Glob3 Mobile: Sistemas de Información Geográficos 3D en entornos de Movilidad

A. Pedriza(1), M. Citores (1), M. de la Calle(2), D. Gómez(2), A. Trujillo(3), J.M. Santana(3), K. Perdomo(3), J.P. Suárez(4),

(1) COTESA, Área de Sistemas de Información, alfonsopedriza@grupotecopy.es

(2) IGO SOFTWARE, Departamento de I+D - mdelacalle@igosoftware.es

(3) Universidad de Las Palmas de Gran Canaria. Departamento de Informática y Sistemas, atrujillo@dis.ulpgc.es

(4) Universidad de Las Palmas de Gran Canaria. Departamento de Cartografía y Expresión Gráfica en la Ingeniería, jsuarez@dcegi.ulpgc.es

Resumen

El trabajo describe el proyecto de desarrollo de un SIG 3D de código abierto para dispositivos móviles (Apple-iOS y Android) y para navegadores web con tecnología WebGL. En la fase actual, nos centraremos en el diseño e implementación del globo virtual, como elemento esencial que da soporte al SIG 3D y de una IDE que permite la programación de nuevas funcionalidades al globo. Dentro de los objetivos de diseño del globo virtual tenemos (i) simplicidad, con código estructurado que facilita la portabilidad y con una API de código abierto sencilla, (ii) eficiencia, tomando en cuenta los recursos hardware de los dispositivos móviles más extendidos en el mercado, (iii) usabilidad, implementando una navegación intuitiva mediante gestos para la interacción en pantalla y (iv) interoperabilidad, gracias a la incorporación de los estándares OGC. Ante un panorama de clara proliferación de aplicaciones para móviles, Glob3 Mobile pretende ser una apuesta fuerte que llegue a convertirse en un SIG 3D de código abierto que abarque variadas aplicaciones sectoriales.

Palabras clave: SIG 3D, IDE, dispositivo móvil, globo virtual.

1 Introducción

La penetración de los smartphones ha sufrido un fuerte crecimiento a nivel mundial representando el 27% respecto al mercado global y previéndose que evolucionen hasta el 50% a finales del 2012 [1]. En el caso de España este dato es aún más relevante, ocupando el segundo puesto a nivel mundial con un 51% [2].

El rápido avance en prestaciones y servicios de estos dispositivos ha dado un extraordinario impulso a las aplicaciones que sobre ellos se desarrollan. De esta manera es posible encontrar aplicaciones sectoriales muy diversas, en especial en el campo de los Sistemas de Información Geográfica.

Por otro lado la visualización 3D de la información cartográfica cuenta con una gran aceptación entre los usuarios. Destacar la existencia de diversos globos virtuales 3D: Nasa Worldwind [3], WebglEarth [4], ReadyMap, osgEarth y Google Earth.

Sin embargo, uno de los problemas que nos encontramos en el desarrollo de aplicativos para los dispositivos móviles es la multitud de plataformas que existen en la actualidad. Aquellas con mayor penetración en el mercado de la telefonía móvil son iOS (iPhone/iPad), Android, BlackBerry, Windows Phone y Symbian. Cada una de estas plataformas posee su propio entorno de desarrollo y lenguaje de programación.

Una alternativa para el desarrollo multiplataforma de estos aplicativos es el desarrollo de aplicaciones Web que sean soportadas por el navegador de cada uno de estos terminales.

Actualmente no existe ningún globo virtual 3D que funcione en todas las plataformas. El único que posee versiones específicas para iOS, Android y navegadores web es Google Earth pero no es de código abierto ni respeta los principios de la IDE.

Por lo tanto el objetivo del proyecto es el desarrollo de un sistema de visualización de escenas 3D a través de globos virtuales, mediante la generación de un sistema abierto Open Source y asegurando la accesibilidad desde las distintas plataformas de dispositivos móviles existentes [5].

Además merece la pena hacer especial hincapié en acercar los estándares de información recogidos en el Open Geospatial Consortium (OGC) a estas soluciones, con el objetivo de facilitar la interoperabilidad y el acceso a la información proporcionada por las distintas Infraestructuras de Datos Espaciales (IDEs).

2 Glob3 Mobile

Por la naturaleza del proyecto, se ha demostrado necesario tener tres desarrollos en paralelo:

1. Un proyecto en Objective C y C++ con OpenGL ES para dispositivos bajo iOS.
2. Un segundo proyecto en Java con OpenGL ES para Android.
3. Un tercer proyecto en JavaScript con WebGL para navegadores HTML5.

Para dispositivos móviles iOS y Android, Glob3 Mobile se desarrolla directamente en el código nativo de cada dispositivo (Java en Android y C++ en iOS), aumentando así el rendimiento de toda la aplicación, y permitiendo crear aplicaciones completas utilizando el framework de cada plataforma.

Concretamente en la plataforma iOS, no existe un intérprete de Java, no hay soporte WebGL en el navegador Safari para iOS, y no existen versiones de Chrome ni Firefox por el momento. Por lo tanto, es necesario el desarrollo del sistema de forma nativa para su funcionamiento bajo esta plataforma.

Existe una alternativa para Android que consiste en programar en C++ usando el NDK manteniendo las clases principales en java, pero la combinación entre ambos fuentes provocó bastantes problemas en las pruebas iniciales realizadas. Esta alternativa podría parecer la opción más sencilla en un primer momento. Sin embargo, existen ciertas limitaciones en la codificación, compilación y en la depuración que hacen más complicado el proceso de desarrollo.

Los globos virtuales que están escritos con tecnología JavaScript y WebGL pueden funcionar en algunos dispositivos móviles, pero siempre

embebidos dentro de un navegador, y su rendimiento no es del todo eficiente.

WebGL está basado a su vez en la tecnología OpenGL ES 2.0 [6], que es la librería gráfica usada en la mayoría de móviles y tablets actuales. Ambas librerías son muy similares, aunque sólo puede usarse desde JavaScript para poder funcionar dentro del navegador. Existen actualmente numerosos motores basados en WebGL, que permiten programar aplicaciones gráficas a más alto nivel (WebGLU, GLGE, C3DL, Copperlicht). Sin embargo, en este proyecto se ha optado por crear un motor propio empleando directamente WebGL, ya que los motores anteriores no podrían funcionar con OpenGL ES en los móviles.

La Figura 1 muestra la aplicación Glob3 Mobile en iOS y Android.

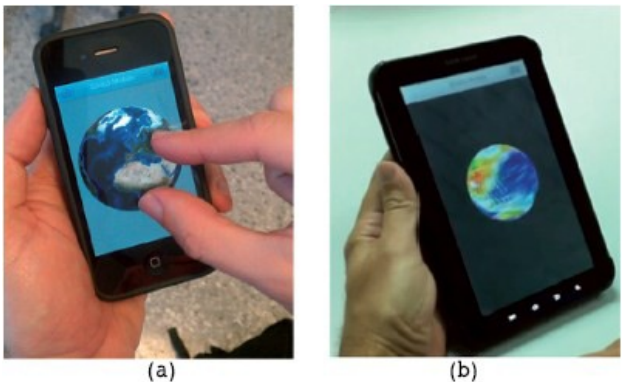

Figura 1. Glob3 Mobile en (a) iPhone de Apple y (b) Samsug TAB.

2.1 Proceso de generación

Aunque mantener tres proyectos de forma paralela exigiría un coste de trabajo muy grande, es imposible programar en un único lenguaje para las tres plataformas planteadas. Por lo tanto, se hace necesario desarrollar en los tres lenguajes de programación.

Para minimizar este problema se ha optado por la siguiente solución:

- El motor del sistema se ha desarrollado inicialmente en C++ estándar. Este motor usará en todo momento clases virtuales para las funciones básicas (acceso al disco, acceso a la red, captura de

eventos, trabajo con la librería gráfica), de tal forma que luego, para cada proyecto específico, se realizarán implementaciones de todas estas clases en el lenguaje correspondiente (Objective C para las clases específicas en iOS, Java para Android y JavaScript para web).
De esta forma, combinando el motor C++ con las clases específicas en Objective-C obtenemos la aplicación nativa para dispositivos iOS.

- A continuación, con este motor escrito íntegramente en C++, se emplea una herramienta software de conversión, que es capaz de convertir código escrito en C++ a código escrito en Java. Si bien hay que usar algunas recomendaciones en el código C++ para que el conversor funcione correctamente.

  Este motor convertido, junto con las clases específicas en Java, constituyen la aplicación nativa para dispositivos Android.

- Finalmente, con el motor escrito en Java, se usa la tecnología GWT de Google para convertir a código JavaScript. Este código, junto con las clases específicas en JavaScript permite obtener la aplicación para navegadores.

  GWT es una herramienta de Google que permite generar código JavaScript a partir de una aplicación escrita en Java.

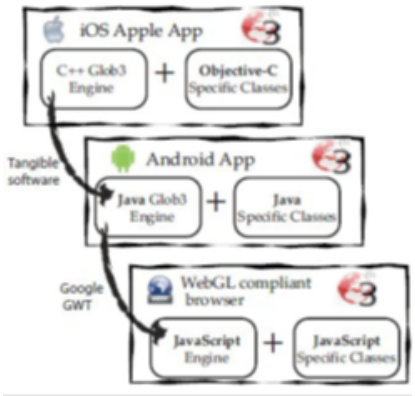

Figura 2. Ciclo de vida de desarrollo de Glob3 Mobile.

La Figura 2 muestra el ciclo de vida del desarrollo del proyecto, donde se indican mediante flechas la migración de código, y los lenguajes de implementación del motor de Glob3 Mobile y de las clases dependientes de cada plataforma.

2.2 Arquitectura del sistema

A continuación se describen los distintos módulos del motor o núcleo del sistema, así como los aspectos más interesantes de su funcionalidad:

- *Ellipsoidal Terrain Engine*: se crea una malla regular de triángulos sobre la superficie del ellipsoide, usando el sistema de proyección WGS84. El sistema debe permitir volar en 3D, hacer drag&drop del globo, hacer zoom, rotar la vista, etc., de forma rápida en cualquier plataforma [7].

- *LOD (Level Of Detail)*: Necesitamos ejecutar una estrategia LOD para representar la superficie con distintos niveles de detalle, atendiendo al criterio de distancia a la cámara, ángulo de inclinación del observador, área en la pantalla, etc).

  Se emplea un algoritmo de niveles discretos llamado Chunk LOD [8]. Para evitar los saltos entre tiles vecinos usamos faldas (skirts), ver Figura 3 de un ejemplo de terreno sin aplicar faldas y a la derecha con su aplicación.

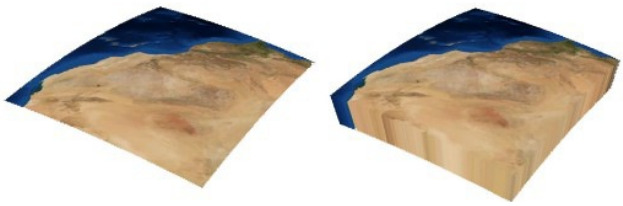

Figura 3. A la izquierda una imagen de textura del globo. A la derecha la misma textura con aplicación de faldas en los bordes.

En la Figura 4 se ilustra la Ilustración de la técnica LOD aplicada en Glob3 Mobile.

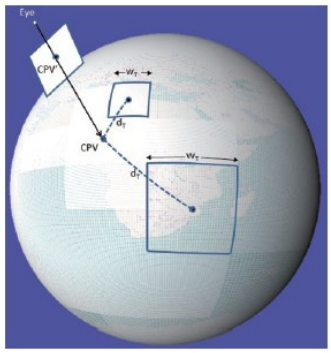

Figura 4: Ilustración de la técnica LOD aplicada en Glob3 Mobile.

- *Out-of-core rendering*: se llama así cuando toda la escena de trabajo no cabe en memoria, con lo cual se mantiene en memoria sólo un subconjunto del terreno (lo necesario para visualizar en cada frame) y hay que traerse desde el disco el resto de la escena a medida que se navega por el globo. Para eso se utilizan políticas de cacheado en disco [9].

- *Conexión a servidores WMS*: para pegar la imagen aérea de cada tile se conecta a servidores WMS (protocolo estándar del OGC) para descargar las texturas de cada tile que se va pintando. El sistema WMS sirve una imagen a partir de las coordenadas de las esquinas del rectángulo (bounding box que se quiere obtener), y por lo tanto las vamos pidiendo a medida que volamos. Hay un montón de servidores públicos, lo que ocurre es que no todos tienen imágenes de todo el mundo.

- *Conexión a los servidores de la NASA*: este servidor cubre todo el mundo con buena resolución (el de Google es mejor pero no es público, sólo a través de google maps o earth). Aunque usa el protocolo WMS para las imágenes más lejanas, usa un sistema de tiles prefijado para las imágenes de más resolución (Virtual Earth), con lo cual se ha desarrollado un módulo para poder realizar las peticiones.

- *Petición de texturas y memoria*: cuando se ha pedido una textura a la red para un tile, se guarda en una cache de disco, para que cuando se necesite otra vez se vaya primero al disco. Tampoco se puede colapsar un servidor pidiendo demasiadas texturas a la vez, y no se pueden mostrar demasiadas texturas simultáneamente, porque no caben en la memoria de la GPU de un móvil. Por estos motivos se ha creado una política de peticiones para poder cumplir todas estas condiciones [10].

- *Ejecución multihilos (multithreading)*: cuando se piden texturas a la red o a la cache se hace en un hilo (thread) aparte, para no interrumpir la tasa de frames por segundo (FPS) del renderizado, con lo cual no se pintan nuevos tiles hasta que no lleguen las texturas.

- *Compresión de imágenes*: para que las texturas ocupen menos espacio se utilizan texturas de 16 bits por cada tile, sin canal alpha. No pueden usarse formatos de compresión ya que el formato manejado por el hardware de los móviles no es el mismo que exporta un servidor WMS, y además el hardware del móvil no puede comprimir y descomprimir en tiempo real a ese formato. Por lo tanto usamos JPG.

- *Manejo de las alturas del terreno*: Para incorporar elevaciones al terreno, se emplea el formato BIL que exportan los servidores de la NASA, que devuelve una matriz de valores dado un contorno o bounding box, que son usados para levantar los vértices de la malla dicha altura sobre la altura del elipsoide y dar la ilusión de elevación del terreno.

- *Capas transparentes*: a veces el usuario quiere mostrar capas WMS con transparencia (como un callejero) sobre la ortofoto, con lo cual deben mostrarse texturas. Normalmente esto se realiza en un equipo sobremesa usando multitextura (dos texturas por tile se le mandan a la GPU y ésta las mezcla). Pero en el caso de un móvil sería demasiada carga de memoria, así que la mezcla se hace en CPU. La Figura 5 muestra un ejemplo de combinación de texturas con transparencias en Glob3 Mobile.

Figura 5: Ejemplo de combinación de texturas con transparencias.

- *Marcadores sobre el terreno*: para poder mostrar una lista de marcadores, que representen posiciones sobre el terreno, se ha implementado usando una técnica de billboards para poder mostrar texturas que tengan tamaño contante en pantalla, y que siempre den la cara al usuario.

3 Manejo de la cache

Otro de los problemas inherentes a los dispositivos móviles es la falta de conexión a la Red. Bien por falta de cobertura o por la ausencia de tarifas de datos contratadas, se producirán multitud de ocasiones en que los usuarios no puedan conectarse a Internet. Por esta razón una de las características fundamentales es el cacheo de la cartografía en el cliente, permitiendo además incrementar el rendimiento.

Glob3 Mobile genera dinámicamente una cache donde va almacenando todas las imágenes que se van descargando, para que cuando haya que volver a usarlas, acuda aquí primero antes de ir a la red.

Sin embargo, cuando se trabaja con JavaScript, no se tiene acceso al disco del cliente, y el navegador maneja su propia cache. En principio esto es una ventaja, porque permite obviar la lógica de preguntar primero antes si está en cache. Sin embargo, ha sido necesario realizar adaptaciones para garantizar que las imágenes que se piden por el protocolo WMS queden almacenadas en la cache del navegador, porque el

comportamiento del sistema de caché es muy variable de un navegador a otro.

Para asegurar que todos los navegadores se comportasen de la misma forma, se forzó a incluir en las cabeceras HTTP el campo “Max-Age”, que indica al navegador que tiene que cachear un recurso y la cantidad de tiempo en segundos de vigencia.

4 Conclusiones y Trabajo Futuro

Disponer de un framework Open Source permite el desarrollo de aplicaciones sectoriales aplicadas a los múltiples campos donde la cartografía y la información geográfica adquieren una especial relevancia. Además dotar a estas aplicaciones de la visualización tridimensional permitirá facilitar la compresión de la información t del entorno, mejorando la experiencia e incrementando el valor ofrecido a los usuarios finales.

Incorporar las fuentes de información y geoservicios a través del uso de estándares OGC garantizará la interoperabilidad y enriquecerá las posibilidades de explotación del sistema. Por este motivo la incorporación de dichos estándares se establece como una premisa en el desarrollo del proyecto. Se persigue la utilización de estándares OGC y formatos de representación de más aceptación en la industria (WMS, WFS, KML).

Glob3 Mobile es un proyecto vivo que crece día a día, por esta razón se está trabajando en la incorporación de los estándares de mayor aceptación en la industria. Ejemplo de ello son los trabajos para la incorporación de WFS, KML y GeoJson, entre otros.

Otro de los requisitos de su diseño es que a corto plazo pueda incluir el acceso a pasarelas a APIs abiertas de Internet (Clima, Geonames, Twitter, Facebook,…), así como la visualización de fotos, vídeos, modelos 3d, etc.

La inclusión de estas características y funcionalidades a las ya existentes, lograrán sin duda constituir al proyecto Glob3 Mobile no sólo como una herramienta de difusión sino como un sistema de uso profesional gracias al uso de estándares y su integración con las IDEs [11].

5 Referencias

2012, MATOS KAPETANAKIS “[Infographic] 100 Million Club – Top smartphone facts and figures in 2011”. Available: http://www.visionmobile.com/blog/2012/02/infographic-100-million-club-top-smartphone-facts-and-figures-in-2011/

2012, Europe, Mobile, U.S. “Smartphone Adoption Approaches Tipping Point Across Markets”. Available: http://www.comscoredatamine.com/2012/02/smartphone-adoption-approaches-tipping-point-across-markets/

2007, D.G. Bell, F. Kuchnel, C. Maxwell, R. Kim, K. Kasraie, T. Gaskins, P. Hogan, and J. Coughlan. Nasa world wind: Opensource gis for misión operations. In Aerospace Conferecnce, 2007 IEEE, pages 1 – 9.

2012, P.Sloup. Webgl earth [online].

2012, Glob3Mobile page project [online]. http://ami.dis.ulpgc.es/glob3m/

2004, Khronos Group. Opengl es: The standars for embeddes accelerated 3d graphics [online]. http://www.khronos.org.

2011, P.Cozzi and K.Ring. 3D Engine Design for Virtual Globes. A. K. Peters, Ltd., Natick, MA, USA, 1st edition.

2002, U.Thatcher. Rendering massive terrains using chunked level of detail control. In Proceedings of the SIGGRAPH.

2011, V. Chandola, R.R. Vatsavai, and B. Bhaduri. Iglobe: an interactive visualization and analysis framework for geospatial data. In Poceeding of the 2nd Inernational Conference on Computing for Geospatial Research & Applications, COM.Geo ’11, pages 21:1-21:6, New York, NY, USA.

2010, J.M. Noguera, R.J. segura, C.J. Ogáyar, and R. Joan- Arinyo. Navigating large terrains using commodity mobile devices. Computers & Geosciences.

2012, A. Trujillo, D. Gómez, M. de la Calle, J.P. Suárez, A. Pedriza, and J.M. Santana. Glob3 Mobile: An open source framework for designing virtual globes on ios and android mobile devices. In 7th International 3DGeoInfo Conference, Quebec, Canada.