UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA
Escuela de Ingeniería Informática

Trabajo Final de Grado en Ingeniería Informática

GESTOR WEB DE RECURSOS GEOESPACIALES CON TECNOLOGÍA OPENLAYERS

INFORMACIÓN

Autor Marco Jesús Umpiérrez Rodríguez

Tutor Agustín Trujillo Pino - profesor Titular de Universidad del área Ciencias de la Computación e Inteligencia Artificial

Grado en Ingeniería Informática

Curso 2013/2014 - *Convocatoria* Julio - extraordinaria

Universidad de Las Palmas de Gran Canaria

AGRADECIMIENTOS

Me gustaría expresar mi agradecimiento a varias personas que me han prestado su ayuda y apoyo durante la realización de este Trabajo de Fin de Grado.

A mi tutor Agustín Trujillo Pino, por darme la oportunidad de realizar este trabajo, guiarme en su desarrollo, aconsejarme y animarme a seguir adelante. Igualmente me gustaría agradecer a Francisco Pérez Rosales, director de seguridad corporativa de la ULPGC, por las aportaciones hechas al proyecto durante su desarrollo para que éste estuviera más completo.

Dar las gracias también a mis amigos y compañeros, por su interés y sus ánimos durante toda esta travesía. Agradecer especialmente a Desirée de las Nieves Cruz Godoy, autora del trabajo Herramienta de ayuda para la gestión de recursos en aplicaciones geoespaciales, que me ofreció su ayuda y consejos en los inicios de este trabajo a través de su experiencia realizando el suyo.

Finalmente desearía agradecer enormemente el apoyo incondicional y constante de mis padres y hermanos que han estado ahí para mí desde el principio y sin cuyos ánimos habría resultado mucho más complicado finalizar este proyecto.

A todos, muchas gracias.

RESUMEN

El objetivo de este Trabajo Final de Grado (TFG) es la creación de un prototipo de aplicación web para la gestión de recursos geoespaciales. Esta propuesta surgió a partir de la necesidad de disponer de una herramienta que no tuviera que ser instalada en un dispositivo, sino servida por un servidor web, permitiendo su acceso desde cualquier parte y dispositivo.

El resultado fue el Gestor Web de Recursos Geoespaciales con Tecnología OpenLayers, una aplicación que combina diversas herramientas (OpenLayers, GeoServer, PostgreSQL, jQuery…) – todas ellas basadas en Software Libre – para cumplir funcionalidades como la creación de primitivas vectoriales sobre un mapa, gestión y visualización de la información asociada, edición de estilos, modificación de coordenadas, etc. siendo todas éstas funcionalidades características de un Sistema de Información Geográfica (SIG) y ofreciendo una interfaz de uso cómoda y eficaz, que abstraiga al usuario de detalles internos y complejos.

El material desarrollado dispone del potencial necesario para convertirse en una solución a las necesidades de gestión de información geoespacial de la ULPGC, especialmente en el campus de Tafira, sobre el que se ha ejemplificado su uso. Además, a diferencia de las herramientas ofertadas por empresas como Google o Microsoft, esta aplicación está por completo bajo una licencia GNU GPL v3, lo que permite que se pueda indagar dentro de su código, mejorarlo y añadir funcionalidades a cualquier persona interesada.

ABSTRACT

The objective of this Final Project Degree is the creation of a web application prototype for managing geospatial resources. This proposal arises from the necessity of having a tool which is served by a web server instead of being installed in a device, allowing its access from anywhere and from any device.

The result was a Web Application Manager for Geospatial Resources with OpenLayers Technology, an application which combine different tools (OpenLayers, GeoServer, PostgreSQL, jQuery...) based on Open Source, to satisfy the characteristic functionalities of a Geographic Information System (GIS) such as creating vectorial features on a map, management and display of associated information, editing styles, modifying coordinates, etc. All of these offered by a comfortable and efficient user interface.

The material developed has the potential to become a solution to the geospatial information management needs of the ULPGC, especially in the campus of Tafira, which has been used as an example along this project. Also, unlike the tools offered by companies like Google or Microsoft, this application is under a GNU GPL v3 license, which allows people to learn its code, improve it and add to it new functionalities.

TABLA DE CONTENIDO

AGRADECIMIENTOS **1**

RESUMEN **2**

ABSTRACT **3**

TABLA DE CONTENIDO **4**

1.INTRODUCCIÓN **7**

1.1 ESTADO ACTUAL 7

1.2 OBJETIVOS 8

2.COMPETENCIAS **10**

IS03 10

IS04 10

CP06 10

SI03 11

TI06 11

3.PLIEGO DE CONDICIONES **12**

3.1 OBJETO DE ESTE PLIEGO 12

3.2 PLIEGO DE CONDICIONES GENERALES 12

3.3 PLIEGO DE ESPECIFICACIONES TÉCNICAS 12

3.3.1 Especificaciones de materiales y equipos *12*

3.3.2 Hardware *13*

3.3.3 Software *13*

3.3.4 Especificaciones de ejecución *13*

3.4 PLIEGO DE CLÁUSULAS ADMINISTRATIVAS 14

3.5 LICENCIA 15

4.APORTACIONES **16**

5.NORMATIVA Y LEGISLACIÓN **18**

5.1 LEY ORGÁNICA DE PROTECCIÓN DE DATOS 18

5.2 DIRECTIVA 95/46/CE 18

5.3 LICENCIAS 20

5.3.1 GNU GPL *20*
5.3.2 BSD *20*
5.3.3 Apache 2.0 ..... *21*
5.3.4 MIT o X11 ..... *21*

6.REQUISITOS 23

6.1 HARDWARE 23
6.2 SOFTWARE 23

7.METODOLOGÍA Y PLAN DE TRABAJO 25

7.1 PLAN DE TRABAJO 25

8.CONCEPTOS 27

8.1 DATOS GEOREFERENCIADOS 27
8.2 INFORMACIÓN RÁSTER 27
8.3 INFORMACIÓN VECTORIAL 28
8.4 BASES DE DATOS ESPACIALES 30
8.5 PROTOCOLOS 30
8.6 SERVIDORES DE MAPAS 32

9.HERRAMIENTAS 33

9.1 GEOSERVER 33
9.2 POSTGRESQL 34
9.3 POSTGIS 37
9.4 OPENLAYERS 38
9.5 JQUERY 40

10.DESARROLLO 41

10.1 ANÁLISIS DEL PROBLEMA 41
10.2 DISEÑO E IMPLEMENTACIÓN 43
10.2.1 Visualización de Capas en OpenLayers *43*
10.2.2 Visualización y Control de Capas WFS-T *45*
10.2.3 Eventos de OpenLayers *49*
10.2.4 PopUps y Gestión de Usuarios *51*
10.2.5 Rediseño de la Interfaz *53*

10.2.6 Tratamiento de datos no Georeferenciados *54*

10.2.7 Estilos *55*

10.2.8 Correcciones y Mejoras *56*

10.2.9 Funcionalidades Adicionales: Añadir iconos *57*

10.2.10 Funcionalidades Adicionales: Añadir Capas *59*

10.3 PRUEBAS Y RESULTADOS 62

11.CONCLUSIONES 64

12.TRABAJOS FUTUROS 65

13.MANUAL DE USUARIO 67

13.1 INTERFAZ 67

13.2 SECCIÓN DE AYUDA 68

13.3 VISUALIZACIÓN DE CAPAS 68

13.4 SECCIÓN DE ATRIBUTOS DE PRIMITIVAS 69

13.5 OPCIONES DE CONTROL 69

13.6 CONTROLES DE EDICIÓN 71

14.BIBLIOGRAFÍA 74

14.1 HERRAMIENTAS 75

1.INTRODUCCIÓN

Los Sistemas de Información Geográfica (SIG o GIS en inglés) son herramientas que permiten la gestión de información geográficamente referenciada, o sea, información asociada a unas coordenadas geoespaciales. En la actualidad su uso está muy extendido; desde las grandes corporaciones hasta las personas individuales hacen uso de esta tecnología constantemente, facilitándoles tareas que de otra forma serían muy tediosas de manejar y que requerirían una importante inversión de recursos y tiempo.

La aparición de los dispositivos móviles y la mejora de las redes de comunicación han incrementado aún más su importancia, permitiendo que cualquiera pueda realizar consultas geográficas en cualquier momento, desde conocer su ubicación exacta hasta localizar el restaurante de comida tailandesa más cercano.

Por todo ello, resulta crucialmente importante hacer hincapié en este tipo de tecnologías, mejorando sus capacidades y su utilidad.

Figura 1: Captura del servidor de aplicaciones de mapas Google Maps

1.1 Estado Actual

Actualmente y salvo que se trate de una empresa interesada en un desarrollo de un SIG propio, los SIGs más utilizados son los proporcionados por Google y Microsoft – Google Maps (ver *Figura 1*) y Bing respectivamente –. Sin embargo, y como resulta

evidente, sus servicios no son gratuitos; ofrecen una serie de servicios estándar para todo el mundo con resoluciones o número de solicitudes limitadas de manera gratuita pero cuando se quiere superar estos límites o se necesitan otros recursos hay que comprar un plan de pago.

Otra desventaja a tener en cuenta es la imposibilidad de modificar su código base. Ofrecen una Interfaz de Programación de Aplicaciones (IPA o API en inglés) con ciertas funcionalidades pero no permiten cambiar su código interno, lo que puede ser una limitación para cualquiera que tenga intereses que no estén contemplados en dicha API.

Es por ello que algunas compañías decidieron ofrecer una alternativa de código abierto como es el caso de OpenLayers [Ho4], una herramienta desarrollada en JavaScript totalmente gratuita que permite a cualquiera desarrollar su propio SIG.

El uso de OpenLayers [Ho4] junto a otras herramientas de gestión de datos geográficos ofrece un número de posibilidades ilimitadas, con una amplia adaptabilidad a múltiples dispositivos, puesto que JavaScript solo necesita de un navegador que le permita ejecutarse independiente del hardware que haya detrás.

1.2 Objetivos

La finalidad de este trabajo consiste en desarrollar una herramienta SIG altamente personalizable, de manera que permita adaptar su uso a cualesquiera que sean las necesidades requeridas. Para ello, se plantean los siguientes objetivos:

- Evaluar la potencialidad de OpenLayers [Ho4] como principal herramienta para la gestión de la información georeferenciada. Probar su compatibilidad con otras herramientas de gestión de datos geográficos, bases de datos, etc. que permitan ofrecer los servicios habituales en un SIG.
- Utilizar herramientas de Software Libre, con código abierto y licencias de uso libre, que reduzcan el coste de desarrollo al mínimo, permitan su modificación a largo plazo según los intereses del momento y cuyo mantenimiento sea sencillo de realizar y ampliamente documentado.

- Desarrollo de la herramienta en un entorno web independiente del dispositivo donde se vaya a ejecutar, permitiendo su consulta en cualquier momento y lugar que sea requerido.
- Simplificar el uso de la herramienta con una interfaz intuitiva, que permita a cualquier persona con nociones básicas de informática usarla y que dicho uso se asemeje a lo ya conocido en otros SIGs como el de Google o Microsoft.

Por último, esta herramienta servirá como prototipo de gestor de datos geoespaciales para su uso en el campus de Tafira de la ULPGC y en cualquiera de sus sucursales en todo el archipiélago canario.

2.COMPETENCIAS

En el desarrollo de este trabajo se han cubierto una serie de competencias más allá de las básicas que todo proyecto debe cumplir. A continuación se listan las distintas competencias específicas satisfechas, su definición y la manera o lugar en la que éstas se ven realizadas.

ISO3

Capacidad de dar solución a problemas de integración en función de las estrategias, estándares y tecnologías disponibles.

Esta competencia ha sido desarrollada en la sección Diseño e implementación. En ese capítulo se muestran las decisiones tomadas con respecto a los problemas surgidos entre las distintas tecnologías.

ISO4

Capacidad de identificar y analizar problemas y diseñar, desarrollar, implementar, verificar y documentar soluciones software sobre la base de un conocimiento adecuado de las teorías, modelos y técnicas actuales.

A lo largo de este documento, especialmente en las secciones de Objetivos, Análisis del problema, Diseño e implementación y Manual de usuario, se analiza la problemática planteada y se propone una solución documentada.

CPO6

Capacidad para desarrollar y evaluar sistemas interactivos y de presentación de información compleja y su aplicación a la resolución de problemas de diseño de interacción persona computadora.

Como se puede ver en la sección Diseño e implementación, se desarrolla una interfaz intuitiva al usuario, cómoda de usar y con el apoyo de OpenLayers para realizar tareas complejas de SIG.

SI03

Capacidad para participar activamente en la especificación, diseño, implementación y mantenimiento de los sistemas de información y comunicación.

Esta competencia está presente a lo largo de la sección Diseño e implementación, donde a través de reuniones se tomaron decisiones conjuntas para mejorar la aplicación, tanto funcionalmente como en su uso.

TI06

Capacidad de concebir sistemas, aplicaciones y servicios basados en tecnologías de red, incluyendo Internet, web, comercio electrónico, multimedia, servicios interactivos y computación móvil.

En diversas secciones de este documento, tales como Objetivos, Aportaciones y toda la sección de Desarrollo, se da un gran uso a las tecnologías de red, creando una aplicación capaz de interactuar con un varios servidores a través de internet y funcional en distintos dispositivos, de escritorio y móviles.

3.PLIEGO DE CONDICIONES

3.1 Objeto de este pliego

El pliego de condiciones técnicas tiene por objeto la definición de condiciones generales, técnicas y económicas asociadas a la ejecución y utilización del software para la gestión de información geoespacial.

3.2 Pliego de condiciones generales

El presente trabajo está destinado al desarrollo de un prototipo SIG para la gestión de información geográfica sobre las distintas sucursales de la ULPGC en el archipiélago canario. Sus características generales son las siguientes:

- Desarrollo sobre herramientas de Software Libre.
- Desarrollo web que no requiere instalación de ningún tipo.
- Capacidad para mostrar capas Web Map Service (WMS) (ver Protocolos) del archipiélago canario.
- Capacidad para mostrar y editar capas Web Feature Service (WFS) (ver Protocolos) sobre una capa WMS base, permitiendo la adición, edición y eliminación de primitivas vectoriales.
- Capacidad de crear nuevas capas WFS de tipo punto, línea o polígono.
- Gestión de información extra asociada a los elementos de las capas WFS.
- Gestión de distintos perfiles de usuarios a través de contraseña.
- Acceso desde múltiples dispositivos.

Cuando el uso del material desarrollado conlleve el tratamiento de datos personales se ha de aplicar la legislación vigente en España. En la actualidad la ley que regula este tema es la Ley Orgánica de Protección de Datos de Carácter Personal (LOPD) [Eo4].

3.3 Pliego de especificaciones técnicas

El pliego de condiciones técnicas dispone de dos apartados diferenciados:

3.3.1 Especificaciones de materiales y equipos

Los materiales de los que se compone el trabajo se dividen en dos categorías: hardware y software. A continuación se muestra un listado general de los requisitos.

3.3.2 Hardware

- Ordenador Portátil Personal. No se requieren características especiales para este equipo.
- Servidor remoto con Windows Server 2008 R2 Estándar e Internet Information Services (IIS) 7.5 suministrado por la ULPGC. Tampoco requiere características especiales aunque deberá ser capaz de gestionar el número de peticiones que se lancen sobre la aplicación. Al tratarse de software privativo, se recomienda usar un servidor con base Linux y Apache para la gestión de las funciones de servidor como se verá más adelante.
- Dispositivo móvil Smartphone para testeo de la aplicación. No se requieren características especiales salvo la capacidad de ejecutar un navegador que incluya soporte para JavaScript y conexión a internet.
- Dispositivo móvil Tablet para testeo de la aplicación. Igualmente, no se requieren características especiales salvo las mencionadas anteriormente para el Smartphone.

3.3.3 Software

- OpenLayers 2.12 [Ho4] para la gestión de la información geoespacial
- Geoserver 2.0.2 [Ho5] para el suministro de los datos de las capas WFS (ver Protocolos)
- Sistema de Gestión de Bases de Datos PostgreSQL [Ho6]

En la sección de Requisitos se detalla con más precisión los elementos usados en el desarrollo de este trabajo.

3.3.4 Especificaciones de ejecución

Los pasos seguidos durante en desarrollo de este trabajo han seguido un desarrollo incremental, con etapas definidas de la siguiente manera:

- Familiarización con la Herramienta OpenLayers [Ho4] realizando diversas pruebas a fin de comprobar la potencialidad de la misma.
- Familiarización con la Herramienta Geoserver [Ho5] para la creación de capas WMS y WFS [Protocolos].

- Desarrollo de un prototipo inicial capaz de obtener información desde Geoserver [Ho5] y PostgreSQL [Ho6].
- Mejora del prototipo con funciones de edición y acceso público al servidor.
- Mejora del prototipo con más funcionalidades y desarrollo de una interfaz adaptable a dispositivos móviles.
- Mejora del prototipo con opciones de administración y gestión de estilos sobre las capas.

Estas etapas se verán en mayor detalle en la sección Diseño e Implementación, dentro del apartado Desarrollo.

3.4 Pliego de cláusulas administrativas

A continuación se hace un desglose de los costes de desarrollo de este trabajo:

Recurso	Coste
Programador (250 horas)	35€/hora
Ordenador Portátil Asus S56C	550,17€
Servidor ThinkServer 70A4001LUX 5U	328,81€
Smartphone HTC Nexus One	85€
Tablet Acer Iconia A1-810	133,91€
Total	9847,89€
Total + IGIC	10537,24€

Tabla 1: Relación de costes

La relación de costes mostrada en la ***Tabla 1*** es en base a los recursos utilizados pero como se ha comentado previamente, no es necesaria la adquisición de esos modelos específicos mientras se cumplan los requisitos mencionados, por lo que la inversión total podría ser más económica. También se ha de tener en cuenta que el uso de un navegador específico sobre un dispositivo específico (por ejemplo Safari en un IPad de Apple) podría mostrar algún tipo de incompatibilidad, por lo que si se deseara garantizar su correcto funcionamiento sobre dicho dispositivo, se debería hacer una inversión al respecto.

3.5 Licencia

Teniendo en cuenta las licencias de las diversas herramientas de las que se hace uso en este trabajo (ver sección licencias en el apartado Normativa y Legislación), se ha buscado una licencia que sea compatible con todas ellas.

El presente trabajo se distribuye bajo una licencia GNU GPL [E06] (GNU General Public License) versión 3. Al ser utilizado, el usuario no adquiere ningún poder ni titularidad sobre el software que contiene. Sin embargo, podrá usarlo, distribuirlo, estudiarlo y modificarlo siempre y cuando así lo desee, tal y como establece este tipo de licencia.

La compatibilidad de dichas licencias se ha podido comprobar en la propia página de GNU, en la sección de Licencias libres compatibles con la GPL [E10].

4.APORTACIONES

El planteamiento inicial de este trabajo se estableció como una solución a las necesidades de geolocalización del campus de Tafira por parte de la ULPGC. Previamente se había desarrollado un trabajo con un objetivo similar – *Herramienta de ayuda para la gestión de recursos en aplicaciones geoespaciales* por Desirée de las Nieves Cruz Godoy -, pero planteaba una serie de inconvenientes al tratarse de un software desarrollado exclusivamente para el sistema operativo Windows y tener que estar instalado en la máquina donde se ejecutara. Por ello, se especificó como requisito que en este trabajo se desarrollara una aplicación web que no fuera necesaria descargar o instalar y que se pudiera ejecutar en diversos dispositivos, no solo en ordenadores personales.

El desarrollo del actual trabajo se hizo pensando en un uso genérico de la aplicación, lo que permite que, con pequeñas variaciones, cualquier usuario pueda utilizarla para su propio consumo. El Gestor Web de Recursos Geoespaciales con Tecnología OpenLayers ofrece las siguientes características:

- Consulta de la información mostrada sobre el mapa desde cualquier lugar y cualquier dispositivo con conexión a internet.
- Posibilitar la creación y edición de información para usuarios con privilegios desde la propia aplicación.
- Permitir la creación ilimitada de capas WFS (ver Protocolos) de tipo punto, línea o polígono.
- Permitir la adición de información extra a las primitivas, de manera pública (a través de PopUps cuyo contenido es altamente modificable) o privada (a través de campos internos).
- Permitir la edición de los estilos de los elementos de las capas WFS facilitando su visualización e identificación.

Todo ello haciendo uso de una interfaz cómoda de usar e intuitiva. Adicionalmente y como se verá más adelante, este trabajo está diseñado de forma que se pueda seguir mejorando y añadiendo funcionalidades en un futuro y según las necesidades de sus usuarios.

El trabajo está realizado bajo una Licencia GNU GPL, permitiendo así unas condiciones de uso favorables para todo tipo de usuarios.

5.NORMATIVA Y LEGISLACIÓN

La legislación vigente que afecta al presente Trabajo de Fin de Grado es la siguiente:

5.1 Ley orgánica de protección de datos

El primer artículo de la Ley Orgánica de Protección de Datos de Carácter Personal (LOPD) [E04] dispone: "La presente Ley Orgánica tiene por objeto garantizar y proteger, en lo que concierne al tratamiento de los datos personales, las libertades públicas y los derechos fundamentales de las personas físicas, y especialmente de su honor e intimidad personal y familiar". Como aplicación, la adición, modificación y eliminación de la información solo se permite a un usuario con privilegios, con un nombre de usuario y contraseña que le permita el acceso a esas funcionalidades. Además, ciertos campos de información solo están visibles a los usuarios con privilegios, protegiendo así su intimidad en caso de incluir información sensible.

Respecto a los usuarios, al permitir la publicación de información adicional asociada a unos datos geoespaciales, se considera a tener en cuenta la protección de dicha información con respecto a las personas físicas y sus derechos fundamentales como se describe en el artículo 2 "Ámbito de aplicación" sección 1 "La presente Ley Orgánica será de aplicación a los datos de carácter personal registrados en soporte físico, que los haga susceptibles de tratamiento, y a toda modalidad de uso posterior de estos datos por los sectores público y privado".

Se deberá prestar especial atención a lo mencionado en el artículo 3 "Definiciones" sección a) "Datos de carácter personal: cualquier información concerniente a personas físicas identificadas o identificables" considerando que los datos incluidos puedan entrar en dicha categoría y requieran un tratamiento adecuado como la autorización de las personas afectadas para formar parte de la información gestionada por la aplicación, conociendo todas las implicaciones de tal acuerdo.

5.2 Directiva 95/46/CE

Igualmente, se deberá tener en cuenta la Directiva 95/46/CE [E05], texto de referencia en materia de protección de datos a escala europea. En síntesis, la directiva expone lo siguiente:

"La presente Directiva se aplica a los datos tratados por medios automatizados (base de datos informática de clientes, por ejemplo), así como a los datos contenidos en un fichero no automatizado o que vayan a figurar en él (ficheros en papel tradicionales).

La Directiva no se aplicará al tratamiento de datos:

- efectuado por una persona física en el ejercicio de actividades exclusivamente particulares o domésticas;
- aplicado al ejercicio de actividades no comprendidas en el ámbito de aplicación del Derecho comunitario, tales como la seguridad pública, la defensa o la seguridad del Estado.

La Directiva tiene como objetivo proteger los derechos y las libertades de las personas en lo que respecta al tratamiento de datos personales, estableciendo principios de orientación para determinar la licitud de dicho tratamiento.

- Los datos personales serán tratados de manera leal y lícita, y recogidos con fines determinados, explícitos y legítimos.
- El tratamiento de datos personales sólo podrá efectuarse si el interesado ha dado su consentimiento de forma inequívoca o si el tratamiento es necesario para ejecuciones de contratos donde el interesado sea parte, el cumplimiento de obligaciones jurídicas, protección de su interés vital, etc.
- Deberá prohibirse el tratamiento de datos personales que revelen el origen racial o étnico, opiniones políticas, convicciones religiosas y la pertenencia a sindicatos, así como el tratamiento de los datos relativos a la salud o a la sexualidad.
- El responsable deberá facilitar datos del tratamiento (identidad, finalidad del tratamiento, destinatarios de los datos, etc) a la persona de quien se recaben los datos.
- Se podrá limitar el alcance de los principios anteriores con objetivo de salvaguardar, por ejemplo, la seguridad del Estado o la seguridad pública.
- El interesado tendrá derecho a oponerse a que los datos que le conciernen sean objeto de tratamiento.

- El responsable del tratamiento deberá aplicar las medidas adecuadas para proteger los datos personales contra la destrucción, accidental o ilícita, la pérdida accidental, la alteración, la difusión o accesos no autorizados.
- El responsable del tratamiento notificará a la autoridad de control nacional con anterioridad a la realización del tratamiento."

5.3 Licencias

A continuación se describen las licencias implicadas en el desarrollo del trabajo por las herramientas usadas y por el propio proyecto en sí:

5.3.1 GNU GPL

La Licencia Pública General de GNU (GNU LPG o GNU General Public License GNU GPL en inglés) [Eo6] garantiza a los usuarios finales (personas, organizaciones, compañías) la libertad de usar, estudiar, compartir (copiar) y modificar el software. Con esto se busca que el software cubierto por la licencia sea software libre y esté protegido de intentos de apropiación que restrinjan sus libertades. Esta licencia fue creada originalmente por Richard Stallman fundador de la Free Software Foundation (FSF) para el proyecto GNU.

La licencia GPL garantiza a los destinatarios de un programa de computador los derechos y libertades reunidos en definición de software libre y usa copyleft para asegurar que el software está protegido cada vez que el trabajo es distribuido, modificado o ampliado. En la forma de distribución (sólo pueden ser distribuidos bajo los términos de la misma licencia) se diferencian las licencias GPL de las licencias de software libre permisivas, de las cuales los ejemplos más conocidos son las licencias BSD (BSD licenses).

Esta licencia es utilizada por Geoserver y también se aplica a este trabajo como se ha mencionado previamente.

5.3.2 BSD

La licencia Berkeley Software Distribution (BSD) [Eo7] es una licencia de software libre permisiva, con menos restricciones que otras como la GPL, de dominio

casi público. Al contrario que la GPL, permite el uso de código fuente en software no libre.

Esta licencia es utilizada por PostgreSQL y por OpenLayers.

5.3.3 Apache 2.0

La licencia Apache (Apache License o Apache Software License para versiones anteriores a 2.0) [Eo8] es una licencia de software libre creada por la Apache Software Foundation (ASF). Esta licencia requiere la conservación del aviso de copyright y disclaimer, sin ser una licencia copyleft, por no requerir la redistribución del código fuente.

Bajo esta licencia, el usuario del software posee la libertad de usarlo para cualquier propósito, distribuirlo, modificarlo, y distribuir versiones modificadas de ese software.

La Licencia Apache no exige la distribución de las obras derivadas bajo la misma licencia, ni que sean software libre o de código abierto. Lo único que exige es que informe a los receptores sobre el uso del código con dicha licencia.

Esta licencia es utilizada por OpenLayers.

5.3.4 MIT o X11

La licencia MIT [Eo9] es una de tantas licencias de software que ha empleado el Instituto Tecnológico de Massachusetts (MIT, Massachusetts Institute of Technology) a lo largo de su historia, y quizás debería llamarse más correctamente licencia X11, ya que es la licencia que llevaba este software de muestra de la información de manera gráfica X Window System originario del MIT en los años 1980. Pero ya sea como MIT o X11, su texto es idéntico.

Su texto es modificable ya que no tiene copyright, aunque no se recomienda. Es una licencia muy parecida a la BSD en cuanto a efectos. Tiene tres puntos bien diferenciados:

- Condiciones, la condición es que la nota de copyright y la parte de los derechos se incluya en todas las copias o partes sustanciales del Software. Esta es la condición que invalidaría la licencia en caso de no cumplirse.
- Derechos, los derechos son muchos: sin restricciones; incluyendo usar, copiar, modificar, integrar con otro Software, publicar, sublicenciar o vender copias del Software, y además permitir a las personas a las que se les entregue el Software hacer lo mismo.
- Limitación de responsabilidad, finalmente se tiene un disclaimer o nota de limitación de la responsabilidad habitual en este tipo de licencias.

Esta licencia está aplicada en jQuery y sus derivados como jQuery Mobile, jQueryTE y en OpenLayers.

6.REQUISITOS

A continuación se listan los requisitos del trabajo de manera más detallada clasificados por hardware y software:

6.1 Hardware

- Ordenador Portátil Personal. No se requieren características especiales para este equipo salvo la capacidad de conectarse remotamente al servidor para actualizar el código de la aplicación y capacidad de ejecutar un navegador actual donde testear la aplicación. Durante el desarrollo de este trabajo se utilizó un Portátil ASUS S56C.
- Servidor remoto con Windows Server 2008 R2 Estándar e Internet Information Services (IIS) 7.5 suministrado por la ULPGC. Tampoco requiere características especiales aunque deberá ser capaz de gestionar el número de peticiones que se lancen sobre la aplicación. Al tratarse de software privativo, se recomienda usar un servidor con base Linux y Apache para la gestión de las funciones de servidor como se verá más adelante.
- Dispositivo móvil Smartphone para testeo de la aplicación. No se requieren características especiales salvo la capacidad de ejecutar un navegador que incluya soporte para JavaScript y conexión a internet. Para este trabajo se utilizó un Smartphone HTC Nexus One.
- Dispositivo móvil Tablet para testeo de la aplicación. Igualmente, no se requieren características especiales salvo las mencionadas anteriormente para el Smartphone. La utilizada en este trabajo fue una Tablet Acer Iconia A1-810.

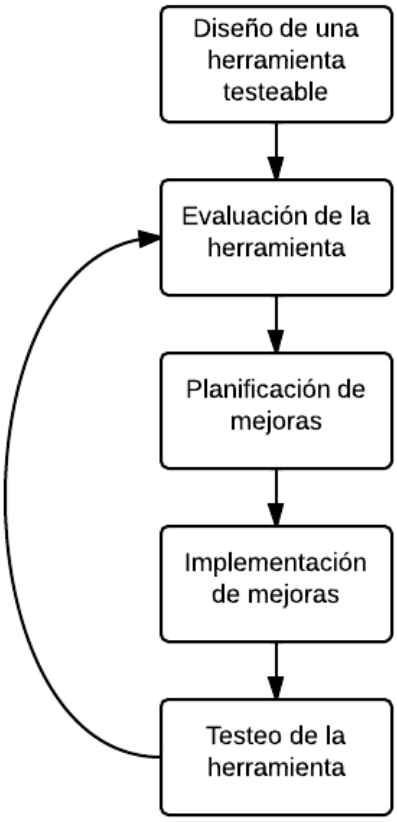
6.2 Software

- Editor de texto con soporte para varios lenguajes de marcado y programación. Se optó por Notepad++[Ho1] por ser de código abierto.
- Navegadores Google Chrome [Ho2] y Mozilla Firefox [Ho3] para el testeo de la aplicación web.
- OpenLayers 2.12 [Ho4] para la gestión de la información geoespacial.
- Geoserver 2.0.2 [Ho5] para el suministro de los datos de las capas WFS (ver Protocolos). Actualmente Geoserver se encuentra en su versión 2.5.1 la cual incluirá una serie de mejoras sustanciales, sobre todo a la hora de gestionar las conexiones seguras a la aplicación.

- Sistema de Gestión de Bases de Datos PostgreSQL [H06]. Se recomienda el uso de pgAdmin III ya que facilita una interfaz con la que realizar tareas sobre la base de datos de manera cómoda.
- Módulo PostGIS [H07] para soporte de objetos geográficos en PostgreSQL.
- jQuery [H08] y jQuery Mobile [H09] para el diseño de la interfaz de la aplicación.
- Editor HTML jQueryTE [H10] para dar formato a la información adicional de las capas WFS (ver Protocolos).

7.METODOLOGÍA Y PLAN DE TRABAJO

La metodología seguida en este trabajo ha sido similar a la del desarrollo Iterativo e Incremental [Eo3] debido a que, al tratarse de un prototipo, cuando se cubrieron los requerimientos mínimos deseados se empezó a añadir funcionalidades, testearlas, verificar el correcto funcionamiento de todo y vuelta a la adición de funcionalidades. Los pasos seguidos fueron los siguientes:



7.1 Plan de trabajo

En relación al plan de trabajo a llevar a cabo, las etapas a cubrir durante el desarrollo del TFG propuesto fueron las siguientes:

- ***Análisis*** Se planteó OpenLayers como solución a las necesidades actuales de la ULPGC para la gestión de información Geoespacial. Se estudió sus características y compatibilidad con otras herramientas de Software Libre para desarrollar una aplicación web que cubriese dichas necesidades.

- ***Desarrollo*** Se implementó una herramienta que realizaba las funciones básicas de un SIG, se desarrolló una interfaz compatible con múltiples dispositivos y se añadieron funcionalidades de uso y gestión, optimizando el código siempre que fuera necesario y comprobando el correcto funcionamiento de la herramienta.
- ***Documentación*** Confección del documento de trabajo de fin de grado a partir de toda la documentación realizada en las etapas anteriores. Preparación de la defensa.

La estimación de las horas trabajadas y las actividades dentro de cada fase son las siguientes:

Fase	Análisis	**Duración Estimada**	150 horas

Actividades

1.1 Estudio inicial de necesidades para cubrir el objetivo del trabajo
1.2 Familiarización con los Sistemas de Información Geográfica, terminología, definiciones, etc.
1.3 Estudio y selección de las herramientas para alcanzar el objetivo propuesto
1.4 Familiarización, testeo e integración entre las distintas herramientas (OpenLayers, Geoserver, Postgres, Postgis, jQuery, jQuery Mobile)
1.5 Planificación del prototipo (uso práctico)

Fase	Desarrollo	**Duración Estimada**	100 horas

Actividades

2.1 Implementación del prototipo
2.2 Corrección de errores y mejora de usabilidad
2.3 Adaptación para diversos dispositivos (PCs, tablets, smartphones…)
2.4 Corrección de errores y mejora de usabilidad

Fase	Documentación	**Duración Estimada**	50 horas

Actividades

3.1 Documentación de la información generada en las fases previas
3.2 Preparación de la defensa

8.CONCEPTOS

A fin de facilitar la comprensión del trabajo realizado, se establecerán algunos conceptos:

8.1 Datos Georeferenciados

El Gestor Web de Recursos Geoespaciales con Tecnología OpenLayers trabaja principalmente con dos tipos de datos: datos comunes y datos georeferenciados. Los datos georeferenciados se diferencian de los comunes por tener asociadas unas coordenadas de latitud y longitud que indican su ubicación en la Tierra. Estos datos facilitan calcular distancias o conocer localizaciones precisas de primitivas sobre un mapa como pueden ser edificios, parques, estatuas, etc.

La información espacial se presenta fundamentalmente en dos tipos de formatos: ráster y vectorial.

8.2 Información ráster

Según la Wikipedia [E11], un tipo de datos ráster es, en esencia, cualquier tipo de imagen digital representada en mallas. El modelo de SIG ráster o de retícula se centra en las propiedades del espacio más que en la precisión de la localización. Divide el espacio en celdas regulares donde cada una de ellas representa un único valor. Se trata de un modelo de datos muy adecuado para la representación de variables continuas en el espacio.

Cualquiera que esté familiarizado con la fotografía digital reconoce el píxel como la unidad menor de información de una imagen. Una combinación de estos píxeles creará una imagen, a distinción del uso común de gráficos vectoriales escalables que son la base del modelo vectorial. Si bien una imagen digital se refiere a la salida como una representación de la realidad, en una fotografía o el arte transferidos a la computadora, el tipo de datos ráster reflejará una abstracción de la realidad. Las fotografías aéreas son una forma de datos ráster utilizada comúnmente con un sólo propósito: mostrar una imagen detallada de un mapa base sobre la que se realizarán labores de digitalización. Otros conjuntos de datos ráster podrán contener información referente a las elevaciones del terreno (un Modelo Digital del Terreno), o de la reflexión

de la luz de una particular longitud de onda (por ejemplo las obtenidas por el satélite LandSat), entre otros.

Figura 2: Representación ráster del campus de Tafira

8.3 Información vectorial

En los datos vectoriales, el interés de las representaciones se centra en la precisión de localización de los elementos geográficos sobre el espacio y donde los fenómenos a representar son discretos, es decir, de límites definidos. Cada una de estas geometrías está vinculada a una fila en una base de datos que describe sus atributos. Por ejemplo, una base de datos que describe los lagos puede contener datos sobre la batimetría de estos, la calidad del agua o el nivel de contaminación. Esta información puede ser utilizada para crear un mapa que describa un atributo particular contenido en la base de datos. Los lagos pueden tener un rango de colores en función del nivel de contaminación. Además, las diferentes geometrías de las primitivas también pueden ser comparadas, permitiendo mostrar por ejemplo los lagos que superan una cantidad exacta de contaminación o cuya área es menor que una definida previamente.

El punto, la línea y el polígono son los elementos digitales que permiten representar las entidades del mundo real.

- Los ***puntos*** indican una referencia exacta sobre un plano o mapa. Normalmente se representan con un par de valores Latitud/Longitud. Por ejemplo, la localización de un restaurante, de una entrada o de una parada de taxi se representarían con un punto. Los puntos pueden estar representados por iconos o símbolos y su tamaño puede llevar información asociada. De esta forma, podríamos indicar no solo los puntos negros de una carretera sino destacar su peligrosidad con el tamaño del punto como indicador del número de accidentes ocurridos en él.
- Las ***líneas*** unidimensionales o polilíneas conectan un punto con otro. Suelen ser usadas para representar ríos, carreteras, caminos, etc. Cada punto de la línea se considera un vértice de la misma y gracias a su información georeferenciada se puede calcular la distancia real entre ambos puntos o de la línea total.
- Los ***polígonos*** bidimensionales se utilizan para representar elementos geográficos que cubren un área particular de la superficie de la tierra. Con ellos se pueden representar edificios, ciudades, países, etc. Al igual que la línea está compuesto por vértices y habitualmente se aplica una tonalidad a su área cubierta para dar algún tipo de información asociada.

Figura 3: Edificios del campus de Tafira resaltados con polígonos

8.4 Bases de datos espaciales

Consisten en bases de datos especializadas que añaden columnas de geometría para la ubicación de los elementos, además de aportar funcionalidades extras de consulta. Se debe establecer previamente un Sistema de Referencia Espacial (SER) para definir la localización y relación entre objetos, ya que no son valores absolutos.

Una base de datos geográfica requiere de un proceso de abstracción que permita representar la complejidad del mundo real de manera simplificada, permitiendo así que las computadoras puedan procesar esta información. Este proceso de abstracción es multinivel y se estructura en la base de datos mediante capas de temáticas definidas.

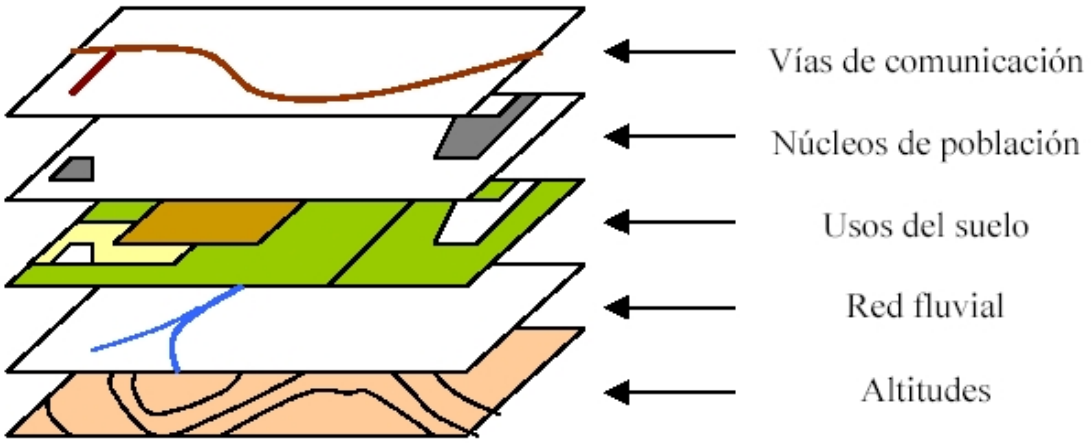


Figura 4: Información del mundo real representado en capas temáticas (extraído de la Wikipedia [E12])

8.5 Protocolos

En los SIG existen una serie de protocolos estandarizados para la transmisión de información espacial en la red. Los usados en este trabajo son los Web Map Service (WMS) para el manejo de información ráster y Web Feature Service (WFS) para la gestión de información vectorial.

- Servicio ***WMS*** [E13] definido por el Open Geospatial Consortium (OGC) produce mapas de datos referenciados espacialmente, de forma dinámica a partir de información geográfica. Este estándar internacional define un "mapa" como una representación de la información geográfica en forma de un archivo de imagen digital conveniente para la exhibición en una pantalla de ordenador.

Los mapas producidos por WMS se generan normalmente en un formato de imagen como PNG, GIF o JPEG.

- Servicio ***WFS*** [E14] también del OGC, es un servicio estándar, que ofrece una interfaz de comunicación que permite interactuar con los mapas servidos por el estándar WMS, como por ejemplo, editar la imagen que nos ofrece el servicio WMS o analizar la imagen siguiendo criterios geográficos. Para realizar estas operaciones se utiliza el lenguaje GML que deriva del XML, que es el estándar a través del que se transmiten las órdenes WFS. Este servicio no es transaccional, por lo que se limita a realizar consultas y recuperar elementos geográficos.
- Servicio ***WFS-T*** o Web Feature Service Transactional, es una extensión del WFS que además, permite la creación, eliminación y actualización de los elementos gráficos del mapa. Una petición transaccional contiene una serie de operaciones sobre los elementos vectoriales de la capa WFS. Al solicitarla, su resolución se puede dar directamente en el servicio WFS – enviando las modificaciones realizadas al almacén de datos del servicio –, o se puede traducir al lenguaje del almacén de datos y que sea él quien se encargue de ejecutar la operación solicitada.

Figura 5: ejemplo de edición de un elemento vectorial en una capa WFS-T

8.6 Servidores de Mapas

Muchas organizaciones públicas o privadas ofrecen servicios de mapeo en web como es el caso de la Infraestructura de Datos Espaciales de Canarias (IDECanarias) [E15], que pone a disposición de sus usuarios la información geográfica producida por el Gobierno de Canarias a través de su visor y de servicios estándares definidos conforme a las especificaciones del OGC.

Los servicios ofrecidos por IDECanarias son de carácter público y garantizan la interoperabilidad de la información geográfica de Canarias, englobando metadatos, conjuntos de datos espaciales, los servicios de datos espaciales y tecnologías de red entre otros.

Este trabajo hace uso de sus servicios WMS de mapas para mostrar el archipiélago canario.

A nivel nacional se encuentra disponible el Instituto Geográfico Nacional (IGN) [E16] que proporciona servicios similares a los descritos anteriormente.

A nivel mundial existen diversas organizaciones sin ánimo de lucro o iniciativas que aportan servicios de mapeo en web, tales como Gebco para la batimetría de los océanos u OpenStreetMap para los callejeros del mundo.

9.HERRAMIENTAS

En capítulos anteriores de este trabajo se ha hecho mención a diversas herramientas utilizadas en su desarrollo, las cuales se verán en profundidad a lo largo de esta sección. Se especificarán detalles sobre su uso, funcionalidades, características particulares y cualquier otro aspecto que facilite la comprensión de este trabajo.

9.1 GeoServer

GeoServer [Ho5] es un servidor de información geoespacial *Open Source* escrito en Java. Diseñado para la interoperatividad, permite a los usuarios compartir, procesar y editar una gran cantidad de datos espaciales, haciendo uso de la mayoría de formatos estándar de información geoespacial.

GeoServer ha evolucionado hasta llegar a ser un método sencillo de conectar información existente a globos virtuales tales como Google Earth y NASA World Wind o mapas basados en web como OpenLayers, Google Maps y Bing Maps.

Entre sus características se encuentran:

- Enteramente compatible con las especificaciones WMS, WFS (ver Protocolos) y Web Coverage Service (WCS), testados por el test de conformidad Compliance and Interoperability Testing Initiative (CITE) de la OGC, la mayor autoridad de estándares geoespaciales.
- Fácil utilización a través de la herramienta de administración vía web sin necesidad de acceder a grandes y complicados archivos de configuración.
- Soporte amplio de formatos de entrada PostGIS, Shapefile, ArcSDE y Oracle. VFP, MySQL, MapInfo y WFS en cascada también están entre los formatos de entrada soportados.
- Soporte de formatos de salida tales como JPEG, GIF, PNG, SVG y GML.
- Imágenes con antialiasing.
- Soporte completo de SLD, como definiciones del usuario (POST y GET), y como uso de configuración de estilos.
- Soporte para edición de datos de banco de datos individuales a través del protocolo WFS transactional profile (WFS-T), disponible para todos los formatos de datos.

- Basado en servlets Java (JEE), puede funcionar en cualquier servlet contenedor.
- Proyectado para ser compatible con extensiones.
- Facilidad de escritura de nuevos formatos de datos con la interfaz de almacenamiento de datos GeoTools - una biblioteca de sistemas de información geográfica - y clases de ayuda.

Su principal funcionalidad en este trabajo será la de servir de comunicación entre la aplicación donde se muestran los datos y la base de datos donde se almacenan.

Figura 6: Visualización de la aplicación Web de gestión de GeoServer

9.2 PostgreSQL

PostgreSQL [Ho06] es un sistema de gestión de base de datos relacionales, orientado a objetos y libre, publicado bajo la licencia BSD.

Como muchos otros proyectos de código abierto, el desarrollo de PostgreSQL no es manejado por una empresa y/o persona, sino que es dirigido por una comunidad de desarrolladores que trabajan de forma desinteresada, altruista, libre y/o apoyados por organizaciones comerciales. Dicha comunidad es denominada el PostgreSQL Global Development Group (PGDG).

Características:

Alta concurrencia Mediante un sistema de Acceso Concurrente Multiversión (MVCC en inglés) PostgreSQL permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. Cada usuario obtiene una visión consistente de lo último a lo que se le hizo commit. Esta estrategia es superior al uso de bloqueos por tabla o por filas común en otras bases, eliminando la necesidad del uso de bloqueos explícitos.

Amplia variedad de tipos nativos Provee nativamente soporte para:

- Números de precisión arbitraria.
- Texto de largo ilimitado.
- Figuras geométricas (con una variedad de funciones asociadas).
- Direcciones IP (IPv4 e IPv6).
- Bloques de direcciones estilo CIDR.
- Direcciones MAC.
- Arrays.

Adicionalmente, los usuarios pueden crear sus propios tipos de datos, que pueden ser por completo indexables gracias a la infraestructura Generalized Search Tree (GiST) de PostreSQL. Algunos ejemplos son los tipos de datos GIS creados por el proyecto PostGIS.

Otras características

- Claves ajenas también denominadas Llaves ajenas o Claves Foráneas (foreign keys).
- Disparadores (triggers), definidos como acciones específicas realizadas en relación a eventos específicos, ocurridos dentro de la base de datos. En PostgreSQL esto significa la ejecución de un procedimiento almacenado basado en una determinada acción sobre una tabla específica. Se definen por seis características:

- **Nombre** El nombre del disparador o trigger
- **Evento** El momento en que el disparador debe arrancar
- **Acción** Acción sobre la que se disparará el evento

- **Área de actuación** La tabla donde el disparador se activará
- **Objetivo** La frecuencia de la ejecución
- **Ejecución** La función que podría ser llamada

CREATE TRIGGER nombre { BEFORE | AFTER } { INSERT | UPDATE | DELETE [OR ...] }
ON tabla [FOR [EACH] { ROW | STATEMENT }]
EXECUTE PROCEDURE nombre de función (argumentos)

Entonces combinando estas seis características, PostgreSQL le permitirá crear una amplia funcionalidad a través de su sistema de activación de disparadores (triggers).

- Vistas.
- Integridad transaccional.
- Herencia de tablas.
- Tipos de datos y operaciones geométricas.
- Soporte para transacciones distribuidas. Permite a PostgreSQL integrarse en un sistema distribuido formado por varios recursos gestionado por un servidor de aplicaciones donde el éxito ("commit") de la transacción global es el resultado del éxito de las transacciones locales.

Para PostgreSQL podemos encontrar una serie de herramientas administrativas que realizan múltiples operaciones de manera visual como es el caso de pgAdmin III o PhpPgAdmin para su versión web.

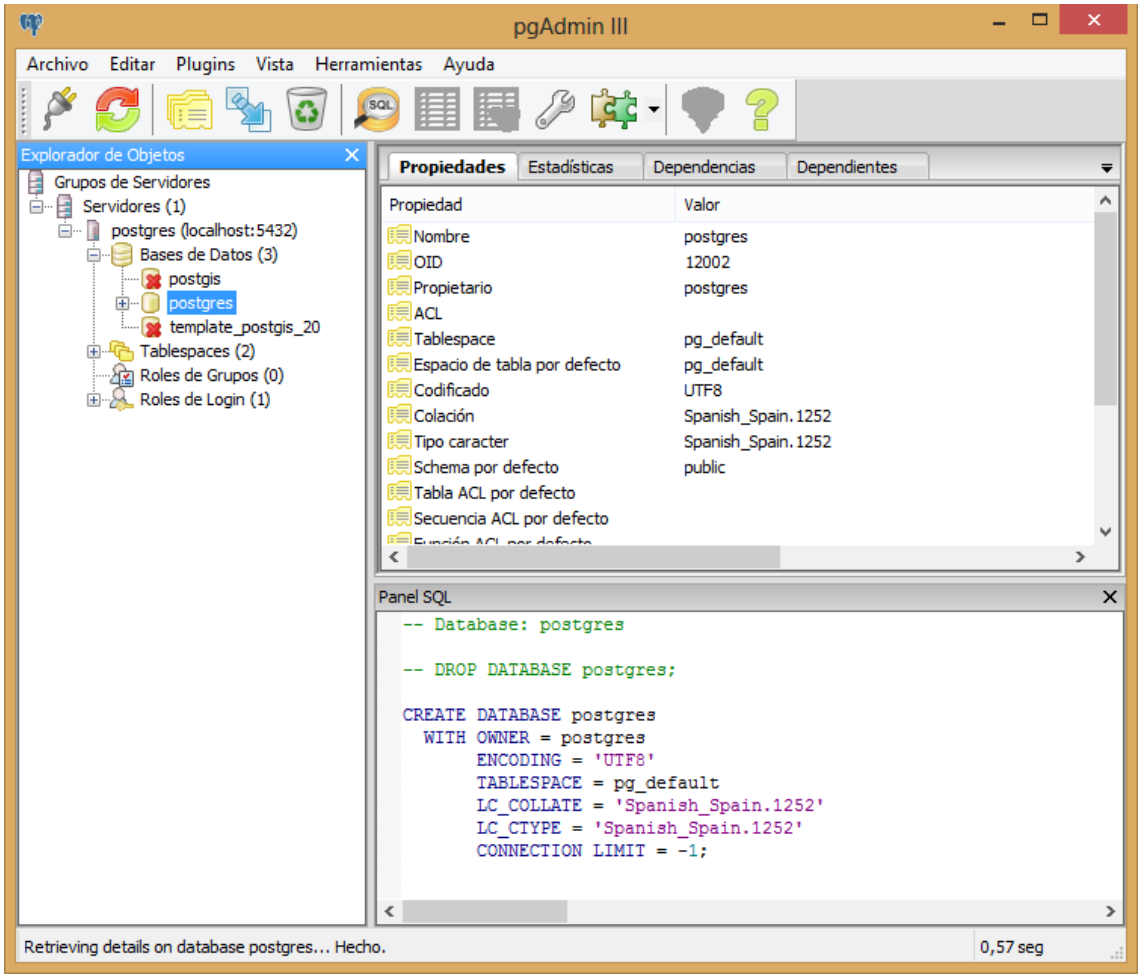


Figura 7: visualización de la herramienta pgAdmin III

9.3 PostGIS

Como se ha mencionado anteriormente, PostGIS es un módulo que añade los tipos de datos GIS, que aportan soporte de objetos geográficos al sistema de gestión de base de datos relacionales PostgreSQL, convirtiéndola en una base de datos espacial para su utilización en un SIG. Se encuentra bajo la Licencia Pública General de GNU.

PostGIS ha sido desarrollado por la empresa canadiense Refraction Research, especializada en productos *Open Source*. Versión tras versión ha ido demostrando su eficiencia, en algunos casos superando a otras herramientas similares como la extensión geográfica de la nueva versión de MySQL o compitiendo con la versión geográfica de la base de datos Oracle.

Otro aspecto interesante es que PostGIS ha sido certificado por el OGC, lo que garantiza la interoperabilidad con otros sistemas también interoperables. PostGIS

almacena la información geográfica en una columna del tipo GEOMETRY, que es diferente del homónimo "GEOMETRY" utilizado por PostgreSQL, donde se pueden almacenar la geometría en formato Well-Known Binary (WKB), aunque hasta la versión 1.0 se utilizaba la forma Well-Known Text (WKT).

Gracias a esta extensión se pueden almacenar datos espaciales y realizar consultas sobre ellos. Por ejemplo, permitiría almacenar el área cubierta de todos los parques de una zona y saber a qué distancia se encontraría una persona respecto a su posición actual.

9.4 OpenLayers

OpenLayers [Ho4] es una biblioteca de JavaScript de código abierto bajo una derivación de la licencia BSD para mostrar mapas interactivos en los navegadores web. Ofrece un API para acceder a diferentes fuentes de información cartográfica en la red: Web Map Services, Mapas comerciales (tipo Google Maps, Bing, Yahoo), Web Features Services, distintos formatos vectoriales, mapas de OpenStreetMap, etc. Entre sus ventajas se encuentran:

Simplicidad de uso Una de las características más destacadas de OpenLayers es su simplicidad. De cara al código HTML de una web, solo se necesita seleccionar y definir un elemento DIV como visor de mapas. Esto permite una alta versatilidad a la hora de diseñar interfaces o aplicaciones web, ya que lo referente al mapa ocurrirá dentro de esa área definida, siendo totalmente independiente del resto de elementos de la web.

Independencia A diferencia de otras APIs como la de Google Maps o la de Bing Maps, OpenLayers no está limitado a un servidor o mapas exclusivos. Ofrece una amplia gama de opciones para los servicios de mapas más conocidos, haciendo que su uso sea simple e incluso permite la mezcla de información obtenida desde distintos servidores. También ofrece una configuración genérica por protocolos, como es el caso de WMS o WFS.

Filosofía Software Libre Al estar basado en código abierto, todas las operaciones realizadas por OpenLayers pueden ser analizadas - e incluso modificadas - por cualquier desarrollador en cualquier momento. Además, como muchas otras

herramientas de Software Libre, OpenLayers está soportado por una gran comunidad de usuarios y desarrolladores que día a día mejoran esta biblioteca, solventando errores en poco tiempo y compartiendo conocimientos entre ellos.

La API de OpenLayers la forman principalmente dos elementos: Mapa y Capas. Un mapa de OpenLayers almacena información sobre la proyección por defecto, la extensión, unidades y demás propiedades del mapa. Dentro del mapa, los datos se muestran a través de las Capas. Una Capa es una fuente de datos que tendrá maneras específicas de solicitar la información y de representarla. Además, OpenLayers suministra una serie de herramientas de gran utilidad a la hora de interactuar con el mapa como son el zoom, la escala, el control de capas visibles, etc. que pueden ser activadas independientemente según interese al desarrollador.

En el caso de este trabajo y, debido a los objetivos específicos planteados, muchas de estas herramientas formarán parte de una interfaz externa, en vez de estar incluidas directamente en el mapa, como suele ser común en la mayoría de ejemplos disponibles en la red.

Figura 8: Ejemplo de visualización de mapa con OpenLayers

En contraposición a las APIs de mapeo de pago como la de Google o Microsoft tenemos que, a diferencia de ellas, OpenLayers sí puede tener control sobre el backend -lado del servidor-, permitiendo su configuración y personalización, además de evitar que se produzcan cambios inesperados en el futuro. Otro aspecto interesante es que no

presenta restricciones comerciales ya que no es una API de pago y su flexibilidad y personalización es total al ser de código abierto.

9.5 jQuery

jQuery [Ho8] es una biblioteca de JavaScript que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM - donde se encuentran todos los objetos de representación del documento -, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web. Fue presentada el 14 de enero de 2006 en el BarCamp NYC.

jQuery es software libre y de código abierto, posee una doble licencia, la Licencia MIT y la Licencia Pública General de GNU v2, permitiendo su uso en proyectos libres y privativos. Al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código, es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio.

Características

- Selección de elementos del DOM.
- Interactividad y modificaciones del árbol DOM, incluyendo soporte para CSS 1-3 y un plugin básico de XPath.
- Eventos.
- Manipulación de la hoja de estilos CSS.
- Efectos y animaciones.
- Animaciones personalizadas.
- AJAX.
- Soporta extensiones.
- Utilidades varias como obtener información del navegador, operar con objetos y vectores, funciones para rutinas comunes, etc.
- Compatible con los navegadores Mozilla Firefox 2.0+, Internet Explorer 6+, Safari 3+, Opera 10.6+ y Google Chrome 8+.5

10.DESARROLLO

A lo largo de este capítulo se explicarán tanto el software desarrollado como la naturaleza de su diseño. Se mostrará el análisis del problema actual que originó este trabajo, seguido de las características de su diseño e implementación. Se hará mención a las pruebas y resultados obtenidos y por último se abordarán las conclusiones, realizando una prospección de la herramienta a partir de su etapa final de desarrollo.

10.1 Análisis del problema

Como se ha comentado previamente, se buscaba realizar un SIG con mucha capacidad de personalización en un entorno web, de fácil acceso para diversos dispositivos y basado en software libre. Para ello fue necesario adquirir los conocimientos básicos del tratamiento de datos georeferenciados, gran parte de ellos extraídos del libro GIS for Web Developers: Adding Where to Your Web Applications [E01].

La selección de las herramientas presentadas se debió principalmente a su renombre en la web con respecto al manejo de datos geoespaciales y su relación con el software libre. Además, GeoServer, PostgreSQL y PostGIS formaron parte del trabajo realizado por la alumna Desirée de las Nieves Cruz Godoy – *Herramienta de ayuda para la gestión de recursos en aplicaciones geoespaciales* –, lo cual permitiría una mayor compatibilidad con su herramienta desarrollada, facilitando así el que pudieran coexistir bajo el mismo entorno. Igualmente, se prefijo el entorno de trabajo del servidor – Windows Server 2008 R2 Estándar –, pues había sido suministrado por la ULPGC. Esto a la larga supuso un inconveniente, especialmente en el caso del servidor web Internet Information Service (IIS) de Microsoft, debido a que la mayor parte de la documentación disponible hace referencia al uso de un sistema Linux y a un servidor web Apache. Se requirió, además, una familiarización con las herramientas para comprender su funcionamiento, con diversas pruebas como la creación de capas en GeoServer o la visualización de mapas WMS en OpenLayers.

Finalmente se establecieron las bases de cómo debía ser la aplicación, lo que generó un desarrollo tal como muestra el siguiente esquema:

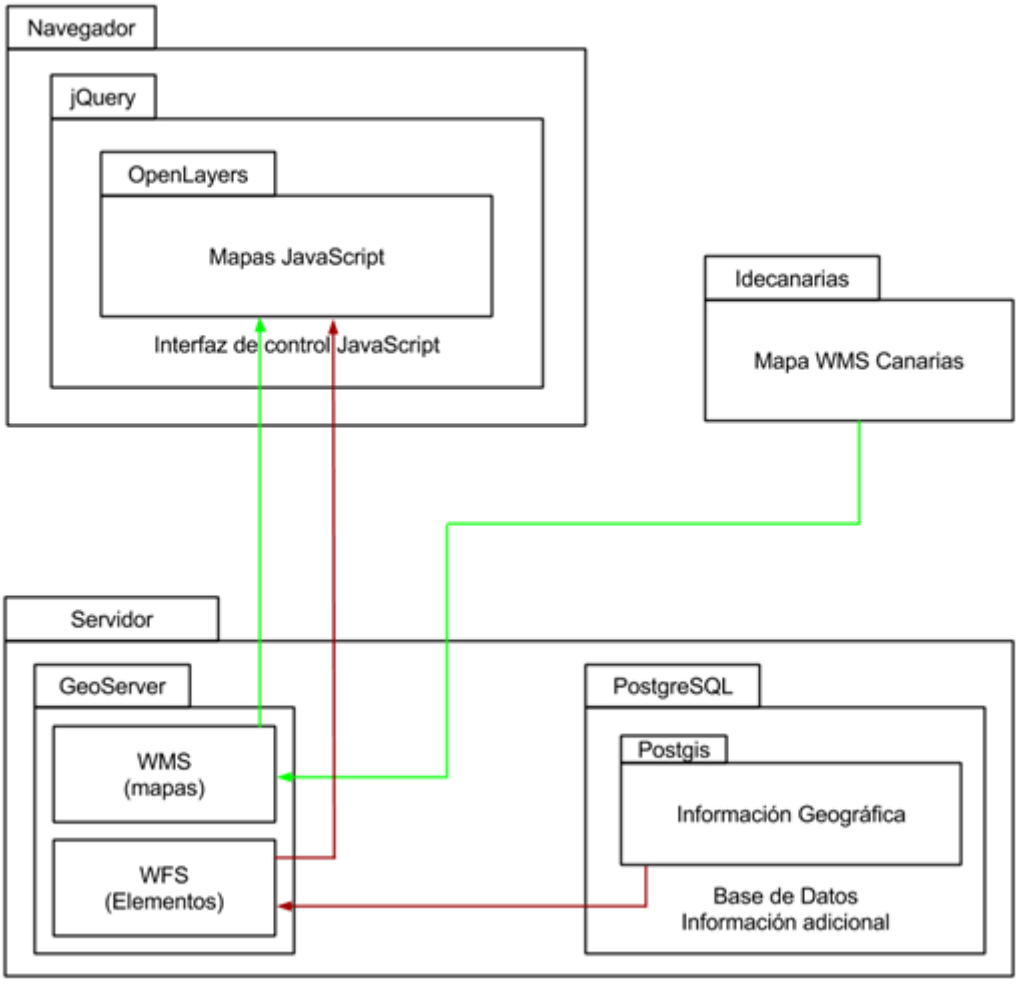


Figura 9: Esquema básico del funcionamiento del Gestor web de recursos geoespaciales con tecnología OpenLayers

Para el seguimiento del trabajo se estableció una metodología incremental, con planteamiento de objetivos que serían revisados en una posterior reunión con el tutor, evaluando lo obtenido y planteando nuevos objetivos o correcciones sobre lo ya realizado. Es por ello que, aunque su estructura principal no varió en demasía a lo largo del desarrollo, muchas de las decisiones no formaron parte de un análisis previo, sino que se tomaron en dichas reuniones.

10.2 Diseño e implementación

En este apartado se verá el proceso seguido durante el desarrollo del trabajo. Como se mencionó anteriormente está basado en una metodología incremental, formado por una serie de reuniones para definir objetivos, las cuales dirigirán el orden con el que se formará la estructura de esta sección.

10.2.1 Visualización de Capas en OpenLayers

En una primera reunión, se planteó el objetivo de mostrar varias capas con mapas WMS y WFS de IDECanarias desde OpenLayers. Configurar un mapa en OpenLayers es un proceso sencillo que requiere de los siguientes elementos:

- Proyección geográfica
- Unidades de medida
- Escala máxima (el mayor zoom que se puede hacer)
- Escala mínima (la mayor distancia que se puede alcanzar)
- Área del mapa

El principal problema surgido fue que IDECanarias no aportaba específicamente el nombre de las capas de sus mapas, solo la dirección del servicio, lo cual era un requisito indispensable de configuración para OpenLayers, ya que no es capaz de extraerla automáticamente. Un ejemplo de una petición sería el siguiente:

http://idecan1.grafcan.es/ServicioWMS/OrtoExpress?layer=OrtoExpress&SERVICE=WMS&VERSION=1.1.1&SRS=EPSG:32628&BBOX=(...)&REQUEST=GetMap

Analizando la petición se observa que existe más información aparte de la URL.

http://idecan1.grafcan.es/ServicioWMS/OrtoExpress	?	Layer= OrtoExpress
URL del servicio prestado por IDECanarias	Marca que indica el comienzo de atributos	Par atributo/valor

Como se observa en el ejemplo, existen una serie de atributos asociados a la petición:

- **Layer** Indica el nombre de la capa que se desea obtener
- **Service** Especifica el servicio sobre el que se ejecutará la petición
- **Version** Indica la versión del servicio
- **SRS** Código del sistema identificador de referencia geoespacial
- **BBOX** área del mapa solicitada

(…)

- **Request** Lo que se está pidiendo

Es posible que OpenLayers proporcione funciones para la obtención de dichas capas a través de la URL pero desconociendo cuál era el problema y en una etapa tan temprana del desarrollo era complicado identificar el error. La solución apareció por otros derroteros: otras plataformas como Gaia 3 creada por The Carbon Project, son capaces de realizar consultas a la dirección para obtener esos datos y además los muestran, permitiendo así poder consultarlos y trasladarlos a OpenLayers. No obstante, tal como se verá más adelante, esta funcionalidad no es necesaria puesto que las capas que se manejarán serán las aportadas por GeoServer, del cual se tiene administración completa.

```
var wmslayerOrtoUrb = new OpenLayers.Layer.WMS(
    "GranCanaria",
    "http://idecan1.grafcan.es/ServicioWMS/OrtoUrb?",
    {layers: 'OU'},
    {
       transitionEffect: 'resize',
       isBaseLayer: true
    }
);
map.addLayer(wmslayerOrtoUrb);
```

Figura 10: Configuración de una capa WMS en OpenLayers

Como se ve en el ejemplo, el nombre de la capa “OU” no aparece en la ruta “http://idecan1.grafcan.es/ServicioWMS/OrtoUrb”, por lo que su identificación requiere de otros medios.

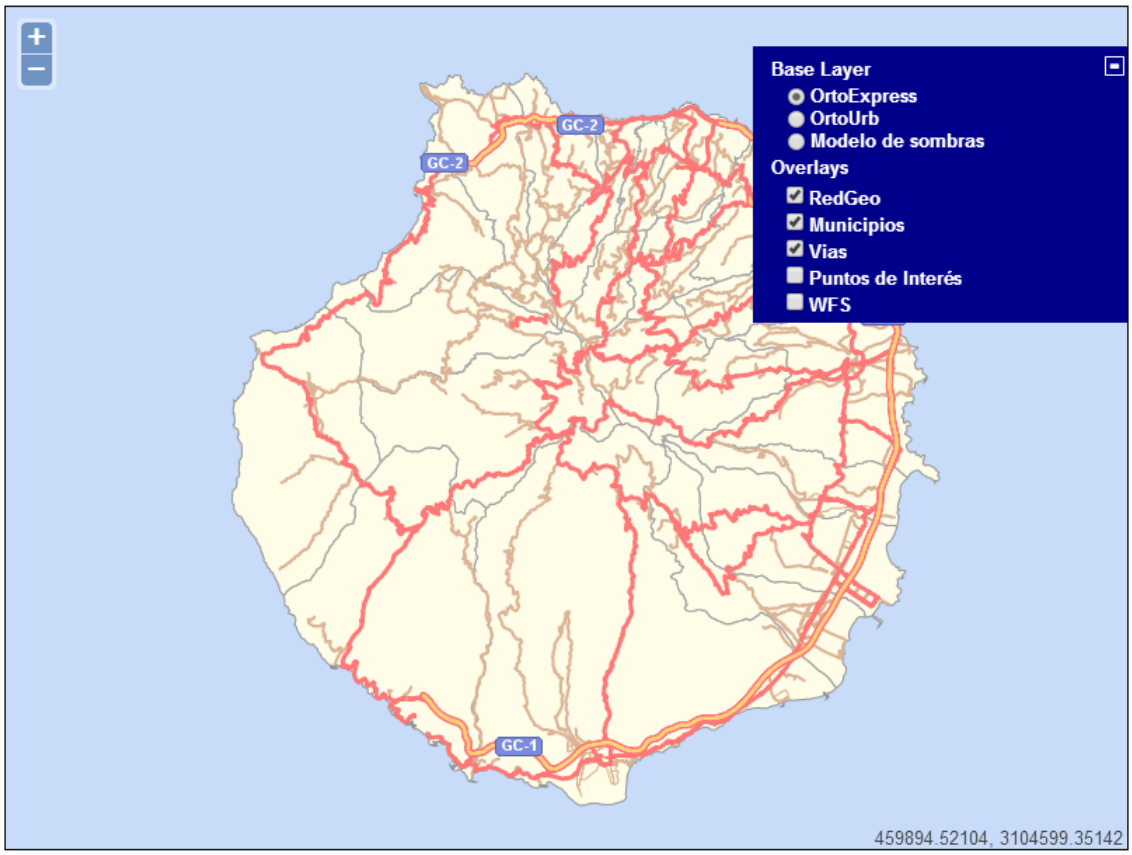


Figura 11: Visualización de diversas capas del IDECanarias a través de OpenLayers

10.2.2 Visualización y Control de Capas WFS-T

En la siguiente reunión se comprobó el correcto funcionamiento de lo desarrollado y se plantearon como objetivos el uso de capas WFS-T que permitan transacciones, crear capas de ejemplo para cada tipo de elemento (punto, línea y polígono) y permitir el acceso a la aplicación desde el exterior del servidor.

Hasta ahora la aplicación solo había requerido el uso de OpenLayers pues la información de los mapas partía del servidor de IDECanarias. El uso de capas WFS-T implicaba tener control de administrador sobre ellas para poder realizar modificaciones y salvados, por lo que se hacía necesario incluir GeoServer y PostgreSQL a la interactuación con la aplicación.

Acceder a la aplicación desde el exterior requería tener instalado un servidor web en el servidor. En este caso habían instaladas dos posibilidades: El IIS de Microsoft

o el propio GeoServer. Por simplicidad y evitar incompatibilidades se optó por el ofrecido por GeoServer. Otro requerimiento fue cambiar las URLs proporcionadas por GeoServer, que hasta ese momento se les hacía referencia como *localhost*. Como OpenLayers se ejecuta en el cliente, requiere que las URLs proporcionadas incluyan el dominio de donde provienen – al igual que IDECanarias con su *idecan1.grafcan.es* –, en este caso la IP pública del servidor.

Para poder visualizar capas WFS-T primero se debía comprobar que no había problemas de comunicación entre OpenLayers y GeoServer para los protocolos WMS y WFS. Se crearon dos capas en GeoServer, una WMS de Gran Canaria y otra WFS del campus de Tafira. En este caso GeoServer proporcionaba todos los datos necesarios para la configuración de OpenLayers, por lo que se pudo establecer la conexión sin problemas. El siguiente paso fue crear una capa que obtuviera los datos desde la base de datos PostgreSQL.

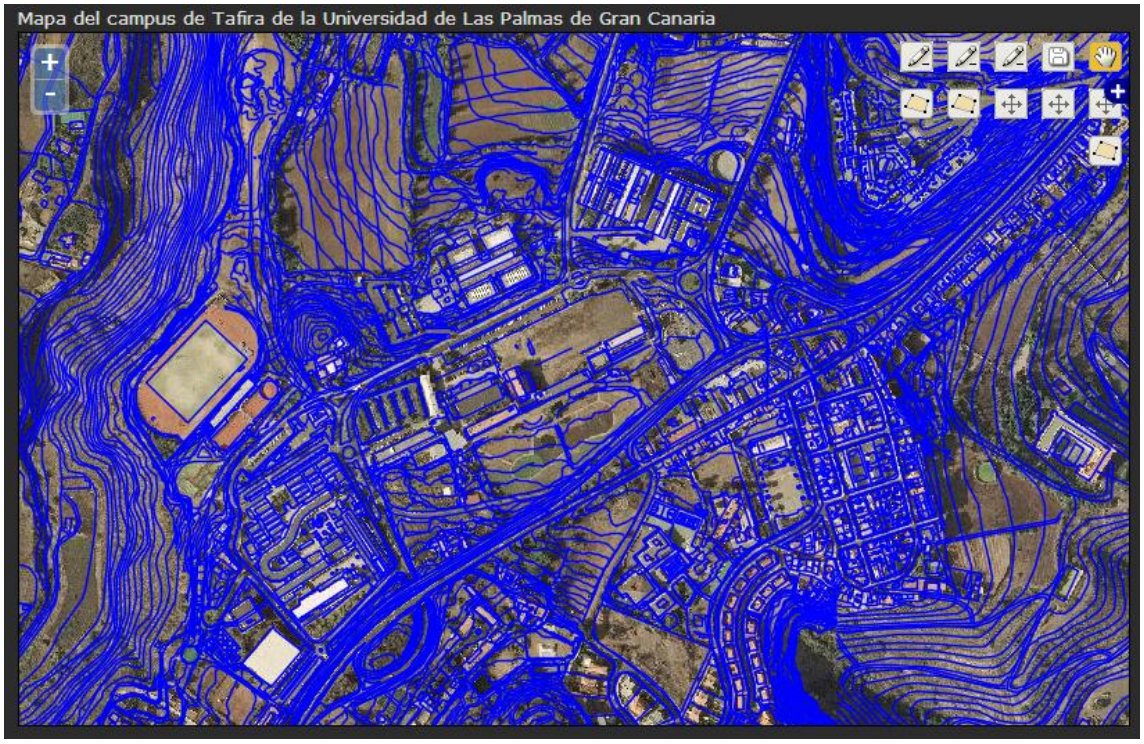


Figura 12: Visualización de capas WMS y WFS en el campus de Tafira

GeoServer proporciona herramientas para conectarse con PostgreSQL pero éste debe tener las estructuras necesarias para que pueda obtener la información correcta. Por cada capa creada, PostgreSQL – en conjunción con PostGIS – debe tener una tabla que, como mínimo, posea un campo de geometría y otro como clave primaria. Para la creación del campo de geometría PostGIS ofrece la función AddGeometryColumn con los siguientes parámetros:

- Nombre de la tabla
- Nombre de la columna
- Proyección geográfica
- Tipo de elemento (punto, línea o polígono)
- Número de dimensiones del mapa (2, 3, 4…)

El nombre de la columna es muy importante de cara a OpenLayers – que requiere que sea especificado durante la configuración de las capas WFS – y cambiarlo puede dar muchos quebraderos de cabeza. No es obligatorio usar uno específico aunque comúnmente se usa “geom” o “the_geom”.

```
layersWFS[i] = new OpenLayers.Layer.Vector(dataWFS[i][0], {
strategies: [new OpenLayers.Strategy.Fixed(), saveStrategies[i]],
protocol: new OpenLayers.Protocol.WFS({
    url: "http://193.145.155.8:8080/geoserver/wfs",
    featureType: dataWFS[i][1],
    featureNS: "http://www.mapmap.org/g4wd",
    maxExtent: bounds,
    geometryName: "location"
}),
styleMap: dataWFS[i][2],
renderers: renderer
});
```

Figura 13: Configuración de una capa WFS

Comprobado el correcto funcionamiento e intercomunicación entre PostgreSQL y GeoServer quedaba por permitir a OpenLayers interactuar con ellos a través del protocolo WFS-T.

OpenLayers no solo proporciona herramientas de edición, también proporciona un tipo de estrategia, llamado *save,* que ejecuta la actualización en la base de datos a través de GeoServer de los cambios realizados sobre el mapa. Para ello, el usuario que realiza los cambios debe estar conectado a GeoServer y tener permisos para modificar dicha capa. Esta conexión, cuando la web se encuentra dentro del servicio de servidor web de GeoServer, se realiza justo en el momento del salvado, saltando un mensaje de login. Cuando se utiliza otro servidor web se requiere realizar la conexión de manera distinta, como se verá posteriormente.

Los controles de OpenLayers son herramientas (añadir, editar, eliminar…) que se crean y se añaden al mapa durante su inicialización. Cada capa debe tener las suyas propias o simular que son únicas eliminando las de una capa y creando las de la siguiente que se va a usar. También permiten ser activadas o desactivadas durante cualquier momento de la ejecución, principalmente porque deben evitar entrar en conflicto entre ellas. OpenLayers ofrece un panel que se sitúa sobre el mapa y controla automáticamente todos estos cambios de selección de herramientas. Si se desea hacerlo desde una interfaz propia, como es el caso de este trabajo, ese control se realizará a través del código desarrollado.

Figura 14: visualización de líneas y polígonos sobre el campus de Tafira

Además de los objetivos planteados, se empezó a trabajar en el diseño de la interfaz para extraer los controles del área del mapa. Éstos se dividieron en dos secciones principales:

- **Capas del Mapa** Permite activar o desactivar la visualización de las capas a través de un botón asociado a cada una de ellas.
- **Controles** Incluyen cuatro categorías según las herramientas que ofrece OpenLayers: Navegación, Adición, Edición y Eliminación. Para cada una de estas herramientas hay que especificar sobre qué capa se desea actuar.

Figura 15: Sección de control de la aplicación

10.2.3 Eventos de OpenLayers

La manera que ofrece OpenLayers de controlar las acciones del usuario sobre las primitivas es a través de eventos. Cada capa tiene los siguientes eventos asociados:

- **BeforeFeatureModified** Se lanza la función asociada antes de modificar una primitiva.
- **FeatureModified** Se activa cuando la primitiva está siendo modificada.
- **AfterFeatureModified** Se activa después de que la primitiva haya sido modificada.
- **BeforeFeatureAdded** Responde cuando se ha añadido un elemento (no solo por el usuario, también al inicio de la aplicación cuando se muestran todas las primitivas de todas las capas).
- **FeatureAdded** Se ejecuta mientras está siendo añadido un elemento.

Existen más eventos, pero estos son los únicos que han sido necesarios para el control de la aplicación.

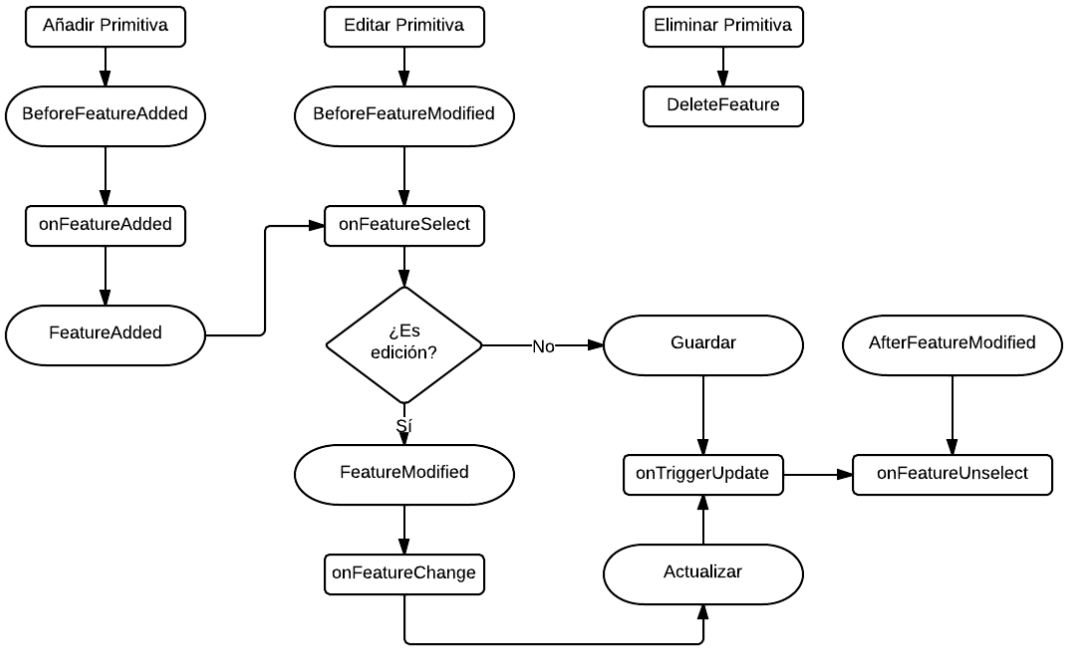


Figura 16: Diagrama del funcionamiento de las operaciones de primitivas

Cuando se añade una primitiva, salta el evento **BeforeFeatureAdded** y llama a la función **onFeatureAdded**. Esta función realiza una inicialización que es obligatoria porque OpenLayers, en sus objetos, ignora los atributos que están vacíos, lo que ocasionaría que no se pudiera modificar posteriormente algunos campos o que los relacionados con el estilo – como el color de una primitiva – no se vieran inicialmente, dejando elementos erróneos o invisibles. Una vez se tienen los campos con valores por defecto, se le da la oportunidad al usuario de definirlos según considere. Esto sucede en el evento **FeatureAdded** y llama a la función **onFeatureSelect**, cuya funcionalidad – mostrar los atributos asociados – es compartida también durante la edición y la visualización de PopUps. Finalmente, los cambios realizados son guardados a través de la función **onTriggerUpdate**.

Durante la edición de primitivas se pueden activar tres eventos; primero se ejecuta **BeforeFeatureModified** lo que lista los atributos asociados del elemento seleccionado para que el usuario pueda modificarlos. Posteriormente se activa **FeatureModified**, que llama a la función **onFeatureChange** y que tiene en cuenta el refresco de las coordenadas de los elementos cuando los cambios se hacen

directamente en vez de escribirlas. Por último, cuando el usuario está satisfecho con el cambio tiene dos opciones: si ha modificado solo la forma o posición del elemento directamente sobre el mapa puede hacer clic fuera de la primitiva, forzando así el salvado automático y lanzando el evento **AfterFeatureModified.** Si por el contrario realizó cambios en los atributos a través del panel puede pulsar el botón *Actualizar*, que llamará a la función **onTriggerUpdate** y posteriormente a **onFeatureUnselect.**

El proceso de eliminar es el más sencillo de todos y lo cubre una funcionalidad de OpenLayers que se encarga de comunicar la orden de eliminar la fila que representa a la primitiva dentro de la tabla de la capa en la base de datos.

10.2.4 PopUps y Gestión de Usuarios

Logrados estos objetivos, se plantearon unos nuevos en la siguiente reunión. La aplicación permitía la edición de los datos geoespaciales pero hasta ahora no se había conseguido nada respecto a los datos extra asociados a las primitivas. Se propuso mostrar PopUps o bocadillos con dicha información. Además, había que diferenciar lo que los usuarios básico y administrador podían ver.

Entre las características de OpenLayers están disponibles los PopUps, los cuales hay que crear, configurar, añadir al mapa y, una vez que no son necesarios, eliminarlos. Consisten en un diseño con forma de bocadillo que contiene un DIV de HTML. Dentro de ese DIV se puede definir cualquier tipo de contenido usando código HTML, lo cual permite una gran flexibilidad a la hora de mostrar información.

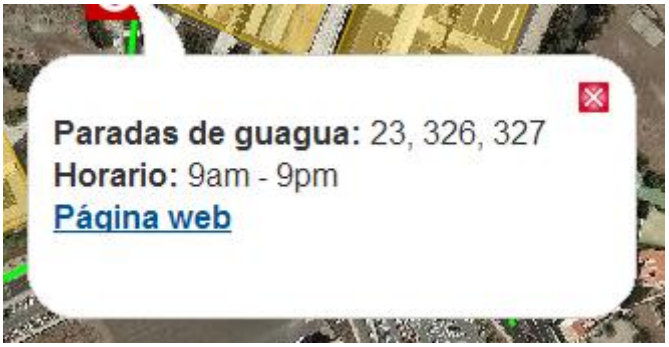


Figura 17: PopUp generado desde OpenLayers

Hasta ahora la interacción con las capas WFS se limitaban a la creación, modificación o eliminación de las prmitivas y en el caso de las capas WMS, a desplazarse sobre el mapa. Con la incorporación de los PopUps había que diferenciar cuándo se clicaba sobre un elemento para visualizar su PopUp y cuando se hacía para modificarlo. OpenLayers no permite tener activas al mismo tiempo las herramientas de selección y de edición, lo que obligó a identificar en qué modo se encontraba el usuario y qué pretendía hacer con el elemento WFS.

```
var popup = new OpenLayers.Popup.FramedCloud(
    "PopUp",
    feature.geometry.getBounds().getCenterLonLat(),
    null,
    content,
    null,
    true,
    onPopupClose);
feature.popup = popup;
map.addPopup(popup);
```

Figura 18: Configuración de un PopUp

El primer planteamiento que se realizó para la administración de usuarios fue a través de PHP, generando una web básica y sin controles para un usuario anónimo o estándar y una web mucho más completa para un administrador que se hubiera identificado previamente. Como el servidor web de GeoServer es limitado, se buscó la manera de permitir la ejecución PHP en el IIS de Microsoft y trasladar la aplicación a esa nueva ruta. Esto generó un problema de cruce de dominios que requería asociar un proxy a OpenLayers para solventarlo, pero la documentación sobre OpenLayers comunicándose a través de IIS es muy pobre – como se ha comentado anteriormente, comúnmente se utiliza el servidor web Apache – y aunque se consiguió que un proxy mostrara los elementos WFS, no se consiguió realizar la identificación del usuario en GeoServer como administrador para poder guardar cambios a través de WFS-T. La alternativa elegida fue aprovechar la potencialidad de JavaScript junto a jQuery para crear una interfaz que, dependiendo del usuario identificado, mostrara los controles o no. Esto requirió controlar el login del usuario con JavaScript para saber cuándo está conectado como administrador, cuya solicitud se resolvió mediante una consulta AJAX al propio GeoServer. Además, para facilitar la navegación del usuario, se utilizó un sistema de Cookies con JavaScript para saber cuándo el usuario estaba conectado, de tal forma que en caso de refrescar la aplicación o cerrar accidentalmente la pestaña, no se perdiera la conexión.

10.2.5 Rediseño de la Interfaz

En un primer momento se pensó en crear una web alternativa para smartphones y tablets que se cargara cuando la aplicación detectara que se accedía desde uno de éstos, pero durante el desarrollo de la interfaz en jQuery se observó que jQuery Mobile presentaba una gran potencialidad para desarrollar interfaces para dispositivos móviles que también podían funcionar adecuadamente en entornos de escritorio. Por ello se hizo un completo rediseño de la interfaz con el framework jQuery Mobile, que tuviera en cuenta los inconvenientes de uso a través de pantallas táctiles o con ratón y teclado para llegar a un compromiso entre ambos.

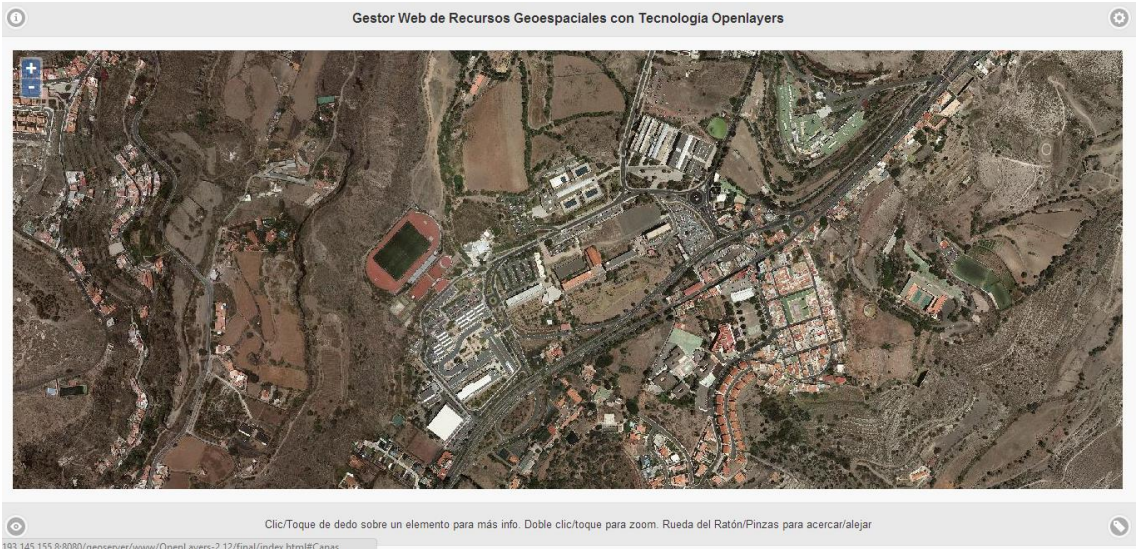


Figura 19: diseño de la interfaz implementada con jQuery Mobile

La nueva interfaz trajo consigo varias correcciones, entre las que se encontraban mejoras de usabilidad. Al poder manejarse con pantallas táctiles los controles fueron distribuidos a lo largo de la pantalla, evitando así que un toque en la pantalla afectara a más de un control a la vez. En cada esquina se situó un botón para un grupo de funcionalidades concretas:

- En la esquina superior izquierda se situó un panel desplegable de ayuda explicando el manejo de la herramienta.

- En la esquina inferior izquierda se estableció un panel desplegable con las opciones de visualización de capas.
- En la esquina superior derecha se colocó un panel desplegable con las opciones de inicio de sesión que, una vez usadas, cambian automáticamente a las opciones de administración del mapa.
- En la esquina inferior derecha se puso un último panel desplegable para la administración de la información asociada a un elemento WFS y cuyo contenido aparece solo cuando se ha seleccionado o creado un elemento.

El sistema de administración de capas también se simplificó. En vez de seleccionar la capa cada vez que se cambia de herramienta se dio la opción de definir primero la capa sobre la que se iba a trabajar y luego cambiar a cualquier herramienta que se desease, lo cual resulta mucho más intuitivo para el usuario.

10.2.6 Tratamiento de datos no Georeferenciados

Todavía quedaba por resolver el tratamiento de la información asociada a los elementos WFS de cada capa (por ejemplo el nombre de un edificio definido por el área de un polígono) y la opción de modificar las coordenadas de los elementos – vértices – a través de campos escritos y no manualmente con el ratón. Estos añadidos implicaban varias cosas:

- Al añadir un elemento, se debe añadir también la información extra.
- Al modificar un elemento, se debe obtener la información de ese elemento seleccionado y dar la opción de modificar su información.
- Para cada capa se deben mostrar dinámicamente los campos de esta información, ya que variarán de una a otra.

El salvado de esta información se realiza de la misma forma que las modificaciones de la geometría de los elementos, a través de la estrategia de salvado proporcionada por OpenLayers.

El primer cambio necesario para conseguir este objetivo fue modificar las tablas en postgreSQL para contener un campo por cada atributo extra. Al tratarse de tablas existentes, también era necesario recargar el Feature Type en GeoServer, para que reconociera los nuevos campos. El rediseño de la interfaz y el uso de jQuery Mobile facilitó la labor de añadir la información no georeferenciada, ya que los campos de información se podían generar dinámicamente tras la selección de una primitiva.

10.2.7 Estilos

Otro añadido importante fueron los estilos. OpenLayers ofrece una configuración de estilos de puntos, líneas y polígonos bastante variada, permitiendo que elementos de una misma capa tengan estilos propios. Conseguirlo requirió de añadir más atributos a la tabla que gestionaba la capa, de tal forma que cada fila que representaba un elemento, pudiera tener descrita su apariencia en las sucesivas columnas. De esta forma se consiguió que los puntos pudieran tener iconos asociados, las líneas grosor, color y trama, y los polígonos distintos colores internos y externos.

Para incluir estos elementos a la interfaz se aprovechó las ventajas de jQuery Mobile, usando barras deslizadoras cuando el atributo requería de un número o menús desplegables para la elección del icono que representaría el punto. Para los colores se editó la funcionalidad de las barras deslizadoras, consiguiendo que éstas mostraran una casilla de color a su izquierda que cambiara según se movía la barra.

Figura 20: Edición de estilos sobre una línea

Respecto al mapa WMS, se volvió a usar el de IDECanarias ya que permitía una mayor resolución y abarcaba a todo el archipiélago canario, no solo a la isla de Gran Canaria.

10.2.8 Correcciones y Mejoras

Las posteriores reuniones consistieron en la corrección de errores y mejora de la interfaz. En el caso de los PopUps, que hasta ese momento se le pasaba información a través de un cuadro de texto, se le asoció un editor *Lo Que Ves Es Lo Que Obtienes* (del inglés What You See Is What You Get o WYSIWYG) para que su contenido pudiera ser mucho más dinámico y permitiera al usuario añadir los elementos comunes en un texto – negrita, cursiva, subrayado, etc. – de manera más cómoda, sin tener que escribir el código HTML directamente. Para ello se barajaron varias opciones de editores basados en JavaScript y se optó por jQueryTE [H10], cuya compatibilidad con jQuery Mobile era la más alta y además funcionaba correctamente tanto en PC como en Tablets.

Figura 21: Editor WYSIWYG para los PopUps

10.2.9 Funcionalidades Adicionales: Añadir iconos

Cuando la interfaz terminó de cumplir las expectativas esperadas, se procedió a añadir funcionalidades adicionales de SIG. Se propuso permitir al usuario registrado con derechos de administrador subir iconos a la aplicación, para luego usarlos en cualquier capa de puntos que se tenga y también la adición de nuevas capas WFS para definir más información sobre el mapa. Ambas tareas requerían de una comunicación directa con el servidor.

Para los iconos se desarrolló un código PHP de subida de archivos. Debido a que seguía existiendo el problema de cruce de dominios, se optó por desarrollar una sección nueva para estos procesos de administración más delicados y enlazarla desde la aplicación principal. Se creó una carpeta en el servidor para el almacenamiento de los iconos y una tabla dentro de la base de datos que incluyera los nombres de todos los iconos subidos.

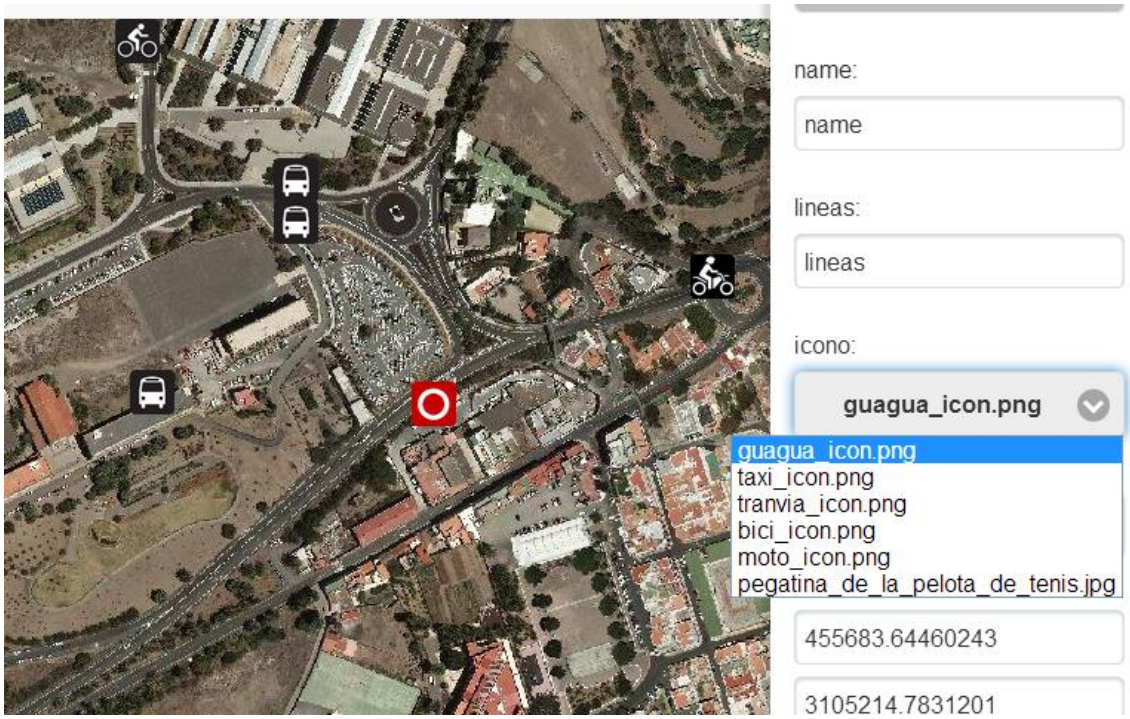


Figura 22: lista de iconos disponible para capas con puntos

La lista de iconos, al haberse convertido en un elemento variable con el tiempo, requirió cambios en la aplicación que obtuvieran al inicio de la misma la lista de iconos que debía mostrar. Esto se consiguió a través de la combinación de PHP, AJAX y el formato JavaScript Object Notation (JSON). Desde PHP se generaba la lista de iconos en formato JSON – expuesto en modo ristra – y a través de AJAX se obtenía dicha lista que era convertida a vector de elementos gracias a las funciones asociadas de JavaScript para JSON. Como esta lista no era necesaria desde el inicio – solo se mostraba cuando se creaba o editaba un punto – la petición AJAX se realizó asíncrona, de tal forma que evitase ralentizar la carga de la aplicación.

```
$cnx = pg_connect($conex) or die ("<h1>Error de conexion.</h1> ".
pg_last_error());

$result = pg_query('SELECT archivo FROM iconos');
$jsonString = "[";
while ($row = pg_fetch_row($result)) {
    $jsonString = $jsonString . '"' . $row[0] . '",';
}
$jsonString = rtrim($jsonString,",") . "]";

echo $jsonString;
```

Figura 23: Consulta para la obtención de la lista de iconos en PHP

10.2.10 Funcionalidades Adicionales: Añadir Capas

Hasta ese momento, solamente se habían definido tres capas WFS como ejemplo y el tratamiento que se realizaba sobre ellas era específico. Permitir que el usuario pudiera añadir capas a su antojo implicaba que todo el código desarrollado funcionase de manera genérica con cada capa, lo que requería una reestructuración del núcleo principal del código.

El flujo básico de la aplicación es el siguiente:

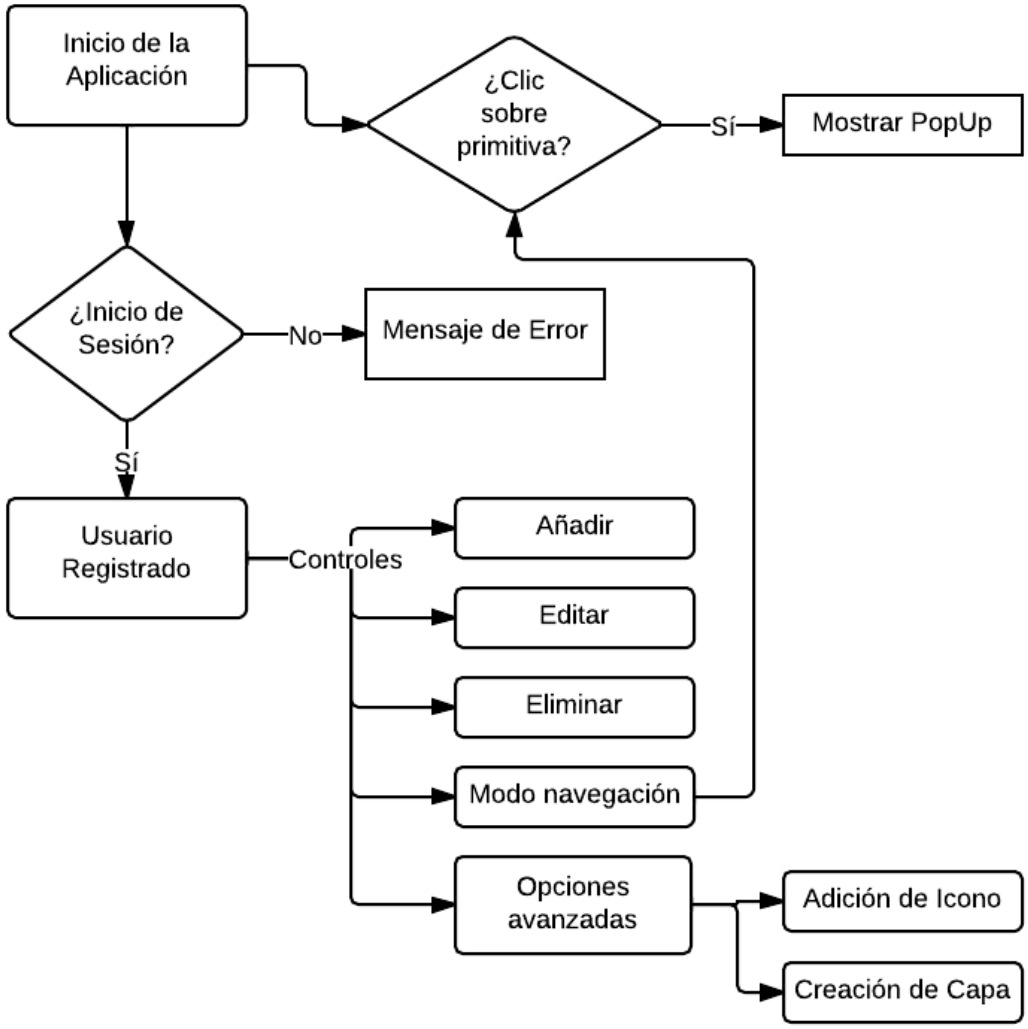


Figura 24: Diagrama del flujo de la aplicación

El tratamiento de las capas, que al principio eran variables independientes, se trasladó un vector cuya dimensión y contenido había que obtener al inicio de la aplicación. Se siguió un procedimiento igual al de los iconos; se creó una tabla en la base de datos que contenía la siguiente información:

- **Nombre de la capa.**
- **Descripción de la capa.**
- **Tipo de capa** punto, línea o polígono.
- **Tipo de estilo** por defecto o especial (con iconos) solo para puntos.
- **Campos asociados** listado de campos de información extra que tiene la capa en formato JSON.

Esta tabla se solicita al inicio de la aplicación con una petición AJAX síncrona, ya que es necesaria desde el comienzo, y con sus datos se configuran las capas WFS en OpenLayers.

```
$.ajax({
    url: "http://193.145.155.8/consulta_wfs.php",
    async: false,
    success: function(result) {
       dataWFS = jQuery.parseJSON(result);
       for (var i = 0; i< dataWFS.length; i++) {
            if(dataWFS[i][2] === "myStyles")
            (…)
       }
    }
});
```

Figura 25: Código AJAX para la obtención de la información necesaria para configurar las capas WFS

Al haber incluido las capas dentro de un vector, su gestión se hace de manera más automática, ya que nos permite trabajar con índices y bucles. Las capas ya no son identificadas por sus nombres sino por el tipo de primitivas que usan gracias al campo **Tipo de capa**, ya que dependiendo de ello se deberá realizar una gestión particular, principalmente referida a los estilos – los puntos no tienen tramas o las líneas no tienen color interior por ejemplo –. La lista de **campos asociados** permite conocer qué atributos asociados tiene cada capa y sobre todo, el nombre al que hay que hacer referencia para modificar su contenido.

Una vez automatizado este proceso se podía incluir la funcionalidad de crear capas, ya que la aplicación respondería con cualquier cantidad de capas que se le dé. Para ello se optó por recurrir a PHP, que se comunicaría directamente con PostgreSQL y GeoServer para la creación de tablas, almacenes de datos y capas. La comunicación específica con GeoServer se hace con las funciones de la librería cURL de PHP.

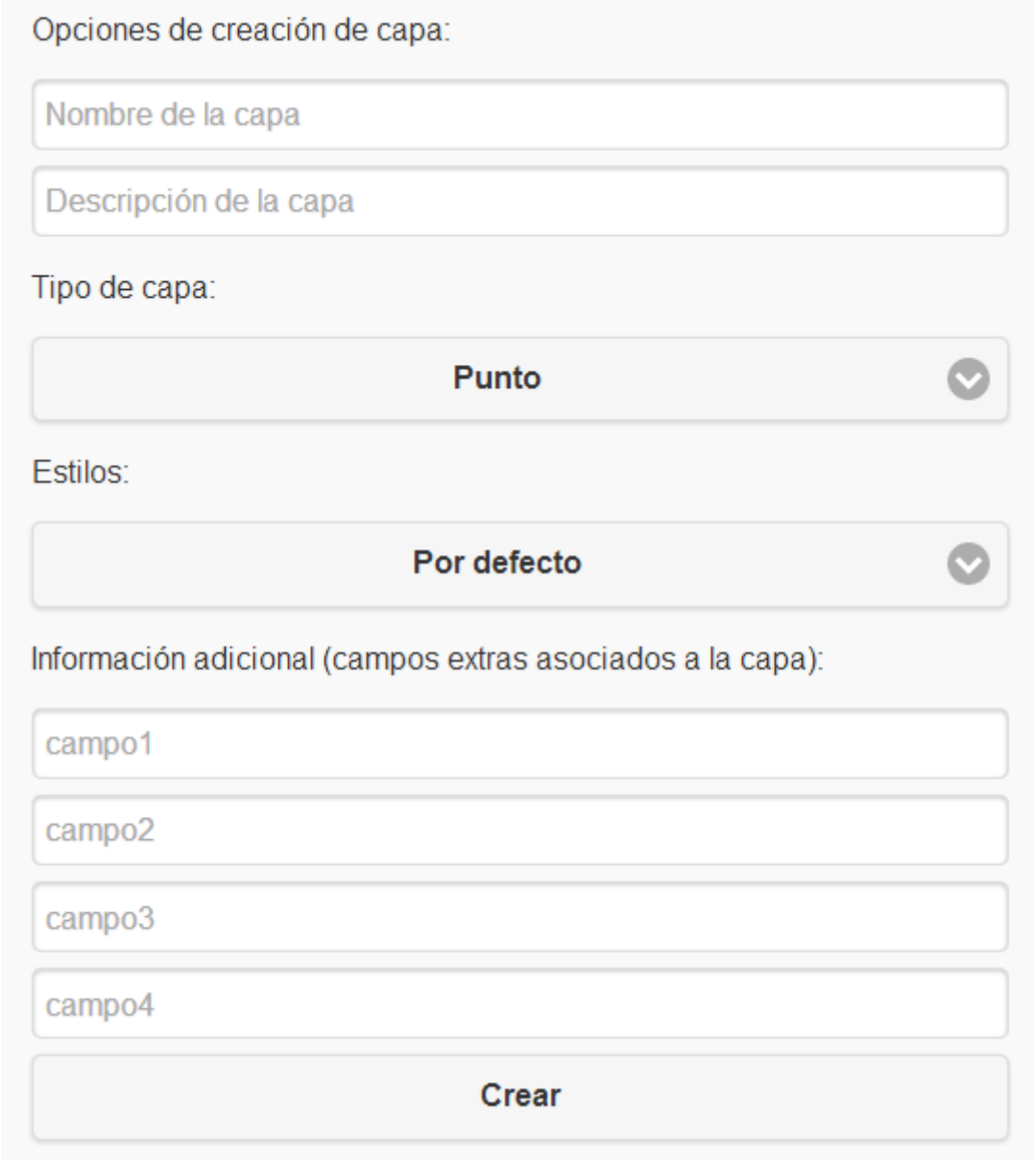


Figura 26: Opciones de creación de capas con PHP

10.3 Pruebas y resultados

Las pruebas realizadas, como se ha comentado en las secciones anteriores, se han ido ejecutando a lo largo del desarrollo de la aplicación, especialmente durante las reuniones de evaluación y mejora.

Disponer de una interfaz facilitaba las labores de testeo de la aplicación. Tareas como activar o desactivar la visualización de una capa solo requiere la pulsación de un botón. También resultaba sencillo comprobar los datos, ya que éstos no se guardan en JavaScript, sino que lo hacen en PostgreSQL y todo dato obtenido llega a OpenLayers a través de una petición a GeoServer. Simplemente con salvar los datos y recargar la página se podía comprobar que el salvado había sido efectivo (otra forma de comprobarlo era mirando directamente la información en la base de datos).

Respecto a las coordenadas, evaluarlas resultaba sencillo al poder comparar las primitivas con las entidades del mapa que representan y sobre las que deben encajar perfectamente. Además, tras cambiar el mapa de Gran Canaria por el proporcionado por IDECanarias – ambos con las mismas proyecciones – se comprobó que coincidían las coordenadas.

La depuración de los errores de JavaScript – OpenLayers y jQuery – se realizó a través de las herramientas de navegación de Chrome [E30] y de Firefox [E31].

Las pruebas no solo se centraron en la funcionalidad, también se evaluó la usabilidad de la aplicación, buscando que la interfaz fuera intuitiva y cómoda de usar, y si requiriese de un aprendizaje, que este fuera sencillo de asimilar y constante durante todo su uso. Es por ello que se añadieron opciones como el “Modo Navegación” cuando se está en modo administrador, para poder usar la aplicación como un usuario básico sin necesidad de cerrar la sesión. También se buscó un tratamiento genérico de las primitivas, independientemente del tipo, mostrando en el mismo panel desplegable toda la información asociada a ellas.

En cuanto a los dispositivos móviles, se comprobó su visualización y funcionalidad, tanto en un Smartphone como en una Tablet.

Figura 27: Visualización de la aplicación en diversos dispositivos

Se encontraron ciertas incompatibilidades a la hora de usar una pantalla táctil. Por ejemplo, cuando los puntos no están definidos por iconos sino por formas geométricas, en el caso de los cuadrados, daba problemas al seleccionarlo con el dedo. Sin embargo, cambiándolo por un círculo, funcionaba sin problemas.

Otro problema descubierto fue el de las líneas, que cuando tenían el ancho mínimo se hacía muy difícil seleccionarlas en una pantalla táctil, así que se dio la posibilidad de definir el grosor de la línea en sus atributos.

11.CONCLUSIONES

El resultado obtenido con respecto a los objetivos planteados ha sido satisfactorio. Se ha conseguido implementar un prototipo de herramienta SIG en formato web basada por completo en Software Libre y cuya compatibilidad con los dispositivos móviles, a falta de testearla en más modelos y navegadores, es bastante alta.

Tras el desarrollo de este trabajo, se extrae como conclusiones la necesidad de unos conocimientos profundos sobre las comunicaciones web, protocolos y configuraciones de servidores web, a fin de conseguir la mayor integración posible entre las herramientas usadas.

Por otra parte, resulta muy importante el diseño de una interfaz con una gran usabilidad, que permita al usuario desenvolverse sobre ella con total normalidad, relacionando acciones con otras aplicaciones similares, de manera que la curva de aprendizaje sea mínima.

Se ha demostrado la potencialidad de OpenLayers para cumplir las funciones básicas de un SIG con eficiencia, y sobre todo, adaptable a las necesidades de cualquier usuario. También se ha probado su compatibilidad y capacidad de integración con otras herramientas.

12.TRABAJOS FUTUROS

Al tratarse de un prototipo, existen una serie de mejoras sobre la aplicación y funcionalidades extras que la completarían aún más o la volverían más robusta.

- **Corrección del IIS para el funcionamiento de la aplicación a través de un proxy o reemplazo por un servidor Apache** Esto permitiría unificar la aplicación bajo el mismo directorio, reducir la carga de JavaScript – ya que se podrían hacer diferentes webs que fueran generadas a través de PHP – y gestionar las sesiones de usuarios con cookies internas de PHP.
- **Opciones de configuración de la aplicación** Actualmente los parámetros como el zoom o la BBOX – la caja que indica el contenido total del mapa solicitado al servidor – están predefinidos. Se podría dar la opción al usuario de configurar estos parámetros u otros que le permitan un uso más cómodo de la aplicación.
- **Selección de primitivas por área** Openlayers ofrece una herramienta de selección por área – dibujando un cuadro de selección – que permite seleccionar varias primitivas a la vez. Editando la funcionalidad de esa herramienta se podría solventar el problema a la hora de seleccionar líneas muy finas desde dispositivos móviles. Otra opción sería implementar botones de primitiva siguiente o anterior, que permitan saltar de un elemento a otro sin necesidad de clicar sobre ellos.
- **Más funcionalidades PHP** A partir del código creado se pueden desarrollar muchas más funcionalidades. Desde modificar información relativa a una capa (nombre de la misma, nombre de campos, añadir campos, etc.) hasta la inserción automática de capas a través de archivos DXF o Shapefiles.
- **Ubicación de usuario** A través de Geolocalización, OpenLayers permite indicar la ubicación del usuario en el mapa. No se consideró una necesidad, pero su implementación daría aún más información a los usuarios que intenten desplazarse por el campus de Tafira, por ejemplo.
- **Opciones de búsqueda** Siendo accesible la información no geoespacial de las primitivas, realizar búsquedas sobre ellas resulta bastante sencillo. Se podría añadir la funcionalidad de un cuadro de búsqueda para resaltar una primitiva en concreto dependiendo de su información asociada.

- **Actualización de herramientas** En este trabajo se utilizó GeoServer 2.0.2 y OpenLayers 2.12 entre otros. Actualmente existen versiones más avanzadas de estas herramientas. Se podrían actualizar dichas herramientas para aprovechar sus nuevas funcionalidades teniendo en cuenta que mantengan su compatibilidad.
- Respecto a las Capas WMS y WFS, éstas son una serie de posibles aplicaciones:
    - Destacar los extintores de un edificio dentro de un plano (a través de una capa WMS e indicando los extintores con puntos WFS).
    - Indicar la cobertura WIFI de los edificios del campus a través de áreas (polígonos) o superponiendo una capa WMS sobre el mapa.
    - Mostrar eventos temporales a través de marcas y PopUps como por ejemplo los exámenes de PAU sobre el edificio correcto, evitando así la confusión de muchas personas que desconocen la ubicación de los edificios.
    - Indicar las distintas facultades que componen el campus y añadir enlaces a las páginas webs de cada una.
    - Mostrar tramos del mapa cerrados por obras
    - Localización de comedores, bibliotecas, salas de estudio, etc.

13.MANUAL DE USUARIO

A continuación se explicará el método de uso de las distintas características de la aplicación. Aunque es bastante intuitiva, hay ciertos aspectos que, si no se han usado SIG anteriormente, pueden ser desconocidos para el usuario.

13.1 Interfaz

Figura 28: Elementos de la interfaz principal

La interfaz principal se compone de cuatro secciones distribuidas en las diferentes esquinas de la aplicación:

1. Sección de ayuda
2. Visualización de capas
3. Sección de atributos de primitivas
4. Opciones de control

13.2 Sección de ayuda

Al clicar sobre la sección de ayuda se desplegará un menú con una explicación acerca de los diversos apartados de la interfaz.

Figura 29: Vista del menú de ayuda desplegable

Este menú está disponible en todo momento para todas las consultas que requiera el usuario.

13.3 Visualización de Capas

El panel desplegable para la visualización de capas permite mostrar o esconder las capas con un simple clic o toque sobre el botón correspondiente. Cada una de las capas está marcada con una casilla que se seleccionará o deseleccionará dependiendo de su estado actual.

Figura 30: Panel para la visualización de las capas

13.4 Sección de atributos de primitivas

Este panel se utiliza principalmente cuando se está modificando una primitiva, en cuyo interior se cargarán todos los atributos asociados a la misma, permitiendo su edición.

13.5 Opciones de control

El panel principal muestra las opciones de logueo para los usuarios administradores. Una vez identificado correctamente, el panel mostrará las opciones de control de la aplicación, que son las siguientes:

- Salir
- Opciones Avanzadas
- Modo Navegación / Modo Edición
- Selección de Capa
- Selección de Herramienta

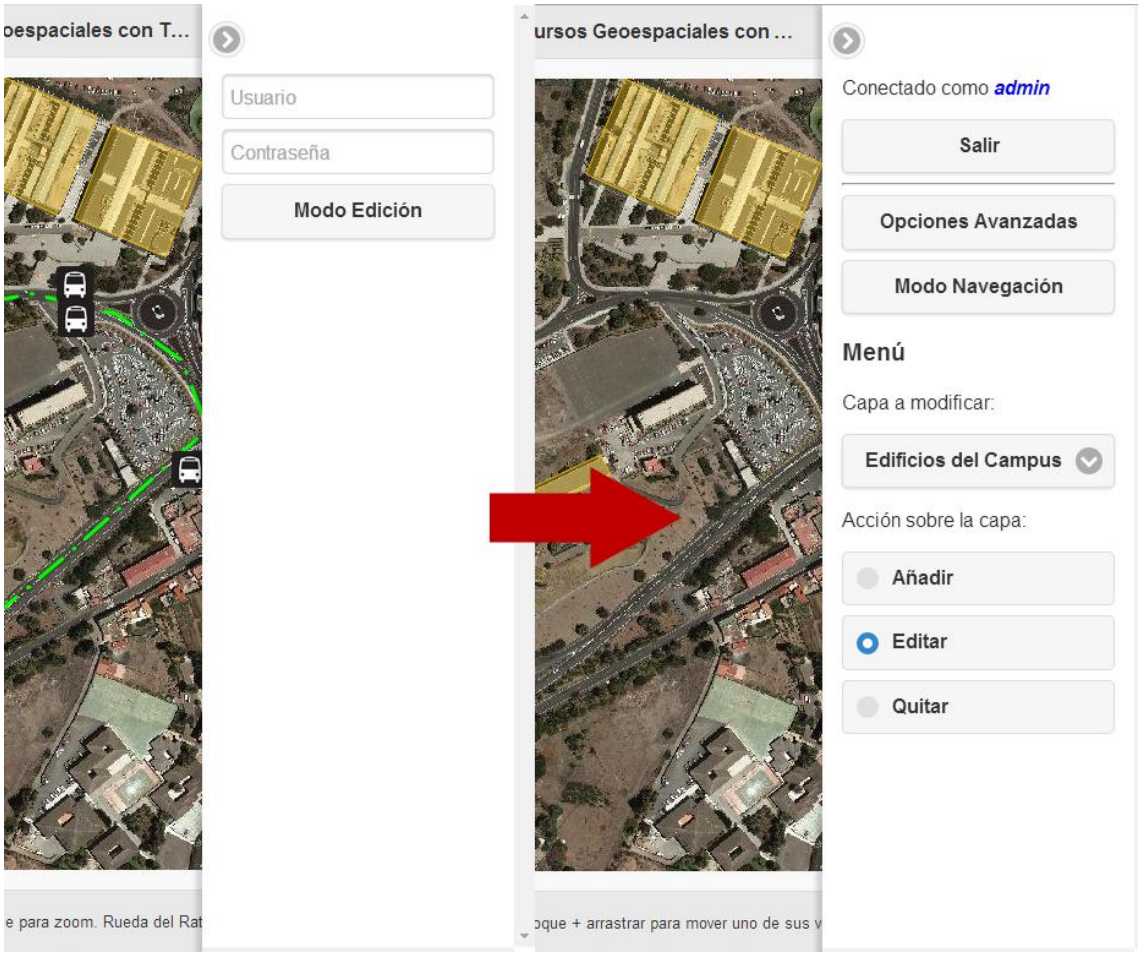


Figura 31: Panel de control antes y después de la identificación del usuario

La funcionalidad del botón de Salir es evidente, cierra la sesión del usuario. La opción de Modo Navegación / Modo Edición permite al administrador usar la aplicación como un usuario básico sin cerrar la sesión, de tal forma que los controles de edición no interfieran en su uso (que no modifique una primitiva accidentalmente) y que pueda volver al Modo Edición con un solo clic.

Las Opciones Avanzadas muestran las funcionalidades extras desarrolladas que son las de añadir un icono a la base de datos y la de añadir una nueva capa.

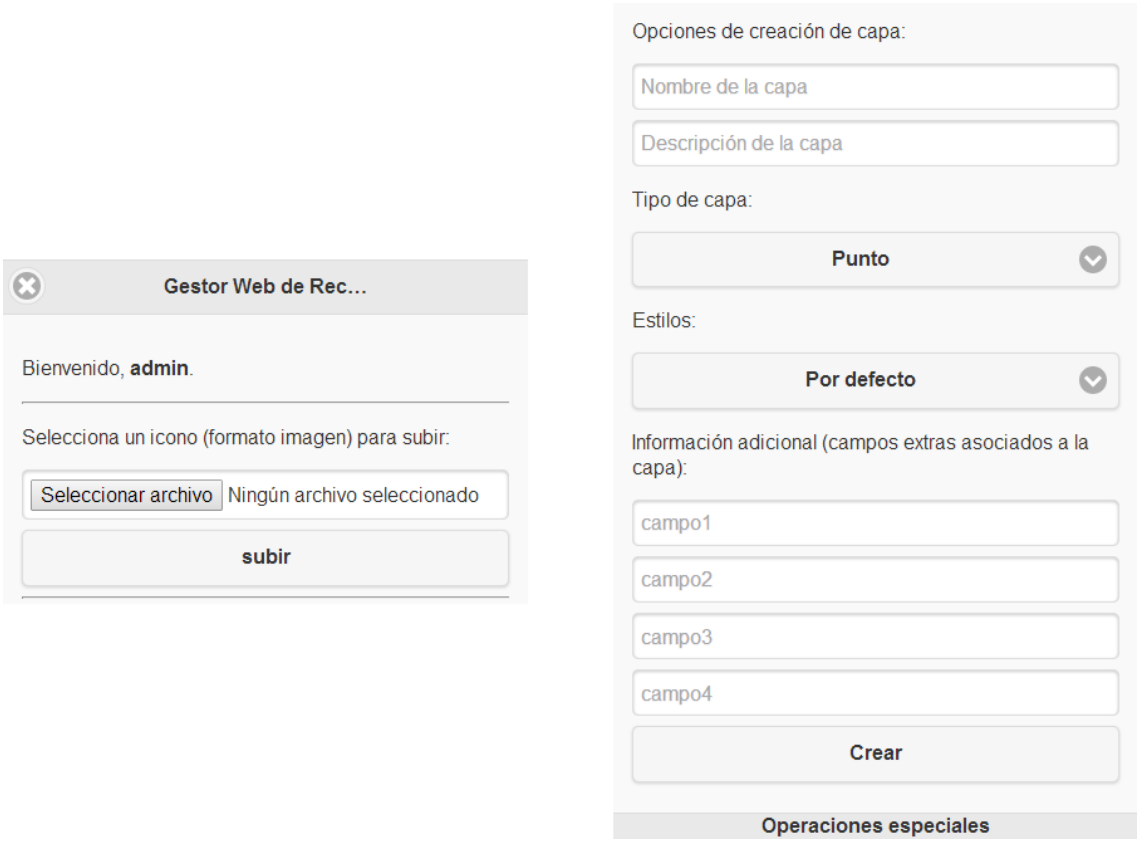


Figura 32: Opciones de inserción de Icono y creación de Capa

Para la inserción de un icono, solo hay que seleccionar un archivo imagen y cargarlo al servidor con el correspondiente formulario. La creación de capas permite algunas opciones más como son la definición del tipo de capa – Punto, Línea o Polígono – y la selección de estilo, si se desea que sea por defecto o que permita mostrar un icono personalizado (solo disponible para la capa de puntos). Además, permite hasta cuatro campos adicionales de atributos no georeferenciados totalmente personalizables.

13.6 Controles de edición

Los controles de edición se dividen en tres:

- Añadir
- Editar
- Quitar

La aplicación reconoce automáticamente el tipo de capa del que se trata, lo que ajustará los controles para que trabajen sobre Puntos, Líneas o Polígonos según el caso del que se trate. Además, en este modo solo se mostrará la capa seleccionada, de tal forma que no confunda al usuario con información adicional.

Cuando se añade un punto, basta con hacer clic o dar un toque en el caso de las pantallas táctiles sobre el mapa. En el caso de las líneas y polígonos se harán tantos clics como vértices tenga la forma, realizando un último doble clic para indicar que se ha terminado de crear. Tras la creación se abrirá automáticamente el panel de atributos para completar la información adicional de la nueva primitiva. Una vez se está satisfecho con el resultado, se ha de clicar en el botón de guardar para que se guarden los cambios.

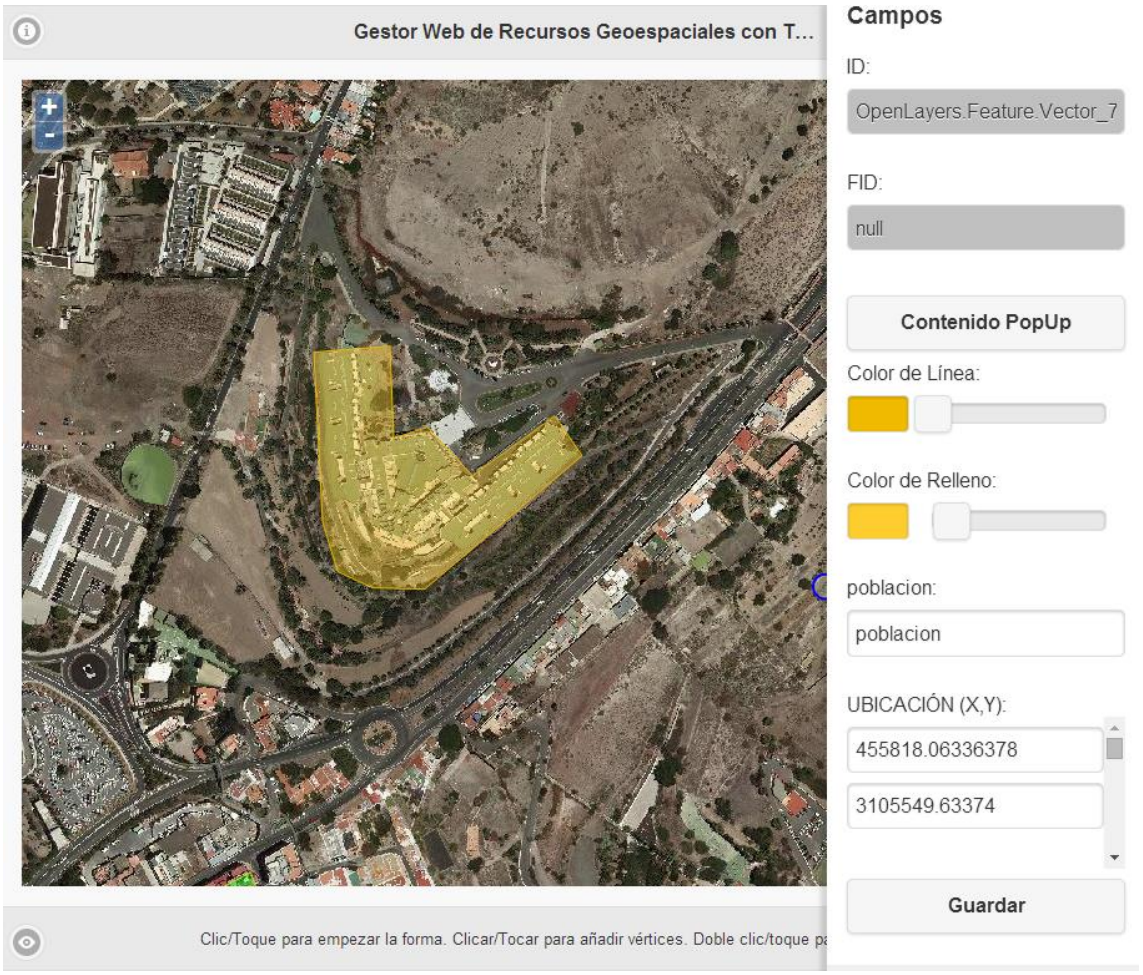


Figura 33: Creación de una nueva primitiva

Todas las capas poseen un botón de Contenido PopUp, que tras clicarlo abrirá una ventana donde definir el contenido que mostrará en su PopUp la primitiva seleccionada.

En la edición de primitivas ocurre algo similar, tras hacer clic sobre la primitiva deseada se abrirá el panel de atributos mostrando la información asociada para su modificación. La forma o posición de la primitiva se puede modificar de dos formas distintas: a través de sus coordenadas en el panel de atributos o directamente sobre la primitiva arrastrando sus vértices (o el vértice en el caso de un punto).

La herramienta quitar permite borrar una primitiva. Para evitar que se elimine accidentalmente una primitiva, se mostrará un mensaje de confirmación antes de cada eliminación.

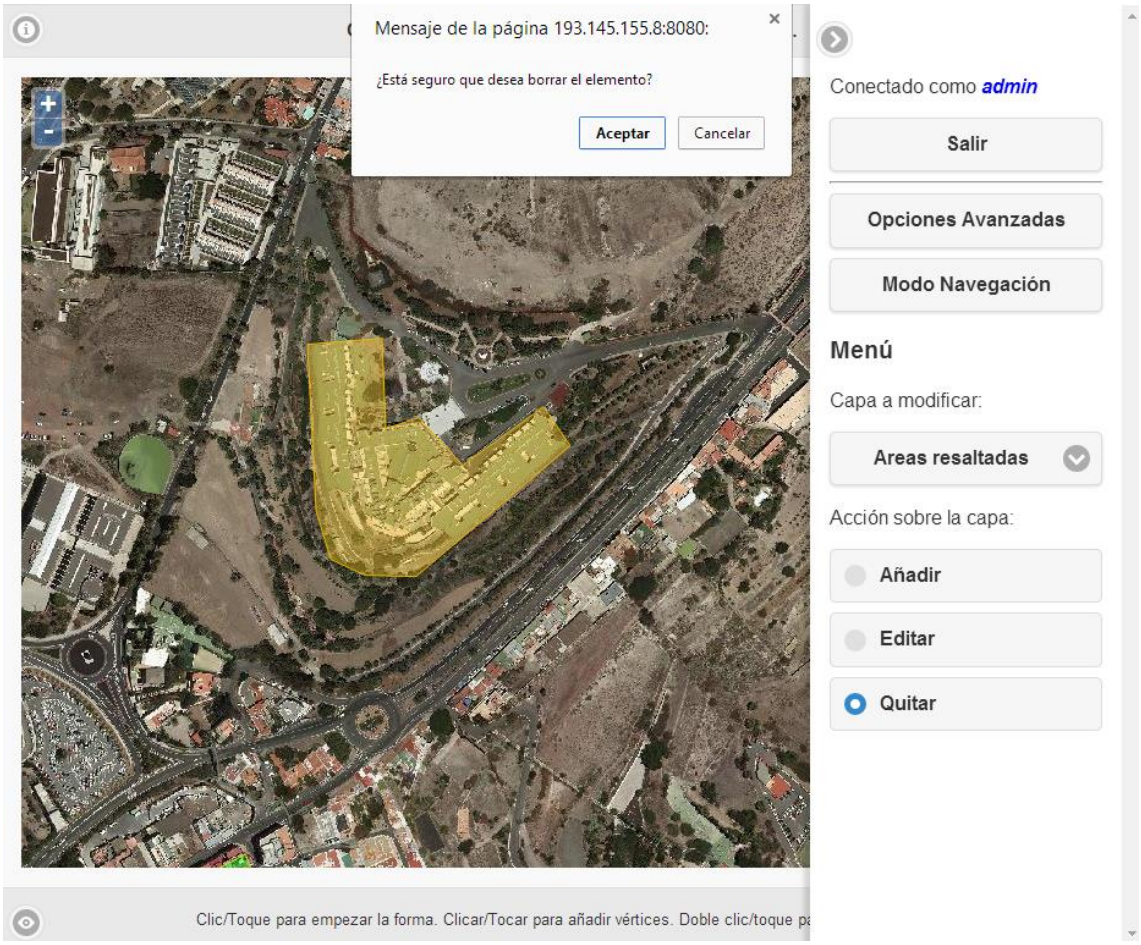


Figura 34: Mensaje de confirmación para borrar una primitiva

14.BIBLIOGRAFÍA

[E01]. *Libro* The Pragmatic Bookshelf | GIS for Web Developers: Adding Where to Your Web Applications by Scott Davis (ISBN-10: 0974514098)

[E02]. *Libro* OpenLayers Cookbook by Antonio Santiago Perez (ISBN: 1849517843)

[E03]. Metodología de desarrollo Iterativo e Incremental

- http://es.wikipedia.org/wiki/Desarrollo_iterativo_y_creciente

[E04]. Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal

- http://www.boe.es/buscar/doc.php?id=BOE-A-1999-23750

[E05]. Directiva 95/46/CE

- http://europa.eu/legislation_summaries/information_society/data_protection/l14012_es.htm

[E06]. Licencia GNU General Public License

- http://www.gnu.org/copyleft/gpl.html

[E07]. Licencia Berkeley Software Distribution (BSD)

- http://www.linfo.org/bsdlicense.html

[E08]. Licencia Apache 2.0

- http://www.apache.org/licenses/LICENSE-2.0.html

[E09]. Licencia MIT

- http://opensource.org/licenses/MIT

[E10]. Licencias Libres Compatibles con la GPL

- http://www.gnu.org/licenses/license-list.html

[E11]. Sistema de Información Geográfica

- http://en.wikipedia.org/wiki/Geographic_information_system

[E12]. Base de datos espacial

- http://es.wikipedia.org/wiki/Base_de_datos_espacial

[E13]. Web Map Service

- http://es.wikipedia.org/wiki/Web_Map_Service

[E14]. Web Feature Service

- http://es.wikipedia.org/wiki/Web_Feature_Service

[E15]. IDECanarias

- http://www.idecan.grafcan.es/idecan/

[E16]. Instituto Geográfico Nacional

- http://www.ign.es/ign/main/index.do

[E17]. Documentación oficial de OpenLayers
- http://docs.openlayers.org/

[E18]. Documentación PostgreSQL
- http://www.postgresql.org/docs/9.0/interactive/index.html

[E19]. Manual de PostGIS
- http://postgis.net/docs/manual-2.0/

[E20]. Manual de usuario de Geoserver
- http://docs.geoserver.org/2.0.2/user/

[E21]. W3school
- http://www.w3schools.com/

[E22]. Stackoverflow
- http://stackoverflow.com/

[E23]. CSS cheat sheet
- http://overapi.com/css/

[E24]. jQuery API Documentation
- http://api.jquery.com/

[E25]. jQuery Mobile API Documentation
- http://api.jquerymobile.com/

[E26]. Documentación del editor HTML de jQuery JQTE
- http://jqueryte.com/documentation

[E27]. Manual de JSON para PHP
- http://www.php.net/manual/es/book.json.php

[E28]. Manual de PHP
- http://www.php.net/manual/es/

[E29]. IDECanarias
- http://www.idecan.grafcan.es/idecan/

[E30]. Chrome Developer Tools
- https://developer.chrome.com/devtools/index

[E31]. Firefox Developer Tools
- https://developer.mozilla.org/en-US/docs/Tools

14.1 Herramientas

[H01]. Notepad++
- http://notepad-plus-plus.org/

[H02]. Navegador Google Chrome

- http://www.google.es/intl/es/chrome/browser/

[H03]. Navegador Mozilla Firefox

- http://www.mozilla.org/es-ES/firefox/new/

[H04]. Página Oficial de OpenLayers

- http://www.openlayers.org/

[H05]. Página oficial de Geoserver

- http://geoserver.org/

[H06]. Página oficial PostgreSQL

- http://www.postgresql.org.es/

[H07]. Página oficial PostGIS

- http://postgis.net/

[H08]. Página oficial jQuery

- http://jquery.com/

[H09]. Página oficial jQuery Mobile

- http://jquerymobile.com/

[H10]. Página oficial jQueryTE

- http://jqueryte.com/

[H11]. Comprobador de código JSON

- http://jsonlint.com/