   No hay desploys disponibles
      1. En el paso 4 el sistema detecta que no hay ningún deploy disponible. El sistema muestra un mensaje de advertencia con esta circunstancia y el caso de uso finaliza al no ser posible acceder a ningún deploy.

6. **Postcondiciones**
   El usuario está identificado con los datos proporcionados al sistema y se ha cargado un curso.

5.4.2 Caso de uso 2: Navegación hacia un punto de interés

1. **Descripción**
   El caso de uso describe como el alumno navega por el mapa hasta alcanzar un punto de interés introducido por el profesor en GLUE!-PS

2. **Actores**
   Alumno

3. **Precondiciones**
   El alumno se encuentra identificado en el sistema

4. **Secuencia normal**
   1. El caso de uso comienza cuando el sistema carga el plug-in de Google Earth con los puntos de interés correspondientes al deploy actual.
   2. Mientras el usuario no ha alcanzado su destino.
      1. El alumno puede usar comandos para avanzar en diferentes direcciones con el objetivo de llegar a un punto de interés y contemplar el entorno, el usuario también puede cambiar al modo Street View.
      2. El sistema modifica apropiadamente la vista en función de los comandos del alumno simulando el movimiento esperado.
      3. El sistema registra la posición del alumno periódicamente para monitorizarlo.
   3. El alumno se acerca finalmente al punto de interés que quería alcanzar.
   4. El sistema muestra el punto de interés que se había cargado previamente ya sea un icono o un modelo 3D.

5. **Excepciones**
   Cerrar
   Entre el paso 1 y 5 el alumno cierra la página y el caso de uso finaliza

6. **Postcondiciones**
   El alumno se encuentra en las cercanías del punto de interés que había seleccionado.

5.4.3 Caso de uso 3: Acceso a un recurso geolocalizado

1. **Descripción**
   El caso de uso describe como el alumno accede a un recurso una vez que ha localizado un punto de interés.

2. **Actores**
   Alumno

3. **Precondiciones**
   El alumno está identificado y tiene visualizado un punto de interés

4. **Secuencia normal**
   1. El caso de uso comienza cuando el sistema está mostrando un punto de interés en pantalla.
   2. El alumno selecciona el punto de interés pulsando sobre el icono que representa el punto de interés.
   3. El sistema detecta la pulsación a través del plug-in de Google Earth y usa el mismo para mostrar la información acerca del punto de interés seleccionado en forma de enlace al recurso.
   4. El alumno pulsa el enlace.
   5. El sistema despliega el recurso en una nueva página mientras deja el sistema en el mismo estado que se encontraba. El sistema también registra que se accedido al recurso con el fin de monitorizar la actividad.
   6. El caso de uso finaliza con el alumno visualizando el recurso.

5. **Excepciones**
   cancelar
   En el paso 4 el alumno puede cerrar el cuadro de información pulsando en cualquier área del plug-in o realizando un movimiento sin acceder al recurso y el caso de uso finaliza.

6. **Postcondiciones**
   El alumno está visualizando el recurso que ha seleccionado.

5.5 Diagrama de clases de diseño

Los diagramas de clases de diseño se han dividido en dos, por un lado el diagrama del lado del servidor y por otro el diagrama del código javascript.

5.5.1 Diagrama de clases del servidor GLUE!-PS

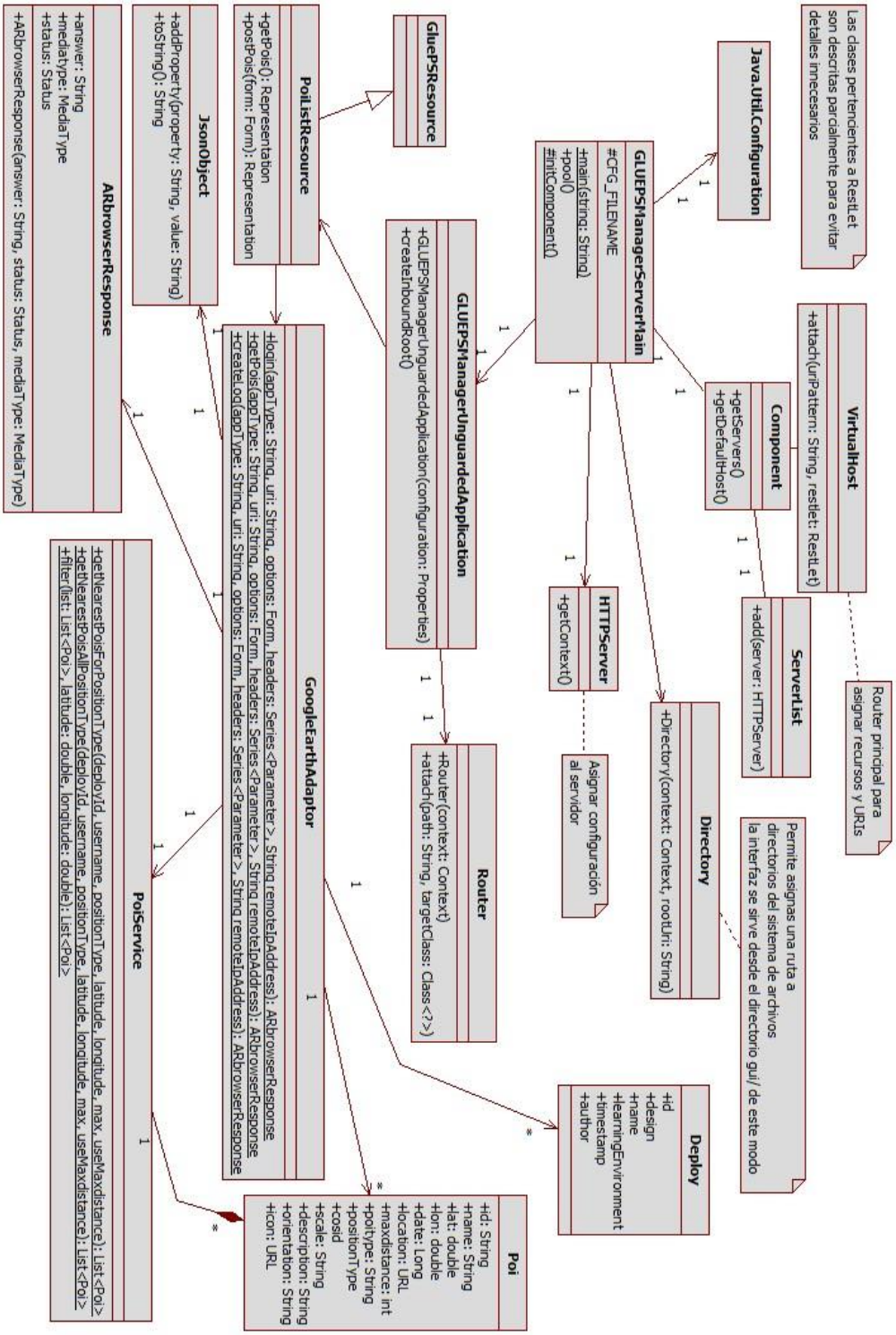

Ilustración 11 Diagrama de clases del servidor

5.5.2 Diagrama de clases del cliente

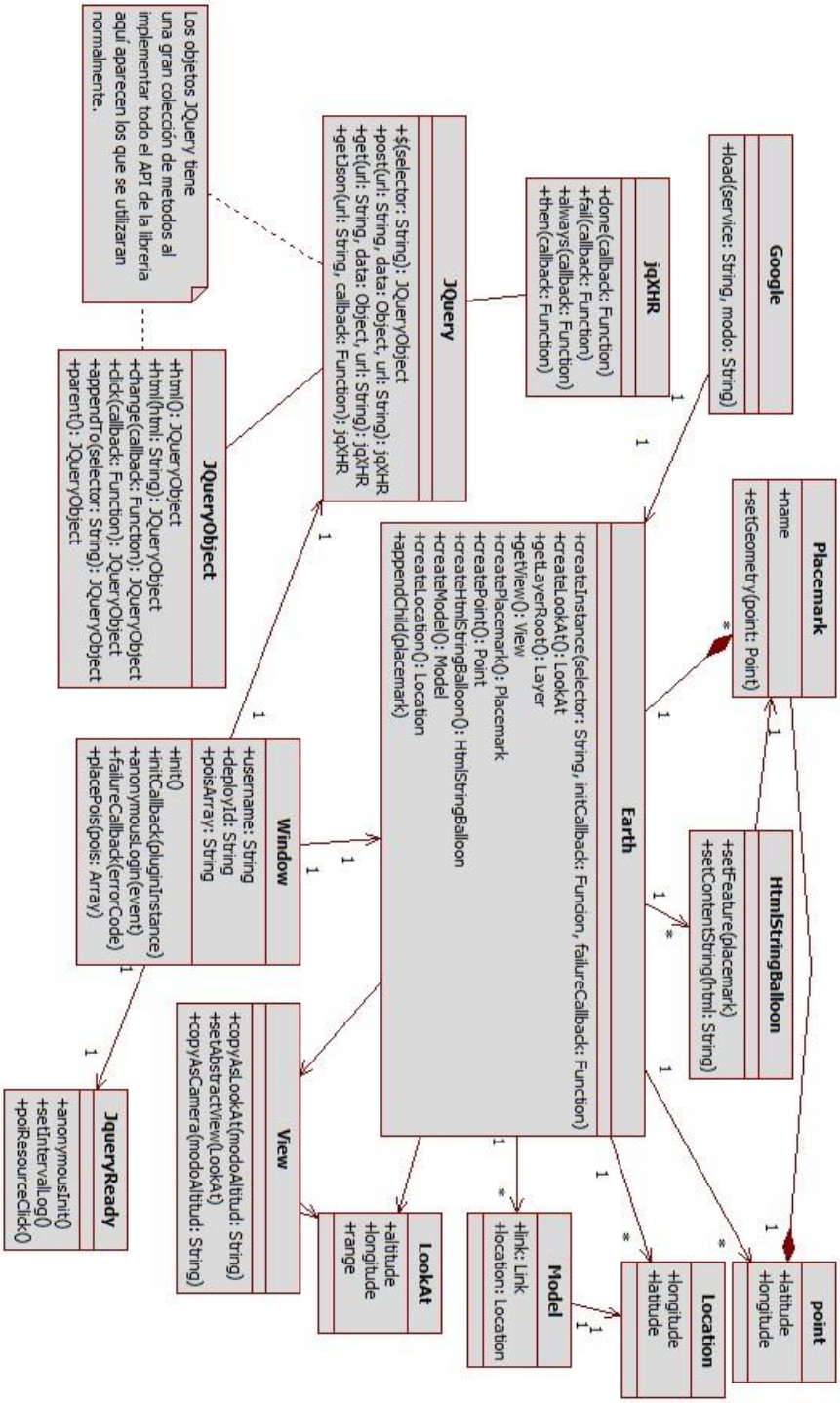

Ilustración 12 Diagrama de clases del cliente

5.6 Diagramas de secuencia de diseño

5.6.1 Caso de uso 1: Login

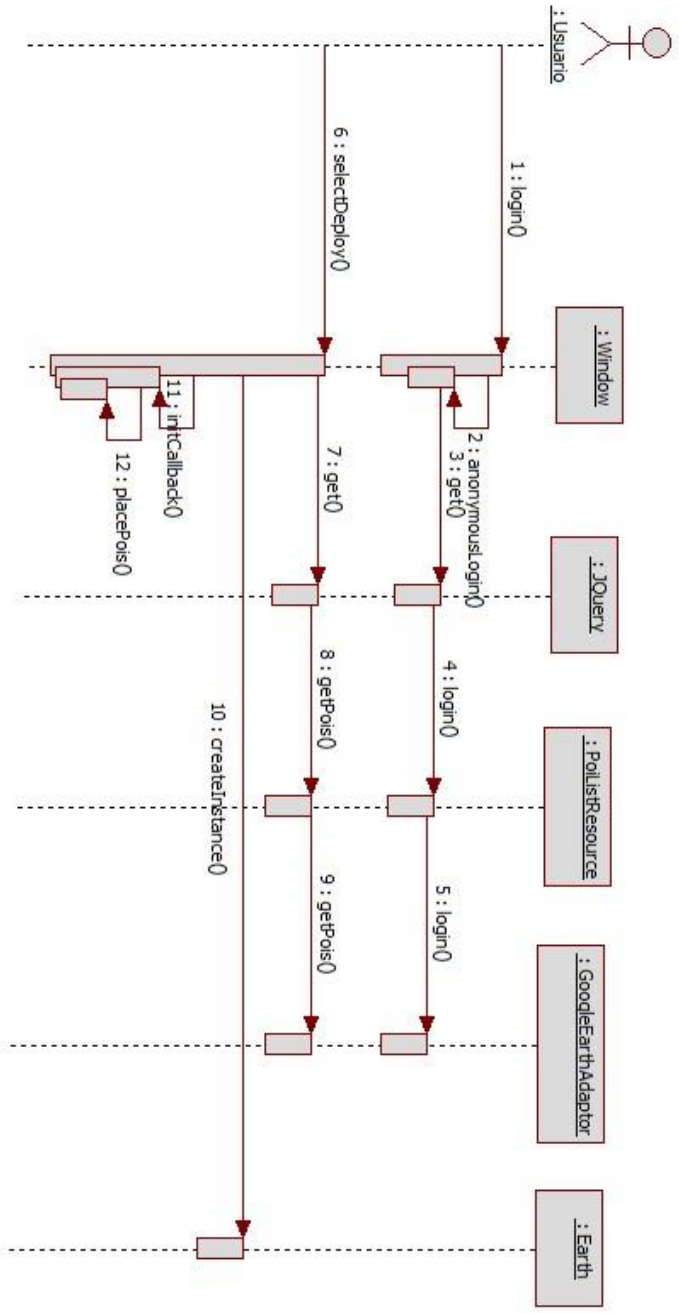

Ilustración 13 Diagrama de secuencia de diseño 1 Login

5.6.2 Caso de uso 2: Navegación hacia un punto de interés

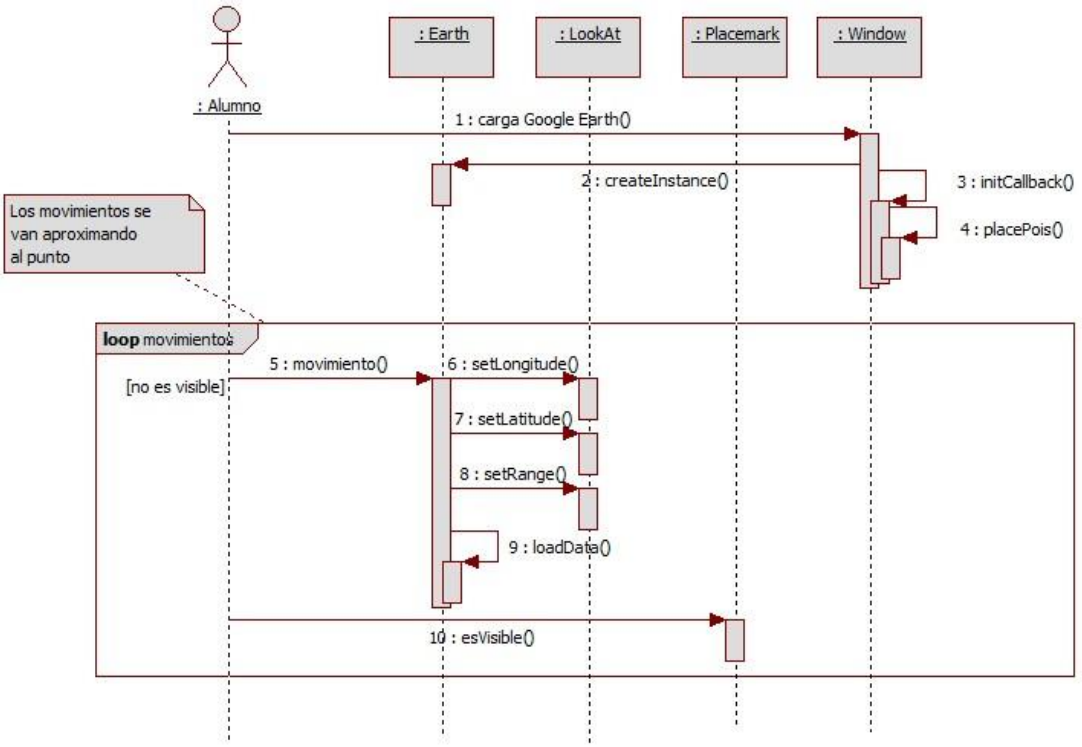

Ilustración 14 Diagrama de secuencia de diseño 2 navegación hacia un punto de interés

5.6.3 Caso de uso 3: Acceso a un recurso geolocalizado

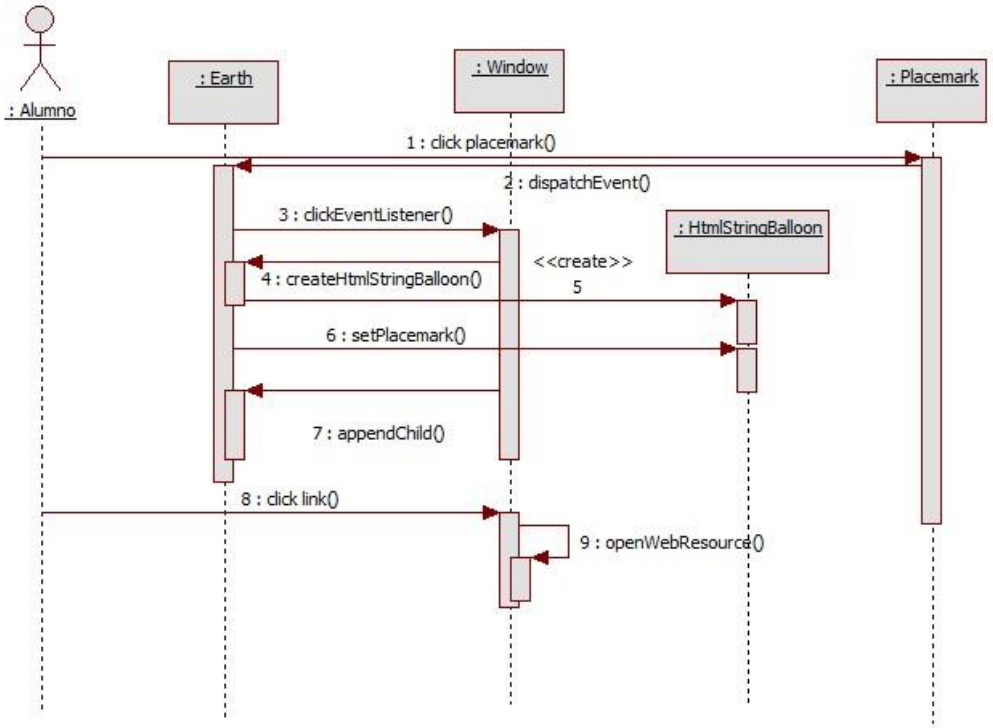

Ilustración 15 Diagrama de secuencia de diseño 3 Acceso a un recurso

Capítulo 6

Implementación

6 Implementación

En este capítulo se describe el proceso de implementación del sistema y las herramientas utilizadas durante el mismo.

6.1 Tecnologías de asistencia al desarrollo

Entre las tecnologías utilizadas para el desarrollo de este proyecto destacan tres: Eclipse[25], Subversion[26] y las herramientas de desarrollador de Google Chrome[27].

Eclipse: El entorno de desarrollo escogido para el desarrollo de proyecto ha sido Eclipse, la decisión se tomó por recomendación de los tutores debido a que es el entorno usado habitualmente en el desarrollo de GLUE!-PS, está disponible en los equipos del laboratorio y ofrece una serie de ventajas para ser usado con el resto de herramientas seleccionadas.

Eclipse está basado en *plug-ins* que complementan el entorno en función de las necesidades del desarrollador y existen varias distribuciones que incorporan un conjunto de *plug-ins* orientados a un tipo de desarrollo específico, en el caso de este TFG se empleó Eclipse para desarrolladores de Java EE[28], orientado a aplicaciones empresariales pero que se adapta muy bien a cualquier desarrollo Web basado en Java.

Entre las ventajas de este entorno de desarrollo en el caso de este proyecto concreto se encuentra la posibilidad de actualizar el código ejecutándose en el servidor Jetty sin necesidad de reiniciar el mismo. Eclipse cuenta con soporte para lenguajes de tipo XML, Javascript y HTML además del mencionado Java también utilizados en el desarrollo. Debido al uso de un servidor completamente desarrollado en Java como es Jetty el depurador de código integrado funciona de forma correcta y permite desplazarse a cualquier punto del procesado de una petición lo cual suele ser más problemático en el caso de otros servidores con peor integración.

La plataforma GLUE!-PS está conformada por varios proyectos Eclipse, en nuestro caso es necesario GLUEPSManager donde se realiza el desarrollo del sistema y GLUECommon que se enlaza como una dependencia.

Subversion: es el sistema de control de versiones utilizado para el desarrollo del proyecto. Se trata de un software heredero de CVS. Se trata de un sistema de control de versiones centralizado y por tanto el repositorio está ubicado en un servidor y cada uno de los clientes mantiene una copia de trabajo que debe de sincronizar regularmente.

[25] http://www.eclipse.org/ (último acceso: 4/9/2013)
[26] http://subversion.apache.org/ (último acceso: 4/9/2013)
[27] https://developers.google.com/chrome-developer-tools/ (último acceso: 4/9/2013)
[28] http://www.eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/keplerr (último acceso: 4/9/2013)

Para simplificar el uso de subversión existe un *plug-in* para Eclipse que integra ambas aplicaciones y permite realizar las tareas de sincronización del repositorio sin abandonar el entorno de desarrollo.

En el caso de este proyecto se empleó al comienzo el repositorio de subversión público de GLUE!-PS[29] posteriormente y cuando aumentaron las tareas de programación se asignó un repositorio específico para el desarrollo del adaptador con permisos de escritura.

Herramientas de desarrollo de Google Chrome: Estas utilidades integradas en Google Chrome proporcionan una gran ayuda en el desarrollo de código Javascript para el navegador. Entre las más destacables se encuentra el depurador de código que permite depurar en el entorno final de ejecución. Se complementan las herramientas de depuración de una consola que permite escribir mensajes desde el código de la aplicación y evaluar pequeños fragmentos de Javascript escritos directamente en el navegador y que se ejecutan en el contexto de la aplicación pudiendo acceder a sus funciones y eventos.

Para la comunicación entre cliente y servidor las herramientas incluyen un monitor de red que identifica todas las peticiones realizadas desde el navegador, los parámetros incluidos en las mismas y los recursos obtenidos como respuesta o en su caso el error que ha impedido completar la petición con éxito.

La última herramienta empleada frecuentemente durante el desarrollo ha sido el explorador del modelo de objetos del documento (DOM). Este explorador permite identificar cualquier elemento HTML presente en un documento web y mostrar el código asociado en el árbol del documento. También se pueden consultar las propiedades del mismo.

6.2 Técnicas de programación

En esta sección se discutirán algunos detalles de programación no habituales en la programación orientada a objetos. Estos se han dado durante el desarrollo en el lenguaje Javascript.

Callbacks: Consiste en un fragmento de código que es utilizado como parámetro en la llamada a una función y presumiblemente será ejecutada en un momento posterior por parte de la función llamada. Por ejemplo:

```
google.earth.createInstance('map3d', initCallback, failureCallback);
```

En la llamada a función `createInstance` los parámetros `initCallback` y `failureCallback` son funciones que serán ejecutadas cuando se finalice la creación de un mapa 3d por parte del plug-in de Google Earth en el elemento 'div' de la página web que tiene como identificador map3d. En el caso de que el mapa se haya podido instanciar se llama a `initCallback` y en caso de error a `failureCallback` de este modo se puede ejecutar código condicionalmente de forma asíncrona y prescindiendo de la presencia de instrucciones condicionales alrededor de la llamada.

[29] http://pandora.tel.uva.es/svn/GLUE (Último acceso: 4/9/2013)

Los *callbacks* proporcionan una forma sencilla de delegar en otra función la ejecución de un fragmento de código. Un problema que se detecta rápidamente cuando se usa repetidamente este mecanismo es lo tedioso de crear una función para fragmentos de código a veces muy reducidos o que sólo son usados en un punto del programa. Por eso los lenguajes que los soportan suelen ofrecer una sintaxis para funciones anónimas que se pueden definir en cualquier punto donde se escribiría el nombre de la función de *callback.*

```
google.earth.createInstance('map3d', initCallback, function(){

      alert("No se ha podido instanciar Google Earth");

});
```

En este caso se ha sustituido la función `failureCallback` por una función anónima que muestra un mensaje de error.

Eventos: Los eventos se han convertido en un elemento imprescindible de la programación web. Su presencia en el DOM ha contribuido notablemente a su popularización y posteriormente se han extendido a otras interfaces como es el caso del *plug-in* de Google Earth.

Esta popularidad también se ha visto influida por la sencillez de implementar un sistema basado en eventos a partir de *callbacks*. De forma básica los eventos son disparados ante un suceso determinado detectado por el programa y son capturados y manejados por un fragmento de código establecido para tal fin. Por ejemplo:

```
google.earth.addEventListener(placemark, 'click', function(event) {

alert ("click");

});
```

En este fragmento de código se ha asignado un manejador de eventos al objeto `placemark` (representado por un icono en la interfaz) cuando se detecte un click sobre el mismo. El manejador del evento en este caso es una función anónima que muestra un mensaje por pantalla a modo de prueba cuando se hace click sobre el icono.

Selectores JQuery: Son una de las grandes ventajas de esta librería respecto a usar directamente el lenguaje Javascript para interaccionar con el DOM. Utilizan una sintaxis estandarizada por la W3C[30]. Hasta que la mayoría de los navegadores los soportaron nativamente al incluirse en librerías eran capaces de agilizar notablemente el desarrollo de código Javascript y aún hoy en día cuando se necesita soportar navegadores antiguos son de gran utilidad. Un ejemplo de selector básico sería:

```
$('div .poiResource');
```

[30] http://www.w3.org/TR/CSS2/selector.html

Este selector devolverá todos los elementos del DOM que sean de tipo 'div' y tenga `poiResource` en su lista de atributos de clase. La sintaxis de selectores cubre todas las propiedades básicas del lenguaje HTML y se pueden hacer consultas en función del tipo de elemento, clase, identificador, atributos específicos o su posición relativa dentro del DOM. Además a medida que el lenguaje HTML evoluciona las posibilidades de los selectores se han ido incrementando gradualmente.

Una característica especial de los selectores JQuery y no presente en los nativos es su tratamiento de colecciones de elementos. Un selector puede encontrar varios elementos y es muy habitual la necesidad de aplicarles una misma función a todos ellos por lo que JQuery agiliza este proceso haciendo que muchas de sus funciones trabajen con colecciones además de con elementos individuales. Por ejemplo:

```
$('div .poiResource').hide();
```

Si este selector encontrará diez elementos aplicaría la función `hide,` que los oculta visualmente, a todos ellos. En caso de no encontrar ningún elemento la función no se aplica en absoluto.

Capítulo 7

Validación y pruebas

7 Validación y Pruebas

En este capítulo se describen una batería de pruebas de caja negra que cubren la funcionalidad del sistema desarrollado con el fin de detectar y subsanar el mayor número posible de errores.

7.1 Prueba 1: Visualización de puntos de interés

Entrada	Salida esperada	Salida obtenida
Curso desplegado con 3 puntos de interés en la ciudad de Valladolid.	3 iconos en las posiciones donde los 3 puntos fueron geolocalizados.	3 iconos en las posiciones donde los 3 puntos fueron geolocalizados. De forma aleatoria y ocasional alguno de los puntos parpadea, el problema parece atribuible al plug-in de Google Earth.

Tabla 4 Resultados prueba 1

7.2 Prueba 2: Visualización de modelos 3D

Entrada	Salida esperada	Salida obtenida
Curso desplegado con 1 modelo 3D de un cubo con textura situado en la ciudad de Madrid.[31]	El modelo 3D se visualiza de forma fiel en la posición apropiada.	El modelo 3D se visualiza de forma fiel en la posición apropiada.

Tabla 5 Resultados prueba 2

7.3 Prueba 3: Identificación de usuario

Entrada	Salida esperada	Salida obtenida
El usuario introduce sus datos de acceso.	El sistema da la bienvenida al usuario y se muestran los cursos que tiene disponibles.	El sistema da la bienvenida al usuario y se muestran los cursos que tiene disponibles.

Tabla 6 Resultados prueba 3

[31] http://earth-api-samples.googlecode.com/svn/trunk/examples/static/splotchy_box.dae (último acceso: 01/09/2013)

7.4 Prueba 4 Selección de un curso

Entrada	Salida esperada	Salida obtenida
El usuario selecciona un curso.	Se cargan los puntos de interés del curso que ha sido seleccionado a los que el usuario pertenezca y de actividades activadas en GLUE!-PS.	Se cargan los puntos de interés del curso que ha sido seleccionado a los que el usuario pertenezca y de actividades activadas en GLUE!-PS. El retraso de tiempo hasta la carga del plug-in cuando se acaba de iniciar el navegador puede ser de 5-7 segundos.

Tabla 7 Resultados prueba 4

7.5 Prueba 5 Monitorización de eventos

Entrada	Salida esperada	Salida obtenida
Cada 5 minutos, cuando se accede al sistema y cuando se accede a un recurso se registra el evento.	Los eventos son registrados en el registro correspondiente y con los parámetros correctos de tiempo y posición en el mapa.	Los eventos son registrados en el registro correspondiente y con los parámetros correctos de tiempo y posición en el mapa. En caso de fallo de red los eventos de acceso a recurso y temporales se pueden perder no así el de acceso al sistema.

Tabla 8 Resultados prueba 5

7.6 Prueba 6: Selección de puntos de interés

Entrada	Salida esperada	Salida obtenida
Se pulsa sobre uno de los puntos de interés.	Se despliega un cuadro con el enlace de acceso al recurso contenido en el punto de interés.	Se despliega un cuadro con el enlace de acceso al recurso contenido en el punto de interés. Debido al parpadeo ocasional descrito en la prueba 1 puede ser necesario pulsar varias veces sobre un mismo punto de interés para que se muestre el cuadro.

Tabla 9 Resultados prueba 6

Capítulo 8

Conclusiones y trabajo futuro

8 Conclusiones y trabajo futuro

En este último capítulo se exponen las conclusiones obtenidas y las posibles mejoras del sistema desarrollado durante este TFG.

En el apartado de conclusiones conviene separar entre las aplicables al alumno personalmente y las correspondientes al trabajo desarrollado.

Para las conclusiones del trabajo realizado me centraré en primer lugar en los objetivos propuestos. De los tres subobjetivos puede asegurarse que el primero de ellos, el estudio de tecnologías, se realizó satisfactoriamente obteniendo una visión general de las herramientas más apropiadas para el trabajo.

Del mismo modo la implementación del adaptador para GLUE!-PS y una interfaz de usuario para el mundo virtual 3D realista se han conseguido en lo más básico y con la funcionalidad esperada. Es cierto no obstante que no se ha logrado crear un mundo virtual completo tal y como se definía en la introducción al no haberse desarrollado un sistema de comunicación entre los distintos participantes.

En las conclusiones personales me gustaría destacar en primer lugar la oportunidad de haberme sumergido en un campo como es del aprendizaje mejorado por tecnología del que no tenía conocimientos previos y que me ha resultado muy interesante y sobre todo amplio y variado. Existen una gran variedad de tecnologías y técnicas desarrolladas en otros ámbitos y que cuenta con unas posibilidades de aplicación en el aprendizaje sorprendentes. Es también destacable cómo el trabajo realizado es muchas veces de aplicación de ideas y técnicas ya conocidas a nuevos entornos, dejando en segundo plano la invención de nuevas técnicas.

Por otra parte he desarrollado una parte del trabajo en el laboratorio del GSIC en la universidad de Valladolid lo que me ha permitido estar en contacto y ver la labor investigadora de sus miembros. Dado que una vez finalizado este TFG volveré al mundo laboral, abandonando el ámbito universitario al menos durante un tiempo, me ha encantado tener la posibilidad de ver este aspecto de la universidad que nunca había conocido siendo estudiante.

Concluyendo las conclusiones personales y en un plano más técnico destacaría lo aprendido al extender un sistema ya existente, desarrollando sobre una base de código en evolución en contraposición a un proyecto de duración y tamaño limitado. Me ha sido posible ver un sistema de un tamaño considerable y que va creciendo incorporando nuevas funcionalidades que lo mejoran globalmente. La otra área relacionada con lo anterior en la que he adquirido nuevos conocimientos es en la programación contra interfaces programables y servicios en vez de interfaces de usuario. Tanto los protocolos empleados como la gestión de errores y excepciones me han resultado novedosos respecto otros trabajos realizados anteriormente.

Por último conviene analizar las posibilidades y limitaciones del sistema desarrollado para poder explorar las posibilidades de futuras mejoras. A medida que la plataforma GLUE!-PS continúe incorporando características y nuevas funcionalidades debería ser posible incorporarlas modificando el adaptador desarrollado.

La interfaz de usuario y la integración con Google Earth también pueden ser mejoradas en el aspecto estético tanto de la parte externa al plug-in de la aplicación web como del mapa mostrado en el plug-in mediante la configuración del mismo o aprovechando mejoras futuras por parte de Google.

En un futuro y dependiendo de la dirección que Google decida tomar sobre la tecnología de Google Earth sería posible desarrollar una versión del sistema con tecnologías web estándar prescindiendo del plug-in, siendo la principal ventaja la compatibilidad con un mayor número y variedad de plataformas. En este caso la cantidad de trabajo adicional necesario dependerá fundamentalmente de las modificaciones que se introduzcan en las interfaces que se han utilizado en el desarrollo del sistema.

Capítulo 9

Referencias bibliográficas

9 Referencias bibliográficas

Bell, M. W. (2008). "Toward a Definition of "Virtual Worlds"." Journal of Virtual Worlds Research 1(1): 1-5.

Eckel, B. (2005). Thinking in Java (4th Edition), Prentice Hall PTR.

Gamma, E., R. Helm, et al. (1995). Design patterns: elements of reusable object-oriented software, Addison-Wesley Longman Publishing Co., Inc.

Larman, C. (2004). Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development (3rd Edition), Prentice Hall PTR.

Muñoz-Cristóbal, J. A., L. P. Prieto-Santos, et al. (2012). "Orchestrating TEL situations across spaces using Augmented Reality through GLUE!-PS AR." Bulletin of the Technical Committee on Learning Technology 14(4): 14-16.

Prieto, L. P., J. I. Asensio-Pérez, et al. (2011). GLUE!-PS: a multi-language architecture and data model to deploy TEL designs to multiple learning environments. Proceedings of the 6th European conference on Technology enhanced learning: towards ubiquitous learning, Berlin, Heidelberg, Springer-Verlag.

Sommerville, I. (1995). Software engineering (5th ed.), Addison Wesley Longman Publishing Co., Inc.

Anexos

Anexos

Manuales

En estos anexos se incluyen los manuales del sistema, el manual de usuario es reducido debido a los pocos pasos necesarios para la utilización del sistema. El manual de instalación se ha desarrollado parcialmente a partir de los ficheros del proyecto GLUEPSManager /doc-usr/INSTALL.txt y /doc-dev/ guia_acceso_configuracion_gluepsmanager.pdf en ellos se puede encontrar documentación adicional y detallada del proceso de instalación.

Anexo 1: Manual de usuario

Para acceder al sistema el primer paso consiste en introducir la URL de la aplicación en el navegador. Esta URL es de la forma dominio/GLUEPSManager/gui/glueps/google/earth.html

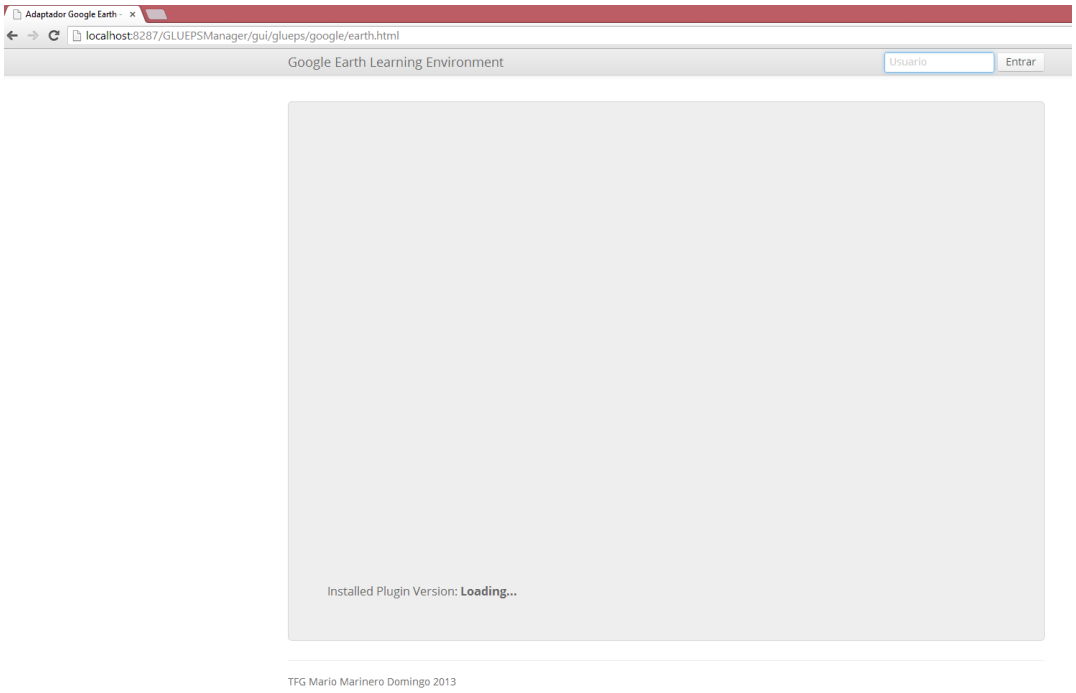

Ilustración 16 Captura inicio del sistema

Una vez en la página principal es siguiente paso es introducir los datos de acceso en el campo reservado al usuario en la esquina superior derecha y pulsar entrar.

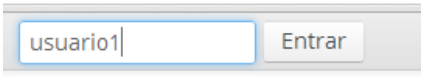

Ilustración 17 Detalle de acceso de un usuario

Sin cambiar de página el sistema dará la bienvenida al usuario y mostrar el cuadro de selección de deploy entre los disponibles para ese usuario.

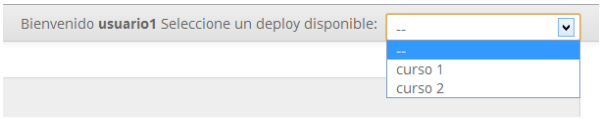

Ilustración 18 Detalle selección de curso

Cuando ya se ha seleccionado un curso se despliega el mismo siendo posible navegar por el mismo mediante los controles de Google Earth ya sea mediante la acción de arrastre del ratón o mediante

los controles de la parte derecha, también se puede cambiar en la parte superior derecha a la vista Google Street View. En esta última captura se puede ver cómo se puede visualizar un modelo 3D introducido en un curso determinado dentro de la aplicación.

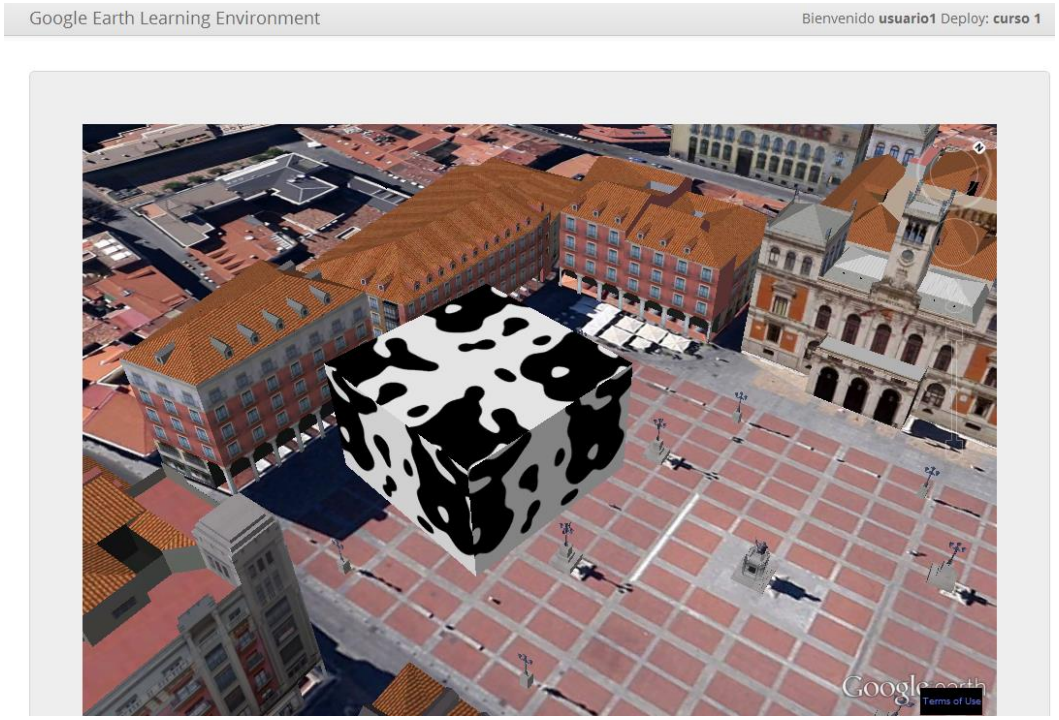

Ilustración 19 Modelo de prueba en la Plaza Mayor de Valladolid

Anexo 2: Manual de instalación

Requisitos del sistema

La plataforma GLUE!-PS ha sido probada en sistemas Linux y Windows XP, Vista, 7 y 8.

El sistema consiste en un programa Java desarrollado para Java SE 6 y es compatible con máquinas virtuales Java JRE 6.

Contenido del software

El software entregado contiene la aplicación GLUEPSManager junto al adaptador para Google Earth, este software incluye tanto la aplicación Java como la interfaz HTML estática de la aplicación.

Dependencias

- Java JRE 6
- Servidor de base de datos MySQL con los datos de configuración modificables en el fichero conf/META-INF/persistence.xml y con la base de datos inicial siguiendo el esquema incluido en conf/create_glueps_database.sql
- El cliente debe contar con acceso a internet para la obtención de datos por parte del plug-in de Google Earth.

Instalación en plataformas Linux

1. Descomprimir la distribución binaria de la aplicación en un directorio que llamaremos GLUEPS_HOME y definir una variable de entorno GLUEPS_HOME que apunte a la ruta completa del directorio creado.
2. Dar permisos de ejecución las los scripts *.sh encontrados en la carpeta GLUEPS_HOME/manager/bin
3. Ejecutar el fichero GLUEPS_HOME/manager/bin/install-glueps.sh
4. Copiar el fichero bin/utils/init.d/svfb en /etc/init.d/ y dar permisos de ejecución al fichero.

Configuración

Antes de iniciar la aplicación deben ser configurados los siguientes parámetros:

En GLUEPS_HOME/manager/conf/app.properties

- - port: Puerto que escucha la aplicación GLUE!-PS Manager
- - logging: On para guarder información de cada petición en el log.
- - app.path: Ruta a GLUEPS_HOME/manager
- - app.external.uri: URI externa de GLUEPSManager
- - gluelet.uri: URI externa de GLUELetManager
- - gluelet.uri.interna: URI Interna de GLUELetManager
- mw.gluepsuse: Nombre de usuario de MediaWiki para desplegar wikis

- - mw.gluepspwd: Contraseña de MediaWiki

En el fichero GLUEPS_HOME/manager/conf/META-INF/persistence.xml se debe modificar

- <property name="eclipselink.jdbc.user" value="user"/> Usuario con acceso a la base de datos GLUE!-PS
- <property name="eclipselink.jdbc.password" value="password"/> Contraseña del usuario de base de datos
- <property name="eclipselink.jdbc.driver" value="com.mysql.jdbc.Driver"/> Driver JDBC del sistema.
- <property name="eclipselink.jdbc.url" value="jdbc:mysql://localhost:3306/glueps"/> URL de la base de datos.

Ejecución:

Una vez completada la instalación se deben emplear los comandos start-glueps y stop-glueps para iniciar y parar el servidor.