SERVEI DE SISTEMES D'INFORMACIÓ GEOGRÀFICA I TELEDETECCIÓ
Universitat de Girona

III JORNADAS DE SIG LIBRE

Capaware: Plataforma de desarrollo de software libre para aplicaciones geográficas 3D multicapa

M. Castrillón[4], P.A. Jorge[2], I.J. López[1], A. Macías[2], D. Martín[2], R.J. Nebot[1], I. Sabbagh[1], J. Sánchez[2], A.J. Sánchez[2], J.P. Suárez[3], A. Trujillo[2]

[1]Instituto Tecnológico de Canarias.
[2]Departamento de Informática y Sistemas. Universidad de las Palmas de Gran Canaria (ULPGC).
[3]Departamento de Cartografía y Expresión Gráfica en la Ingeniería. ULPGC.
[4]Instituto Universitario SIANI. ULPGC.

RESUMEN

En este trabajo se presenta Capaware, una plataforma de software libre para el desarrollo de aplicaciones geográficas 3D multicapa, que surge a partir de la iniciativa del Instituto Tecnológico de Canarias en colaboración con la Universidad de Las Palmas de Gran Canaria. Este entorno simplifica la creación de aplicaciones 3D sobre territorios geográficos extensos, disponiendo de una herramienta muy visual que aporta un nuevo punto de vista muy importante para una toma de decisiones eficaz.
Capaware proporciona una interfaz fácil de usar y muy flexible que simplifica el desarrollo de nuevas aplicaciones, permitiéndonos crear rápidamente entornos virtuales con múltiples capas de información sobre el terreno. Con las capacidades clásicas de un Sistema de Información Geográfica (SIG), Capaware permite actualmente la carga de capas WMS sobre entornos 3D, añadir objetos 3D sobre el terreno, y visualizar elementos dinámicos, ofreciendo una nueva perspectiva de la información analizada. Así mismo, podemos administrar las capas de recursos y elementos que se pueden representar sobre la zona geográfica en cuestión.
Capaware ha sido desarrollado en C++, utilizando el motor gráfico de código abierto Open Scene Graph, y la librería de interfaz WxWidget igualmente libre, con lo cual se convierte en un software multiplataforma. Con un sistema de plugins, permite que cualquier desarrollador pueda completar el sistema con nuevas funcionalidades específicas.
En la actualidad, como ejemplo de uso de este sistema de desarrollo, se están articulando los medios para la implantación de un sistema virtual de seguimiento de emergencias para el Cabildo de la isla de La Palma. Como funcionalidad principal tendrá la de seguimiento y predicción de los incendios forestales. Con las capacidades gráficas de esta herramienta, la nueva aplicación incluye un plugin de simulación de incendios, que aportará a los técnicos de emergencias forestales de la isla nuevos elementos para la toma de decisiones en el caso de los incendios, tanto en tiempo real como de forma preventiva.

Palabras clave: *Plataforma de desarrollo, SIG 3D, incendio forestal.*

ABSTRACT

In this work we present Capaware, a free software platform to develop 3D multilayer geographical applications. The project is an initiative of the Technological Institute of the Canary Islands with the active collaboration of the Universidad de Las Palmas de Gran Canaria. The platform makes it simple the development of 3D applications over large geographics areas and providing very visual tool contributing with a new point of view.
Capaware gives a flexible and easy to use interface that simplifies the development of 3D geographical applications. With Capaware anyone may build visual environment with many layers of terrain informations in a fast manner. The software has the classics features of a GIS software and it permits the integration of WMS layers over the 3D land and also 3D designed objects. An added feature of the software allows he visualization of dynamics objects in the terrain, providing so of a new perspective to analyze the information. In addition, Capaware allows the user to manage the resources and objects in the terrain.
Capaware is a cross-platform software that has been developed in C++ using the graphics toolkit Open Scene Graph and the tools library WxWidget. With the plugin system capability the software developer increases the functions and possilities.
Currently, the project using Capaware is oriented to an application that implement a virtual system of emergencies monitoring for the Cabildo of La Palma island, in particular, it is focused on the forest fire predictions. With the graphics capabilities of the developed tool a new plugin for fire simulations is included. That plugin offers to forest fire engineers in the island new tools for the decision making process in forest fire field, in a real time environment and also in a preventive manner.

Key words: *development platforms, SIG 3D, wildfire.*

INTRODUCCIÓN

El panorama de las aplicaciones geoespaciales ha crecido enormemente en estos últimos años, en particular gracias al empuje de la comunidad libre que ha desarrollado numerosas herramientas y librerías de código abierto sobre las cuales se han sostenido ([1] y [13]). En este artículo se describe el proceso del desarrollo de una herramienta open source, llamada Capaware, orientada al desarrollo de aplicaciones geográficas multicapa 3D.

Inicialmente se resumen los pasos que nos han hecho llegar a este punto, a continuación se analiza la herramienta presentada, posteriormente se expone la forma de generar un terreno para poder navegar sobre él de forma realista, y finalmente se comenta la funcionalidad de la aplicación, y la forma de desarrollar plugins sobre el sistema.

HISTORIA

Proyecto inicial: El Hierro Virtual

A finales del año 2004, el Instituto Tecnológico de Canarias (ITC) propició el proyecto ”El Hierro Virtual”, donde se creó un navegador virtual en una zona concreta de las Islas Canarias, en la isla de El Hierro. Dicho simulador permitía una experiencia de vuelo real, con el fin de servir de presentación de la isla con distintos propósitos,

tanto turísticos como industriales (la figura 1 muestra una captura de pantalla).

Figura 1: *Screenshot de la aplicación "El Hierro Virtual"*

Los objetivos buscados al inicio del desarrollo, y que deberían permitir considerar la aplicación como diferenciadora de otras aportaciones similares, eran los siguientes:

- Elección de una plataforma de desarrollo potente, flexible y de licencia pública. En este sentido hay que señalar que existían plataformas de desarrollo para realizar vuelos virtuales con costes muy elevados que eran prohibitivas de amortizar. En nuestro caso, el motor Crystal Space [2] fue la elección.
- Suavidad del movimiento. En el vuelo, el usuario debía experimentar la sensación más suave en el movimiento libre sobre la isla, evitando así, saltos o discontinuidades en el vuelo.
- Realismo y fidelidad de las ortofotos. Se debería poder incorporar las mejores ortofotos digitales y convenientemente mapeadas en el sistema para lograr la sensación más real del vuelo sobre la isla.

Todos los objetivos se cumplieron satisfactoriamente ([3] y [4]). Como curiosidad cabe destacar que el popular navegador Google Earth, software que incorporaba algunas de las características requeridas por el ITC y la ULPGC, aunque con mucho menor rendimiento al tratarse de una aplicación web, no había aparecido todavía (lo hizo en Julio de 2005).

Nuevo proyecto: Geviemer

Tras el éxito obtenido en el visualizador anterior, y aprovechando la experiencia adquirida por el equipo de desarrollo, se decidió emprender un proyecto más ambicioso, que abarcara la totalidad de la superficie del archipiélago canario, y que además tuviera una utilidad concreta: que fuera capaz de realizar simulaciones de incendios para su uso en un sistema de emergencias ([5]) (ver figura 2).

Los objetivos que se plantearon para este nuevo proyecto fueron los siguientes:

- Desarrollar un motor que permitiera la visualización realista de terrenos extensos.
- Poder insertar elementos 3D sobre el terreno, y poder animarlos en el tiempo.
- Capaz de conectarse a servidores estándar OGC y mostrar sus capas de información sobre dicho terreno.
- Permitir varios usuarios conectados simultáneamente en tiempo real, de forma que las modificaciones de uno sean vistas inmediatamente por el resto.
- Generar una API libre para poder desarrollar nuevas aplicaciones sobre ella.
- Permitir la creación de plugins para añadir nuevas funcionalidades al sistema.

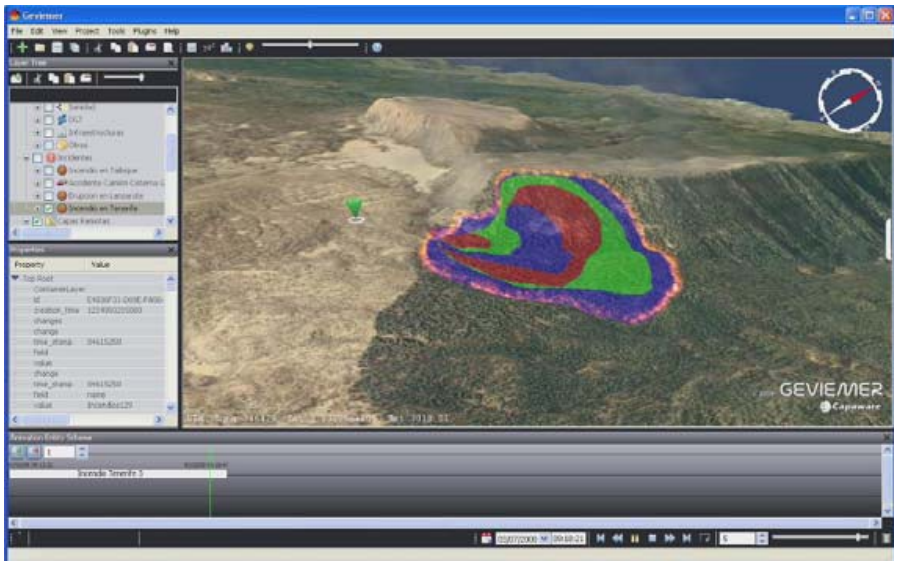

Figura 2: *Screenshot de la aplicación "Geviemer (Gestor Virtual de Emergencias"*

Separando la API de la aplicación final: Capaware

Desde el principio se observa la necesidad de crear una plataforma base a partir de la cual construir la aplicación final. De esta manera se cumplirían dos objetivos simultáneamente: por un lado se obtendría una herramienta tipo SDK que incluyera toda la representación gráfica y el manejo del terreno en tiempo real, desde la cual poder desarrollar aplicaciones geográficas en 3D, y por otro lado la aplicación final se obtendría como un ejemplo de uso de dicha herramienta. A esta herramienta le dimos el nombre de Capaware (software de capas).

También desde ese momento se tomó la decisión por todas las partes implicadas de que el SDK fuera código abierto, principalmente porque todas las entidades implicadas eran administraciones públicas así como los fondos dedicados al proyecto. Así pues uno de los muchos retos del proyecto era también el poder poner al alcance del resto de la comunidad todo el software desarrollado en el mismo.

ANÁLISIS DEL PROYECTO

Elección del motor gráfico: OpenSceneGraph

El proyecto de El Hierro Virtual se desarrolló utilizando la librería Crystal Space. Esta librería está orientada al desarrollo de aplicaciones gráficas 3D, con especial énfasis hacia el área de los videojuegos. Con el modelo de la isla de El Hierro su funcionamiento fue bastante aceptable, ya que el terreno no era excesivamente grande para la capacidad de memoria de una tarjeta gráfica de gama media.

Teniendo en cuenta que la superficie aproximada de la isla es de 270 Km^2, y que la malla poligonal que representaba el terreno tenía una resolución de 10 metros por vértices, y las ortofotos eran de 1 metro por píxel, el modelo final tenía un tamaño inferior a los 200 Mb, por debajo de los 256 Mb de una tarjeta media.

Sin embargo, para el nuevo proyecto el terreno abarcado era mucho mayor, ya que comprendía las 7 islas canarias, las cuales suman aproximadamente 7.500 Km^2 (sin considerar la superficie de mar entre ellas). La librería Crystal Space quedaba corta para este objetivo. Era necesario utilizar un nuevo motor que incluyera niveles de detalle (LOD) para trabajar con superficies tan grandes.

Se estudiaron varias alternativas y dos se ajustaban perfectamente a lo que se quería: OpenSceneGraph (OSG) [6], y Virtual Terrain Project (VTP) [7]. Mientras que OSG es de propósito general para cualquier aplicación gráfica, VTP está orientada expresamente para la visualización de terrenos. De hecho, en su página web se mencionan diferentes usos que la comunidad internacional de desarrolladores está realizando actualmente.

Sin embargo, después de hacer varias pruebas de rendimiento, el funcionamiento no era el esperado, al contrario que OSG que se comportaba perfectamente en todas las pruebas que hicimos. Por otro lado, la generación del modelo del terreno a partir del modelo digital y de las ortofotos no era del todo adecuada en VTP, mientras que en OSG podíamos controlar varios parámetros como la simplificación de la malla o el tipo (regular o irregular).

La prueba final que se hizo fue generar un terreno en forma rectangular que abarcara la superficie de las 7 islas, suponiendo que no existiera mar, es decir, como si fuera una extensión sobre el continente. Dicho rectángulo tenía una superficie total de 100.000 Km^2 (500 x 200), la cual se discretizó en una malla de resolución 10 metros por vértice (en total mil millones de vértices, 2 mil millones de triángulos), y el funcionamiento con una tarjeta gráfica de gama media fue bastante aceptable.

Arquitectura de Capaware

El núcleo de Capaware es una **API multiplataforma**, llamada **CPW**, la cual puede utilizarse para el desarrollo de aplicaciones de visualización de terrenos. Este núcleo se construye sobre el motor gráfico elegido, OSG, el cual está soportado por un lado por la librería gráfica multiplataforma OpenGL, y por otro lado, por una librería de interfaces de usuario genérica, que en nuestro caso ha sido WxWidgets (ver figura 3).

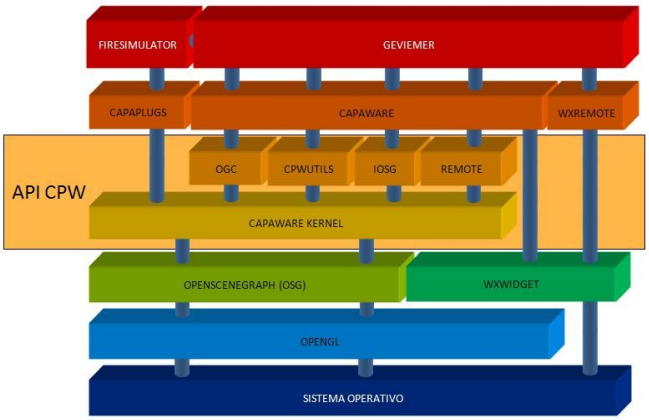

Figura 3: *Diagrama de bloques de Capaware*

La API CPW está compuesta por un módulo base, llamado **Capaware Kernel**, que incluye toda la funcionalidad básica, y otros módulos adicionales en niveles superiores para tratar las conexiones remotas, los protocolos OGC, utilidades varias, etc.

Por encima de la API CPW existe la aplicación general **Capaware**, que es la que integra la llamada a la mayoría de la funcionalidad de la API. El objetivo de esta nueva aplicación es servir para usuarios creadores de contenido, y para usuarios que desarrollen plugins para añadir nuevas funcionalidades. Por otro lado, también pretende servir como ejemplo de partida para los desarrolladores de aplicaciones.

Finalmente, en lo más alto del diagrama de la figura 3, aparece una aplicación más elaborada, ya con contenido específico y con un plugin concreto de simulación de incendios añadido, que era la aplicación pedida inicialmente por el ITC. Este último bloque es donde aparecería cualquier aplicación desarrollada con la API partiendo del código fuente de Capaware.

GENERACIÓN DE UN TERRENO

Hoy en día estamos acostumbrados a visualizadores como Google Earth, que nos permiten navegar por cualquier lugar del planeta prácticamente en tiempo real, y sin necesidad de generar ningún terreno con antelación. Simplemente viajamos hacia la zona que queramos y el terreno va a apareciendo en la pantalla a medida que nos va llegando a nuestro equipo desde internet. Realmente, el terreno ya ha sido generado en los servidores de Google, y llega hasta nosotros al hacer la conexión.

Sin embargo, en nuestra aplicación la visualización realista era muy importante, y no se deseaba que se notasen esos pequeños desfases cuando la información debe ser transferida desde internet. Por otro lado, se quería poder disponer de un control total sobre el terreno, de forma que se pudieran elegir mallas de distintas resoluciones. Por ejemplo, hoy en día, que ya existen sistemas LIDAR para obtener modelos digitales con resolución muy alta (de varios centímetros) [8], podría haber una necesidad para representar una zona específica a dicha resolución. Esto no es posible con visualizadores tipo Google Earth, en donde el modelo que representa la superficie es el decidido por Google (el cual además no indica qué resolución tiene).

La solución elegida por tanto es la que propone la librería OSG, que consiste en representar todo el terreno en una estructura quad-tree, en donde los nodos de cada nivel se corresponden con un nivel de detalle diferente. De esta manera, el nodo raíz o nivel cero representaría un modelo de toda la superficie a una resolución muy baja; a continuación, el siguiente nivel serían 4 nodos que representarían los cuatro cuadrantes del terreno, a una resolución el doble de precisa, y así sucesivamente hasta llegar a los nodos hoja, en donde se representarían zonas muy pequeñas del terreno con la resolución máxima que se requiera, como se muestra en la figura 4.

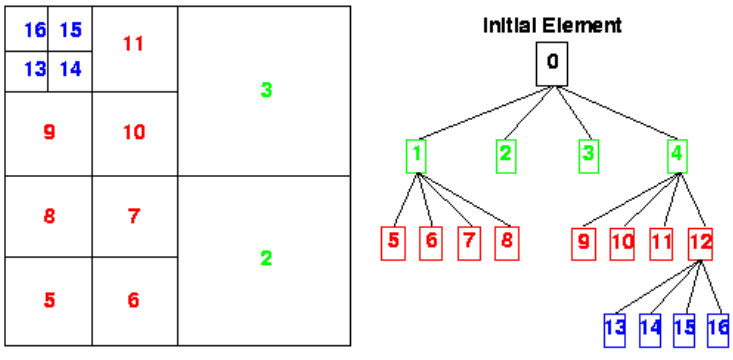

Figura 4: *Ejemplo de un quad-tree para representar un terreno*

De esta forma, a medida que se navega sobre el terreno, el sistema decide para cada zona con qué nivel de detalle debe mostrarse, y un algoritmo de carga de disco y liberación en memoria va decidiendo, en función de la distancia del observador a cada zona, qué nodos cargar y cuáles liberar. De esta forma se consigue un número de polígonos relativamente constante en cada fotograma, en función de lo cerca o lejos que estemos sobre el terreno, como se ve en la figura 5.

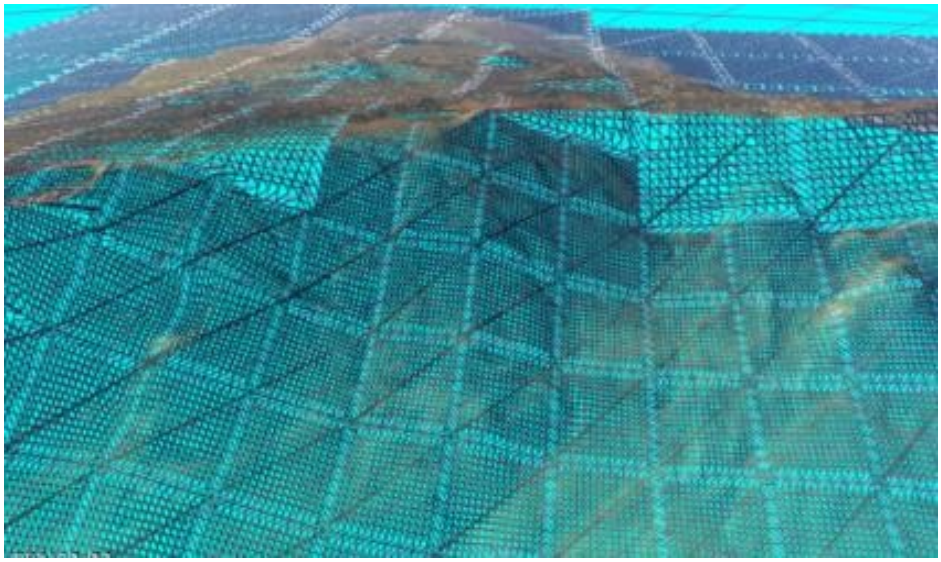

Figura 5: *Visualización de nodos del quad-tree durante la navegación*

El problema es que para generar un quad-tree a partir del modelo digital y de las ortofotos se necesitan dos cosas importantes: tiempo, para crear en disco toda la información necesaria para cada nodo (malla poligonal y textura asociada), y espacio en disco. Por ejemplo, para la isla de Tenerife, cuya extensión es próxima a los 2.000 Km², con una resolución de ortofoto de 1 metro por pixel, y una resolución de malla de 10 metros por vértice, necesitamos aproximadamente 15Gb de espacio en disco.

Con el sistema actual, y teniendo en cuenta que para nuestro caso particular, nuestra escena estaba compuesta de 7 islas independientes, hemos generado un quad-tree para cada isla, y el mar se ha representado como una malla plana de muy poca resolución que abarca toda la escena.

FUNCIONALIDAD DE LA APLICACIÓN

Capaware es un sistema de visualización geográfica 3D multicapa, que permite insertar elementos tridimensionales sobre el terreno, conecta con servidores WMS, y permitar conectar varios usuarios simultáneamente, y todo ello desarrollado sobre un SDK (CPW) que integra toda esta funcionalidad. Analicemos cómo ha sido el desarrollo de estas características

Inserción de modelos tridimensionales

Una vez estemos sobre el terreno, podemos insertar cualquier objeto tridimensional sobre él. Actualmente los formatos con los que se trabaja son 3DStudio y OSG, aunque el motor gráfico también permite otros formatos adicionales. Dichos objetos son trasladados, reorientados y escalados a petición del usuario mediante un gizmo como se aprecia en la figura 6, hasta posicionarlo en la ubicación deseada, pudiendo estar incluso elevados sobre la superficie (como en el caso de helicópteros o aviones para el ámbito de las emergencias).

Estos modelos pueden representar cualquier objeto que se desee sobre el terreno: vehículos, edificios, postes de luz, etc. lo que vendría en función de las necesidades de cada aplicación.

Figura 6: *Reorientando un modelo 3D sobre el terreno*

Visualización del fuego

Teniendo en cuenta que la aplicación final requerida iba a estar orientada a la gestión de incendios, se desarrolló primeramente una técnica basada en un sistema de partículas genérico sobre el terreno, con el cual se pudiera generar no sólo fuego, sino también otros objetos típicos de representarse con sistemas de partículas, como pueden ser nubes o humo.

A partir de esto, se concretó en un sistema para el caso de una llama, y otro para el humo. En ambos casos se hicieron numerosas pruebas hasta lograr una calidad visual aceptable. Hay que tener en cuenta que la calidad es un parámetro subjetivo, y que además dependía también de la distancia de visualización. No es lo mismo ver la llama a 5 metros de distancia que desde una altura de mil metros. Por lo tanto, los parámetros de ambos sistemas se hicieron también dependientes de la distancia al observador, para un mayor realismo, como se ve en la figura 7.

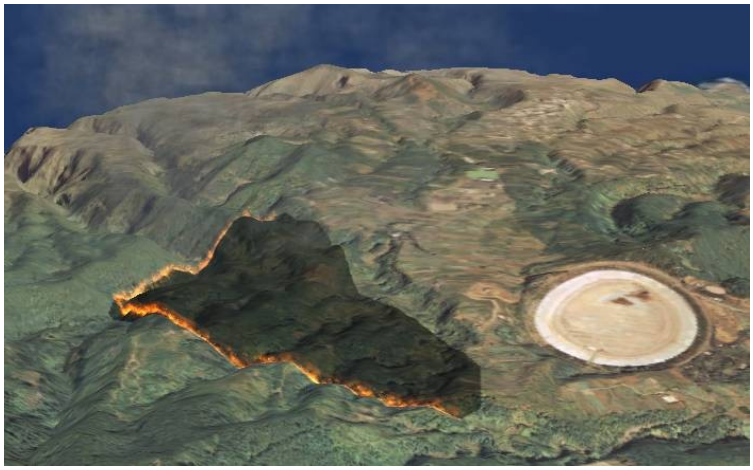

Figura 7: *imagen de un incendio forestal, que ha quemado la zona interior*

A su vez, se creó un método para poder “quemar” la ortofoto, y dar la apariencia visual de por dónde el fuego ha pasado, oscureciendo la región interior, que también se puede apreciar en la misma figura.

Animación temporal

Cualquier objeto insertado en la escena puede estar animado en el tiempo. Para ello, la aplicación maneja internamente un eje temporal, que el usuario puede hacer mover hacia delante o atrás, a mayor o menor velocidad, para ir viendo la evolución.

Donde es más evidente el efecto es en el caso de los incendios. Cuando se establecen los distintos perímetros, cada uno lleva asociado un instante temporal, con lo cual, desde el panel de animación, como se ve en la figura 8, podemos ver la evolución del incendio durante un intervalo de tiempo.

También está pensado para cuando un vehículo tenga una trayectoria previamente grabada, poder recuperar dónde se encontró en cada momento.

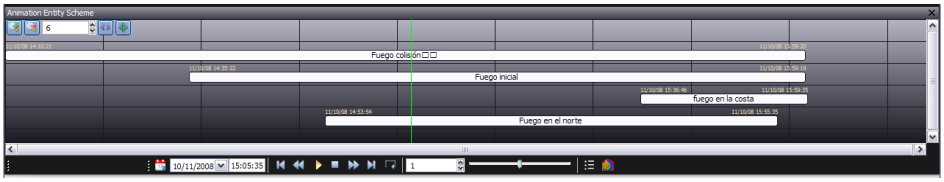

Figura 8: *panel de animación para controlar el instante que se está viendo*

Conexión a servidores WMS

El terreno generado consiste en el modelo digital con la ortofoto incrustada. Pero una vez generado, es posible colocar sobre él todas las capas WMS que se deseen, conectando a uno o varios servidores públicos. A su vez, dichas capas pueden verse de una en una, o bien superpuestas con transparencias sobre el terreno, como se aprecia en la figura 9.

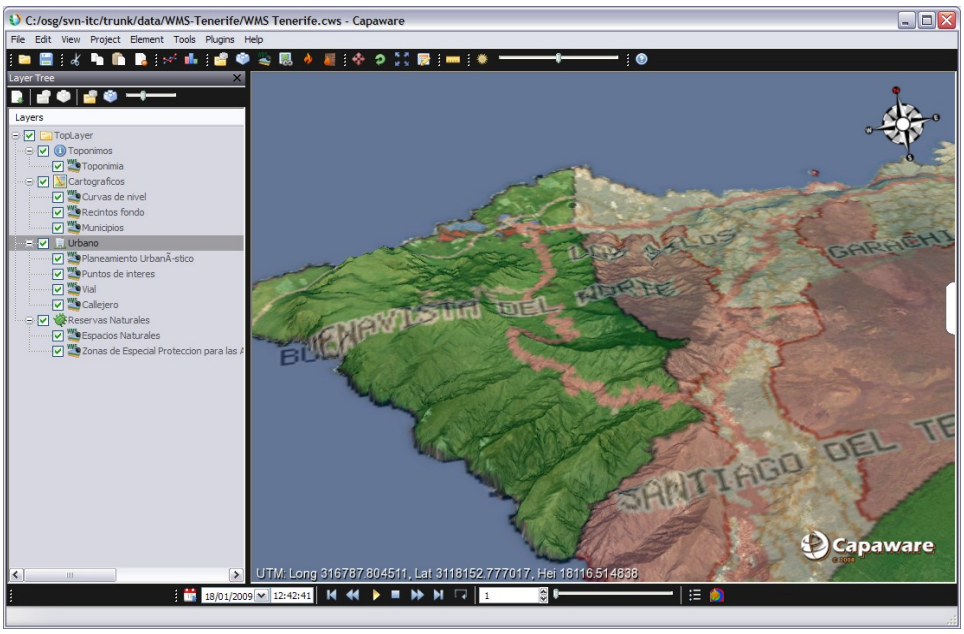

Figura 9: *Vista de varias capas WMS simultáneas*

Internamente hay que mencionar que las peticiones de imágenes individuales a los servidores es totalmente diferente a un software SIG tradicional 2D. Normalmente en estos el usuario recibe una vista inicial del *bounding box*, y luego con herramientas de lupa va acercándose o alejándose de la vista inicial.

Otros más avanzados, como OpenLayers o Google Maps usan sistemas de mosaicos (tiles) pregrabados para lograr más velocidad, y a medida que hacemos zoom o arrastramos la imagen con el ratón, el sistema va haciendo nuevas peticiones ya cacheadas y se van mostrando en pantalla.

Sin embargo, nuestro sistema debe realizar peticiones en función de la posición y orientación de la cámara durante el vuelo (ver figura 5). Se hicieron pruebas con la aplicación libre OssimPlanet [9], basada también en OSG, pero no era muy eficiente. Su comportamiento era muy lento durante la navegación, y pronto fue desechada.

Por este motivo, se desarrolló un algoritmo propio con varios hilos en paralelo que fuera realizando peticiones en tiempo real durante la navegación, empezando por nodos de nivel de detalle más altos (que abarcan un área mayor, para que cuanto antes se cubra una zona extensa de la que se ve en pantalla), y finalizando en los nodos que realmente se están mostrando en pantalla.

Este algoritmo puede generar muchas peticiones por minuto, con lo cual se va procesando también en todo momento la carga de la red para optimizar el número de ellas. Otro aspecto que también optimizamos es el garantizar que nunca haya un nodo sin imagen (aunque sea de poca resolución), para que el resultado visual sea mucho más agradable. Por ejemplo, esto no ocurre cuando en Google Earth se conecta a un servidor WMS, como comentan los autores en [10], donde se nota los saltos entre niveles de detalle.

Conexión entre múltiples usuarios

Otra característica requerida era la posibilidad de que varios usuarios remotos pudieran conectar con una misma escena, y añadir o modificar los elementos en ella, y que el resto de usuarios pudieran ver en tiempo real todos esos cambios a medida que se producían.

La filosofía que elegimos para la transmisión de información es la de *Peer to Pee*r (P2P). Esto favorece que todos los equipos conectados entre sí se comuniquen de igual a igual, sin la necesidad de incorporar servidores que controlen el intercambio de información. Cualquier usuario se puede conectar con cualquier otro usuario directamente y compartir su información. Desde el LayerTree, se pueden seleccionar las entidades a publicar en red.

El módulo de comunicación remota permite que un usuario se pueda conectar a una máquina remota usando la dirección IP del equipo. Una vez indicada la dirección IP, la aplicación descarga las entidades publicadas por el equipo remoto y se las muestra al usuario. Este selecciona las entidades que quiere descargar y la aplicación descarga automáticamente una copia de las mismas y las inserta en su LayerTree.

A partir de ese momento, ambos usuarios comparten esas entidades y las modificaciones que se realicen en cualquiera de los equipos, se transmiten automáticamente al otro usuario. De igual forma, otros usuarios se pueden conectar en cadena a estos usuarios y se compartirán las entidades entre todos ellos, viendo cada uno las modificaciones que realicen los demás.

El equipo guarda un registro con las entidades que son remotas y los equipos que están conectados a éstas. Cuando la aplicación se reinicia, se realiza una resincronización de la información con todos los equipos conectados para tener los datos actualizados.

Todas las modificaciones que se realizan a las entidades se transmiten en tiempo real, definiéndose un protocolo de envío y recepción de datos a través de sockets, de

manera que la comunicación sea lo más eficiente posible. En cada mensaje solo se envía la información del campo de la entidad que ha sido modificada, con lo que las modificaciones llegan prácticamente en tiempo real al resto de los usuarios.

USO DEL SDK

Toda la funcionalidad comentada en la sección anterior se encuentra integrada dentro de la API CPW. La aplicación Capaware consiste básicamente en un ejemplo de llamadas a esta API, utilizando una interfaz de usuario basada en WxWidgets.

Por tanto, existen tres formas de desarrollar a partir del SDK:

- Desarrollando un plugin a través del sistema de plugins que ha sido desarrollado
- Partiendo del código fuente de la aplicación Capaware y añadiendo o modificando la funcionalidad que ya tiene incluida
- Comenzar desde cero creando una nueva aplicación que incluya la API CPW

La más aconsejable es la primera de ellas, ya que es la que más facilidad supone al desarrollador, si bien es verdad que existen ciertas acciones que no se pueden hacer. Los plugins están orientados a añadir una nueva opción en el menú plugin de la aplicación principal, y realizar algún proceso que añada o modifique cualquier elemento de la escena. Toda la documentación concerniente a cómo desarrollar en Capaware se colocará en la página web del proyecto [11].

Ejemplo de plugin: La simulación y predicción de Incendios Forestales

El mejor ejemplo de plugin es el que se ha desarrollado para la simulación de incendios, con el cual se ha construido la aplicación Geviemer ya mencionada [5]. Dicho plugin conecta a un servidor externo donde se halla un motor de simulación llamado Farsite [12], al cual se le pasa la información de los puntos de ignición conocidos en un instante dado, información sobre el tipo de vegetación que hay en la zona, y que a su vez se conecta con un servidor meteorológico para obtener la previsión del tiempo en dicho instante (humedad, temperatura, viento), dando como resultado una serie de perímetros temporales.

Dichos perímetros son mostrados dentro de la aplicación, de forma tridimensional sobre el terreno, y de forma temporal en el panel de animación, para que el usuario puede ver la evolución de la simulación.

TRABAJO FUTURO

Al tratarse de una primera versión, quedan muchas funcionalidades que añadir para futuras versiones. Las más destacadas pueden ser:

- Añadir nuevos estandáres OGC al sistema, además del WMS, como pueden ser WFS, KML, CityGML, WCS, WPS, etc.
- Añadir la posibilidad de asociar datos GPS en tiempo real a cualquier elemento 3D de la escena, de forma que pueda verse su posición en tiempo real
- Teniendo en cuenta que empiezan a surgir servidores públicos con información de las alturas, tratar de generar el terreno en tiempo real mientras se está navegando, al estilo de los visualizadores más populares, pero evitando perder mucho rendimiento con respecto a la situación actual.

CONCLUSIONES

Se ha presentado un sistema de desarrollo de entornos geográficos 3D basado en software libre con grandes capacidades para su uso en aplicaciones concretas y suficientemente genérico para ser adaptado a muchas necesidades. Mediante el uso de plugins, Capaware es capaz de convertirse en un sistema altamente especializado y concreto. Como ejemplo de esta capacidad se ha desarrollado un caso de uso concreto en el campo de la gestión de los incendios forestales con la capacidad de simulación y predicción del comportamiento del fuego. El sistema puede servir de mucha ayuda en el análisis y la toma de decisiones durante un incidente en general.

Se ha dedicado un gran esfuerzo en el diseño e implementación de esta plataforma que posee algunas características notables como su arquitectura robusta, funcional y flexible así como su naturaleza de software libre. Asimismo, se hace uso de estándares como el Open Geospatial Consortium (OGC) y de la infraestructura de datos espaciales de la comunidad canaria para garantizar una plena integración y vigencia de los datos usados.

AGRADECIMIENTOS

Para el desarrollo de este proyecto ha sido necesario el trabajo de los equipos de ambas instituciones (ITC y ULPGC), pero sin duda éste habría sido infructuoso sin la colaboración de diferentes organismos que han apoyado este trabajo. Cabe destacar a la Agencia Canaria de Investigación, Innovación y Sociedad de la Información (ACIISI) del Gobierno de Canarias, la Consejería de Medio Ambiente del Excmo. Cabildo Insular de La Palma, el Centro Coordinador de Emergencias CECOES-112 y Cartográfica de Canarias (GRAFCAN).

REFERENCIAS

- [1] S. STEINIGER Y E. BOCHER, “An Overview on Current Free and Open Source Desktop GIS Developments”, International Journal of Geographical Information Science (Sept. 2008) http://www.geo.unizh.ch/publications/degen/sstein_foss_desktop_gis_overview.pdf
- [2] Crystal Space. http://www.crystalspace3d.org
- [3] M. CASTRILLÓN, E. DELGADO, C. GUERRA, M. PADRÓN, Y. RODRÍGUEZ, J.P. SUÁREZ Y A. TRUJILLO (2006). “Simulated Flight over El Hierro Island”, Computer Graphics and Visualization (CGV 2006) IADIS Virtual Multi Conference on Computer Science and Information Systenms (MCCSIS 2006) http://www.iadis.org/Multi2006/Papers/19/P015_CGV.pdf
- [4] M. CASTRILLÓN, E. DELGADO, C. GUERRA, M. PADRÓN, Y. RODRÍGUEZ, J.P. SUÁREZ Y A. TRUJILLO (2006). "Hierro Virtual: Implementación de un vuelo virtual interactivo sobre la isla de El Hierro", XVIII Congreso Internacional de Ingeniería Gráfica, http://www.ingegraf.es/XVIII/PDF/Comunicacion17213.pdf
- [5] M. CASTRILLÓN, P. JORGE, I. LÓPEZ, A. MACÍAS, D. MARTÍN, R. NEBOT, I. SABBAGH, J. SÁNCHEZ, A. SÁNCHEZ, J.P. SUÁREZ Y A. TRUJILLO (2008). “Aplicación para la gestión de emergencias en la Comunidad Canaria: el caso de predicción y simulación de incendios forestales”, V Jornadas Técnicas de la IDE de España (JIDEE 2008) http://www.orzancongres.com/ideart/064.pdf
- [6] OpenSceneGraph. http://www.openscenegraph.org
- [7] Virtual Terrain Project. http://www.vterrain.org/
- [8] J.C. GARCÍA GONZÁLEZ (2008), “DielmoOpenLidar: Análisis de datos Lidar sobre gvSIG”, IV Jornadas internacionales gvSig, http://www.jornadasgvsig.gva.es/fileadmin/conselleria/Documentacion/4asJornadas/articulos/Sesion4/DielmoOpenLiDAR.pdf
- [9] OssimPlanet. http://www.ossim.org/OSSIM/ossimPlanet.html

- [10] J. ROSALES Y J. RODRIGO (2008), “Difusión de IDECanarias a través del estándar OpenGIS KML Encoding Standard”, V Jornadas Técnicas de la IDE de España (JIDEE 2008), http://www.orzancongres.com/ideart/052.pdf
- [11] Capaware. http://www.capaware.org/
- [12] FarSite. http://www.firemodels.org/content/view/112/143/
- [13] M. MONTESINOS Y J. GASTAR SANZ, “Panorama actual del ecosistema de Software Libre”, I Jornadas de SIG Libre. Gerona, 2006, http://www.sigte.udg.es/jornadassiglibre2007/comun/1pdf/12.pdf