IDELab MapstractionInteractive: API Universal y Políglota

P. López Escobés [(1)], *R, García Martín* [(1)] *y J.P. de Castro Fernández* [(2)]

[(1)] Laboratorio de Infraestructuras de Datos Espaciales (IDELab), Escuela Técnica Superior de Ingenieros de Telecomunicación, Campus Miguel Delibes, Universidad de Valladolid, Camino del Cementerio s/n, 47011 Valladolid, plopesc@gmail.com, {ricgar,juacas}@tel.uva.es.

RESUMEN

Continuamente aparecen nuevas plataformas de gestión de cartografía web, con el inconveniente de que cada una de ellas utiliza un API propia. Dada la gran heterogeneidad de APIs de Mapas existente, sería conveniente disponer de una librería de mapas capaz de abstraer al desarrollador de las pequeñas diferencias entre ellas. Este es el objetivo de la librería Javascript de código abierto Mapstraction. Este tipo de API recibe el nombre de «API Universal y Políglota».

Gracias a Mapstraction se pueden desarrollar aplicaciones en las que el usuario puede visualizar la información cartográfica con varios proveedores, pero presenta el inconveniente de no proporcionar mecanismos de creación y/o edición. En este documento se recogen las principales novedades que presenta la librería IDELab MapstractionInteractive, una extensión de Mapstraction que ofrece nueva funcionalidad para solventar las carencias de ésta.

Las nuevas funcionalidades implementadas para los proveedores que se incluyen dentro de la librería brindan al usuario la posibilidad de poder editar y crear geometrías sobre el mapa (puntos, líneas y polígonos). Por otra parte, también se implementan dentro de la librería nuevos eventos para los mapas, de forma que el programador puede tener un mayor control de lo que el usuario hace sobre éstos.

Palabras clave: Mapas, API Universal, Web 2.0, software libre, abstracción, interactividad, Mapstraction.

ABSTRACT

Constantly emerging new platforms for managing web mapping, with the drawback that each uses their own API. Given the great diversity of existing Maps APIs, it is desirable the appearance of a maps library which can abstract the developer from the small differences between them. This is the goal of the JavaScript OpenSource Mapstraction library. This kind of API is called «Universal and Polyglot API».

Thanks to Mapstraction, developers can make their applications where users can visualize cartographic information with several map providers. However, this library doesn't provide mechanisms for creating or editing maps. In this document are compiled the principal improvements provided by the IDELab MapstractionInteractive project, which attempts to address these limitations.
New functionalities implemented give the user the ability to generate new geometries on the map (points, linestrings, polygons) by just clicking on the map, and to edit or delete those in the same way. This project also provides new events with which developer can have greater control over what the user clicks on the map.

Key words: Abstraction, interactivity, maps, Mapstraction, OpenSource, Universal API, Web 2.0.

INTRODUCCIÓN

El fenómeno de la Web 2.0 cada día está más integrado dentro del mundo de Internet y con él, las nuevas posibilidades que ofrece. El usuario puede acceder a un buscador, a un cliente de correo electrónico, a un lector de suscripciones Web o incluso, en los últimos tiempos, a completas aplicaciones de oficina, retoque de imágenes, etc. Además, la integración de la información geográfica y de las tecnologías 3D está empujando el desarrollo tecnológico hacia nuevas y prometedoras formas de mostrar y entender los contenidos que integran los mecanismos sociales de generación de contenidos y las técnicas tridimensionales de modelado del mundo virtual y de la organización de la información.

Uno de los elementos que más éxito ha tenido dentro de esta evolución han sido los *mashups* de mapas [1]. Multitud de iniciativas han surgido para intentar convertirse en el referente a la hora de integrar sus contenidos geográficos dentro de cualquier página Web. Esta competición entre los diferentes proveedores de información geográfica ha traído consigo una simplificación a la hora de ofrecer sus diferentes APIs y servicios adicionales, que unidos a su fácil integración en cualquier página web, han dado lugar a un nuevo término, la *neogeografía* [2].

Un interesante ejemplo es el servicio Panoramio[1], en el que los propios usuarios aportan sus fotografías georreferenciadas en un mapa. Esta aproximación permite que una ingente cantidad de material multimedia se almacene en el servicio, y que la propia comunidad de usuarios añadan valor al servicio. Un mecanismo de búsqueda muy visual y sencilla, basada en un mapa rápido y de calidad, hace que este servicio tenga un éxito de aceptación ejemplar.

En el éxito de este desarrollo también ha influido la disponibilidad de pequeños componentes incrustables en páginas Web (también conocidos como *Web widgets*), y a algunas API (*Application Programming Interface*) comerciales asociadas a los mismos (como las de Google, Yahoo! o Microsoft). Esto permite que, hoy en día, cualquier desarrollador o incluso un usuario sin demasiados conocimientos, puedan incluir en sus sitios Web uno de estos componentes (mapas, galerías de fotos, entradas de una suscripción Web, ventanas de mensajería instantánea, etc.).

El problema que históricamente ha perjudicado el desarrollo y evolución de tecnologías incipientes es la competencia descoordinada entre alternativas

[1] http://www.panoramio.com

tecnológicas. Actualmente hay una serie de APIs para la elaboración de aplicaciones de neogeografía que obligan a los desarrolladores a la adopción de un estilo de programación determinado y una serie de decisiones de diseño que hacen que el resultado esté fuertemente ligado al proveedor original. De esta forma se produce una competición despiadada por imponer un determinado proveedor de *widgets* que a la larga genera usuarios cautivos [3].

Por este motivo surgió la idea de la creación de un API "universal y políglota", que sea el encargado de introducir una capa de abstracción para poder controlar todos los clientes de mapas con un mismo API y de esta forma permitir el intercambio entre los clientes de mapas para poder utilizar el más apropiado en cada caso, sin necesidad de aprender un nuevo API. Este API universal y políglota ya existe y se llama Mapstraction[1], utilizado como base para llevar a cabo este trabajo. Sin embargo el diseño de este API todavía no está definido y todavía son necesarios más esfuerzos de investigación en este sentido.

Ante los nuevos retos que planteaba esta alternativa, se eligió trabajar en el diseño de una extensión que dotara a la librería de la posibilidad de poder interactuar con los mapas, de manera que fuera posible añadir, modificar y borrar geometrías en los mismos de forma dinámica, tan sólo interactuando con el ratón sobre el mapa. Derivados de este objetivo, también se definen otros, cómo la definición de eventos relacionados con estas operaciones o la posibilidad de añadir u obtener las geometrías del mapa mediante objetos de tipo GeoJSON[2].

En este documento se estudia en primer lugar la estructura y el estado del desarrollo de la librería Mapstraction, para conocer sus principales puntos fuertes y debilidades. A continuación se describe el proceso de implementación de la extensión sobre la que se centra este documento, para luego mostrar los resultados de las pruebas de compatibilidad y uso de la extensión. En la siguiente sección se muestran otros retos que se plantean a la librería para poder mejorar sus funcionalidades y hacer que aumente su notoriedad. Finalmente se muestran las conclusiones obtenidas tras la realización de este trabajo.

EL ESTADO ACTUAL DE MAPSTRACTION

En esta sección se pretende mostrar el estado de la técnica en primer lugar de los clientes de mapas en general para después adentrarse en el mundo de las APIs universales y políglotas. Este concepto de API universal y políglota es de nueva acuñación y está estrechamente relacionado con la neogeografía y su concepto de alta usabilidad.

El término neogeografía viene a definir aquellas herramientas y técnicas geográficas utilizadas para realizar actividades personales o por un grupo de usuarios no expertos en el análisis geográfico, ya que su fin inicial es el uso informal y no analítico. Es una expresión, en su más reciente sentido, que deriva del nuevo uso que se está dando a cartografía en Internet por parte de usuarios masivos. Esta nueva geografía es el resultado de la libertad de acceso a la georreferenciación de lugares, la geoetiquetación de contenidos, la fácil integración de recursos en entornos web mediante el uso de APIs y la utilización cada vez más cotidiana de GPS y de aparatos de posicionamiento (teléfonos móviles, PDAs, navegadores) en la vida cotidiana.

Al albor de esta tendencia de generación de *mashups* basados en mapas, han surgido gran cantidad de clientes ligeros incrustables en cualquier Web. Esta

[1] http://www.mapstraction.com
[2] http://www.geojson.org

proliferación de clientes ha desencadenado una feroz batalla por hacerse con un hueco en el mercado, en función de los servicios que deseen ofrecer. Sin embargo, la mayoría de los creadores de *mashups*, suelen optar por la solución más conocida y utilizan el API de Google Maps, que es de gran calidad y ofrece multitud de servicios, pero con la limitación de tratarse de una solución propietaria [3]. Como se puede ver en la página de la fundación OSGeo [4], aparecen hasta 24 alternativas diferentes *OpenSource,* si a esto se añaden las alternativas propietarias como las de Google o Microsoft, las opciones son muy numerosas. Sin embargo, la fragmentación de los esfuerzos en el desarrollo de aplicaciones diferentes, pueden dañar la evolución de esta rama de la técnica.

La problemática reside en la complejidad de conseguir que, un programador acostumbrado a utilizar determinado cliente, cambie de cliente para cada aplicación concreta, así como que un usuario doméstico que realiza sus *mashups* como *hobby* aprenda el funcionamiento de múltiples API.

Toda esta problemática hizo que surgiera la idea de generar un API universal y políglota que fuera capaz de integrar los distintos clientes de mapas. De esta forma sólo sería necesario conocer el funcionamiento de ese API, con la posibilidad de utilizar cualquiera de los clientes disponibles. Para dar respuesta a estos interrogantes surgió Mapstraction [5].

Mapstraction es una librería Javascript para la abstracción de mapas. Ésta proporciona un API único para diferentes clientes de mapas Javascript. Mediante el uso de esta librería, el creador de *mashups* de mapas puede implementar sus aplicaciones tan solo una vez con la posibilidad de cambiar de proveedor de mapas de forma sencilla, si así se requiere.

Además, Mapstraction trata de cubrir ciertas deficiencias que muestran algunos de los clientes, para normalizar el comportamiento de todos ellos. Los clientes principales que se incluyen en la versión estable de Mapstraction son: Poly9Globe[1], Google Maps[2], Map24[3], MapQuest[4], Microsoft Bing Maps[5], MultiMap[6], OpenLayers[7], OpenSpace[8], OpenStreetMap[9], ViaMichelin[10] y Yahoo Maps[11].

Mapstraction en su versión original estaba muy limitada porque su arquitectura era bastante inmadura y resultaba tediosa la colaboración con el proyecto. Sin embargo en los últimos meses, la librería está sufriendo un nuevo empuje y se está trabajando en la refactorización de la misma para darle la orientación a objetos de la que carecía inicialmente [6]. Con esta refactorización es muy sencillo añadir nuevos proveedores, así como incluir nuevas funcionalidades a la librería.

Mapstraction es una iniciativa *OpenSource* y que se distribuye con licencia BSD. Por el momento no hay una *release* estable de la nueva versión refactorizada, pero se pueden descargar las versiones que hay en el repositorio de la librería. Se puede colaborar integrando nuevos clientes de mapas que hasta ahora no están incluidos en

[1] http://globe.poly9.com
[2] http://maps.google.es
[3] http://www.map24.com
[4] http://www.mapquest.es
[5] http://www.bing.com/maps
[6] http://www.multimap.com
[7] http://openlayers.org
[8] http://www.openspaceworldmap.org
[9] http://www.openstreetmap.org
[10] http://www.viamichelin.es
[11] http://maps.yahoo.com

la librería, o añadiendo funcionalidades GIS que por el momento no se incluyen pero sería interesante que lo hiciese.

De esta forma, ya se comienza a vislumbrar una nueva manera de generar *mashups* de mapas independientes del tipo de mapa que se utilice, lo que dará una nueva visión de este campo y fomentará la utilización de otros clientes además de los más habituales.

Sin embargo, en esta nueva arquitectura de Mapstraction no se implementa nueva funcionalidad en la librería, por lo que sigue siendo un tanto limitada. Por ello, desde el IDELab (Laboratorio de Infraestructuras de Datos Espaciales) de la Universidad de Valladolid, se está colaborando con la comunidad a través del diseño de la extensión *IDELab MapstractionInteractive*.

IMPLEMENTACIÓN

Para llevar a cabo esta extensión, se siguió un proceso que consistió en primer lugar en definir las operaciones que se iban a proporcionar, los eventos asociados y el ciclo de vida de las geometrías de los mapas. Posteriormente se diseñó el sistema para conseguir que esta extensión fuera universal y pudiera ser utilizada de forma independiente por cualquier proveedor integrado en Mapstraction.

La nueva interfaz

El diseño de la interfaz que se ofrece en esta extensión se ha llevado a cabo según los requerimientos que surgen a la hora interactuar con el mapa. De esta forma, se definen los siguientes métodos:

- **addFeature(featureType,data)**: Con este método se activa el modo de incluir nuevas geometrías en el mapa haciendo clic sobre el mismo. Como parámetro se le pasa el tipo de geometría a añadir (*marker*, *linestring*, *polygon*) y los parámetros de la misma (color, anchura, opacidad, etc.).
- **activateEdition()**: Con este método se activa el modo de edición de geometrías. Se permite interactuar con las ya existentes en el mapa pudiendo modificarlas o borrarlas.
- **deactivateEdition()**: Este método es el opuesto al anterior, al ejecutarlo, las geometrías permanecen inalterables y no pueden ser modificadas ni borradas.
- **updateProprietary()**: Con este método se reflejan en el objeto *Javascript* los cambios que se hacen en la geometría que el mapa tiene asociada.
- **addJSON(GeoJSONObject)**: Con este método se permite añadir nuevas geometrías al mapa a partir de un objeto de tipo GeoJSON.
- **getJSON()**: Con este método se obtienen un objeto de tipo GeoJSON en el que se recogen todas las geometrías incluidas en el mapa.

Con esta nueva interfaz es posible proporcionar todos los servicios deseados por la extensión. Para poder controlar las nuevas operaciones que el usuario puede realizar, es necesario también proporcionar nuevos eventos que notifiquen las acciones que se llevan a cabo sobre el mapa.

Los nuevos eventos

Para poder controlar las acciones que ahora el usuario es capaz de llevar a cabo sobre el mapa, surge la necesidad de añadir nuevos eventos a la lista ya existente. Los eventos que se han añadido son los siguientes:

- **markerChanged** y **polylineChanged**: Se genera al modificar una de las geometrías presentes en el mapa.

- **markerSelected** y **polylineSelected**: Se genera cuando una geometría es seleccionada haciendo clic en ella.
- **markerUnselected** y **polylineUnselected**: Se genera al deseleccionar una geometría que esté seleccionada o al seleccionar otra diferente mientras ésta esta seleccionada, dejando de estarlo.

El ciclo de vida de las geometrías

Una vez definidos los métodos y los eventos que se implementan en la extensión, se debe desarrollar el ciclo de vida que tendrán las geometrías. Para diseñar este *workflow* de forma más sencilla, se definen, en primer lugar, por separado las diferentes funcionalidades requeridas, antes de integrarlas en un esquema más complejo:

- Activar un modo de edición que mientras esté activo permita realizar las siguientes funciones:
  - Al seleccionar una geometría se genere el evento de selección.
  - Al seleccionar otra o deseleccionarla, se genere el evento de selección.
  - Si mientras está seleccionada se pulsa *Supr* o *Del*, se elimine, generándose el evento correspondiente.
  - Mientras esté seleccionada se pueda modificar, generándose el evento de modificación.
- Desactivar este modo de edición, dejando de estar disponibles todas las funcionalidades anteriormente listadas.
- Activar un modo de edición en el que sea posible añadir nuevas geometrías, generándose el evento correspondiente.

Una vez definidas las funciones, se definen los estados del *workflow*. A tenor de lo anterior, se pueden definir los siguientes estados:

- **Latente**: Cuando el modo de edición está desactivado.
- **Activo**: Cuando el de edición está activado, pero no es la geometría seleccionada.
- **Seleccionado**: Cuando es el nodo seleccionado y se puede actuar sobre él.

A continuación se definen las transiciones, indicando de que estado a qué estado se llevan a cabo y los eventos que se generan. En la Tabla 1 se puede observar el resumen de estas transiciones.

Tabla 1: La tabla de estados del ciclo de vida de las geometrías

Desencadenante	Estado Inicial	Estado Final	Evento
Crear Geometría	----------	Latente / Activo	Creación
Activar edición	Latente	Activo	----------
Seleccionar Geometría	Activo	Seleccionado	Selección
Deseleccionar Geometría	Seleccionado	Activo	Deselección
Eliminar Geometría	Seleccionado	----------	Borrado
Modificar Geometría	Seleccionado	Seleccionado	Modificación
Desactivar edición	Activo / Seleccionado	Latente	----------

Una vez definidos los estados y las transiciones, ya queda definido el ciclo de vida que tendrán las geometrías de la librería desde que son creadas hasta que son eliminadas del mapa. Este ciclo de vida se puede observar en la Figura 1.

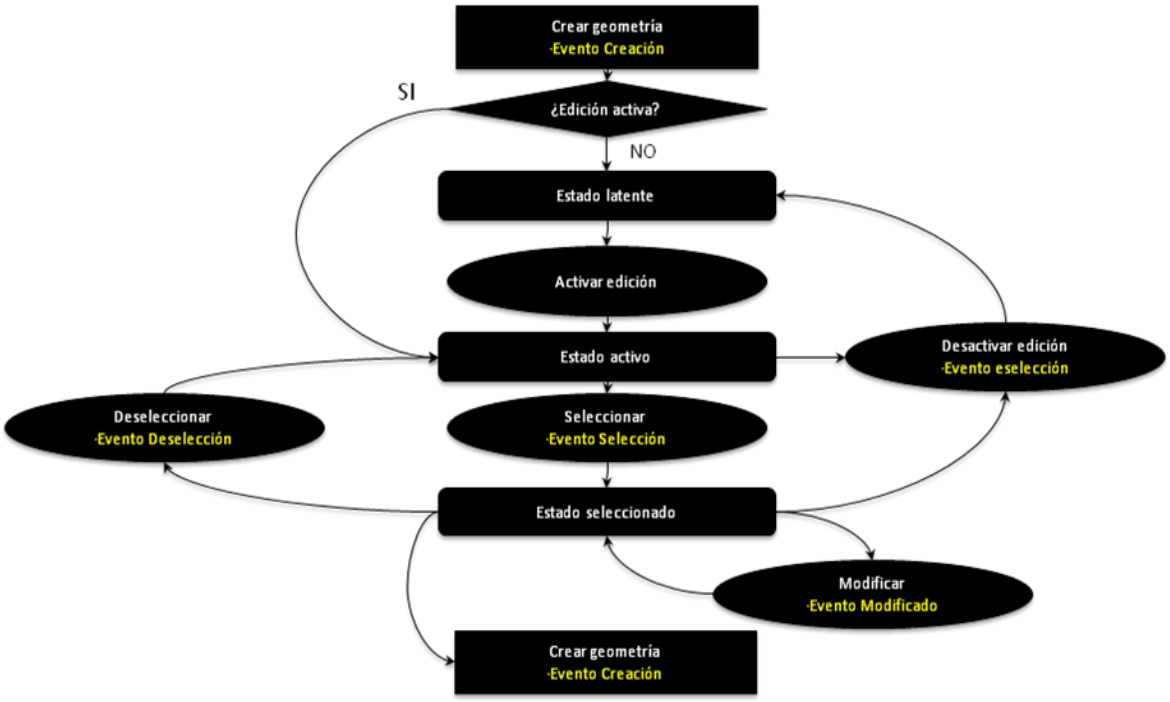

Figura 1: Ciclo de vida de las geometrías del mapa

La interactividad universal

Para poder llevar a cabo esta implementación de forma universal, el bloque en el que se implementan estos métodos debe utilizar únicamente métodos pertenecientes al API de Mapstraction. De esta manera, una única implementación es capaz de proporcionar de forma transparente el mismo servicio a cualquiera de los proveedores que está integrado dentro de la librería. En la Figura 2 se puede observar un esquema de cómo ha sido implementada esta funcionalidad.

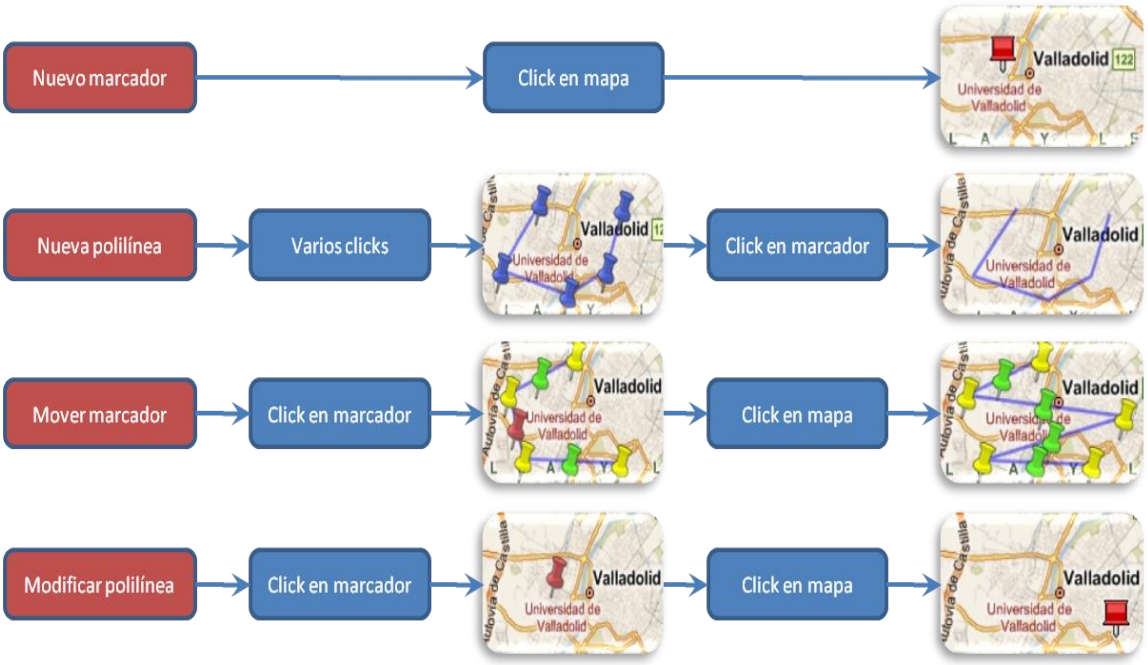

Figura 2: Ciclo de vida de creación y edición de geometrías

Sin embargo, hay clientes de mapas que ofrecen de forma nativa la interactividad sobre los mapas. Dado que esta forma nativa es más natural para los clientes que la que se diseña en el bloque universal, se opta por una doble versión. Para los clientes que ofrecen de forma nativa estas funcionalidades, se pueden diseñar implementaciones propias, mientras que se mantiene el bloque universal para el resto de los proveedores que no la implementen.

RESULTADOS

Una vez llevada a cabo la implementación anterior se han llevado a cabo diversas pruebas para comprobar el buen funcionamiento de la propuesta. Se ha verificado el correcto funcionamiento de esta implementación mediante el bloque de funcionalidad universal para los proveedores Yahoo! Maps y Bing Maps, mientras que se han implementado bloques propios para los proveedores Google, OpenLayers y Cartociudad. Los resultados han sido los esperados y las pruebas han dado resultados satisfactorios en todos los casos.

Dado que se trata de una librería Javascript y que este lenguaje es interpretado en el navegador, cabe la posibilidad de que se muestren comportamientos erróneos en alguno de los navegadores. Se ha comprobado que la aplicación funciona correctamente en los navegadores Web Firefox 3, Internet Explorer 8 y Chrome. Sin embargo en Ópera 10.10 se observa un comportamiento defectuoso con el proveedor Google Maps. Se ha comprobado, no obstante, que este fallo no es de la librería, sino del navegador, ya que este error se da en todas las aplicaciones del mismo.

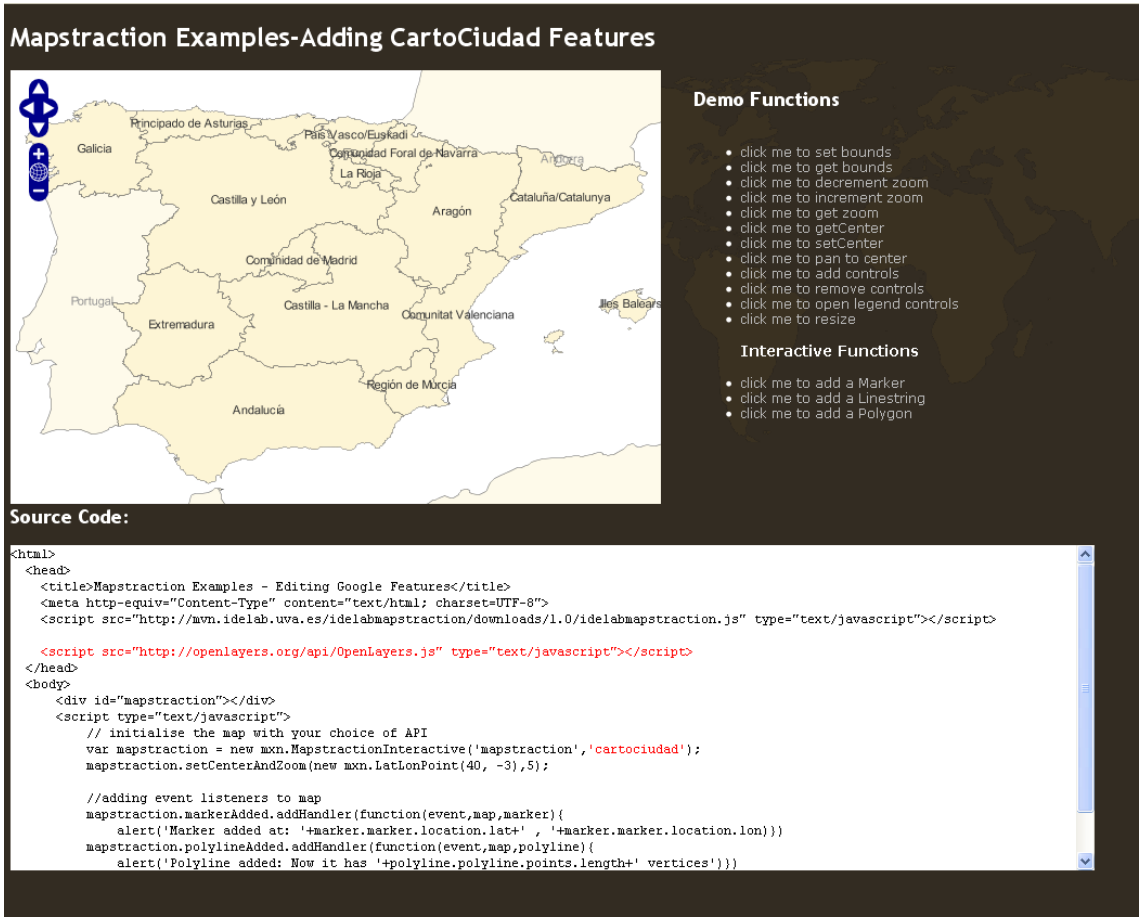

Figura 3: Página de demostración de la librería IDELab MapstractionInteractive

Se ha publicado un demostrador[1] desde el que se pueden comprobar todos los casos de prueba realizados y disfrutar de la nueva experiencia de uso que ofrece esta extensión. En estos ejemplos, se puede ver el código fuente empleado para generar cada uno de ellos. En este se puede observar cómo *IDELab MapstractionInteractive* proporciona un nivel superior de abstracción, de forma que todos los ejemplos se ejecutan con el mismo código fuente. La única diferencia a la hora de generar el mapa reside en la especificación del nombre del proveedor de mapas a utilizar en cada caso. En la Figura 3 se muestra una captura de estas páginas de demostración.

Los resultados de todas estas pruebas han sido satisfactorios. Se ha realizado un gran esfuerzo en conseguir que, cuando un desarrollador integre un nuevo proveedor en la librería, pueda integrar estas nuevas funcionalidades sin trabajo adicional. En caso de que incluya de forma nativa la interactividad, será decisión suya optar por la definición de un nuevo bloque propio o la reutilización del ya existente.

TRABAJO FUTURO

A la vista de los buenos resultados obtenidos con esta extensión se están considerando nuevas líneas de trabajo que ayuden a conseguir que Mapstraction sea un referente a la hora de generar *mashups* de mapas. Su principal reto es seguir atrayendo la atención de los desarrolladores de estos. Haciendo que su uso sea cada vez más generalizado, será más sencillo que la comunidad adopte esta librería como

[1] http://mvn.idelab.uva.es/idelabmapstraction/interactive_examples/

la básica para su construcción. Los desarrolladores que utilicen clientes todavía no incluidos en la librería tratarán de incluirlos para no quedarse fuera de esta iniciativa, aumentando la universalidad de la librería.

Otro reto que se le presenta a Mapstraction es el de los globos virtuales. Estos clientes ofrecen una experiencia más espectacular que los mapas tradicionales y con ellos puede obtenerse una visualización mucho más exacta de la orografía del terreno. Sin embargo tienen más parámetros a gestionar y más casos de uso por definir, y con el API actual de Mapstraction no se pueden controlar todos ellos. Aunque se están realizando intentos para integrar globos virtuales dentro de Mapstraction, éstos tienen funcionalidades muy básicas y no se aprovecha toda su potencia. Resulta muy interesante, por tanto, el reto de añadir éstas nuevas funcionalidades específicas para la integración de globos virtuales dentro de Mapstraction.

Además de la integración de nuevos clientes de mapas y el reto de los globos virtuales, otro reto que se plantea es la integración dentro de la librería de funcionalidades genéricas GIS para hacer que los desarrolladores puedan utilizar formatos estandarizados. De este modo se conseguiría, de forma transparente, la unión entre la librería y las fuentes de datos estandarizados. En este sentido se han hecho algunos avances, como puede ser la utilización de formatos GeoJSON tanto para añadir nuevas geometrías a los mapas como para extraer la información de las que hay en los mapas. Sin embargo, todavía quedan varias deficiencias por cubrir en este sentido.

Finalmente, para una mejor integración de Mapstraction dentro de las páginas Web, convendría presentar los mapas de la librería integrados dentro de un componente, como sucede en las alternativas de GeoExt[1] o GeoMajas[2]. Para llevar a cabo este trabajo, se contempla la utilización del *framework Google Web Toolkit* (GWT)[3] [7].

Como se puede observar, las alternativas para la mejora de este API son muy variadas, lo que puede dar lugar a numerosos retos de investigación. Los retos que se pretenden solventar son muy ambiciosos y todavía queda mucho camino por recorrer.

CONCLUSIONES

Con el auge del fenómeno de la Web 2.0 se ha generado una gran comunidad de usuarios, que a la vez son desarrolladores de páginas Web utilizando las distintas alternativas que se ofrecen, tanto libres como comerciales. En este contexto, los *mashups* de mapas han adquirido una importancia inusitada, haciendo que aparezcan gran cantidad de alternativas con orientaciones muy diferentes.

La aparición de tantas y tan dispares alternativas, puede hacer que los desarrolladores se saturen y decidan centrarse en diseñar sus aplicaciones utilizando tan sólo una de ellas. En este contexto las APIs universales y políglotas, como Mapstraction ofrecen una alternativa muy útil, ya que permiten al desarrollador de *mashups* de mapas implementar sus aplicaciones tan sólo una vez con la posibilidad de cambiar de proveedor de mapas de forma sencilla si así se requiere, sin necesidad de variar el código de la aplicación.

La extensión *IDELab MapstractionInteractive* permite dotar a los mapas proporcionados por la librería Mapstraction de una interactividad de la que anteriormente carecían. De esta manera se da un impulso a la librería, con la que se pueden ofrecer nuevos servicios que anteriormente no podían ser implementados, por lo que era necesario utilizar un API concreta.

[1] http://www.geoext.org
[2] http://www.geomajas.org
[3] http://code.google.com/intl/es/webtoolkit/

Existen diversos retos a los que esta librería puede enfrentarse en un futuro cercano para convertirse en referente a la hora de la generación de *mashups* de mapas. La solución a estos retos puede hacer que la librería adquiera una importancia mucho mayor que la actual y atraer a nuevos desarrolladores.

El proceso de creación de un API universal y políglota es una tarea de investigación no resuelta como puede ser el desarrollo de ontologías en otras áreas científicas o tecnológicas, lo que hace que sea una vía de investigación atractiva. El hecho de que esta iniciativa esté incluida dentro de un proyecto *OpenSource* invita a que muchos desarrolladores interesados puedan probarla y ofrecer realimentación, así como unirse y colaborar incluyendo nuevos proveedores o funcionalidades.

AGRADECIMIENTOS

El desarrollo de este trabajo ha sido posible gracias a la financiación por parte del Instituto Geográfico Nacional en el marco del Proyecto Conjunto al amparo del convenio de colaboración entre la dirección general del Instituto Geográfico Nacional y la Universidad de Valladolid en su edición de 2009.

REFERENCIAS

[1] J. Yu, B. Boualem, F. Casati, y F. Daniel, “Underestanding mashup development,” *IEEE Internet Computing*, vol. 12, Sep. 2008, págs. 44-52.
[2] A. Turner, *Introduction to Neogeography*, O'Reilly, 2006.
[3] William Cartwright, “Google Maps and mobile devices: Can just one generic design work?.”
[4] “OSGeo.org | Your Open Source Compass.”
[5] “Mapstraction - a javascript library to hide differences between mapping APIs.”
[6] P. López Escobés, *Aplicación de técnicas de neogeografía en un sistema de gestión de contenidos web*, Universidad de Valladolid, 2009.
[7] Pieter De Graef, “Web GIS: from Javascript to GWT,” 2009.