UNIVERSIDAD DE LAS PALMAS
DE GRAN CANARIA

ESCUELA DE INGENIERÍA DE

TELECOMUNICACIÓN Y ELECTRÓNICA

PROYECTO FIN DE CARRERA

DISEÑO DE APLICACIÓN MÓVIL MEDIANTE TÉCNICAS MULTI-PLATAFORMAS Y NATIVAS ORIENTADA AL SENDERISMO CON MAPAS OFF-LINE.

Autor: Serguey Gilberto Gómez Belyaeva

Tutor: José María Quinteiro González

Fecha: Junio, 2017

UNIVERSIDAD DE LAS PALMAS
DE GRAN CANARIA

ESCUELA DE INGENIERÍA DE

TELECOMUNICACIÓN Y ELECTRÓNICA

PROYECTO FIN DE CARRERA

DISEÑO DE APLICACIÓN MÓVIL MEDIANTE TÉCNICAS MULTI-PLATAFORMAS Y NATIVAS ORIENTADA AL SENDERISMO CON MAPAS OFF-LINE.

HOJA DE FIRMAS

Alumno/a

Fdo: Serguey Gilberto Gómez Belyaeva

Tutor/a

Fdo: José María Quinteiro González

Fecha: Junio, 2017

UNIVERSIDAD DE LAS PALMAS
DE GRAN CANARIA

ESCUELA DE INGENIERÍA DE

TELECOMUNICACIÓN Y ELECTRÓNICA

PROYECTO FIN DE CARRERA

DISEÑO DE APLICACIÓN MÓVIL MEDIANTE TÉCNICAS MULTI-PLATAFORMAS Y NATIVAS ORIENTADA AL SENDERISMO CON MAPAS OFF-LINE.

HOJA DE EVALUACIÓN

Calificación:_____

Presidente

Fdo:

Vocal **Secretario**

Fdo: Fdo:

Fecha: Junio, 2017

A mi madre y a mi mujer por haber estado siempre ahí.

A mis profesores, por su paciencia y dedicación, en espacial a mi tutor, D. José María Quinteiro González.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

I. MEMORIA

1 . INTRODUCCIÓN

En la sociedad en la que nos ha tocado vivir uno se puede dar cuenta de cómo una gran variedad de artilugios electrónicos forman parte primordial de nuestra rutina diaria. Cuesta creer esto cuando no hace ni medio siglo que la mayoría de todos los seres humanos vivían en la total ignorancia con respecto a este hecho; se utilizaban radios, televisores, los ordenadores personales comenzaban a hacer acto de presencia.

Aquella época "analógica" ha mutado, por así decirlo, en una sociedad completamente digitalizada. Ahora, en los países más industrializados del planeta, la amplia mayoría posee un ordenador personal, Internet a gran velocidad, televisiones con la posibilidad de ver contenido mediante Internet, vehículos en los cuales el mayor reclamo es el equipamiento electrónico que tenga como GPS o cámaras de visión trasera para facilitar las maniobras de aparcamiento.

Otra de las novedades tecnológicas que se ha ido consolidando a través de estos años ha sido la del teléfono móvil. Desde que hizo su aparición el primer teléfono móvil[1] hasta hoy en día muchas cosas han cambiado en cuanto a su arquitectura, tanto por fuera como por dentro. Dentro de toda esta travesía, ha habido un punto de inflexión que ha hecho que estos aparatos electrónicos se hayan convertido en imprescindibles para todos nosotros. Hablamos de los llamados coloquialmente smartphones o teléfonos inteligentes.

Aunque no fue el primer smartphone en salir al mercado, el primero fue el IBM Simon Personal Communicator[2], la irrupción de Apple con su iPhone en el año 2007[3], el cual fue proclamado con invento del año por la prestigiosa revista Time[4], marcó el camino a seguir en cuanto al futuro de estos teléfonos. Desde ese año, y en prácticamente una década, el ascenso en cuanto al uso y desarrollo de estos dispositivos ha sido cuanto menos meteórico. Actualmente, es normal que las personas caminen por las calles de las grandes ciudades prestando atención únicamente y exclusivamente a las pantallas de sus respectivos teléfonos inteligentes. Se ha llegado a decir que casi

estamos abducidos por este aparato debido al largo período de tiempo que le destinamos[5] a lo largo del día. La venta de estos smartphones la aglutinan gigantes tecnológicos, entre los que sobresalen dos, Apple y Samsung[6].

Lo que hace diferente estos dispositivos de los teléfonos móviles tradicionales es el sistema operativo que incorporan. Con ellos los smartphones alcanzan la capacidad de ser inteligentes (smart) ya que pueden realizar tareas que con los antiguos aparatos eran poco más que quimeras. Los sistemas operativos más utilizados por estos smartphones son iOS y Android. Para realizar todas esas tareas vienen equipados por procesadores que poco o nada tienen que envidiar a los ordenadores de sobremesa, una gran variedad de sensores (GPS, giróscopo o acelerómetro), capacidad para hacer fotos a grandes resoluciones, etc. Una de las ventajas más llamativas de dichos smartphones son las llamadas tiendas de aplicaciones. En ellas se pueden alojar aplicaciones (software de cualquier índole). Cada sistema operativo posee una de estas tiendas, en las que solo encontraremos aplicaciones hechas con el sistema operativo de turno, no podremos encontrar *apps* hechas con Android en la App Store ni *apps* hechas con iOS en Play Store. Resta mencionar que las más famosas son las pertenecientes a iOS y Android, AppStore y Play Store respectivamente. Es esta versatilidad de poder crear cualquier tipo de software para smartphones la que da sentido a este Proyecto Fin de Carrera.

En un principio, las aplicaciones que se creaban con el objetivo de alojarlas en las diferentes tiendas se tenían que hacer bajos los kits de desarrollo que ofrecían las distintas compañías propietarias de dichas tiendas. Con el paso del tiempo este hecho se ha ido flexibilizando para dar entrada a otras maneras de conseguir crear una aplicación para teléfonos inteligentes. Las nuevas fórmulas logran la creación de software para smartphones por diversas vías tales como tomando como base todo aquello que enrola la creación de páginas web. Otra manera es la de basarse en un solo lenguaje distinto a los nativos para luego mediante diversas técnicas llegar a estos lenguajes nativos. Otra vía, completamente distinta a las demás es la de poder tener la aplicación en servidores desde los cuales ir tomando los datos.

Todas estas formas de crear contenido para los smartphones se le han dado el nombre de multi-plataforma, debido a que el contenido creado con ellas puede ser "exportado" a las distintas plataformas nativas.

1.1 . Objetivos del proyecto

Este proyecto de fin de carrera cuenta con dos grandes objetivos a cubrir. Por un lado se dedica la parte inicial de este documento a explicar de una forma lo más clara y concisa posible dos de las más populares vías para crear aplicaciones multi-plataformas que existen además de comentar el entorno de trabajo diseñado para el mismo fin en el departamento del IUMA de la ULPGC. Se muestran en qué consisten, sus ventajas y desventajas así como ejemplos de su uso.

Específicamente se estudia las siguientes herramientas multi-plataformas:

Appcelerator Titanium: Utiliza un único lenguaje, JavaScript. Permite la creación de aplicaciones tanto para Android como para iOS manteniendo sus interfaces de usuarios casi sin cambios.

PhoneGap: Utiliza varios lenguajes, HTML, JavaScript y CSS. Permite aplicar conceptos de la creación de páginas web a la creación de software para smartphones.

Framework del IUMA: No llega a ser una herramienta multi-plataforma como las otras dos pero permite la consecución de software para diversas plataformas nativas a partir del hecho de tener el contenido de las aplicaciones en servidores web desde los cuales la aplicación se nutre.

Por otra parte, el otro gran objetivo de esto proyecto consiste en la ayuda a la creación de una aplicación poniendo en práctica una de estas herramientas. La aplicación en cuestión tiene como objetivo llevar a los smartphones el Proyecto de Fin de Carrera llamado ***Georruta Transgrancanaria*** de la alumna Bárbara del Castillo-Olivares y Suárez. Para ello se estudian diversas variantes para la integración de los mapas en una aplicación para móviles inteligentes.

1.2 . Estructura del proyecto de fin de carrera

Este proyecto de fin de carrera sigue la estructura clásica, conforme al reglamente de proyectos de fin de carrera de la ETSIT del 18 de Marzo de 2010. Siguiendo dicho reglamento, el documento se divide en cuatro secciones:

- Memoria, parte principal del proyecto. Aquí se explica con todo lujo de detalles todo el trabajo realizado por el proyectando.
- Pliego de Condiciones, parte dedicada a mostrar los requerimientos indispensables para el desarrollo del proyecto.
- Presupuestos, aquí se desglosa las cantidades monetarias necesarias para llevar a cabo el proyecto.
- Anexos, formado por información adicional de utilidad para el proyecto.

A su vez la memoria se divide de la siguiente manera:

- El primer capítulo comprende esta introducción.
- En el segundo capítulo, "Herramientas multi-plataformas: Estado del arte, ventajas y desventajas" realizamos una explicación bastante exhaustiva de tres entornos de desarrollo enfocados a la creación de aplicaciones, comentando sus pros y sus contras además de mostrar algunos ejemplos de su uso.
- En el tercer capítulo, "IUMA Framework" se comenta en que se basa esta idea creada en la ULPGC además de echar un vistazo a todas las tecnologías involucradas.
- En el cuarto capítulo, "Fuentes de mapas" mostramos el recorrido llevado a cabo desde la primera idea que se tuvo para la obtención de los mapas necesarios para la aplicación hasta la última de ellas.

- En el quinto capítulo, “Tratamiento de mapas: Trabajo con Mapbox, QGIS, GDAL y TileMill” se hace un seguimiento exhaustivo acerca del impacto de cada una de estas herramientas en la futura aplicación.

- En el sexto capítulo, “Aplicación resultante” se muestra de una manera gráfica el resultado al que se ha llegado a lo largo de este Proyecto .

- En el séptimo capítulo, se exponen las conclusiones del proyecto además de posibles líneas futuras de investigación.

2 . HERRAMIENTAS MULTI-PLATAFORMAS: ESTADO DEL ARTE, VENTAJAS Y DESVENTAJAS.

En un principio la idea de este proyecto de fin de carrera fue la de utilizar, para la puesta en marcha de la aplicación, alguna de las herramientas de creación de aplicaciones multi-plataformas más potentes que teníamos a nuestra disposición.

Actualmente una de estas herramientas multi-plataformas es Appcelerator Titanium[7], otra es PhoneGap[8]. También estudiaremos la viabilidad del entorno de desarrollo que manejan en el IUMA.

2.1 . Appcelerator Titanium

La primera de ellas solamente utiliza un lenguaje de programación para la creación de aplicaciones multi-plataformas, JavaScript[9]. El trabajar solo con un lenguaje, que además es muy conocido por toda la comunidad, otorga una gran facilidad para el desarrollo de aplicaciones lo que a su vez conlleva a un acortamiento en el tiempo que se le dedica a cada proyecto.

Appcelerator Titanium es una herramienta creada por la compañía llamada Appcelerator en el año 2008[10]. Actualmente, con dicha herramienta, podemos desarrollar contenidos para teléfonos inteligentes o tabletas que trabajen con iPhone o Android mayormente. Desde la versión 5.3 de Titanium, lanzada en Mayo de 2016, se ofrece la posibilidad de desarrollar aplicaciones para Windows 10 aparte de Windows 8.1. Aparte de estas plataformas Appcelerator Titanium provee soporte para la creación de las llamadas “mobile web apps”, las cuales no son más que páginas web adaptadas a los smartphones. Las aplicaciones creadas en ésta última plataforma van encapsuladas dentro de los navegadores web de los móviles.

Titanium pone a disposición de los desarrolladores un gran conjunto de APIs (*Application Programming Interface*). Con dichas APIs tenemos acceso a las diversas partes de los dispositivos móviles como pueden ser el acelerómetro, la agenda, podemos manejar los archivos del teléfono/tableta o acceder a Internet entre otros.

Aplicaciones creadas con esta herramienta multiplataforma presentan una interfaz de usuario mucho más cuidada y parecida a las que se pudieran crear usando los lenguajes de las plataformas nativas, como Objetive-C[11] para iOS o Java[12] para Android. Esto se consigue gracias a que la API de Titanium provee casi la totalidad de la interfaz de usuario de las principales plataformas nativas, tales como botones, ventanas, switches, etc. Si no fuera así tendríamos que crear, por ejemplo, los botones aplicándole un estilo que se le pudiera parecer a alguna plataforma nativa, aquí ya tendríamos que utilizar otra herramienta además de JavaScript.

2.1.1 . Appcelerator Studio

El entorno de desarrollo de la herramienta, llamado Appcelerator Studio, nos permite crear todo tipo de aplicaciones de una manera muy intuitiva. Lo podemos descargar fácilmente desde la página principal de Appcelerator, para el caso de Windows el instalador tiene incluido el software adicional necesario para su correcto funcionamiento los cuales son el kit de desarrollo para Java(*Java Development Kit*),el paquete Git para poder trabajar con GitHub[13] y la librería Node.js[14].

Una vez que tenemos lo anterior, el siguiente paso consiste en instalar todo lo requerido en el desarrollo para aplicaciones nativas soportadas por la herramienta. En la siguiente imagen se puede apreciar la pantalla principal de la herramienta, Figura I.1, en el recuadro rojo vemos cuales son las plataformas para las que podemos desarrollar además de corroborar si las tenemos descargadas. Cabe decir que el desarrollo para iOS solo lo podemos hacer desde un ordenador con sistema operativo de Apple a la vez que el desarrollo para Windows tendremos que hacer bajo ordenadores con SO de Windows.

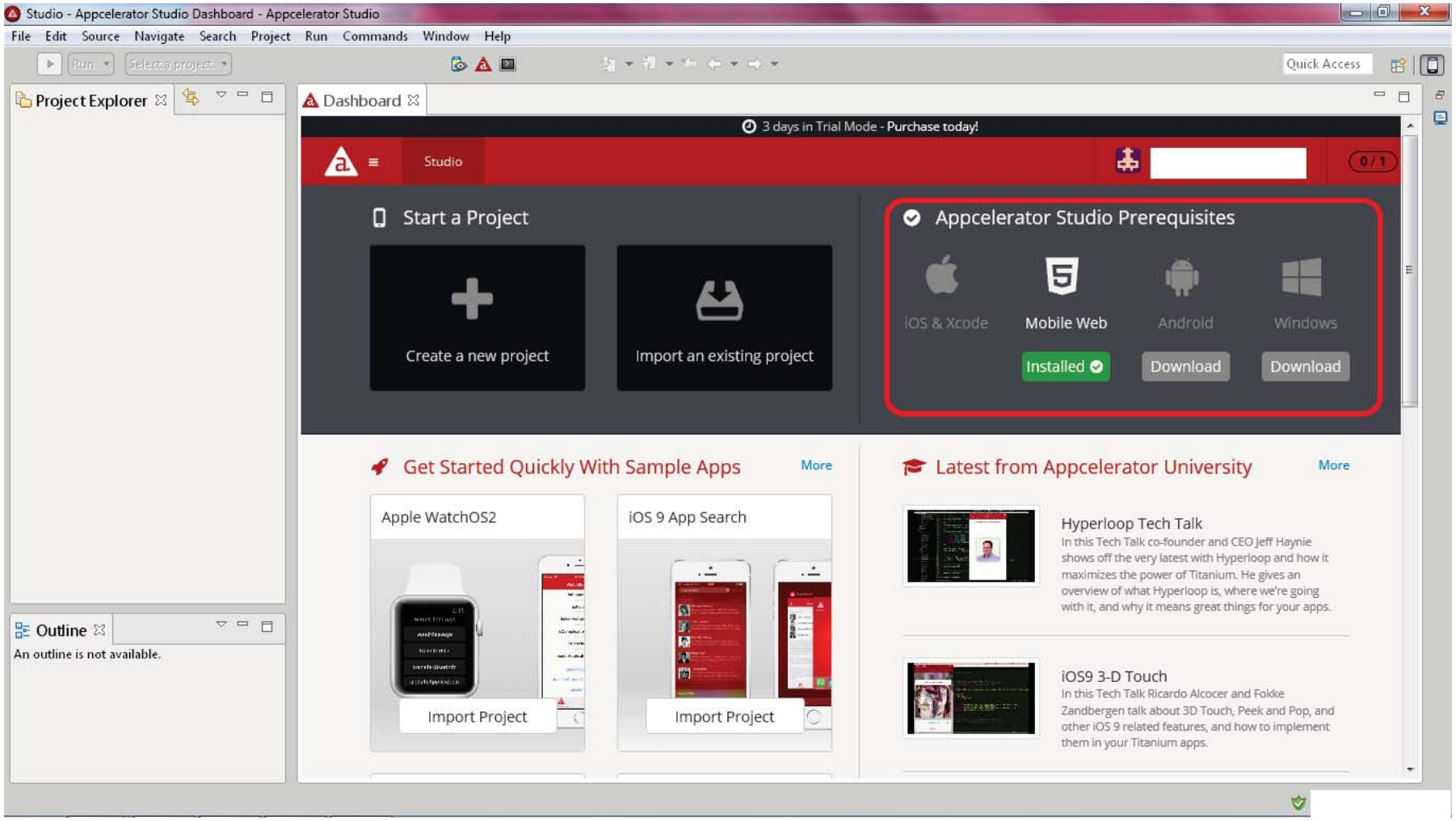


Figura I.1: Entorno de desarrollo Appcelerator Studio

2.1.2 . APIs de Appcelerator Titanium

Appcelerator Titanium provee al desarrollador una amplia variedad de APIs para así poder crear aplicaciones que permitan sacarle el máximo potencial a los smartphones. Ejemplo de ello pueden ser Titanium.Media[15] que permite grabar vídeos y obtener fotos además de poder reproducirlos. Con Ti.Accelerometer[16] (es igual que Titanium.Accelerometer) tenemos la capacidad de saber en cualquier momento en que posición se encuentra el dispositivo. Mediante Ti.Analytics[17] podemos generar información muy valiosa para cualquier desarrollador ya que podremos saber cuántas veces se ha generado un determinado evento por poner un uso. Como último ejemplo me parece interesante hablar de Ti.UI[18], este módulo este el encargado de las interfaces de usuarios nativas y su interacción. Más específicamente, podemos acceder mediante Ti.UI.Android a métodos, eventos o propiedades específicas de esta plataforma. Lo mismo es aplicable a Ti.UI.iOS, Ti.UI.iPAD, Ti.UI.iPhone, Ti.UI.Windows o Ti.UI.MobileWeb.

2.1.3 . Alloy

Por otra parte, Appcelerator facilita otro enfoque para la creación de aplicaciones basadas en el paradigma Modelo-Vista-Controlador o MVC (Model-View-Controller) llamada Alloy[19]. Este framework incorpora además soporte de las librerías Backbone.js[20] y Underscore.js[21]. De esta manera la aplicación queda dividida en tres grandes bloques:

- **Modelo**: provee la lógica de negocio, conteniendo los datos, reglas y estados de la futura aplicación.
- **Vista**: Se encarga de la interfaz de usuario gráfica. Permite al usuario interactuar con el modelo de datos y los muestra.

- **Controlador**: Hace las veces de intermediario entre la Vista y el Modelo. Responde a eventos, envía órdenes a su Vista asociada.

Una ventaja de esta arquitectura es que se puede reusar código mediante la separación de las funciones. Un ejemplo de esto es que podemos tener varias Vistas para diferentes smartphones manteniendo el Modelo y cambiando pequeñas cosas del Controlador.

2.1.4 . Compatibilidad de Appcelerator Titanium

Con respecto a la compatibilidad, Appcelerator Titanium se puede utilizar bajo varios de los más populares sistemas operativos a la vez que nos da la capacidad de crear aplicaciones para las tres grandes plataformas nativas existentes hoy en día. Con la siguiente tabla, Tabla I.1, se podrá apreciar estos hechos con más claridad y exactitud.

Plataforma móvil/SO	Windows	Apple	Linux
Android	Sí	Sí	Sí
iOS	No	Sí	No
Mobile Web	Sí	Sí	Sí
Windows	Sí	No	No

Tabla I.1: Compatibilidad de la herramienta Appcelerator con las diferentes plataformas móviles más usadas

Como se puede apreciar, las plataformas nativas Windows e iOS solo se pueden utilizar bajo los sistemas operativos Windows y Apple, respectivamente.

2.2 . PhoneGap

Una vez que se ha descrito, a grandes rasgos, el funcionamiento de Appcelerator Titanium pasaremos a realizar el mismo estudio sobre la herramienta de código libre PhoneGap[22].

La andadura de PhoneGap comenzó en el año 2008[23] por la compañía Nitobi en un campus de desarrollo de Apple. La idea era la de crear un framework capaz de hacer funcionar aplicaciones web dentro de los smartphones iPhone. Luego de conseguirlo ampliaron sus miras en la plataforma Android y luego en BlackBerry. En octubre de 2011 Adobe anuncia la adquisición de Nitobi y por consiguiente de PhoneGap[24]. También cabe mencionar que PhoneGap forma parte de la Apache Software Foundation[25].

Esta herramienta ofrece al desarrollador crear aplicaciones multi-plataformas con el mismo software usado para la creación de páginas web, léase HTML, CSS y JavaScript. En otras palabras, las aplicaciones resultantes del uso de esta herramienta son páginas web con acceso a todas las partes del smartphone, Figura I.2.

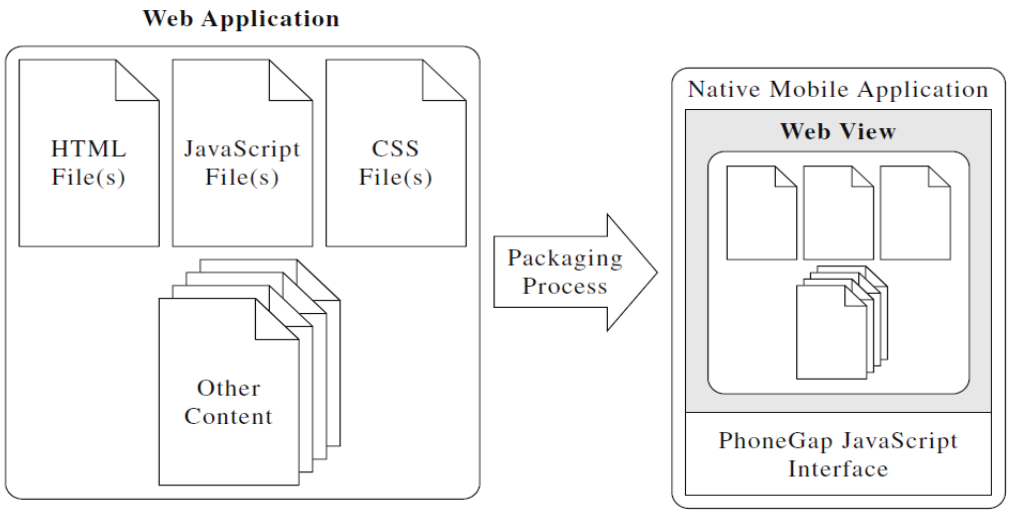


Figura I.2: Arquitectura de la herramienta PhoneGap[26]

Ahondado un poco más, esta herramienta permite incrustar por medio de un *Web View* una aplicación. El *Web View* no deja de ser nada más que un instrumento mediante el cual podemos mostrar una página web[27]. Como se puede apreciar en la imagen anterior, la aplicación nativa esta compuesta de un *Web View* y una interfaz en JavaScript que permite acceder a los componentes propios de los teléfonos inteligentes. Esta interfaz está compuesta de todas las APIs que proporciona PhoneGap, las cuales son capaces de interactuar con las APIs nativas de cada sistema operativo móvil que sea soportado por PhoneGap.

2.2.1 . Herramienta PhoneGap Build

El proceso de compilar las aplicaciones para las distintas plataformas se puede hacer mediante la instalación de los diferentes entornos de desarrollo que hay para cada plataforma además de sus correspondiente SDK o mediante una herramienta online muy novedosa que pone a nuestra disposición PhoneGap llamada PhoneGap Build[28]. De esta manera solo nos tenemos que preocupar de elegir el editor que normalmente utilicemos a la hora de crear páginas web para la creación de las aplicaciones, una vez tengamos la aplicación escrita solo nos queda por subir los archivos que conformen nuestra aplicación (HTML, CSS, JavaScript) a este servicio y dejar que él se encargue de crear las distintas versiones para las plataformas que deseemos y que lógicamente esten soportadas. En este momento las plataformas soportadas por PhoneGap Build son iOS, Android y Windows Phone 8.

Aparte de PhoneGap Build, las aplicaciones basadas en este framework se pueden realizar de la manera más clásica, léase mediante la instalación del software necesario para cada plataforma. Para Android necesitamos instalar Eclipse[29], el cual es un entorno de desarrollo integrado (IDE) bastante popular entre la comunidad y el plugin llamado Android Development Tool (ADT)[30]. Además nos hace falta instalar el SDK de Android, lógicamente, y las librerías necesarias de PhoneGap. En cuanto a iOS tendremos que descargar Xcode[31], que hace las veces de IDE para esta plataforma, y crearnos una membresía en el iOS Developer Program[32] que conlleva

un costo anual de unos 99 dólares. Para Windows Phone tenemos que tener tanto las APIs de PhoneGap como el entorno de desarrollo integrado Visual Studio[33].

2.2.2 . APIs de PhoneGap

PhoneGap ofrece, también, un amplio surtido de APIs[34] que nos ayudaran a enriquecer de gran manera nuestra futura aplicación. Podemos controlar el acelerómetro, utilizar las cámaras de nuestro dispositivo móvil tanto para obtener fotos como vídeos. Tenemos la posibilidad de hacer uso, si lo lleva incorporado, del GPS de nuestro teléfono para así poder ubicarnos con una precisión lo más exacta posible. Por otra parte, garantiza el acceso a los contactos de nuestra agenda además de a otras actividades como la brújula, poder gestionar los archivos y un largo etcétera.

2.2.3 . Plugins en PhoneGap

Poniendo a un lado las APIs que ofrece esta herramienta multi-plataforma, PhoneGap ofrece otra vía para crear contenido adicional que por cualquier razón o motivo no se encuentra en las APIs iniciales. Esta vía, llamada PhoneGap Plugin[35] funciona en todo el entramado de PhoneGap como una extensión, al poder ofrecer una gran variedad de posibilidades. Básicamente cualquier plugin está formado por un mínimo de dos archivos, un archivo JavaScript y otro en el lenguaje nativo correspondiente. El archivo con extensión JavaScript permite la interacción entre la aplicación y el mismo plugin mientras que el archivo escrito en el lenguaje nativo al que vaya dirigido la aplicación hace las veces de interfaz entre PhoneGap y la plataforma nativa. Actualmente existen una gran cantidad de plugins disponibles creados tanto por personal perteneciente a la compañía propietaria de PhoneGap como por desarrolladores externos, a continuación mostramos un listado de estos plugins.

Plugins creados por PhoneGap:

- Device Orientation: Proporciona información acerca de la dirección a la cual apunte el dispositivo gracias a acceder a la brújula del mismo[36].

- Device Motion: Informa de los cambios de posición del teléfono gracias al acelerómetro[37].

Plugins creados por desarrolladores externos:

- Facebook Connect: Como su propio nombre indica, permite el acceso a la famosa red social[38].
- Calendar: Permite añadir eventos al calendario de nuestro smartphone[39].

2.2.4 . Compatibilidad de PhoneGap

Por último pero no menos importante ahondaremos un poco más en el apartado de la compatibilidad que ofrece esta herramienta. Cuando se habla de este tipo de software multi-plataforma lo primero que a cualquier futuro desarrollador interesado en utilizarla se le viene a la mente es saber para cuantas plataformas podremos desarrollar, tener información acerca de las limitaciones que podamos tener con su utilización, etc.

Actualmente, las APIs de PhoneGap pueden ser utilizadas casi en su totalidad por las plataformas nativas más usadas como iPhone u Android, seguidamente mostramos un recuadro dando cuenta de ello.

	Android	Blackberry10	iOS	Windows Phone 8	Windows(8.1, 10, Phone 8.1)	Ubuntu	OS X
Acelerómetro	Sí	Sí	Sí	Sí	Sí	Sí	No
Batería (Estado)	Sí	Sí	Sí	Sí	Windows Phone 8.1	Sí	No
Cámara	No	Sí	Sí	No	Sí	Sí	No
Brújula	Sí	Sí	Sí(3GS+)	Sí	Sí	Sí	No
Archivos	Sí	Sí	Sí	Sí	Sí	Sí	Sí
Geolocalización	Sí	Sí	Sí	Sí	Sí	Sí	No
Reproducción de videos y similares	Sí	Sí	Sí	No	Sí	Sí	No
Redes	Sí	Sí	Sí	Sí	Sí	Sí	No
Notificaciones	Sí	Sí	Sí	Sí	Sí	Sí	No
Vibración	Sí	Sí	Sí	Sí	Windows Phone 8.1	Sí	No
Almacenamiento	Sí	Sí	Sí	Sí	Sí	Sí	No

Tabla I.2: Compatibilidad de la herramienta PhoneGap[40]

En lo referido a las plataformas en las cuales podemos crear aplicaciones basadas en PhoneGap, aparte de incluir el SDK correspondiente a cada plataforma nativa, se puede concluir que para desarrollar aplicaciones para la plataforma nativa Windows 8 necesitaremos un ordenador con el sistema operativo Windows (en sus versiones más actuales tales como Windows 10 o Windows 8) además de cualquier versión de Visual Studio; en el caso de querer desarrollar para Windows pero desde Mac deberemos de tener una máquina virtual desde la cual acceder a Windows.

Para desarrollar para iOS, deberemos de hacerlo bajo ordenadores con sistemas operativos Mac, la última versión de Xcode y ser miembro del programa de desarrolladores de iOS el cual tiene un costo anual de 99 dólares.

El desarrollo para Android lo podremos realizar desde cualquiera de los sistemas operativos más usados hoy en día, tales como Windows, OS X y Linux.

2.3 . Framework del IUMA

El Framework del IUMA tiene su fundamento en la arquitectura MVP (Modelo-Vista-Presentador)[41], la cual es una variante de la arquitectura MVC (Modelo-Vista-Controlador). La idea principal de este paradigma recae en la manera de programar, separándolo todo en tres grandes bloques que son los datos de una aplicación, la interfaz de usuario y la lógica de control. Con este enfoque se ayuda a que la evolución sea mucho más fluida y sencilla. Además permite que se pueda reutilizar código ya que por ejemplo el Modelo podría servir para ser utilizado por muchas Vistas.

Esta arquitectura se utilizó por primera vez en IBM para luego tener más presencia en otra compañía, creada a partir de la primera, llamada Taligent durante la década de los 90. En la actualidad el patrón MVP se utiliza en multitud de frameworks como ASP.NET[42] o Silverlight[43] en entornos .NET y Java Swing[44] o Google Web Toolkit[45] en entornos Java.

Más detalladamente, el patrón de arquitectura MVP se divide en tres partes:

- El **Modelo** es responsable de acceder a la capa de almacenamiento de datos, lo óptimo sería que el **Modelo** fuese totalmente independiente del sistema de almacenamiento. No depende de ningún modo tanto del **Presentador** como de la **Vista**.

- La **Vista** es donde se representa la información proveniente del **Modelo**. No contiene nada de lógica, actúa de forma pasiva. Puede manejar eventos pero no gestionarlos. Esta capa no se limita únicamente a HTML o texto, sino que puede ser utilizada para dar una gran variedad de formatos en función de lo que se necesite en cada momento como por ejemplo vídeo, música y cualquier otro formato que se quiera emplear.

- El **Presentador**, por su parte, se encarga de gestionar las peticiones de los usuarios además de mostrar datos recabados desde el **Modelo** y mostrarlos en la **Vista**. Dicho de otra manera, pueden ser vistos como el administrador ya que tiene la misión de que todos los recursos necesarios para completar una tarea cualquiera se delegue a los trabajadores más cualificados.

El flujo de control de la arquitectura MVP se podría definir por medio de los siguientes pasos, Figura I.3:

- El usuario interactúa con la interfaz de usuario o **Vista** de alguna manera (pulsando un botón o en un enlace).
- El **Presentador** recibe a través de los objetos de la **Vista** la notificación de acción propuesta por el usuario.
- El **Presentador** notifica al **Modelo** la acción del usuario, lo que puede implicar un cambio del estado del **Modelo** a no ser que sólo sea una simple consulta.
- El **Presentador** obtiene los datos provienientes del **Modelo** y los representa en la **Vista**. Aquí el **Modelo** no tiene conocimiento alguno de la **Vista**. Además, el **Presentador** es completamente independiente de la tecnología usada para la creación de la **Vista**.
- La **Vista** espera otra interacción del usuario, lo que dará comienzo a un nuevo ciclo.

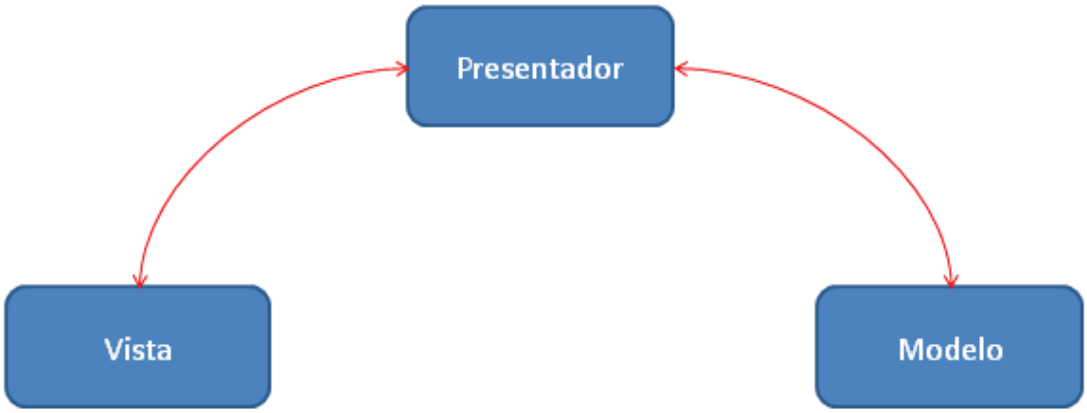


Figura I.3: Representación del paradigma MVP[46]

Una vez realizado este pequeño inciso para comentar de una manera más explícita el modelo MVP, procederemos a ir detallando en qué consiste el entorno de

trabajo creado por los profesores del Instituto Universitario de Microelectrónica Aplicada o IUMA.

A grandes rasgos la creación de aplicaciones bajo este framework se basa en almacenar en la nube toda la información relacionada con la aplicación. En función del contenido ubicado en estos servidores el framework se ocupa de generar toda la estructura de dicha aplicación. Actualmente existe la posibilidad de crear aplicaciones para smartphones que tengan iOS o Android como sistema operativo.

Desde la óptica del paradigma MVP toda la información que se va a utilizar en la aplicación, cómo va a estar estructurada y en donde se va a alojar conforma lo que es el **Modelo**. La **Vista** estará representada por las diferentes interfaces gráficas que cuente la aplicación resultante. Los **Presentadores** serán los encargados de gestionar toda esta información, llevándola desde los servidores hasta los smartphones.

De esta manera y mediante el uso del Framework del IUMA crear una aplicación cualquiera solo se basaría en organizar toda la información de la manera que establece dicho framework para luego hacer uso de los Presentadores creados previamente. En caso de que se necesite hacer uso de alguna función que no este previamente cubierta por los **Presentadores** que conforman esta plataforma de desarrollo se tendría que crear desde cero un nuevo **Presentador**. El nuevo **Presentador** pasaría a formar parte junto con los demás de una especie de base de **Presentadores**, pudiendo ser aprovechado en futuro proyectos.

Como el Framework del IUMA soporta por ahora la creación de aplicaciones moviles tanto para Android como para iOS los **Presentadores** que se necesiten se tiene que construir en los lenguajes soportados por dichas plataformas.

2.4 . Ventajas y Desventajas

En este apartado haremos un analisis lo más objetivo posible acerca de los pros y contras que puedan existir con el manejo de estas herramientas multi-plataformas.

2.4.1 . Ventajas y Desventajas de Phonegap

Comenzaremos diciendo que PhoneGap ofrece todo su framework de manera gratuita, una ventaja nada desdeñable para desarrolladores noveles o para otros a los cuales por el simple hecho de no desembolsar nada le parezca atractiva dicha herramienta. Además, con su utilización nos podemos llegar a ahorra una gran cantidad de tiempo en cuanto al desarrollo de una aplicación debido a que no se tienen que hacer versiones para cada plataforma sino que solo se haría una sola versión. Los lenguajes que se usan en esta herramienta multi-plataforma (JavaScript, HTML5 y CSS) son de los más utilizados por toda la comunidad dedicada a la creación de páginas web dando como consecuencia que la curva de aprendizaje para familiarizarse con PhoneGap sea bastante rápida e intuitiva. Como consecuencia de esto último los costos de desarrollo son significativamente menores a aquellos que se produzcan debido a la utilización de las herramientas nativas.

En cuanto a las desventajas de PhoneGap, se puede decir que presenta varios inconvenientes de gran importancia. Todas las aplicaciones creadas con PhoneGap son en esencia páginas web, esto nos lleva a que cada vez que queramos usarlas en cualquier plataforma tengamos que echar mano un software intermediario como por ejemplo el navegador web de cada plataforma. Esta fuerte dependencia hace que se resienta de manera considerable el rendimiento de la aplicación provocando fallos, no comportarse como debiera o haciendo que la experiencia de uso por parte del usuario sea muy pobre. Las interfaces de usuario son otro gran escollo que presenta el uso de PhoneGap, debido a que cada plataforma nativa presenta una interfaz gráfica bastante diferente de otra. Como consecuencia de esto tendremos que crear las UI desde cero mediante los lenguajes que provee la herramienta para que el resultado final se asemeje lo más posible a lo usado por cada plataforma nativa. De más queda decir que este inconveniente en un porcentaje bastante alto no se llega a resolver del todo. Otra desventaja que presenta, la cual es intrínseca al uso de PhoneGap, es que el acceso al hardware de los dispositivos nunca va ser tan fluido en comparación con las herramientas nativas. Como conclusión de ello en aplicaciones de mayor envergadura y

complejas el rendimiento nunca va a ser el mismo que el que se obtendría con la misma aplicación realizada mediante herramientas nativas como iOS o Android.

2.4.2 . Ventajas y desventajas de Appcelerator Titanium

Con respecto a la segunda herramienta multi-plataforma comentada en este PFC, Appcelerator Titanium, podemos decir que una de sus grandes ventajas es el uso como lenguaje de programación de JavaScript. De sobra esta decir que es un lenguaje bastante conocido por todos los que de una forma u otra hayan realizado algo relacionado con el mundo web. Este hecho facilita en gran medida el salto a esta herramienta por parte de desarrolladores web. Al igual que con PhoneGap, el uso de Appcelerator Titanium acortará los tiempos de creación de una aplicación debido que solo se tendría que realizar una versión y luego obtener las diferentes versiones que queramos, como por ejemplo para iOS o Android. Esta herramienta no presenta el problema que tiene PhoneGap en cuanto al diseño de interfaces de usuario, esto se debe a que con las APIs que proporciona Appcelerator Titanium sea muy fácil crear dichas interfaces. Otra ventaja a tener en cuenta es que podemos hacer uso de Alloy, una herramienta que permite la creación de aplicaciones multi-plataforma desde un enfoque MVC (Modelo-Vista-Controlador). La utilización de Alloy nos hará ganar en versatilidad, ya que usando el paradigma MVC podremos reusar código de nuestras aplicaciones de forma sencilla y cómoda.

En cuanto a las desventajas que podemos inferir del uso de Appcelerator Titanium se podrían destacar varias. En primer lugar se encuentra el hecho de que se tenga que trabajar con los diferentes kits de desarrollo, SDK en inglés, de cada plataforma para la cual se quiera crear una aplicación con Appcelerator. Al tener que descargarlos tendremos que tener cierto cuidado de ir actualizándolos a medida que se necesite. Otra desventaja que afecta a una plataforma nativa en particular es la que para poder realizar aplicaciones para iOS tendremos que estar en posesión de un ordenador con sistema operativo Mac, esto acarrea unos costos que puede que algunos de los que estén interesados en desarrollar con Appcelerator desistan. Por otra parte, el trabajo con esta herramienta multi-plataforma requiere que el individuo adquiera cierto bagaje con

la API de dicho software. Claramente este hecho puede ocasionar un retraso temporal, que para el caso de PhoneGap no existe, que muchos desarrolladores no esten dispuestos a aceptar. De esta última desventaja se puede sacar como conclusión que lo que se ha comentado como una ventaja, en referencia al tratamiento de las interfaces de usuario, se puede convertir en una desventaja debido a la curva de aprendizaje que presenta la herramienta Appcelerator.

2.4.3 . Ventajas y desventajas del Framework del IUMA

La tercera opción comentada en este PFC para la realización de software para telefonos inteligentes tiene varias ventajas. Una de ellas es que al utilizar el código nativo de las plataformas soportadas el resultado es una aplicación que nada tiene que envidiar a las que se cofeccionan solamente utilizado estas plataformas nativas. La experiencia de usuario no se trastoca ni un ápice y el consumo de recursos del telefono movil se optimiza al maximo posible. Gracias al uso del patrón MVP las aplicaciones se pueden crear con una gran rapidez ya que ésta manera de trabajar facilita la reutilizacion de abundante código, esto se traduce en que haya aplicaciones para las cuales no se necesite crear nuevos **Presentadores** al poder ser utilizados los ya creados sin ningún tipo de problema.

Referente a las desventajas que podemos reseñar se encuentra en primer lugar la de que por el momento esta herramienta solo ofrece el poder llevar a cabo proyectos orientados a las plataformas nativas Android e iOS. No sería mala idea el poder ampliar los horizontes al poder añadir nuevos sistemas operativos enfocados a los dispositivos móviles como los creados por Windows, el último es Windows 10 Mobile por proponer alguno.El uso de la estrategia de programación MVP pudiera inducir a tener algunos contratiempos debido a que en algún proyecto en particular no pudiera hacer uso de los **Presentadores** creados con anterioridad, lo que nos llevaría a tener que crearlos desde cero. Esto haría que una de las ventajas principales del modelo MVP como es la reutilización de código no la pudierarmos a aprovechar. Por eso es de vital importancia al utilizar este paradigma de programación ser los suficientemente habilidosos para

obtener el máximo partido utilizando la mayor cantidad de **Presentadores** creados previamente en nuestros proyectos.

2.5 . Conclusiones

A la vista queda que el uso de las dos primeras herramientas multi-plataformas puede resultar muy beneficioso, pero eso sí, en algunos casos. En el desarrollo de pequeñas aplicaciones, que no requieran gran cantidad de recursos del sistema operativo en cuestión pueden ser la opción ideal. Para proyectos de más envergadura, que requieran usos intensivos tanto del hardware como del software de un smartphone, su uso se podría llegar a desaconsejar por los motivos que antes hemos descrito. Es por esta razón principalmente por la que se ha optado por la tercera herramienta en discordia, el Framework del IUMA, como entorno de trabajo principal para la creación de la aplicación móvil.

Nos hemos decantado por este entorno de desarrollo ya que nos facilitará en gran medida la creación de la aplicación para las plataformas Android e iOS. Mediante su uso se ahorrará gran cantidad de tiempo al poder reutilizar gran cantidad de código, previamente escrito para otros proyectos, en la futura aplicación. Además la incorporación de nuevos contenidos a la aplicación se llevará a cabo de una manera eficaz y sin grandes modificaciones en la estructura de dicha aplicación.

En cuanto a la aplicación a desarrollar se ha optado por llevar al mundo de los smartphones(y tabletas) el Proyecto de Fin De Carrera de la alumna Bárbara del Castillo-Olivares y Suárez titulado "*Georruta Transgrancanaria*". En él se lleva a cabo un profundo estudio relacionado con la actividad geológica de una parte de la isla de Gran Canaria. Esta ruta comienza en el Faro de Maspalomas perteneciente al municipio de San Bartolomé de Tirajana y culmina en el Puerto de Las Nieves del municipio de Agaete. El trayecto pasa por diversos puntos de gran interés tanto turístico, geológico o cultural de la isla como el Roque Nublo, las Dunas de Maspalomas o el yacimiento arqueológico de Arteara por poner algunos ejemplos de lugares que se visitan.

La aportación del presente PFC a la aplicación estará basada en todo lo que se incluirá en el llamado Modelo del paradigma *MVP*. Concretamente trataremos con el

manejo de mapas, partiendo desde qué fuentes se obtienen hasta como se incluirán en la aplicación móvil. Todo ese complejo proceso se mostrará de la mejor forma posible en los siguientes capítulos del presente Proyecto de Fin de Carrera.

3 . IUMA FRAMEWORK

Una vez descrito el por qué de la utilización del Framework del IUMA para el desarrollo de la aplicación brevemente expuesta se pasará a comentar en detalle cómo se procederá con dicho entorno de trabajo.

El Framework del IUMA lo dividiremos en tres partes, basándonos en el paradigma MVC, para poder explicarlo de la mejor forma posible. Dichas partes son el **Modelo**, las **Vistas** y los **Presentadores**.

3.1 . Modelo teórico

Empezaremos hablando de todo lo que engloba nuestro **Modelo**. El **Modelo** del Framework del IUMA se ha estructurado en forma de catálogo, de esta forma toda la información contenida en él se puede organizar más eficientemente a la par que su visualización se convierte en algo más claro de cara al usuario final. El catálogo que se genera contiene categorías y subcategorías, teniendo dentro de ellas uno o más productos.

La idea de tratar todo la información como si se tratase de un catálogo proviene de un plugin muy utilizado en el ámbito de creación de páginas web llamado Spider Catalog[47]. Este plugin, creado por la compañía Web-Dorado[48], está disponible actualmente para las plataformas más famosas en cuanto a creación de páginas como son WordPress[49], Joomla![50] o Drupal[51]. Spider Catalog ha sido creado con el objetivo final de mostrar productos en un formato de catálogo.

Cada producto, Figura I.4, en el catálogo puede ser asignado a una determinada categoría, Figura I.5, o subcategoría, haciendo el proceso de búsqueda de productos mucho más fácil para los usuarios. Además, es posible asignar una cantidad ilimitada de parámetros, Figura I.6, para cada categoría para así dar la posibilidad de ofrecer una

información lo más detallada posible de cada producto y acompañar cada producto con una imagen y una descripción que lo identifique. Otra opción que nos parece muy interesante es la de poder importar/exportar los producto en el formato CSV, facilitando así, las tareas de cambiar y actualizar el catálogo.

Figura I.4: Vista de los productos con Spider Catalog

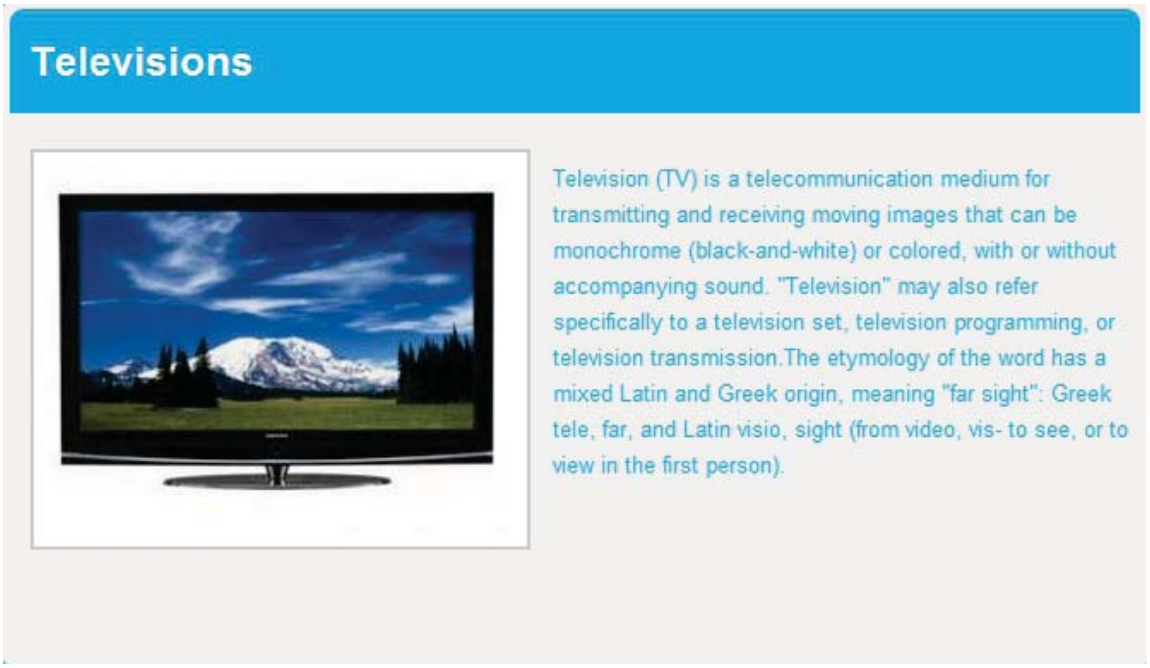


Figura I.5: Vista de las categorías con Spider Catalog

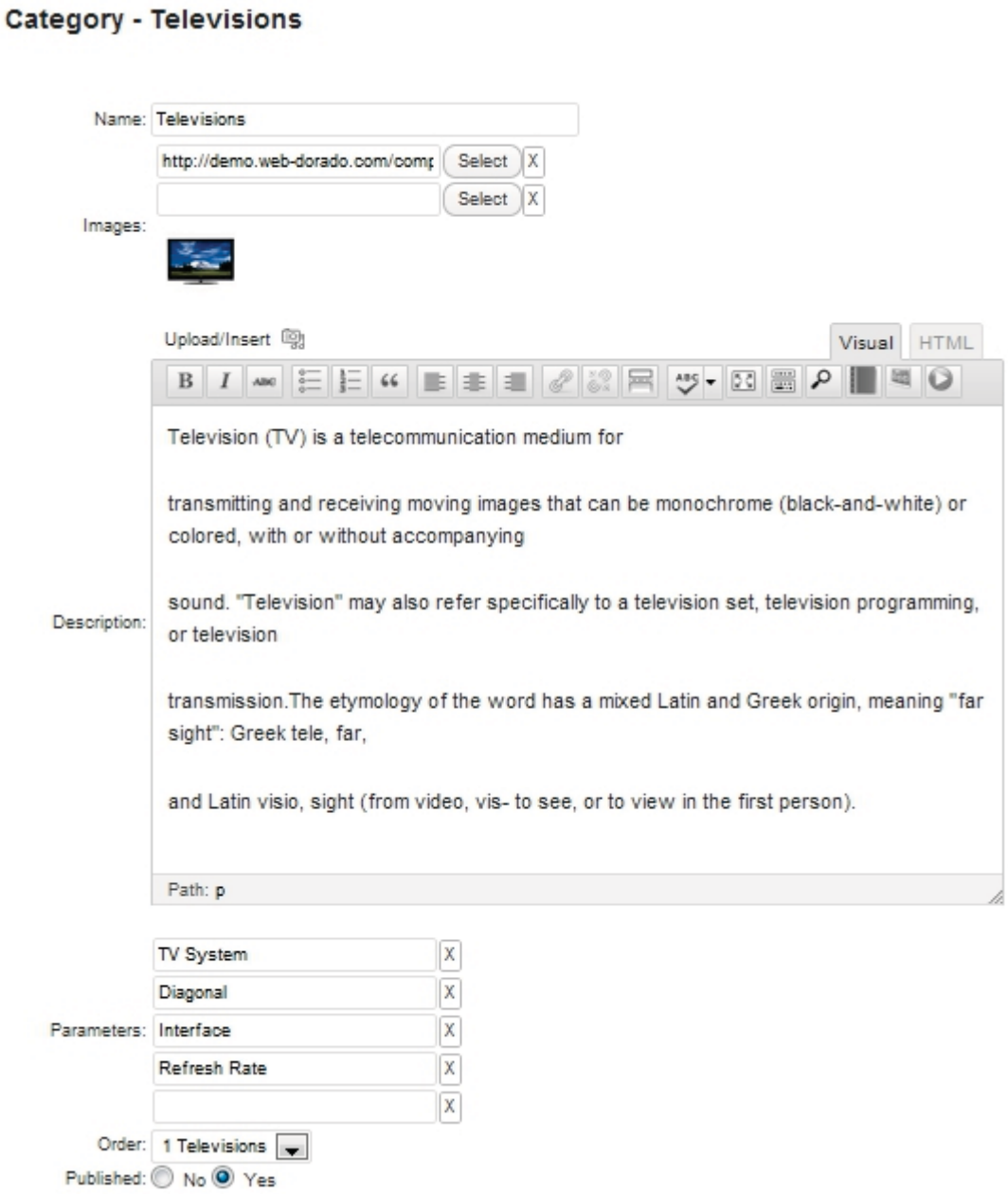


Figura I.6: Modo de creación de categorías con Spider Catalog en WordPress

Toda la información que utilizaremos para la aplicación, configurada de la manera explicada anteriormente, la tendremos que almacenar en alguna parte. Es en este momento donde entra en juego la otra herramienta que utilizamos para el **Modelo**, llamada Parse[52].

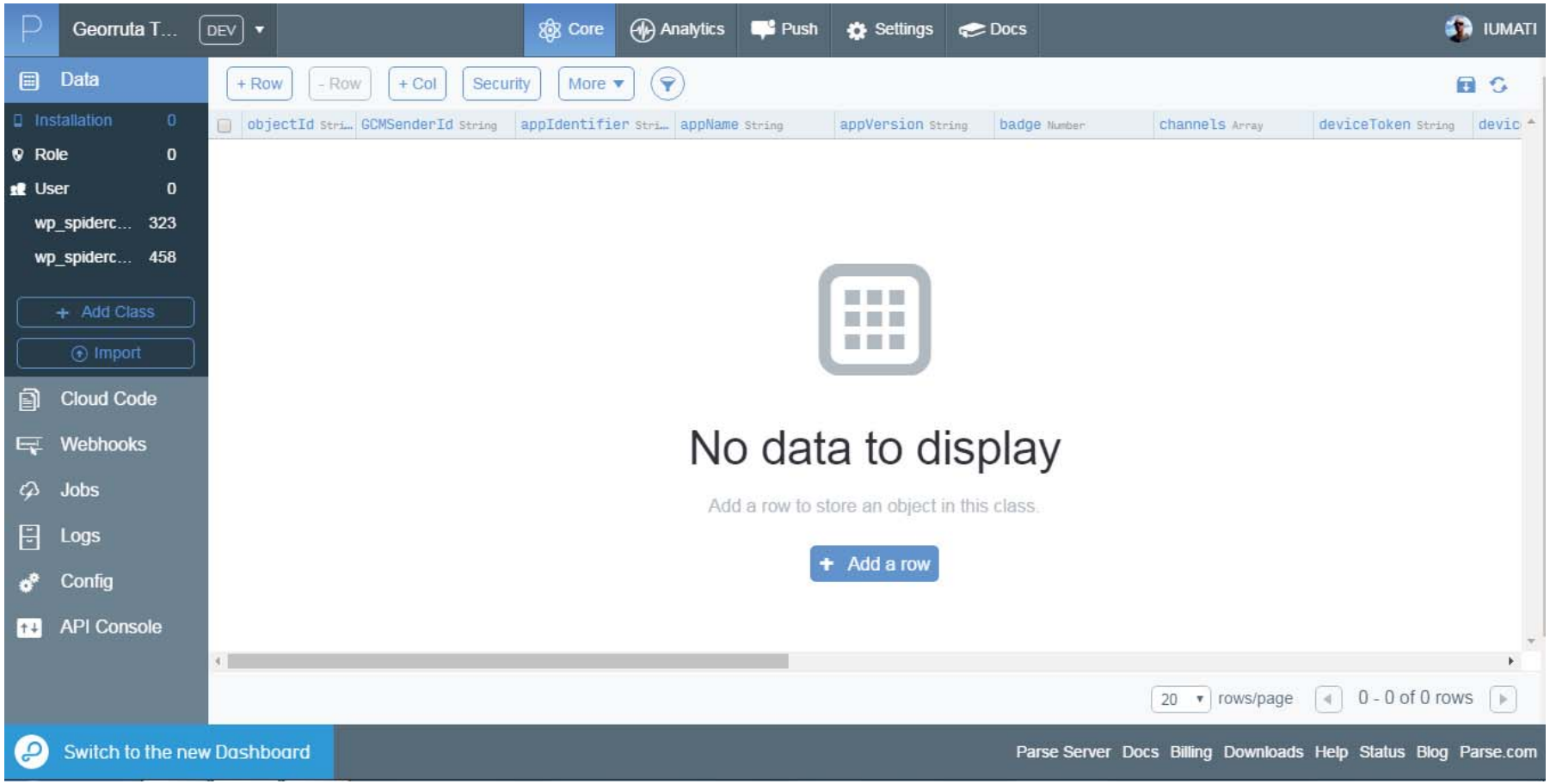


Figura I.7: Vista general de la herramienta Parse

Parse, Figura I.7, es una herramienta que se puede catalogar como **Backend as a Service (BaaS)** muy útil que nos permite hacer uso de multitud de funcionalidades en la nube, favoreciendo el desarrollo de aplicaciones que requieran un backend con relativamente poco trabajo. Los servicios que ofrece Parse son principalmente estos:

- Modelo de datos en la nube. Creación de tablas no-SQL en la nube y capacidades para modificar, insertar o consultar vía API.
- Notificaciones PUSH. Posibilidad de envío de notificaciones a nuestros usuarios, previa aceptación por parte del usuario.
- Código en la nube (Cloud Code). Capacidad para la ejecución de código en el servidor.
- Analytics. Permite saber diversos parámetros importantes sobre el uso de la aplicación por parte de los usuarios.

Además, Parse pone a nuestra disposición SDKs, Figura I.8, para utilizarlo en múltiples plataformas como iOS, OSX, Android, .NET, JavaScricpt o PHP entre otros. Así mismo, permite utilizar REST API de forma estándar.

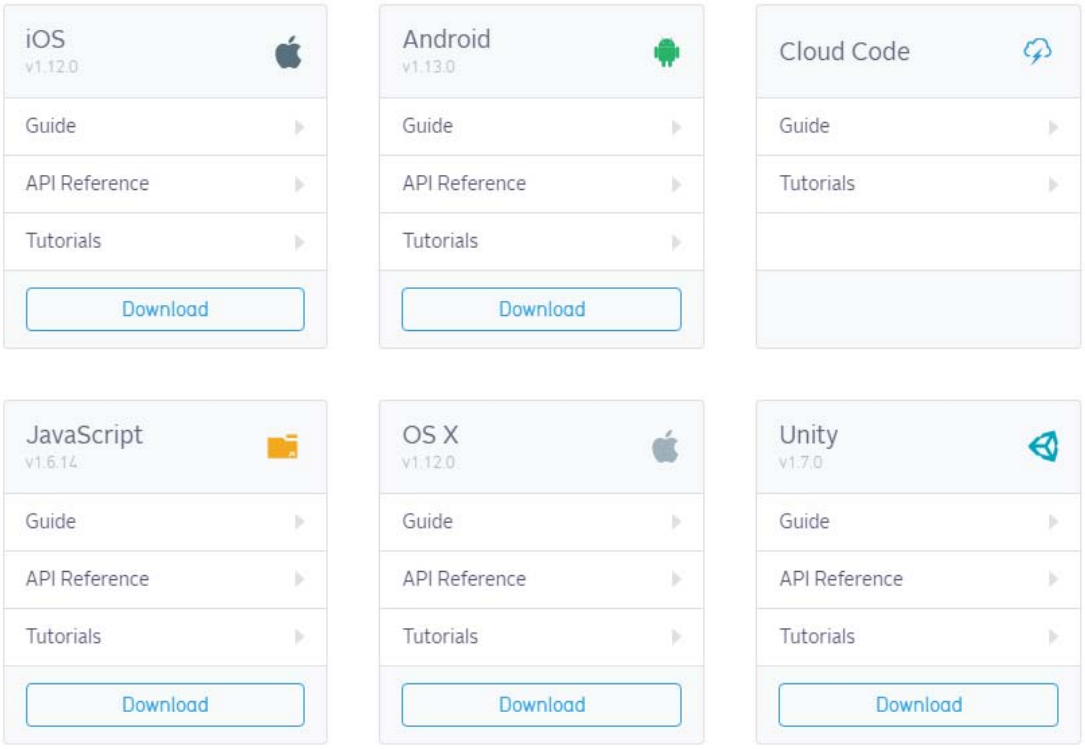


Figura I.8: Representación de algunos de los SDK que ofrece Parse

Parse es una plataforma de pago, pero cuenta con una suscripción gratuita bastante amplia que nos dejará bastante margen de maniobra. Estas son las funcionalidades que incluye esta cuenta gratuita al mes:

- 1000000 peticiones API
- 1000000 notificaciones PUSH
- 1 Gb de espacio en disco
- Un máximo de 30 peticiones por segundo

Como un "plus" podríamos decir que ofrece una gran cantidad de tutoriales y ejemplos de uso para cada una de las plataformas soportadas, Figura I.9 y Figura I.10, lo

que facilita aún más su uso para el desarrollo de aplicaciones que necesiten servicios como los que ofrece Parse.

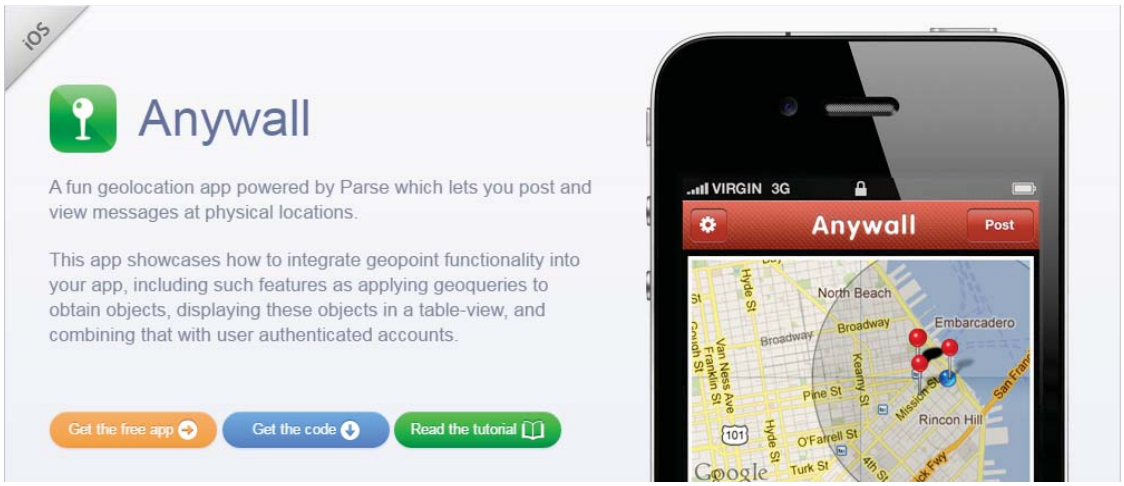


Figura I.9: Tutorial de iOS en Parse

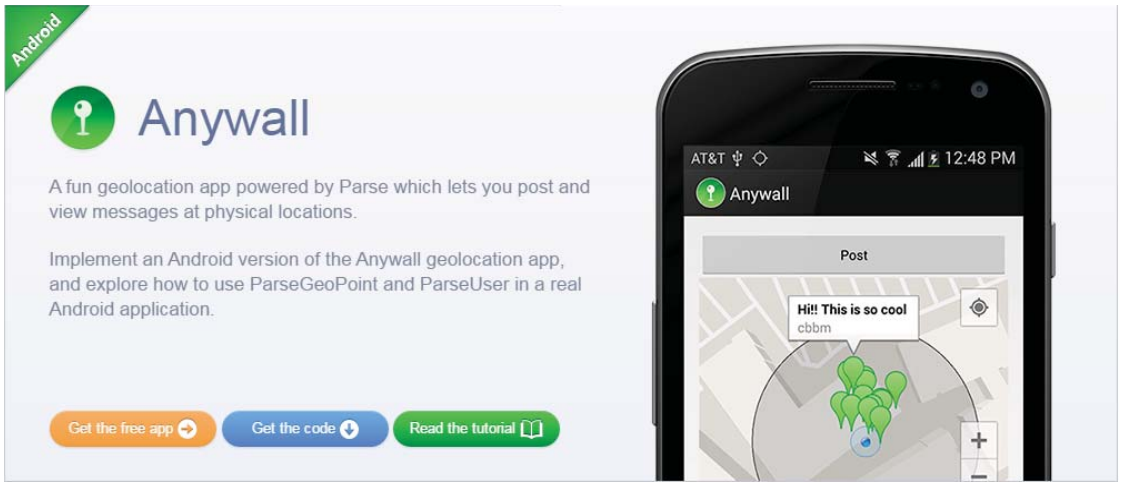


Figura I.10: Tutorial de Android en Parse

La manera en la cual la futura aplicación accede a los contenidos se detalla seguidamente. Los datos o el **Modelo** estan en Parse, organizados en un formato de base de datos noSQL llamado MongoDB, Figura I.11. De esta manera Parse y más

concretamente este tipo de base de datos alberga donde se encuentran los mapas, los POI, etc. Como Parse permite tener múltiples aplicaciones bajo una misma cuenta de usuario tenemos que especificar unos datos cada vez que necesitemos algo de dicha herramienta. Estos datos son el *Application ID* y el *REST API key*.

Figura I.11: Diagrama del acceso a Parse de manera online

Al principio del presente año Parse informó que dejaría de prestar sus servicios en Enero del año 2017. Para sortear este gran inconveniente, la estructura antes mostrada se ha tenido que modificar para adaptarse a un mundo sin la herramienta de la manera en la que se venía trabajando con ella. Lo primero que se llevó a cabo es alojar la base de datos MongoDB en los servidores de la universidad, Figura I.12.

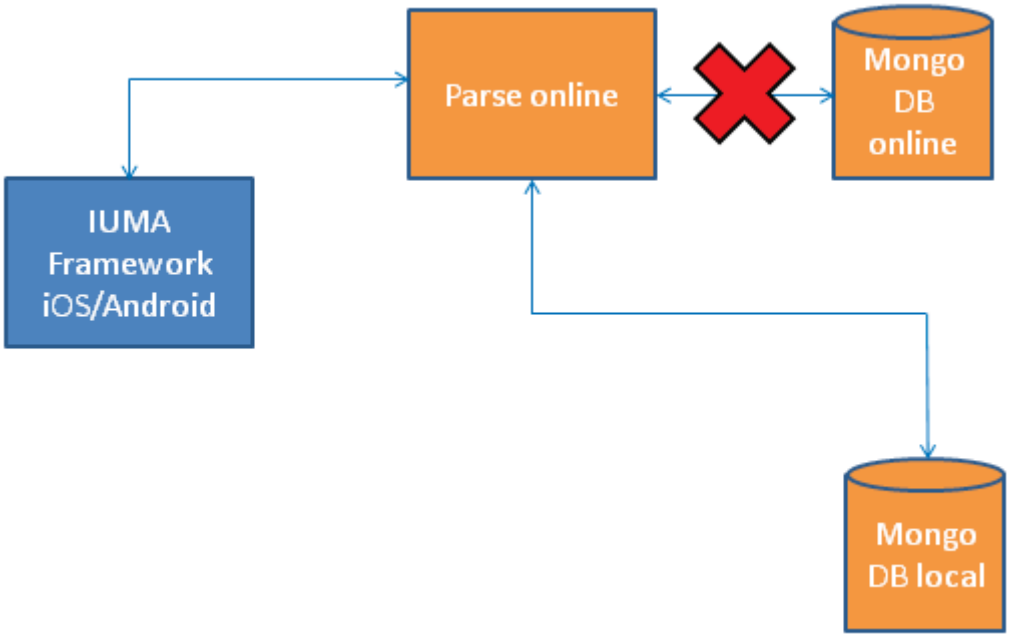


Figura I.12: Diagrama en el que se muestra la primera modificación llevada a cabo para sortear el cierre de Parse.

Posteriormente se llevó a cabo una segunda modificación para salvar de manera satisfactoria el hecho comentado anteriormente. Así, lo que se hizo fue montar de manera local en los laboratorios de la universidad la herramienta Parse, Figura I.13.

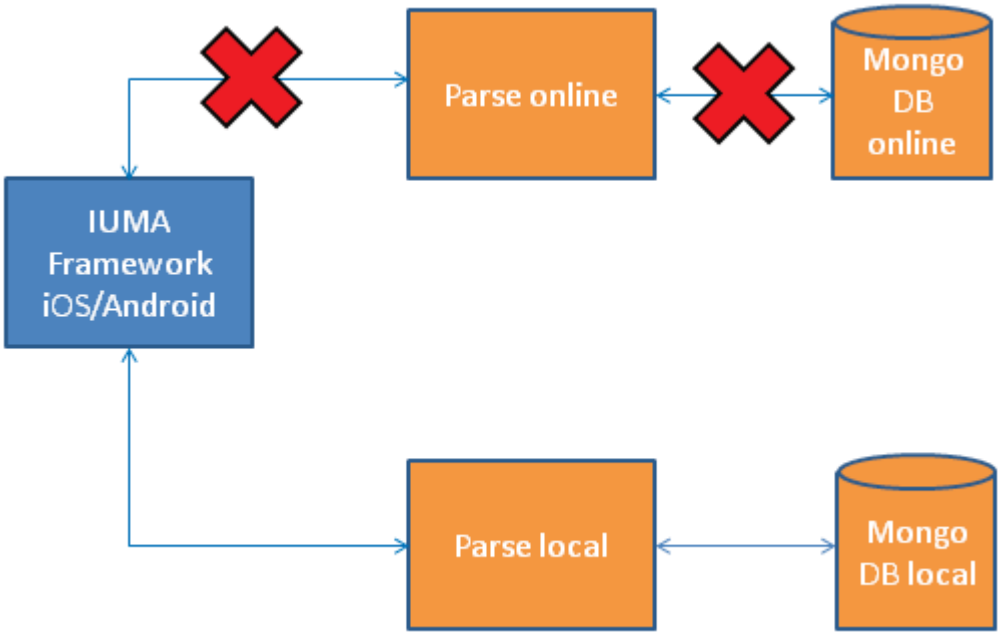
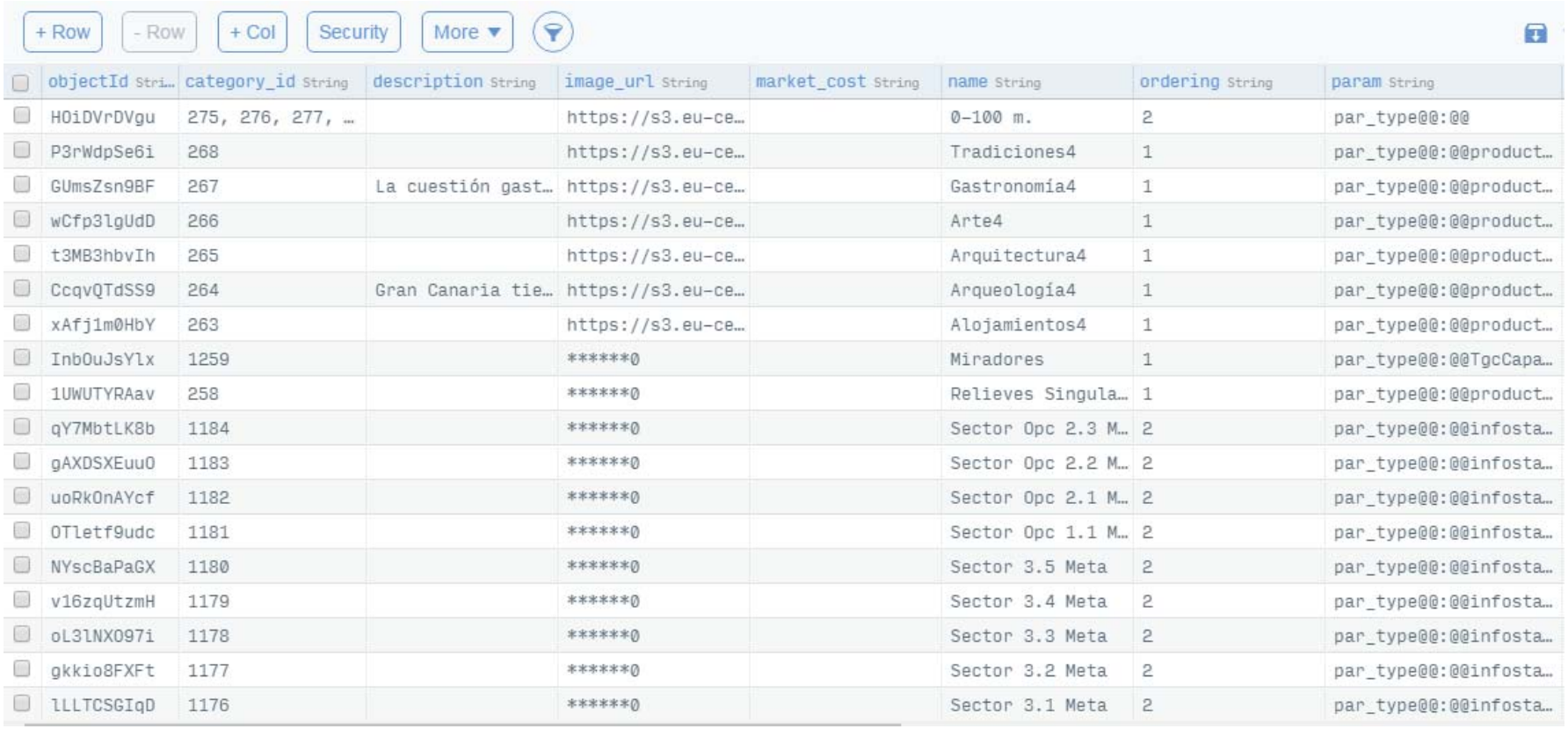


Figura I.13: Diagrama en el que se muestra la primera modificación llevada a cabo para sortear el cierre de Parse.

Con los cambios realizados por parte de los creadores del Framework del IUMA podremos seguir usando la herramienta Parse pero no de manera online. Todos los datos que se alojen se tendran que almacenar en servidores propios.

Uno de los inconvenientes que nos puede afectar en mayor grado acerca del uso de Parse en nuestro Proyecto Fin de Carrera es el de poder visualizar con cierta claridad toda la información que vayamos añadiendo relativa al **Modelo**.



+ Row | - Row | + Col | Security | More

objectId String	category_id String	description String	image_url String	market_cost String	name String	ordering String	param String
HOiDVrDVgu	275, 276, 277, …		https://s3.eu-ce…		0-100 m.	2	par_type@@:@@
P3rWdpSe6i	268		https://s3.eu-ce…		Tradiciones4	1	par_type@@:@@product…
GUmsZsn9BF	267	La cuestión gast…	https://s3.eu-ce…		Gastronomía4	1	par_type@@:@@product…
wCfp3lgUdD	266		https://s3.eu-ce…		Arte4	1	par_type@@:@@product…
t3MB3hbvIh	265		https://s3.eu-ce…		Arquitectura4	1	par_type@@:@@product…
CcqvQTdSS9	264	Gran Canaria tie…	https://s3.eu-ce…		Arqueología4	1	par_type@@:@@product…
xAfj1m0HbY	263		https://s3.eu-ce…		Alojamientos4	1	par_type@@:@@product…
InbOuJsYlx	1259		******0		Miradores	1	par_type@@:@@TgcCapa…
1UWUTYRAav	258		******0		Relieves Singula…	1	par_type@@:@@product…
qY7MbtLK8b	1184		******0		Sector Opc 2.3 M…	2	par_type@@:@@infosta…
gAXDSXEuuO	1183		******0		Sector Opc 2.2 M…	2	par_type@@:@@infosta…
uoRkOnAYcf	1182		******0		Sector Opc 2.1 M…	2	par_type@@:@@infosta…
OTletf9udc	1181		******0		Sector Opc 1.1 M…	2	par_type@@:@@infosta…
NYscBaPaGX	1180		******0		Sector 3.5 Meta	2	par_type@@:@@infosta…
v16zqUtzmH	1179		******0		Sector 3.4 Meta	2	par_type@@:@@infosta…
oL3lNXO97i	1178		******0		Sector 3.3 Meta	2	par_type@@:@@infosta…
gkkio8FXFt	1177		******0		Sector 3.2 Meta	2	par_type@@:@@infosta…
lLLTCSGIqD	1176		******0		Sector 3.1 Meta	2	par_type@@:@@infosta…

Figura I.14: Ejemplo del modo de almacenamiento de datos en Parse

Como podemos ver en la Figura I.14, lo que tenemos en Parse es una gran cantidad de información almacenada en forma de tabla. Esta manera de ver el **Modelo** es demasiado engorrosa ya que no nos proporciona una visión global acerca del mismo además de que se hace muy complicado y costoso en términos temporales el poder encontrar cualquier error que podamos haber cometido al añadir información.

Teniendo este problema en nuestro Proyecto, decidimos decantarnos por una herramienta que nos facilita la tarea de visualizar de una forma global todos los datos referentes a nuestro **Modelo** para así tener una mejor perspectiva de lo que vamos añadiendo conforme avanza el Proyecto Fin de Carrera.

La herramienta en cuestión no es otra que XMind[53]. Es una herramienta *open source* que permite crear una gran variedad de mapas tales como mapas mentales o conceptuales, entre otros, Figura I.15.

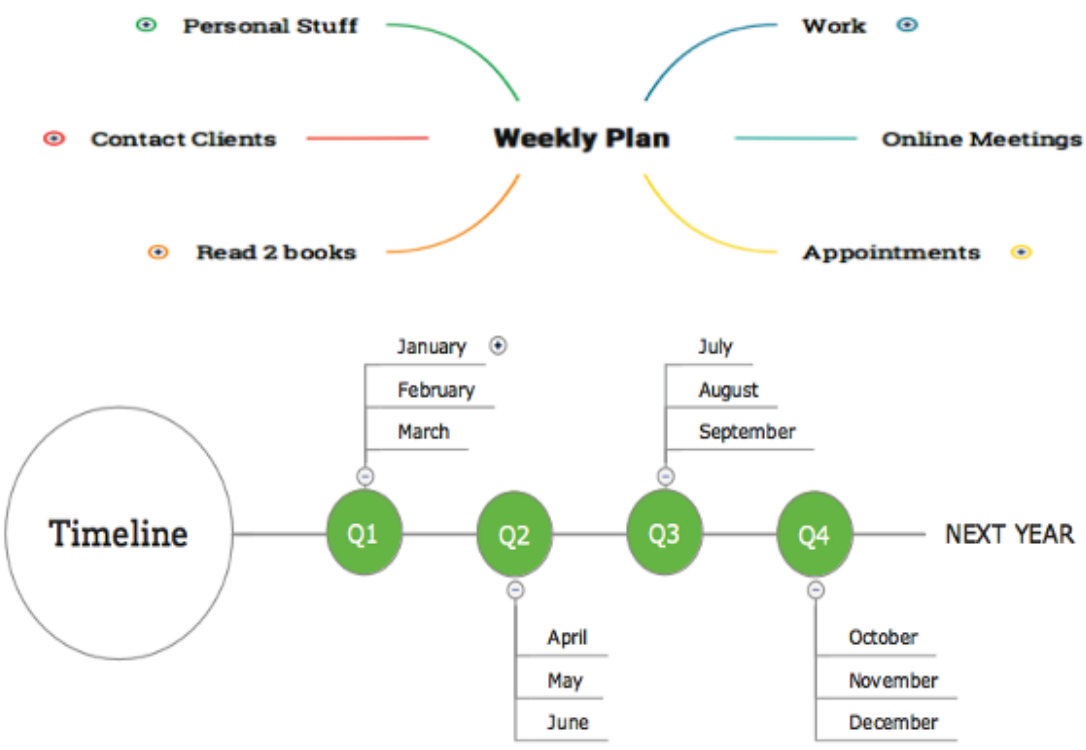


Figura I.15: Ejemplos de mapas mentales que se pueden realizar con Xmind

Además, nos permite exportar los mapas creados a diversos formatos como por ejemplo: *JPEG*, *PNG*, *Word*, *PowerPoint*, *Excel*, *PDF*, *CSV*, *HTML* entre muchos otros más. En el lado opuesto, podemos importar a XMind los siguientes tipos de archivos: FreeMind, Mindjet MindManager, Word y XMind 2008 Workbook.

No obstante a esto último que se ha comentado, el proceso que se realiza para llevar la información almacenada en Parse para poder visualizarla en XMind es un poco más complejo debido a que no existe un nexo de unión entre el formato que podemos exportar con Parse y el que podemos importar con Xmind.

Cabe reseñar que dicho proceso ha sido creado por el tutor de este PFC, José María Quinteiro González.

Para pasar de Parse a XMind lo que se hace es, mediante un software creado por el tutor, generar un fichero con extensión *JSON*[54] para luego con un conversor crear un segundo archivo con extensión *XML* en formato FreeMind. Este segundo fichero sí será válido para poder usarlo en XMind. El software, ahondando un poco más, se encarga de que por medio de un **Visitador** ir recorriendo todas las categorías y productos pertenecientes al **Modelo** almacenado en Parse y en cada paso, generar la estructura del formato FreeMind con extensión JSON.

3.2 . Vista teórico

En cuanto a todo lo referente a la **Vista**, comprende todo el apartado visual del Proyecto Fin de Carrera. Ejemplo de ello puede ser el ver un determinado mapa, acceder a información detallada acerca de un punto de interés o ver una foto, entre otros.

3.3 . Presentador teórico

Acerca del **Presentador**, utilizamos varios presentadores, creados tanto por el personal involucrado en este framework así como por distintos proyectandos. Nos ayudan a mostrar un mapa, buscar información acerca de los sitios de interés que estén alrededor de un determinado punto de nuestra elección o simplemente mostrar imágenes.

3.4 . Modelo real. Contenido de la aplicación

Una vez comentado, de una manera clara, cómo esta formado nuestro modelo MVP; pasamos a entrar en detalle acerca del contenido que tendrá nuestra aplicación móvil multi-plataforma.

Como se ha venido contando a lo largo de este PFC, además de explicar en que se basa el Framework del IUMA en el campo de las aplicaciones móviles, se quiere crear un ejemplo en el cual se vean cómo funcionan todas las partes de dicho entorno de trabajo. Para ello se ha creado una aplicación móvil, apta tanto para iOS como para Android, cuyo objetivo principal ha sido el de describir la historia geológica de la isla de Gran Canaria a través de mapas creados, por un lado por el Departamento de Geomática y por otro lado creados por el proyectando. Además del apartado geológico de la aplicación, incluiremos otras funcionalidades de otros ámbitos como mapas en tres dimensiones o la posibilidad de guardar un recorrido que se haga.

La parte de la aplicación a la que este Proyecto Fin de Carrera hace mención es la que engloba todo el apartado de tratamiento de mapas, dejando a un lado todo lo relacionado con la sección geológica de la aplicación, salvo los mapas de esta parte, y otras posibles secciones que se puedan añadir.

3.4.1 . Rutas

Para la descripción de la historia geológica de la isla de Gran Canaria, nos hemos basado en una ruta que abarca la mayoría de las formaciones geológicas más importantes de dicha isla. Esta ruta parte desde el Faro de Maspalomas, ubicado en el sureño municipio de San Bartolomé de Tirajana, hasta el Puerto de La Nieves, perteneciente al municipio de Agaete. A su paso recorreremos sitios tan emblemáticos de la isla como el Roque Nublo o el Pico de las Nieves, las Necrópolis de Arteara y de Maipez de Arriba o el poblado de Cruz de Tejeda, Figura I.16.

Figura I.16: Georruta completa

Cabe mencionar que la aplicación, a la que hace referencia este PFC, viene de trasladar a un dispositivo móvil otro PFC creado por Bárbara del Castillo-Olivares y Suárez titulado Georruta Transgrancanaria. La aplicación resultante no es más que el resultado de llevar toda la información geológica plasmada en dicho PFC a un smartphone, además de añadir otras funcionalidades y mapas.

Esta ruta geológica se haya dividida en tres etapas que atraviesan de sur a norte la isla de Gran Canaria, mostrándonos a su paso volcanes, yacimientos arqueológicos, asentamientos urbanos, entre otros puntos de interés.

La *Etapa 1,* Figura I.17, comienza en el Faro de Maspalomas. Desde este enclave turístico iremos a través del canal que desemboca en la Charca de Maspalomas hasta la planta depuradora de agua Elmasa. Desde este punto seguiremos un poco más,

bajando por el barranco hasta llegar a la conjunción de los barrancos de los Vicentes y de Fataga. Nuestro camino seguirá por el barranco de Fataga hasta encontrarnos con el pago de La Gitagana. Seguidamente tomaremos el camino que conduce hacia Ayagaures para luego tomar una desviación a la derecha que nos llevará hasta la Necrópolis de Arteara. El siguiente tramo es uno de los más difíciles de toda la Georruta ya que deberemos de superar un gran desnivel para llegar hasta la Degollada de Garito. Una vez llegados a este punto deberemos continuar con nuestra ascensión, pero ya en menor medida, hasta alcanzar la Degollada de la Manzanilla. El siguiente paso será llegar hasta el poblado de San Bartolomé de Tirajana, el que se hará mucho más fácil debido a que este trayecto se hace siempre en descenso. La etapa finaliza en la Degollada de Cruz Grande a la cual se llega a luego de ir ascendiendo durante casi 5 kilómetros por senderos plagados de pinares, por este tramo pasaremos por El Roquillo, privilegiada atalaya desde la que se contempla una bonita panorámica del recorrido. Sobra decir que tendremos que ir siempre extremando las precauciones debido a lo escarpado del trayecto en algunas partes además y la probabilidad de resbalar en algunos tramos.

Figura I.17: Etapa 1 de la Georruta

La *Etapa 2,* Figura I.18, comienza donde termina la anterior etapa, en la Degollada de Cruz Grande. Desde aquí continuaremos hasta el poblado de Ayacata pasando por el denominado Camino de La Plata, en él podremos ver la Caldera de Tirajana o la Ventana del Nublo. Desde Ayacata seguiremos nuestro camino hasta el Roque Nublo, símbolo emblemático de la isla de Gran Canaria, por este camino se podrá observar algunos de los accidentes geológicos más afamados de Gran Canaria como lo son el Fraile y la Rana. Cabe mencionar que al dirigirnos al Roque Nublo estaremos entrando en el Parque Rural del Nublo, estando dentro de dicho parque el grueso de la etapa 2. Además, si el tiempo nos lo permite, estaremos en condiciones de visualizar una parte del Teide, punto más alto de todo el territorio nacional. Desde el Roque Nublo nos dirigiremos hacia una de sus faldas hasta llegar al Risco de la Foguera, punto de interés geológico importante debido a los desprendimientos acaecidos hacia el fondo de la Caldera de Tejeda. Decir que desde dicho risco tendremos una fabulosa panorámica del Roque Bentayga, que se encuentra dentro de la Caldera de Tejeda, lugar sagrado para los habitantes originarios de esta isla como así lo atestiguan los yacimientos arqueológicos que guarda. La siguiente para en nuestra travesía se encuentra en El Garañon, campamento ubicado en los LLanos de la Pez en el que se organizan diversas actividades como senderismo o descenso de barrancos. A continuación tomaremos dirección rumbo al Pozo de las Nieves, luego de recorrer casi todo el trayecto entre pinos canarios nos encontraremos en la zona más alta de la isla de Gran Canaria (a unos 1949 metros de altitud). Luego daremos vuelta e iremos deshaciendo el camino recorrido hasta llegar al Aula de la Naturaleza, ubicada cerca muy de El Garañon. El siguiente destino en nuestra ruta será el Parador Cruz de Tejeda, su entrada esta presidida por una gran cruz de piedra "La Cruz de Tejeda" cuya funcionalidad era la de servir de orientación a los vecinos del siglo XVII. Este enclave es el epicentro geográfico de la isla y el punto de encuentro de la mayoría de los antiguos caminos reales. Desde aquí enfilaremos rumbo a la Degollada de las Palomas. Por esta degollada discurre el límite entre los municipios de Tejeda y Valleseco, además de que dicho límite coincide con el camino real que va desde la Cruz de Tejeda a Artenara en el que encontraremos importantes testimonios arqueológicos de los antiguos

canarios como las Cuevas del Caballero o la Cueva de los Candiles. Esta etapa la terminaremos siguiendo por el camino real hasta llegar a Artenara.

Figura I.18: Etapa 2 de la Georruta

El punto de partida de la *Etapa 3,* Figura I.19, es el poblado de Artenara. Desde aquí nos dirigiremos hacia un sitio llamado Portada de Tirma, adentrándonos a su vez en el Parque Natural de Tamabada. Seguidamente iremos en dirección la Presa de los Pérez, durante el descenso a través de un sendero bien cuidado por el bosque de pinos de Tamadaba tendremos vistas del Roque Nublo parte de a la misma Presa de los Pérez y la Presa de los Lugarejos. A continuación la *Etapa 3* nos llevará hasta Los Berrazales, en el camino podremos observar el barranco de Agaete hacia un lado y hacia el otro el barrio de El Hornillo con sus casas cuevas en la pared que vale la pena de ver. Desde Los Berrazales iremos a la Montaña de Berbique, entrando en una zona de gran interés geológico. Luego, y para terminar, tomaremos rumbo al Puerto de Las Nieves.

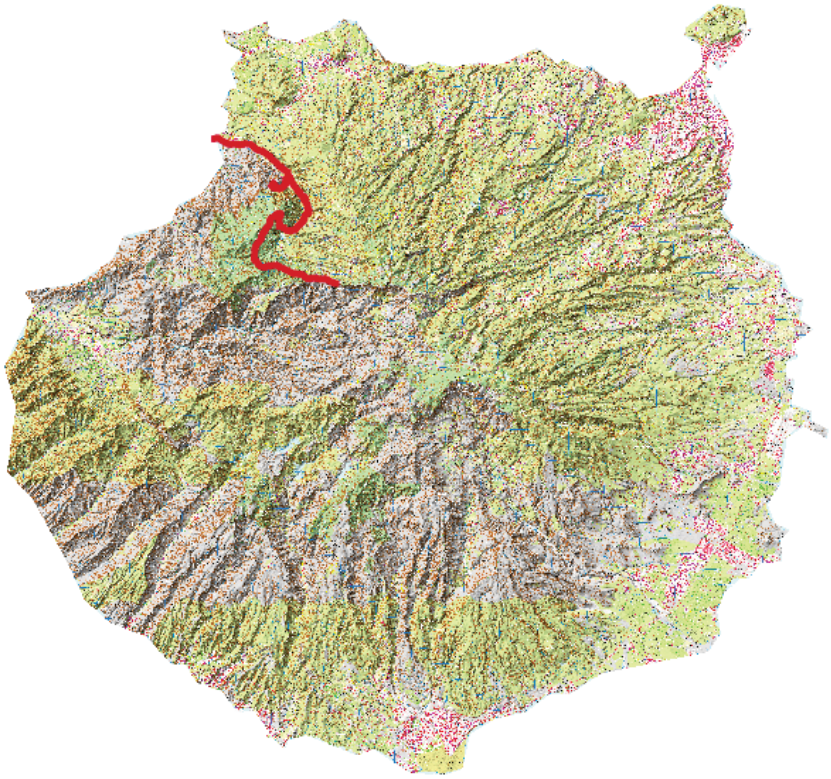


Figura I.19: Etapa 3 de la Georruta

Con respecto a los mapas, algunos de ellos ya se han podido apreciar, en un principio se tenía la idea de tener mapas para cada una de las etapas de las que consta la ruta. De esta manera se tendrían un total de cuatro mapas para cada etapa (ortofotos, líneas isométricas, tintas hipsométricas y geológicos) además de otros cuatro mapas en los cuales se mostraría toda la isla de Gran Canaria para cada uno de los tipos de mapas que ofrecemos en la aplicación. Otra opción que se ha venido estudiando es la de solo mostrar los cuatro mapas de la isla entera, cada uno mostrando los mapas disponibles, y como añadido incluir un tipo de mapa llamado LIDAR[55] que ofrece el GRAFCAN. El LIDAR (Light Detection and Ranging) es un sensor láser aerotransportado que permite obtener una nube de puntos georreferenciada del territorio, Figura I.20. El sensor mide el tiempo que tarda el láser en ir desde el punto de partida hasta el suelo y su vuelta. Conociendo con exactitud la posición y orientación del avión se pueden calcular las coordenadas de cada uno de los puntos.

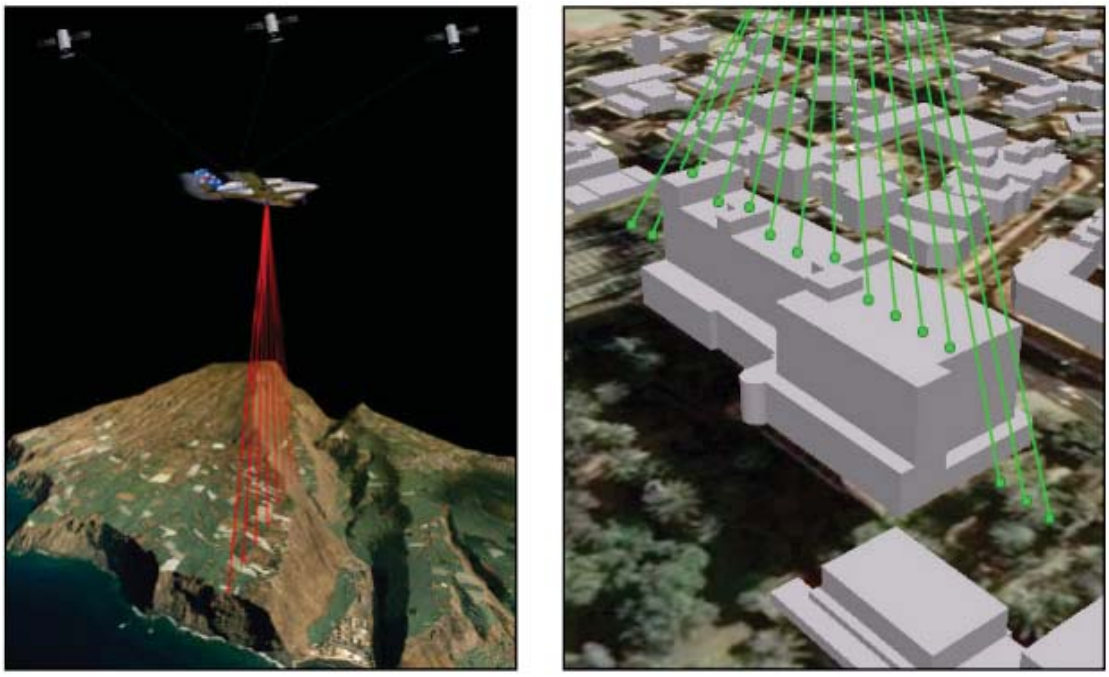


Figura I.20: Tecnología LIDAR[55]

Respecto a los datos y archivos necesarios para la creación de la Georruta, además de diversa información relacionada con puntos de interés geológico o turístico entre otros, han sido dados por parte del departamento de Geomática de la ULPGC y en especial por Bárbara del Castillo-Olivares y Suárez. El formato en el que estuvo toda esta información en un primer momento fue Shapefile.

El Shapefile es un formato de representación vectorial desarrollado por ESRI[56] (Enviromental Systems Research Institute), compañía líder en la creación y comercialización de software para sistemas de información geográfica (SIG o GIS en inglés). La implantación de este formato en toda la gama de productos de la compañía (ArcView, ArcInfo, ArcGis) lo ha vuelto bastante popular hasta tal punto de convertir el Shapefile en el formato más extendido dentro de los SIG vectoriales. Aparte de esto se ha convertido en un estándar a la hora de representar información geográfica de cualquier índole debido que es un formato abierto, permitiendo que cada vez más sean las compañías que desarrollen software compatible con dicho formato.

El Shapefile nos permite representar entidades mediante la utilización de formas geométrica como puntos, líneas o polígonos. Al poder representar datos espaciales, estas entidades están vinculadas a un *datum* y un sistema de coordenadas de referencia que se especifica en el archivo Project (*.prj). También suele aparecer con frecuencia información temática para representar al igual que lo anterior. Esta información se expresa en forma de atributos asociados a los elementos que componen el formato y se recogen en un tabla de datos de tipo dBase (*.dbf).

El Shapefile consta de varios archivos, aunque solo tres de ellos son indispensables para su correcto funcionamiento, los cuales son:

- Shape(*.shp): Es el archivo principal, almacena las características geométricas de los elementos existentes en la capa. Como estamos hablando de un formato vectorial, la información se guarda mediante puntos, polilíneas(sucesiones de puntos unidos) o polígonos(polilíneas cerradas), Figura I.21. Una de las desventajas de este formato es que no guarda información topológica. La contraparte de ello es que permite una edición más sencilla, menores requisitos de almacenamiento y una velocidad de acceso mayor. Otro inconveniente es que en función de la forma de los elementos y su complejidad, se requieren más o menos puntos para representar una entidad, esto supone un mayor volumen de información para representar determinadas tipos de curvas complejas.

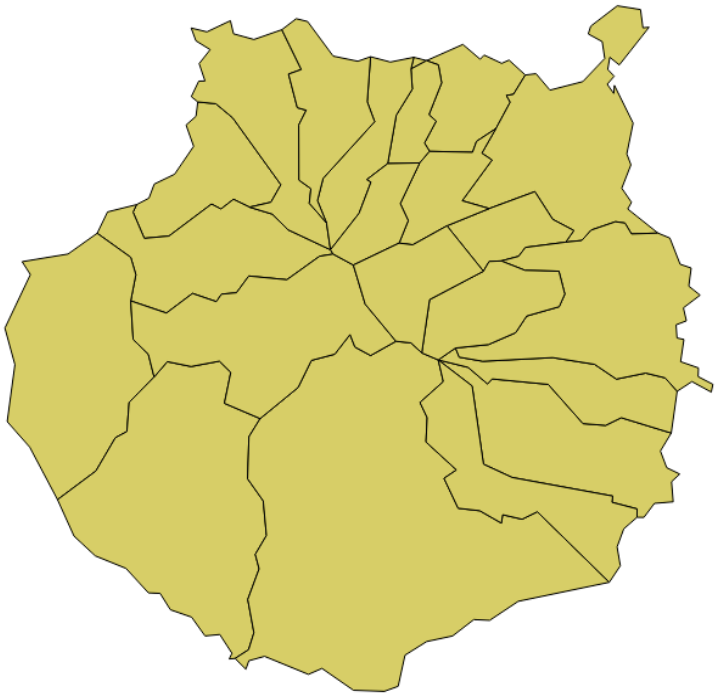


Figura I.21: Representación del archivo Shape en QGIS

- Shape Index (*.shx): Es un índice de las entidades geométricas que posibilita afinar búsquedas dentro del archivo Shape(*.shp), Figura I.22.

ID	Nombre	Tipo	Nombre de tipo	Longitud
0	OBJECTID	int	Integer	9
1	EJERCICIO	QString	String	4
2	MUNICIPIO	QString	String	5
3	LINEA	int	Integer	9
4	CODIGO_INE	QString	String	4
5	NOMBRE	QString	String	60
6	TIPO	QString	String	3
7	Tipo_Nombr	QString	String	50

Figura I.22: Representación del archivo Shape Index en QGIS

- dBase (*.dbf): Se trata de una tabla de datos en la que registran los atributos de cada elemento, Figura I.23. Estos atributos pueden ser tanto numéricos, de texto o de fecha a los registros contenidos en el archivo principal.

	pig_cod	nombre	etapa_cod	m_cod	x	y
0	1	Dunas de Maspalomas	1	1	442747.0...	3068879....
1	2	Deslizamiento de La Gitagana	1	2	442223....	3078595....
2	3	Avalancha Rocosa de Arteara	1	3	443818....	3080577....
3	4	Mesa de Trujillo	1	5	440806....	3083035....
4	5	Deslizamiento de Ayagaures	1	5	439413....	3081313....
5	6	Facies deslizada de la Brecha Roque Nublo	1	4	440154....	3084876....
6	7	Llano de Los Caserones	1	6	444884....	3082440....
7	8	Canchales Camino Real de Tirajana	1	6	442896....	3086624....
8	9	Puntas de Manzanilla	1	6	442926....	3085866....
9	10	Flujo detrítico de La Manzanilla	1	7	443551....	3086558....
10	11	Risco Blanco	1	7	444511....	3091251....
11	12	Deslizamiento de Rosiana	1	8	446414....	3089150....

Figura I.23: Representación del archivo dBase en QGIS

Todos los archivos que componen el formato Shapefile deben tener el mismo nombre, tan solo variará la extensión del archivo.

Aparte de estos archivos, es común que cada Shapefile incluya algunos de los siguientes archivos:

- Spatial Index (*.sbn y *.sbx): Se trata de un formato exclusivo de ESRI que almacena un índice espacial de los elementos. No son estrictamente necesarios ya que el archivo *.shp contiene esta misma información.

- Metadatos (*.xml): Los metadatos guardan información sobre el contenido del archivo y su formato.

- Projection (*-prj): Este archivo es primordial para georreferenciar los datos geométricos que poseemos en el Shape, Figura I.24. Con el archivo *Shape* (*.shp) se definen geométricamente una serie de elementos en un espacio

bidimensional; también se pueden vincular valores de altura. Pero si queremos situar dicho elemento sobre el terreno necesitamos referir los datos a un sistema de coordenadas. Aquí es donde entra en juego este archivo.

```
PROJCS["WGS_1984_UTM_Zone_28N",
        GEOGCS["GCS_WGS_1984",
            DATUM["D_WGS_1984",
                SPHEROID["WGS_1984",6378137.0,298.257223563]],
            PRIMEM["Greenwich",0.0],UNIT["Degree",0.0174532925199433]],
            PROJECTION["Transverse_Mercator_Complex"],
            PARAMETER["False_Easting",500000.0],
            PARAMETER["False_Northing",0.0],
            PARAMETER["Central_Meridian",-15.0],
            PARAMETER["Scale_Factor",0.9996],
            PARAMETER["Latitude_Of_Origin",0.0],
            UNIT["Meter",1.0]]
```

Figura I.24: Representación del archivo Projection

Debido a que usamos la librería Glob3Mobile, de la cual se habla en los ANEXOS, se harán algunos cambios ya que dicho framework solo soporta el formato para datos vectoriales llamado *GeoJSON*[57]. Como se ha venido haciendo durante todo el PFC para el tratamiento de datos geográficos utilizamos la herramienta GDAL, del cual tambien se hace un análisis en profundidad en los ANEXOS. Más específicamente ejecutamos el siguiente comando desde la consola desde la consola de Windows (además debemos situarnos en la carpeta que contiene el archivo con el formato *Shapefile*):

ogr2ogr ***-f*** *GeoJSON* ***-t_srs crs:84*** *[nombre].geojson [nombre].shp* (5.1)

Otra opción para llevar a cabo el cambio de formato es hacerlo mediante la herramienta QGIS, Figura I.25. Para ello tenemos que cargar el archivo que deseemos convertir de formato. Seguidamente hacemos clic derecho sobre el archivo y seleccionamos la opción *Guardar como*. Finalmente, en la ventana que se nos abre,

debemos elegir el formato *GeoJSON* de los disponibles y el sistema de coordenadas *WGS84*.

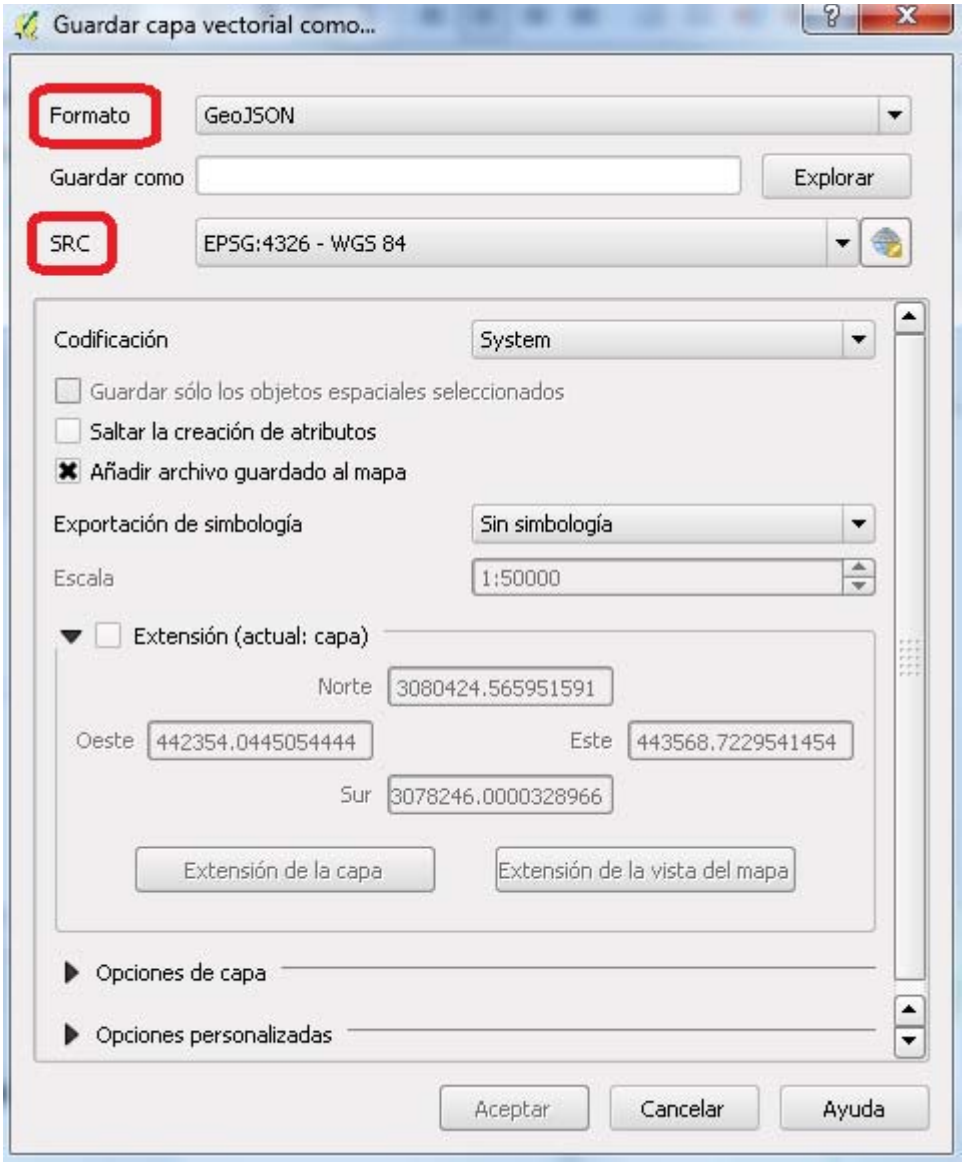


Figura I.25: Cambio de formato de GeoJSON a Shapefile en QGIS

GeoJSON también es un formato que permite visualizar una gran variedad de datos geográficos. La información es representada en base a la estructura de datos JSON, Figura I.26. Soporta los siguientes tipos de geometrías:

- Point, MultiPoint.

- LineString, MultiLineString.
- Polygon, MultiPolygon.
- GeometryCollection. Puede representar una mezcla de los anteriores.

En cuanto a los sistemas de coordenadas de referencia(o CRS por sus siglas en inglés), son asignados por el objeto de *GeoJSON* llamado *CRS*. En caso de que no se asigne ningún sistemas de coordenadas se toma por defecto el *WGS84* o *EPSG:4326*. Además, el *CRS* puede ser adjuntado por su nombre cambiando el tipo del objeto para añadirlo. También se puede añadir un enlace para especificar el *CRS* siguiendo el proceso anterior.

```
{
"type": "FeatureCollection",
    "crs": { "type": "name",
        "properties": { "name": "urn:ogc:def:crs:OGC:1.3:CRS84" } },
    "features": [
            { "type": "Feature",
            "properties": { "tramo_cod": 1,
            "nombre": "Faro de Maspalomas - La Gitagana" },
            "geometry": { "type": "LineString",
            "coordinates": [ [ -15.59, 27.73], [ -15.58, 27.77] ] }
            ]
}
```

Figura I.26: Estructura del formato GeoJSON

Aparte de las rutas para cada una de las etapas de la Georruta, también añadiremos a nuestro **Modelo** más información en el formato *GeoJSON*. Como viene siendo habitual, toda esa información ha sido entregada por parte del departamento de Geomática (Bárbara del Castillo-Olivares y Suárez). Más explícitamente agregamos la siguiente información:

- Relieves singulares
- Miradores

- Alojamientos
- Arqueología
- Arquitectura
- Arte
- Gastronomía
- Tradiciones populares

3.4.2 . Modelos Digitales del Terreno

Por último vamos a hablar acerca de los Modelos Digitales de Terrenos (MDT) que estan incluidos, como no podía ser de otra manera, en nuestro **Modelo**. Estos archivos son los que se encargan del modelado en tres dimensiones de los mapas.

Los MDT constituyen una forma de visualización de la superficie del terreno en un formato ráster[58]. Básicamente son una representación digital de una variable continua sobre una superficie bidimensional por medio de un conjunto de valores referenciados. Los MDT representan, más concretamente, la superficie del suelo desnudo sin ningún objeto. Junto con el Modelo Digital de Superficie (MDS) encargado de la representación de la superficie de la Tierra, incluyendo todos los objetos que incluye, conforman lo que se conoce como Modelo Digital de Elevación(MDE)[59].

Estos modelos se suele construir de diversas maneras tales como a partir de imágenes aéreas (fotogrametría), mediante la digitalización de las curva de nivel, introduciendo directamente las coordenadas de los puntos medidas por un GPS o mediante un sistema láser aerotransportado (LIDAR) entre otros.

Este último modo de obtener los MDT ha sido uno de los métodos que se han utilizado con los modelos que tenemos a nuestra disposición. El otro método que ha sido utilizado es el de los vuelos fotogramétricos.

Para la obtención de los MDT mediante los vuelos fotogramétricos lo que se hace es montar una cámara de gran resolución en una aeronave, orientada hacia el suelo, Figura I.27. Además se dispone de un sistema GPS para la geolocalización y una IMU (Inertial Measurement Unit) para poder corregir posteriormente las imágenes debido a la orientación de la aeronave durante el vuelo. Los datos obtenidos se procesan mediante técnicas de fotogrametría como la autocorrelación automática para obtener el MDT. También se suelen utilizar otros equipos como sensores pancromáticos (trabajan en el espectro visible extendido) o multiespectrales (trabajan en diferentes bandas del espectro electromagnético al mismo tiempo).

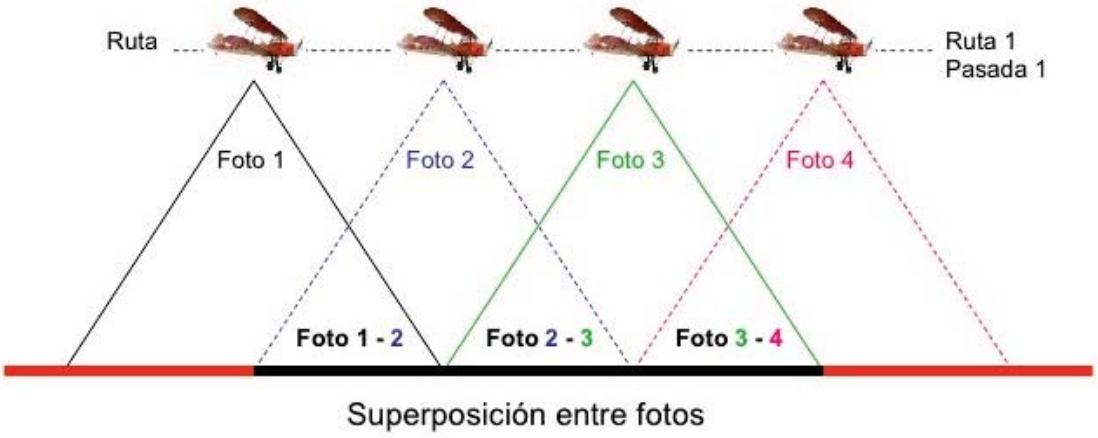


Figura I.27: Vuelos fotogramétricos[60]

El otro método que se ha utilizado para la obtención de los MDT de que disponemos es la técnica LIDAR, la cual ya ha sido comentada en páginas anteriores.

Los Modelos Digitales del Terreno que vamos a utilizar para la aplicación han sido obtenidos desde el IGN. Para ello, como se ha hecho tanto con las ortofotos como con los mapas de líneas isométricas, tenemos que encaminarnos hacia su Centro de Descarga[61]. Dentro de su Catálogo de Productos buscamos el que se llama *MDT05/MDT05-LIDAR*.

Luego de hacer clic en la opción de *Descargar*, nos remite a otra ventana (Búsqueda Avanzada) en la cual podemos descargar los archivos que necesitemos. La búsqueda de dichos archivos la podemos hacer de varias maneras, buscando por

división administrativa (comunidad autónoma, provincia o municipio) o introduciendo los valores que concuerden con lo que queremos referidos a la Hoja del MTN50 (Mapa Topográfico Nacional con escala 1:50000).

Por cada cuadrícula que descarguemos tenemos dos archivos:

- Un archivo que contiene el MDT, con extensión "*ASC*".
- Un archivo que contiene los Metadatos del MDT en cuestión, con extensión "*XML*".

La extensión "*ASC*" proviene de ASCII Grid[62][63], formato propiedad de la compañía ESRI. La estructura del fichero es la siguiente:

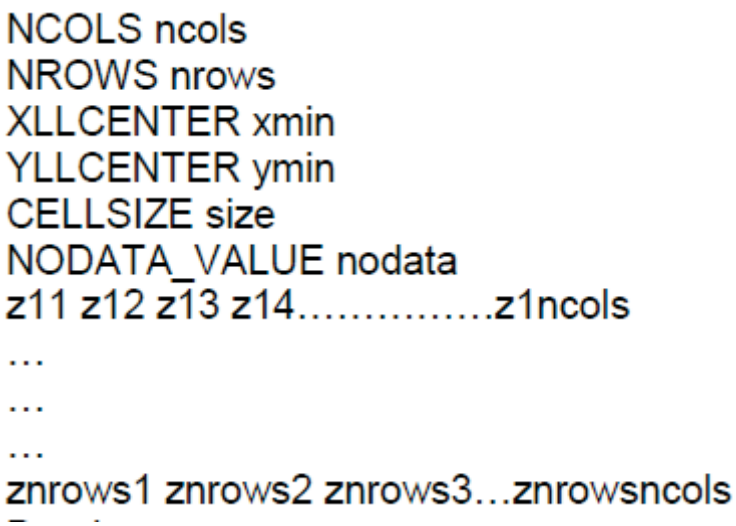


Figura I.28: Formato del archivo ASCII Grid[63]

Dónde:

- NCOLS es el número de nodos por fila.
- NROWS es el número total de filas.
- XLLCENTER y YLLCENTER son las coordenadas suroeste de la malla
- CELLSIZE es el paso de malla.

- NODATA VALUE es el valor asignado a los nodos para los que no se dispone de cota conocida.

Las cotas de la malla están expresadas en metros, con un espacio en blanco entre valor y valor, y describen el terreno de norte a sur y de oeste a este. El sufijo CENTER indica que las coordenadas describen la posición del centro del píxel. Cuando el sufijo es CORNER indica la posición de la esquina inferior izquierda del mismo.

Para obtener los MDT de cada de etapa de la Georruta así como el de la isla de Gran Canaria nos ayudamos de la herramienta GDAL. Estos paso los hacemos casi idénticamente a como se ha venido haciendo con otros tipos de mapas que hemos usado para crear la aplicación.

Utilizamos el comando *gdalwarp* para realizar el correspondiente cambio de sistemas de coordenadas ya que el que descargamos del Centro de Descargas del IGN se encuentra en el sistema geodésico de referencia *REGCAN95* o *EPSG:4083* y el sistema con el que estamos trabajando es el *WGS84* o *EPSG:4326*. El otro comando que vamos a utilizar es *gdal_translate* para crear el MDT tanto de la isla de Gran Canaria como el de las correspondientes etapas de la Georruta.

Un gran inconveniente que surge del uso de estos tipos de ficheros es su tamaño. En función de la resolución y del tamaño del mapa del cual queramos obtener el MDT aumentará el espacio en disco de estos archivos. Por poner algún ejemplo de esta desventaja acerca de los ficheros con extensión "*ASC"*, el MDT creado a partir de la información obtenida del IGN de la isla de Gran Canaria presenta un tamaño de 1,41 GB, Figura I.28.

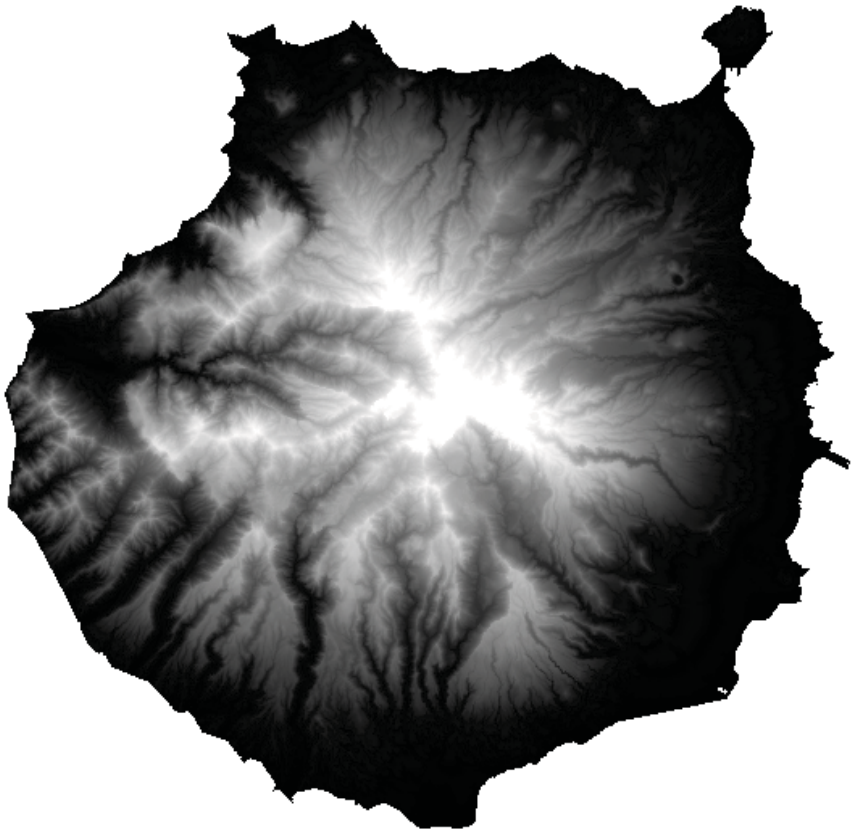


Figura I.29: Representación del MDT en QGIS

Como se puede apreciar por la imagen anterior, las zonas más oscuras representan a áreas de poco relieve, mientras que conforme el color se torna a blanco indica zonas con gran altitud con respecto al nivel del mar.

El Modelo Digital del Terreno, en nuestro Proyecto Fin de Carrera, tiene como finalidad el poder representar en tres dimensiones la isla de Gran Canaria. Para usar estos modelos tenemos que echar mano de la librería Glob3Mobile. De esta herramienta, creada por la ULPGC e IGO Software, se habla más detalladamente en los ANEXOS de este PFC.

Para poder hacer uso de los MDT en G3M tenemos que realizar un cambio de formato debido a que este software solo permite el trabajo con MDT bajo el formato "*BIL"*.

El cambio de formato se llevó a cabo mediante un pequeño software creado en el IUMA. A través de dicho software podemos cambiar drásticamente el espacio en disco de los MDT sin perder ni un ápice su utilidad. El funcionamiento de dicho código comienza con la lectura del archivo de entrada en formato "ASCII Grid". Los datos de entrada se guardan como si fueran de tipo cadena o "string". Seguidamente se utiliza un bucle para ir recorriendo el fichero de entrada e ir cambiando a entero cada dato, añadirlo a los que ya están además de ir comprobando la altura máxima. Finalmente, el archivo resultado de todo el proceso anterior se empaqueta en el formato "*BIL*".

Los ficheros con extensión "*BIL"* (banda intercalada por línea) son archivos binarios utilizados comúnmente para la organización datos de una imagen por imágenes multibanda. No es un formato de imagen pero es un esquema para almacenar los valores de píxel reales de una imagen de un archivo. Almacena la información de píxel banda por banda para cada línea o fila de la imagen.

	1 hasta n columnas	1 hasta n columnas	1 hasta n columnas
Línea 1	Banda 1	Banda 2	Banda 3
Línea 2	Banda 1	Banda 2	Banda 3
…	….	…	…
Línea n	Banda 1	Banda 2	Banda 3

Tabla I.3: Arquitectura de los ficheros BIL[64]

Como se puede apreciar en la tabla anterior, las tres bandas se escriben para la primera línea, luego se hace lo mismo para la segunda línea, y así sucesivamente hasta la última línea.

Para poder ver toda la parte referente al **Modelo** utilizamos la herramienta XMind, descrita con brevedad anteriormente.

3.4.3 . Representación del Modelo

Como se ha descrito, la aportación de este proyecto a la hora de la creación de la aplicación llamada Georruta Transgrancanaria es la de encargarse de todo lo relacionado con la información geográfica. A continuación se visualiza de manera más detallada todo este aporte.

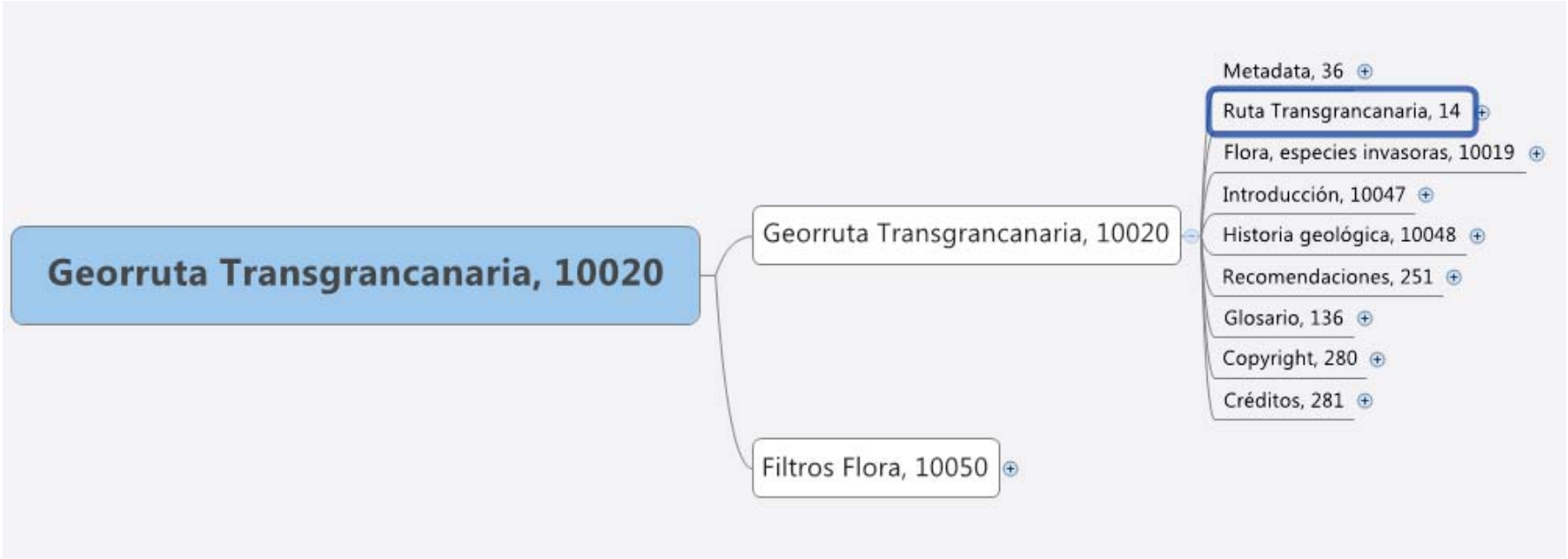


Figura I.30: Vista general de la aplicación desde Xmind

Como se puede apreciar de la imagen, Figura I.30, todo nuestro contenido esta dentro de lo que se llama Ruta Transgrancanaria. Este contenido se haya dividido en cuatro partes, tenemos tres etapas además de otra etapa que no es más que las anteriores etapas unidas, Figura I.31.

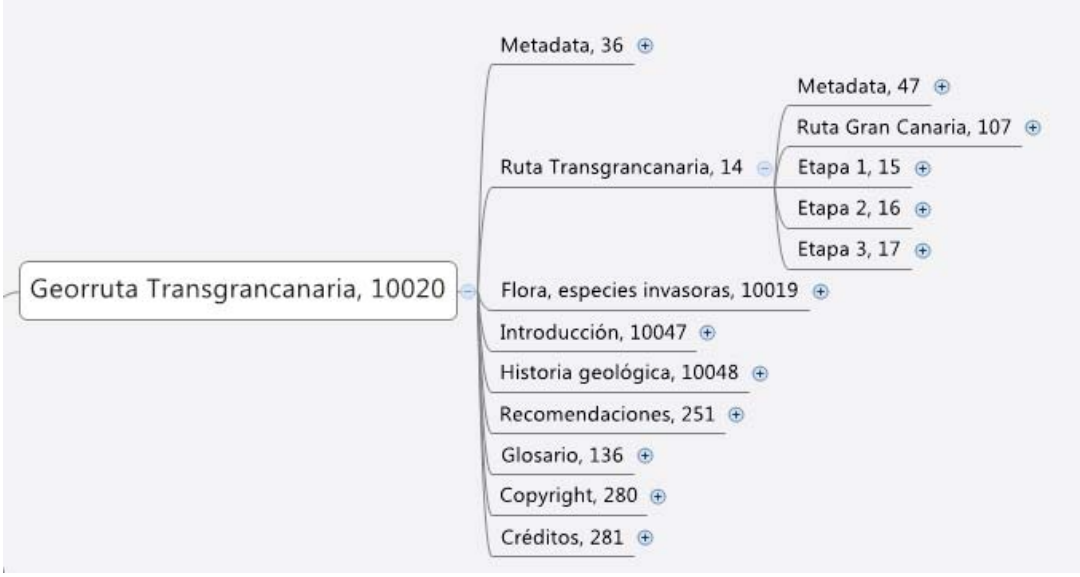


Figura I.31: Vista de las etapas

En este diagrama o mapa, todos los "padres" tienen un "hijo" que ses el encargado de organizar la información del "padre" en cuestión. Este "hijo" llamado *Metadata* irá cambiando en función del "padre".

Como los datos de las cuatro etapas son prácticamente idénticos solo mostramos la información de una de ellas, la de la Ruta Gran Canaria, Figura I.32. Al expandir vemos la información que contiene esta etapa, tenemos los mapas de dicha etapa además de otro tipo de datos que explicaremos a continuación.

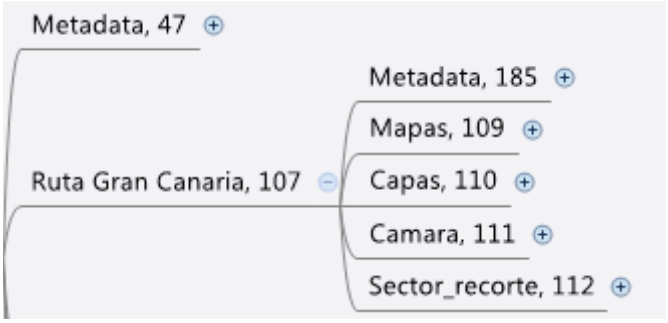


Figura I.32: Información contenida en la etapa Ruta Gran Canaria

El primer tipo de dato llamado *Metadata* contiene información inherente a la etapa. El segundo tipo llamado *Mapas* almacena todo lo referente a los mapas. Como se ha trabajado con cuatro tipos de mapas todos ellos estan incluidos en este apartado, además de esto tenemos valores acerca de su zoom o de como localizarlos mediante los datos que ofrece Mapbox, Figura I.33.

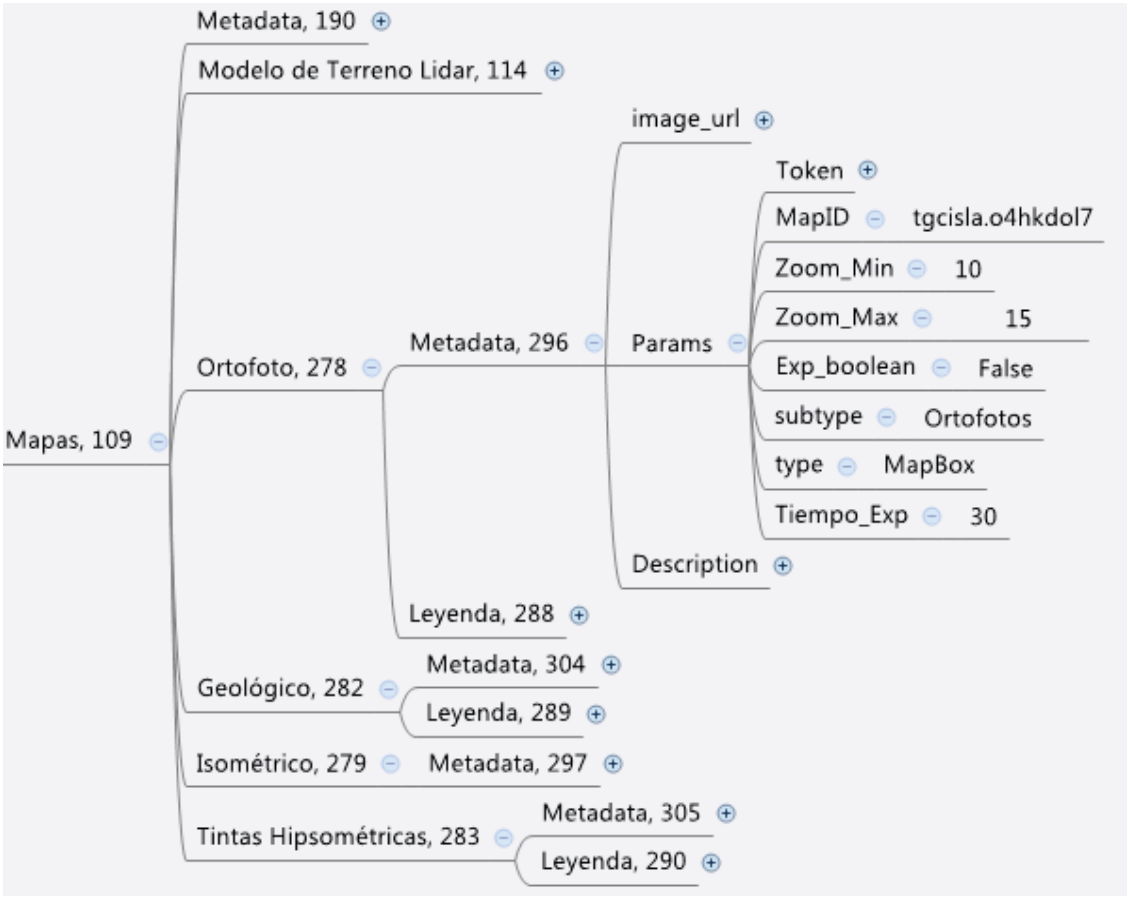


Figura I.33: Contenido de Mapas en Xmind

El apartado *Capas* contiene toda la información referente a las distintas rutas que se han incluido en la aplicación, entre las que se encuentran subetapas geológicas cedidas por la antigua alumna Bárbara del Castillo-Olivares y Suárez. Además contiene información referente a puntos de interés tanto de carácter geográfico como turístico, Figura I.34.

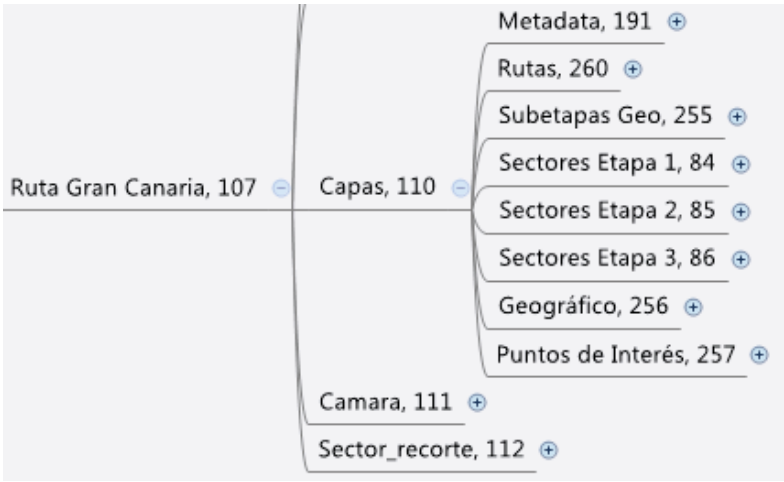


Figura I.34: Contenido de Capas

El contenido de *Cámara* hace referencia a los datos necesarios que se tienen que añadir para ver el mapa desde un punto de nuestra preferencia, Figura I.35. Esta información es usada por parte de la librería G3M. Modificando estos datos cambiaremos el punto de vista inicial del mapa en cuestión.

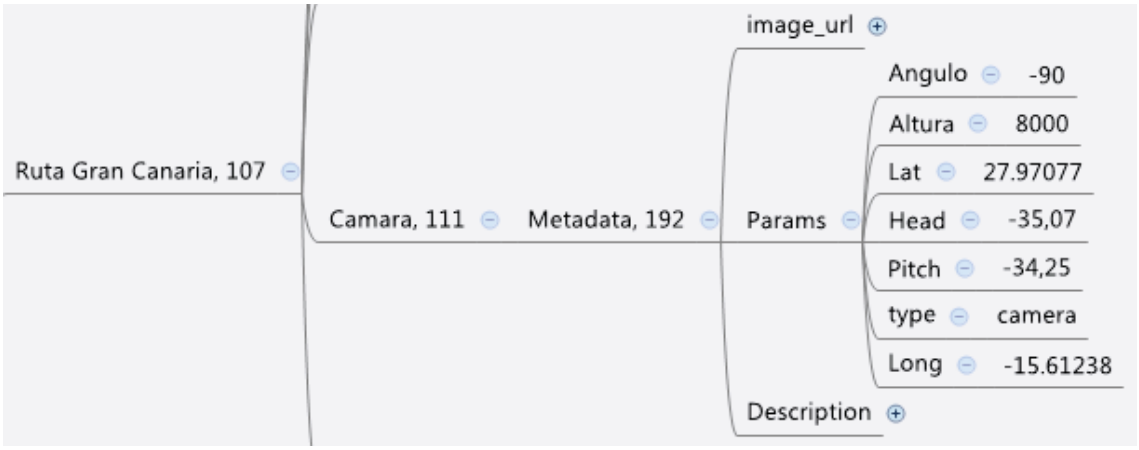


Figura I.35: Parámetros de la cámara

Por último, el contenido de *Sector_recorte* incluye los valores geográficos entre los cuales se encuentra el mapa al que se hace referencia en ese caso, Figura I.36.

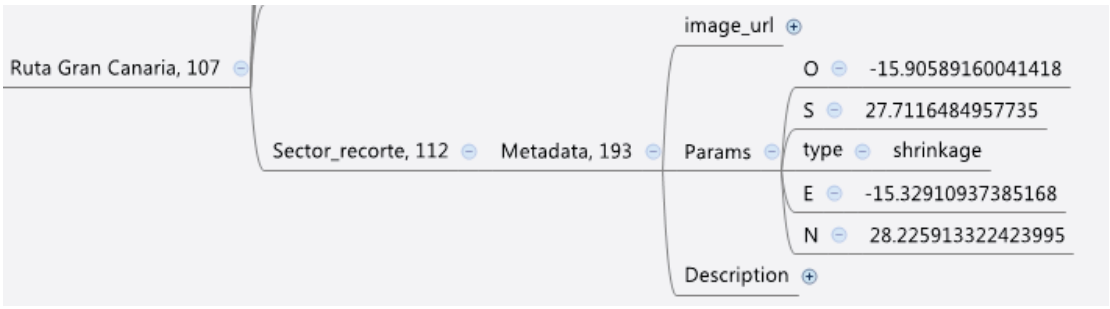


Figura I.36: Información del sector

Cabe hacer un breve inciso para hablar acerca de unos parámetros que siempre aparecen como son *image_url* y *Description*. Es información que solicita la librería G3M. El primer parámetro hace referencia a la dirección de una captura de pantalla del mapa en cuestión mientras que el segundo parámetro permite incluir un pequeño comentario acerca del mapa al que hace referencia. En nuestro caso no es necesario rellenar ambos parámetros.

4 . FUENTES DE MAPAS

En lo referente a la obtención de los mapas, se ha ido buscando la mejor opción para conseguirlos en función del trabajo que necesitemos para hacernos con ellos.

4.1 . Obtención imágenes del visor SIGPAC

La primera opción que se indagó fue la de utilizar el visor de mapas del SIGPAC, Sistema de Información Geográfica de Parcelas Agrícolas[65]. Perteneciente al Ministerio de Agricultura, Alimentación y Medio Ambiente el SIGPAC es una herramienta muy poderosa que nos permite visualizar al completo todo el territorio nacional, su principal uso de forma general es el de localizar los terrenos declarados por ganaderos y agricultores además de proporcionar varias capas vectoriales para diversos usos como pueden ser el de la identificación de los tipos de árboles(Arboles), zonas especiales de protección de aves(ZEPA) o las superficies de interés ecológico(SIE). También provee visores específicos para cada Comunidad Autónoma.

El camino a seguir para la obtención de mapas provenientes del SIGPAC es bastante fácil e intuitivo, toda vez que lo que tenemos que hacer es buscar la zona en la que estemos interesados e indicar en el menú desplegable ubicado en la esquina superior derecha de la Figura I.37 que estamos interesados en la capa llamada Ortofotos, término utilizado para las fotografías aéreas corregidas geométricamente. Cabe mencionar que la opción de ver el mapa mediante fotografías aéreas no se habilita automáticamente, sino que lo hace cuando estamos a un determinado nivel de zoom(nivel 14 para ser más exactos). Seguidamente podemos seleccionar en el menú de la esquina inferior izquierda de la Figura I.37 el *datum*[66] o sistema de referencia espacial, modelo que sirve para describir la forma del planeta en función de unos parámetros ajustables y una forma geométrica que puede ser un elipsoide o una esfera. De entre las opciones que ofrece este visor se encuentra con la que estamos trabajando como sistema de referencia

espacial principal, el *WGS84*[67][68]. Una vez que hemos llevado a cabo estos pasos ya estaremos en disposición de descargar nuestra imagen haciendo click en el icono Imprimir del menú que se encuentra en la zona superior izquierda de la Figura I.37.

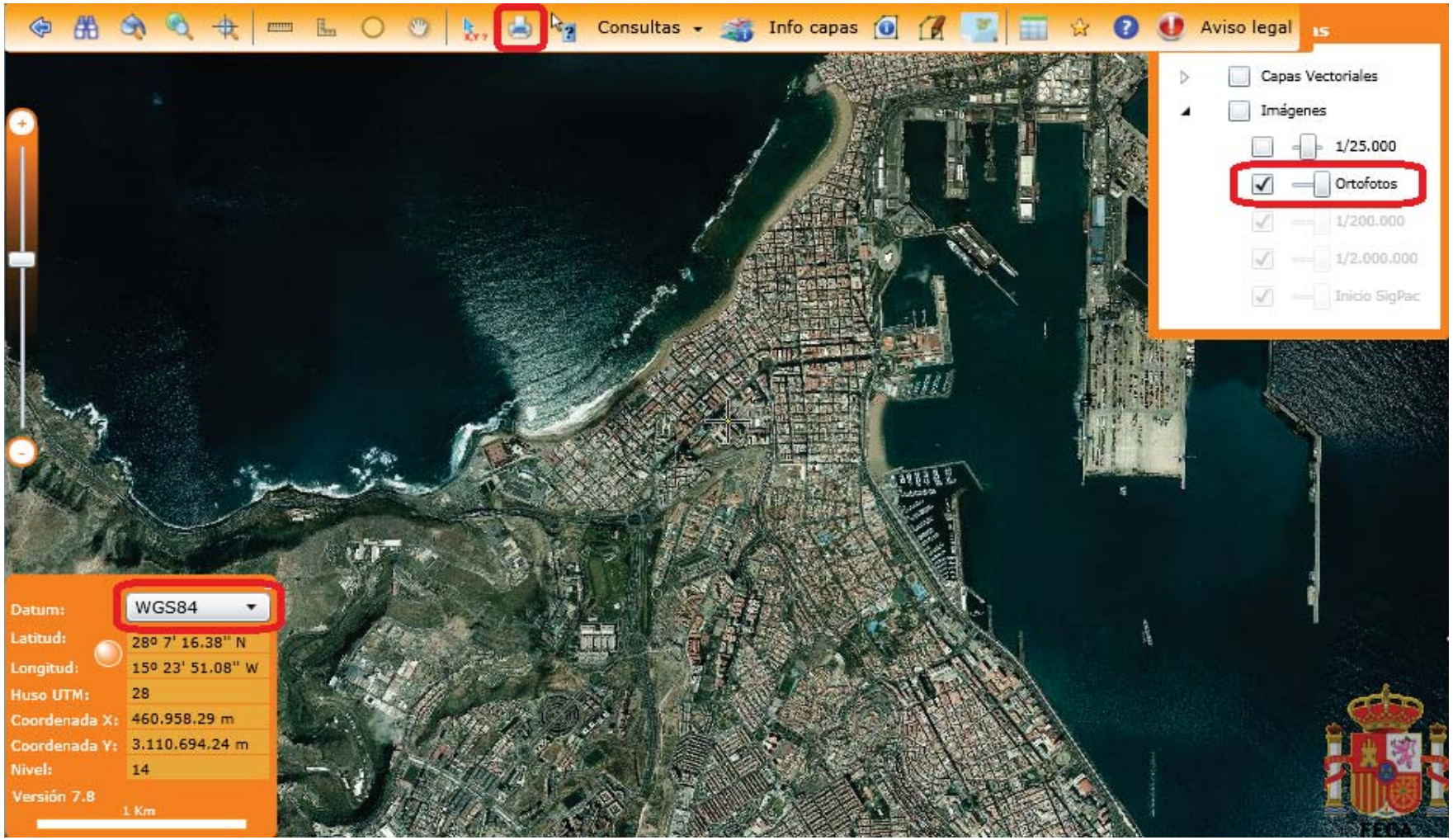


Figura I.37: Visor SIGPAC

Lo que obtenemos es un archivo en formato PDF con nuestra imagen en él, Figura I.38.

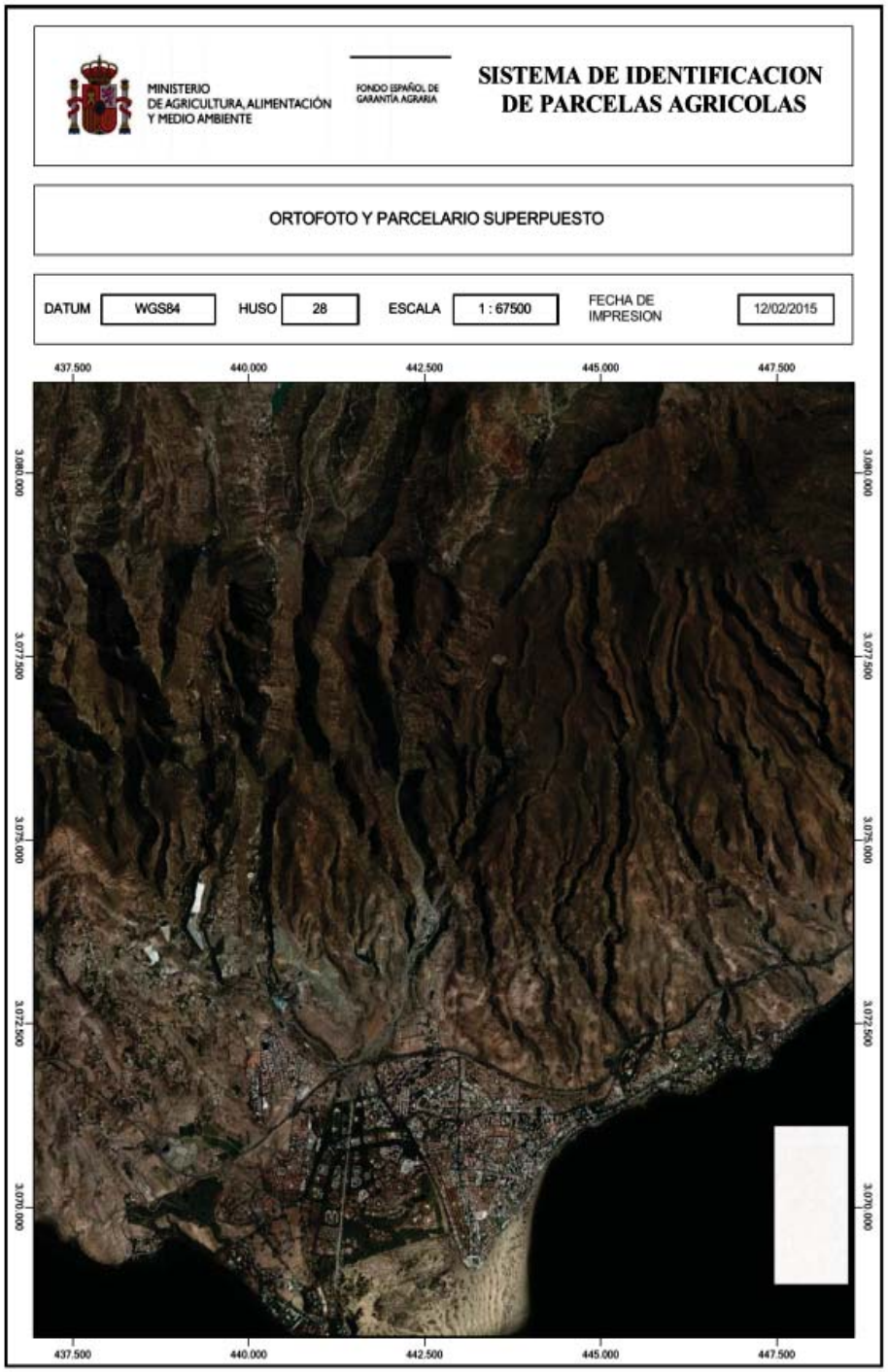


Figura I.38: Imagen obtenida del visor SIGPAC en formato PDF

Lo complicado de este método viene a continuación ya que tenemos que realizar una serie de pasos para poder tener un mapa normal y corriente. El primero de ellos es el de cambiar el formato del archivo a uno más propio de una imagen como pudiera ser TIF, para ellos nos ayudamos del siguiente comando de GDAL.

*gdal_translate –**of** GTiff nombre_archivo.pdf nombre_archivo.tif* (3.1)

Seguidamente lo que hacemos es buscar puntos de los cuales sepamos sus coordenadas en la imagen para utilizarlos a modo de referencia a la hora de georreferenciar. Para ello nos ayudamos de una herramienta que nos permita marcar varios puntos en la imagen, en nuestro caso serán cuatro puntos, Figura I.39, como puede ser GIMP y luego recortar el archivo y así quedarnos solo con la imagen.

Figura I.39: Representación de los cuatros puntos necesarios para georreferenciar una imagen con QGIS

Para la georreferenciación nos ayudamos del software de libre distribución QGIS, [69]. Teniendo operativo dicha herramienta nos vamos a *Ráster* → *Georreferenciador* → *Georreferenciador*.

Una vez en este apartado nos vamos a *Archivo* → *Abrir ráster* y seleccionamos la imagen. A continuación aparece una pantalla en la cual nos pide que especifiquemos el sistema de referencia espacial que tendrá la imagen, en nuestro caso es el *WGS84/UTM zone 28N,* Figura I.40.

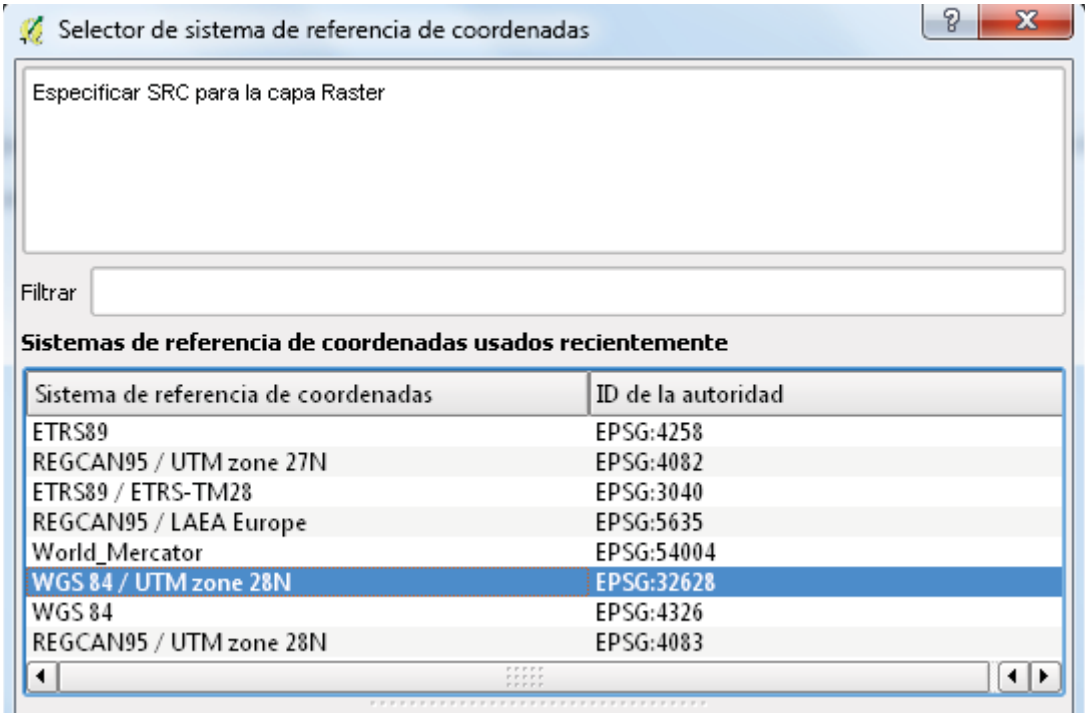


Figura I.40: Selección del sistema de referencia de coordenadas

La georreferenciación la hacemos con los puntos marcados anteriormente. Los vamos añadiendo uno por uno asignándoles el valor correspondiente, Figura I.41 y Figura I.42.

Figura I.41: Asignación de puntos basándonos en las marcas hechas en la imagen

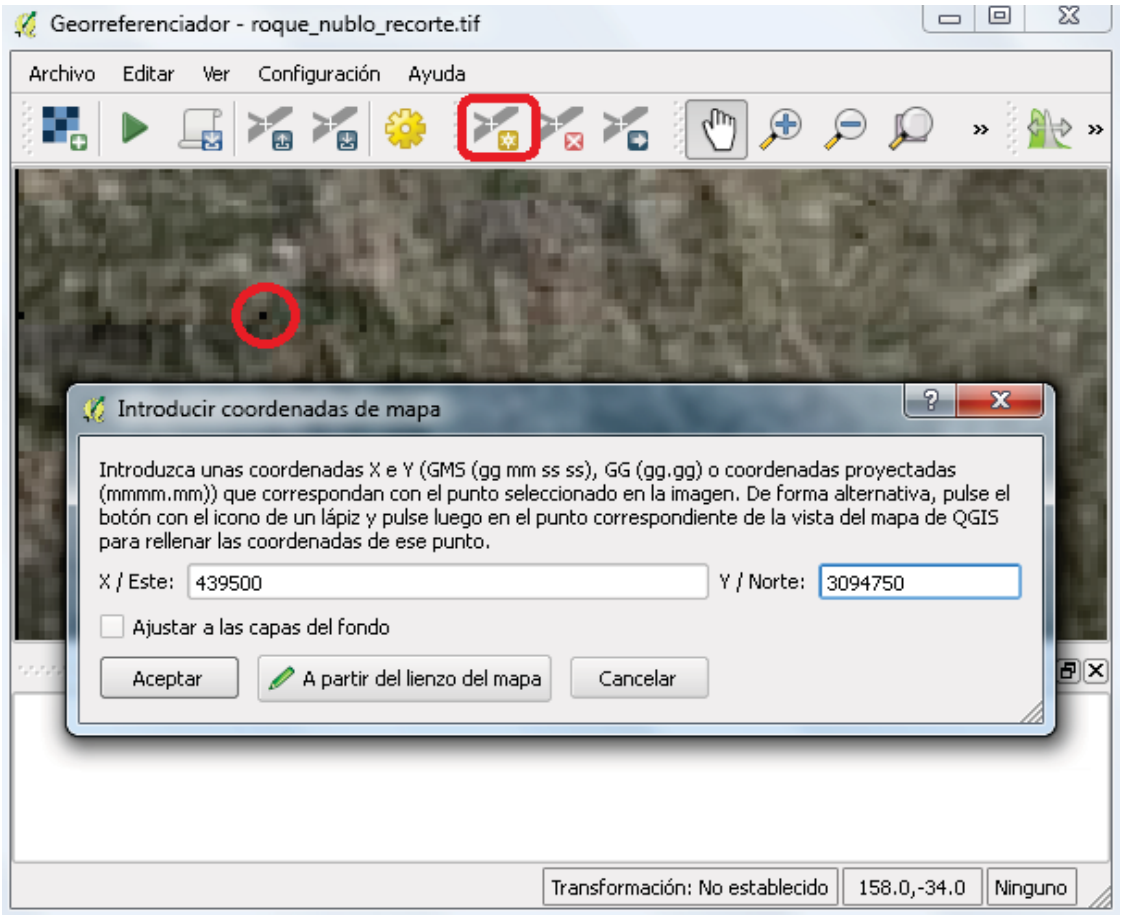


Figura I.42: Representación de los cuatros puntos con sus valores expresados en función del sistema de referencia dado

Como paso previo a terminar, una vez que seleccionamos el botón de comenzar, nos pide que establezcamos un tipo de transformación. Una vez hecho esto tendremos nuestra imagen georreferenciada.

Este método es por decirlo de alguna manera el más erróneo de todos ya que la georreferenciación nunca va a ser del todo exacta en comparación con imágenes ya georreferenciadas. Básicamente no llegaríamos a alcanzar una precisión tan grande, ello conlleva a tener errores de representación a la hora de añadir nuestro mapa como una capa ya que tendríamos problemas de solapamiento al no encajar del todo las coordenadas de nuestro mapa con las de otro servidor de mapas como Google, Figura I.43.

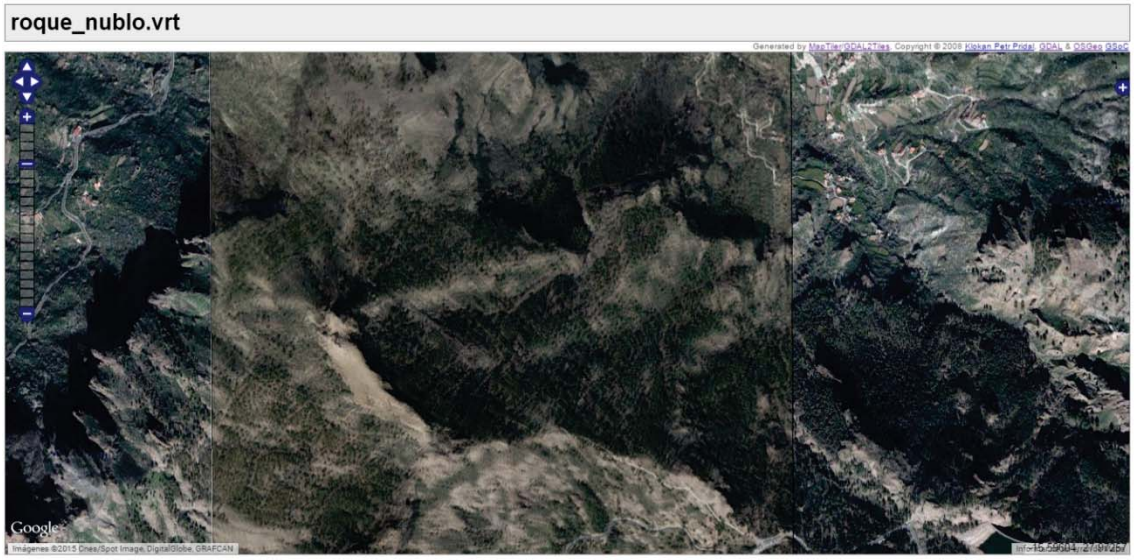


Figura I.43: Representación de los errores aparecidos a causa del uso de este sistema de georreferenciación

Aparte no nos permite a este nivel de resolución abarcar mucha más porción de terreno, por lo que si en el caso de estar interesados en un mapa de mayor tamaño no lo podemos hacer desde este visor.

4.2 . Obtención de imágenes del visor IBERPIX

La segunda opción que descubrimos fue la de utilizar otro visor, el cual en principio te permitía descargar imágenes ya georreferenciadas. Se trata del visor IBERPIX, propiedad del Instituto Geográfico Nacional (IGN). Esta herramienta[70]nos ofrece varias opciones a la hora de la visualización de mapas en función de nuestras necesidades, organizadas mediante capas. Dichas capas van cambiando en función del nivel de zoom que escojamos. En nuestro caso estamos interesados en la capa llamada "IMAGEN(ORTOFOTOS)", Figura I.44.

De la misma manera que con el visor del SIGPAC, el visor IBERPIX nos permite escoger, en nuestro caso *WGS84*, *REGCAN95* y *ED50*[68], entre varios

sistemas de referencia espacial. Nosotros nos decantamos por el *WGS84*. Para la descarga de la imagen solo tenemos que elegir la opción "Descargar Imagen Georreferenciada" disponible en la barra superior de la Figura I.44. A continuación se despliega en la parte superior derecha del visor una ventana en la que podemos elegir el tamaño de la imagen a descargar, Figura I.45.

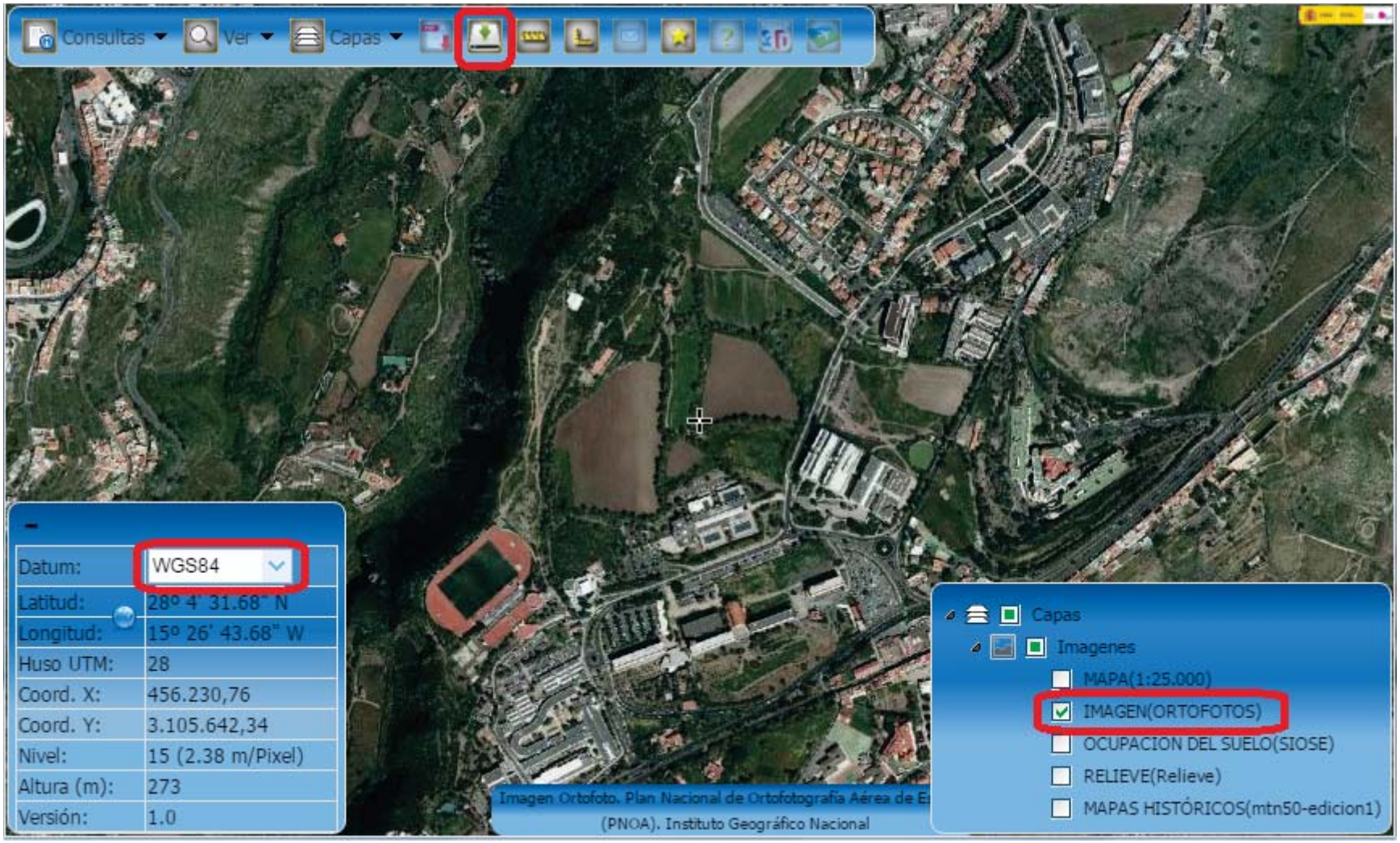


Figura I.44: Visor IBERPIX

Figura I.45: Ventana desde la cual se puede elegir el tamaño de la imagen georreferenciada a descargar desde el visor IBERPIX

La imagen descargada siempre vendrá acompañada de un archivo con extensión "*JGW*" que contiene información relativa a la georreferenciación, dicho archivo siempre tiene que estar en la misma carpeta y con el mismo nombre que la imagen a la cual proporciona los datos de georreferenciación ya que de no ser así no estaría georreferenciada.

Aunque en este caso no tenemos el problema referente a la georreferenciación seguimos sin tener la posibilidad de obtener mapas con mayor amplitud ya que al querer obtenerlo con este visor la capa de "ORTOFOTOS" cambia a otra capa llamada SPOT5(imágenes provenientes del satélites del mismo nombre) la cual ya no nos interesa. En nuestro afán de seguir indagando por alguna fuente que nos proporcionara ortofotos que abarcasen grandes porciones de terreno dimos con la siguiente opción.

4.3 . Obtención de los mapas del Centro de Descargas del IGN

La tercera opción a seguir nos vino gracias, en parte, a la anterior estrategia seguida debido a que los mapas del visor IBERPIX los proporciona el Centro de Descargas del Centro Nacional de Información Geográfica[61]. Por lo tanto, era algo evidente el probar directamente con esta otra fuente de imágenes.

Dicha fuente provee una gran variedad de mapas a distintas resoluciones y tamaños, ordenados, dependiendo del caso, por Autonomías, provincias, etc. Pinchando en la pestaña llamada "Catálogo de productos" podemos acceder a todo la variedad de mapas que ofrecen.

Algunos de los productos que ofrecen son las ortofotos de máxima actualidad o históricas provenientes del Plan Nacional de Ortografía Aérea (PNOA). También nos encontramos con el Mapa Topográfico Nacional (MTN) en dos escalas, 1:25,000 Figura I.46 y 1:50,000 Figura I.47, ambos en su versión ráster.

Figura I.46: Representación del Mapa Topográfico Nacional a escala 1:25,000 en formato ráster

Figura I.47: Representación del Mapa Topográfico Nacional a escala 1:50,000 en formato ráster

Otro ejemplo de los productos que podemos encontrar en el Centro de Descargas son las versiones vectoriales del MTN en las mismas escalas, Figura I.48.

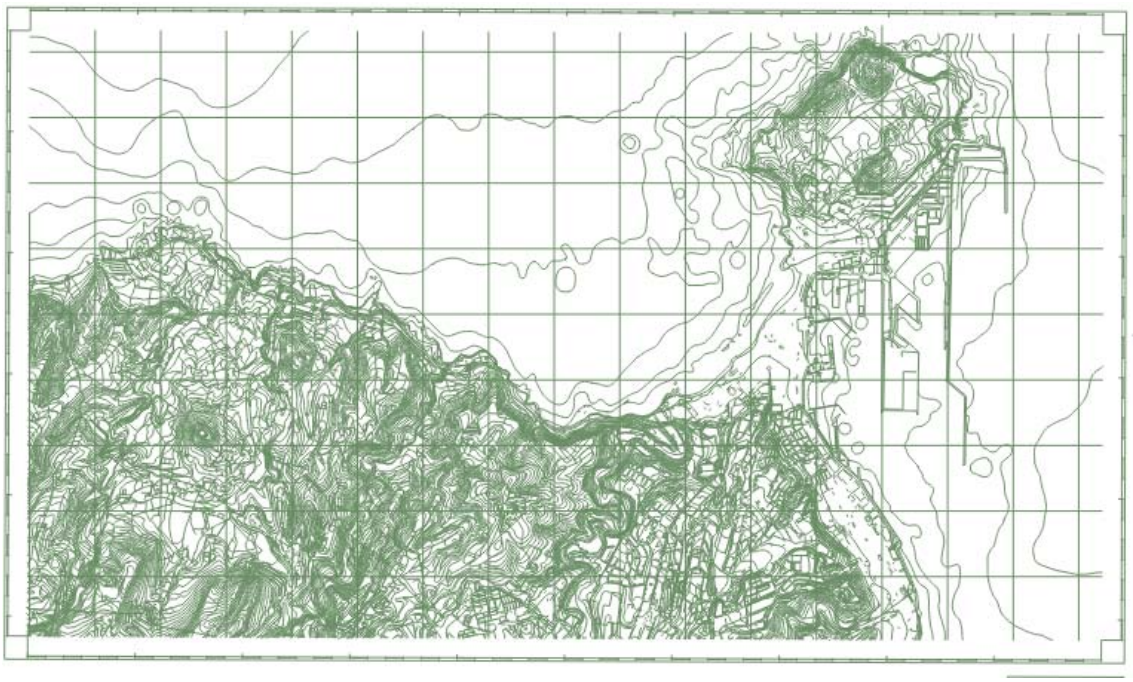


Figura I.48: Representación del Mapa Topográfico Nacional a escala 1:50,000 en formato vectorial

Además tenemos acceso a las versiones, tanto ráster Figura I.49 como vectorial, del mapa provincial a escala 1:200,000.

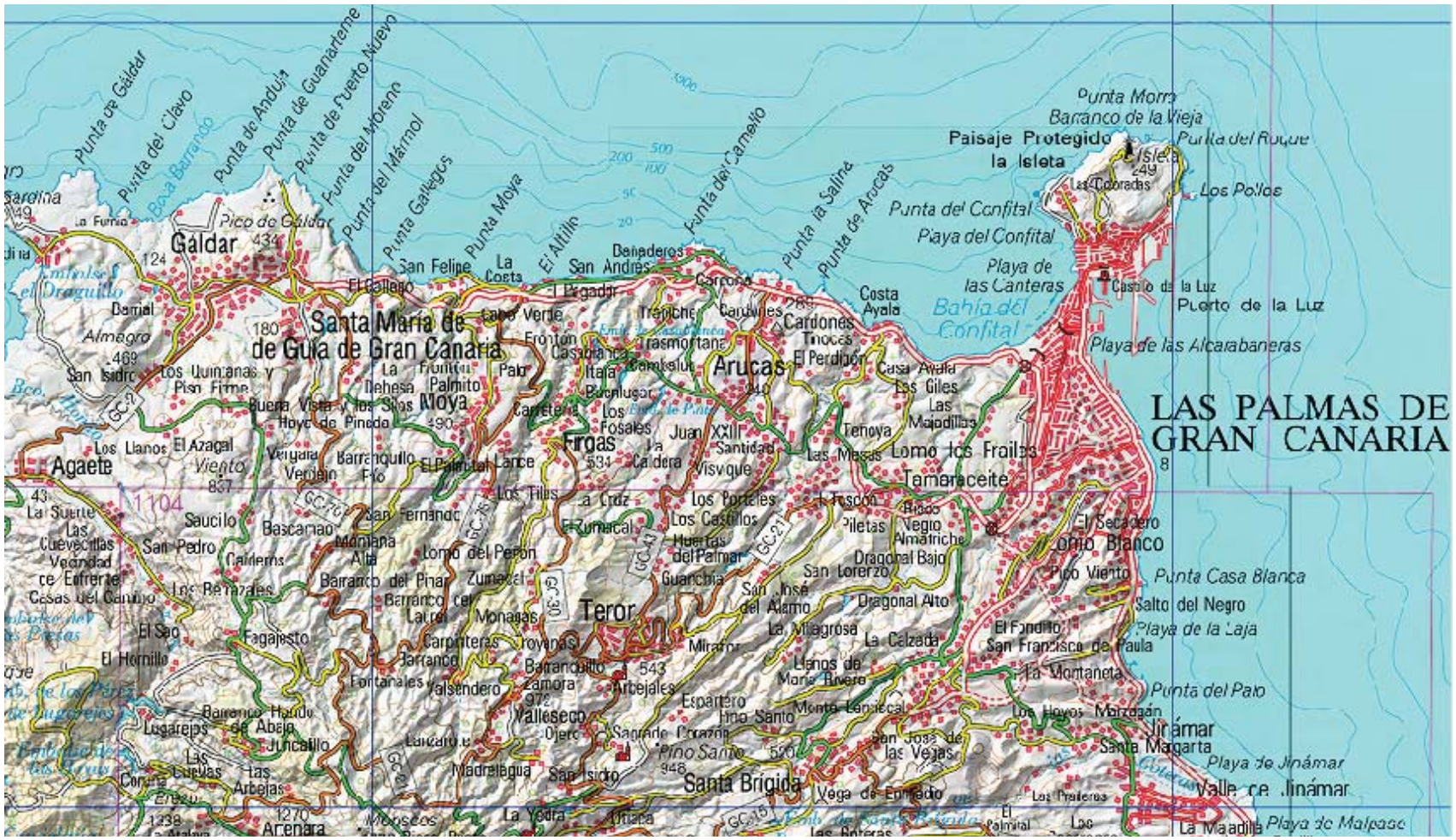


Figura I.49: Representación del Mapa Topográfico Nacional a escala 1:200,000 en formato ráster

A modo de curiosidad podemos citar también un tipo de producto ofrecido como es el de la Cartociudad, mediante el cual podemos acceder a diferentes capas como las que delimitan el terreno en función de su código postal, Figura I.50, del municipio, Figura I.51 o capas que nos muestran los tramos viales, Figura I.52.

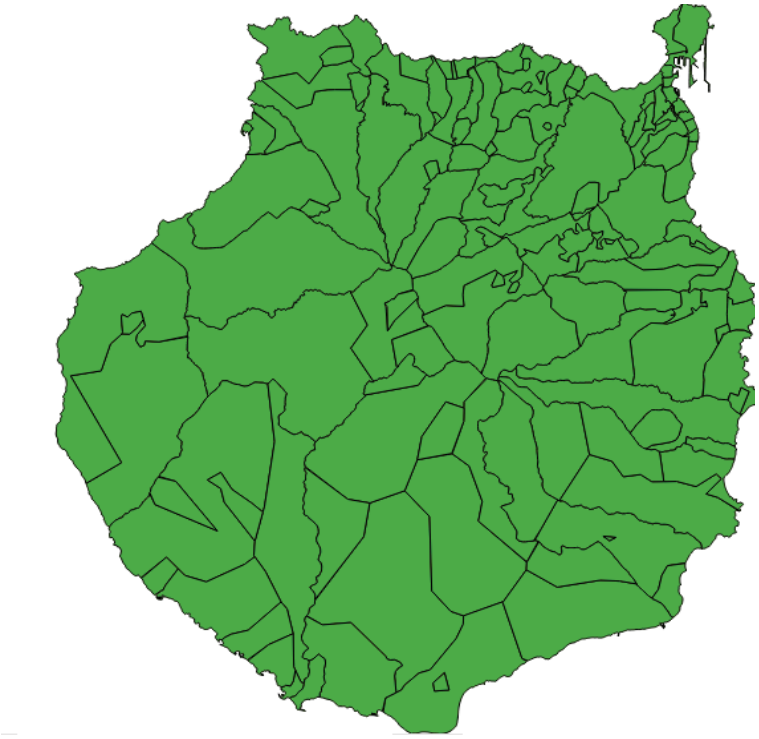


Figura I.50: Representación del producto Cartociudad, en función de los códigos postales existentes en la isla de Gran Canaria, en formato vectorial

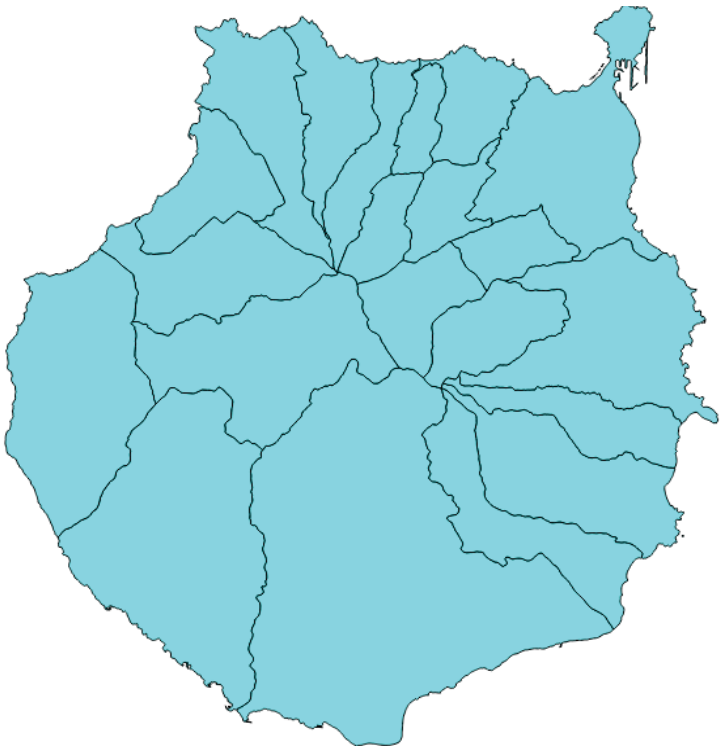


Figura I.51: Representación del producto Cartociudad, en función de los municipios de la isla de Gran Canaria, en formato vectorial

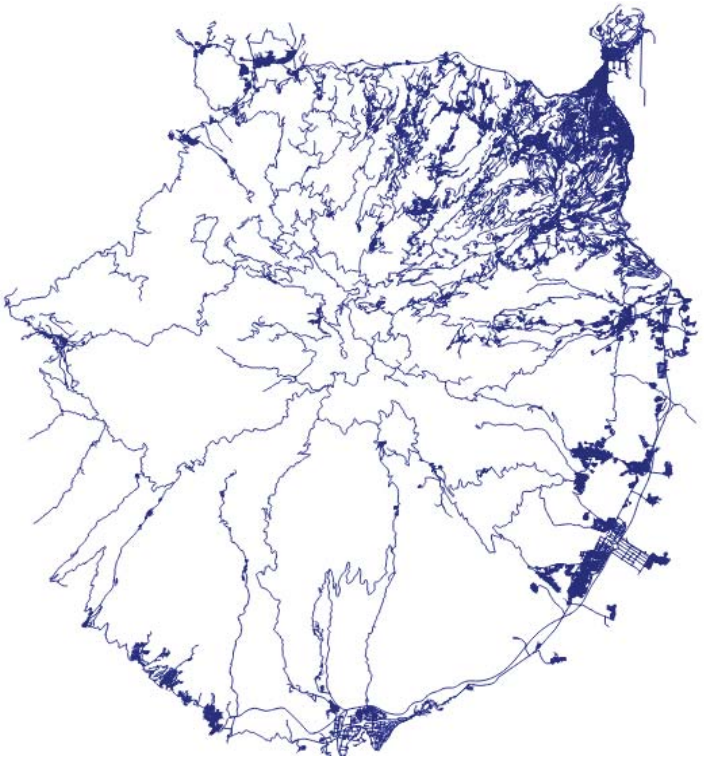


Figura I.52: Representación del producto Cartociudad, en función de los tramos viales de la isla de Gran Canaria, en formato vectorial

Aparte nos permite la búsqueda de toda la información que posee de dos maneras. Por un lado, tenemos la opción de acceder a un visor mediante el cual podemos obtener los productos en función de la escala que queramos entre otras opciones, Figura I.53. La otra opción es la bautizada "Búsqueda avanzada", en ella podemos acceder a dos pestañas desplegables desde las cuales accedemos a toda la variedad de productos que ofrece el CNIG y en la otra tenemos la opción de hacer la búsqueda más específica al poder especificar la zona que deseemos, Figura I.54.

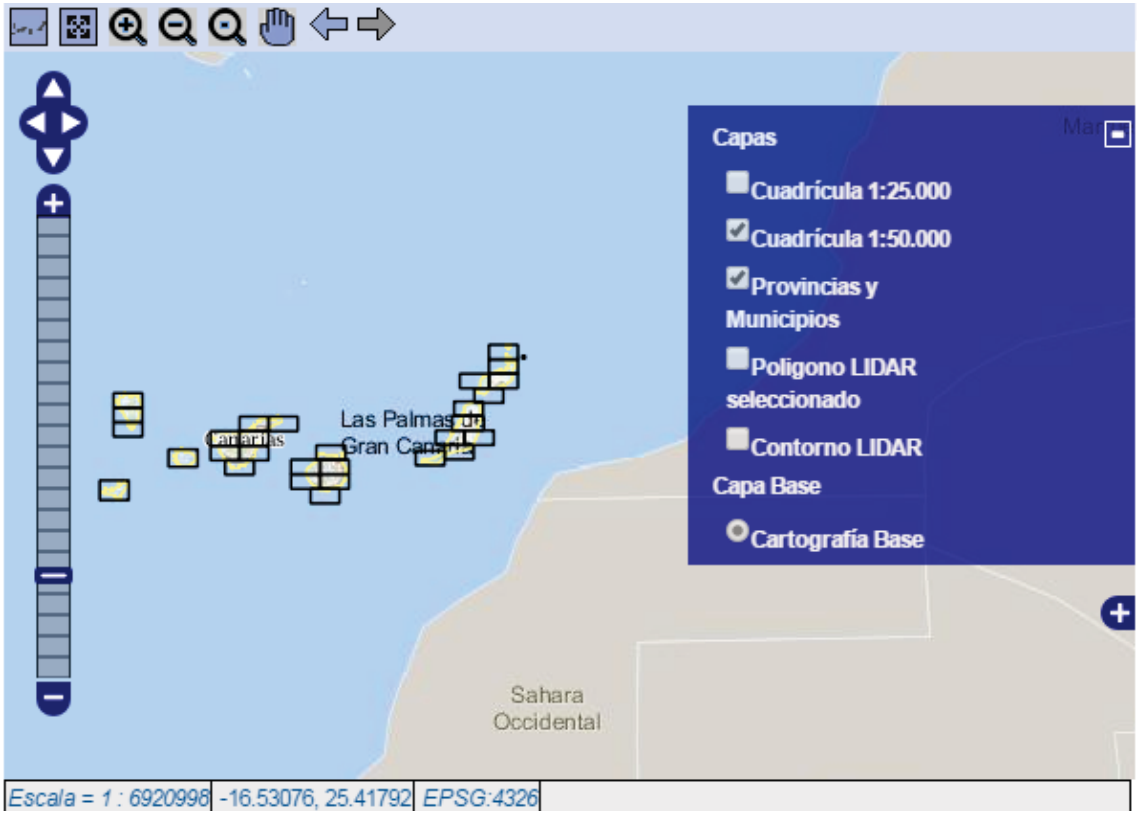


Figura I.53: Representación del visor desde el cual se acceden a los distintos productos del IGN.

Figura I.54: Forma alternativa para acceder a los productos del IGN

5 . TRATAMIENTO DE MAPAS: TRABAJO CON MAPBOX, QGIS, GDAL Y TILEMILL.

En este capítulo hablaremos acerca de todo lo proceso seguido con los mapas una vez que los hayamos obtenido de las fuentes descritas en el capítulo anterior. Comenzamos con una breve introducción de la herramienta Mapbox orientada al diseño y almacenaje de mapas. Luego iremos desgranando los pasos dados con los mapas, las herramientas a utilizar en este proceso así como algunas incidencias que se han tenido a lo largo de dicho proceso.

5.1 . Introducción a la herramienta Mapbox

Mapbox[71] es una herramienta de libre acceso con la que podemos crear multitud de mapas de diferentes. Nos permite el desarrollo tanto para iOS como para Android, algo que nos interesa en particular debido al enfoque de este proyecto. La interfaz de programación de aplicaciones de dicha herramienta esta basada en el lenguaje de programación JavaScript, que a su vez esta íntimamente relacionada con otra librería de desarrollo de mapas llamada Leaflet[72].

A la hora de la creación de mapas, Mapbox nos da la posibilidad de crearlos de dos formas que se podrían catalogar como online y offline. Para la creación de mapas de manera online tenemos a nuestra disposición del llamado “Mapbox Editor”. Dicho editor pone a nuestra disposición vías para añadir datos así como una variedad de estilos para, de esta forma, enriquecer de una forma visual nuestros futuros mapas. Todo esto lo podemos hacer desde la barra herramientas disponible, Figura I.55.

Figura I.55: Barra de herramientas de Mapbox Editor

La opción “Style” nos permite escoger de entre una gran variedad de estilos lo cual nos permite darle un toque más personal a nuestros proyectos, Figura I.56.

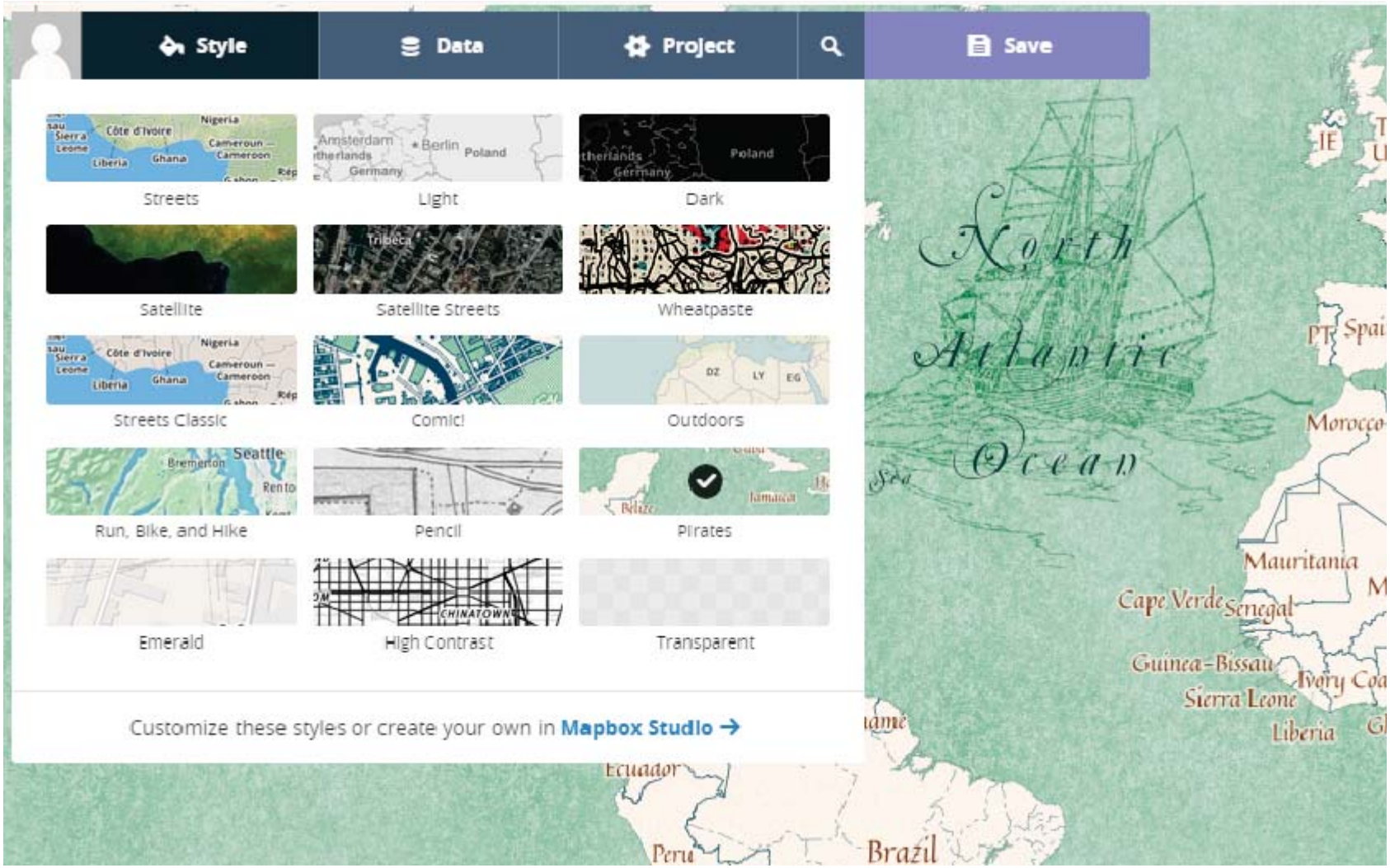


Figura I.56: Representación de los diferentes estilos que ofrece Mapbox Editor para aplicar los mapas.

Seleccionando “Data” de la barra de herramientas podremos añadir toda la información que queramos siempre que sean soportados por Mapbox, algunos de los formatos soportados son “*KML*” o “*GeoJSON*”. Aparte podemos añadir polígonos, líneas o insertar nuestros propios mapas, Figura I.57, previamente cargados en Mapbox.

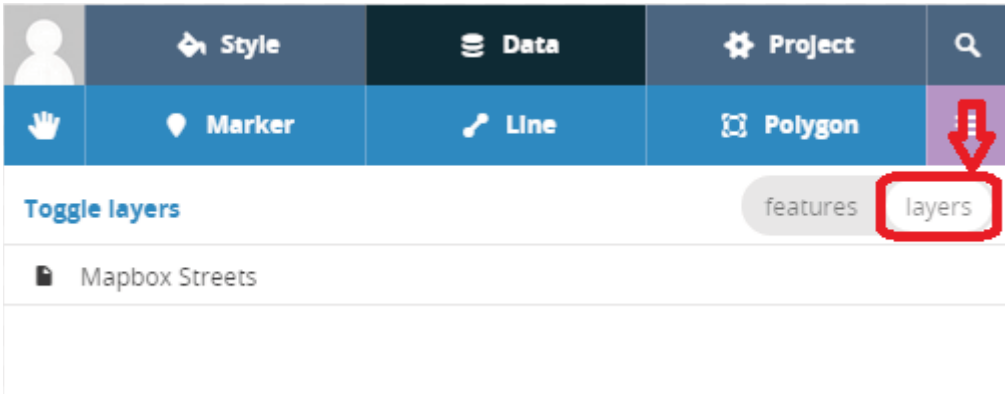


Figura I.57: Opción desde la cual se puede añadir nuestros propios mapas

Por último, la pestaña “Project” nos provee de un conjunto de opciones como la de poder compartir el mapa o empotrarlo en páginas web.

El tratamiento de mapas, creación y diseño, para Mapbox de manera offline se puede acometer por dos vías, las cuales vendrían a ser dos herramientas diferentes. Si nos decantamos por mapas con tiles en formato vectorial la herramienta a manejar es Mapbox Studio, Figura I.59, software enfocado hacia ese formato de tiles que además nos permite diseñar mapas por medio de CartoCSS. Si por el contrario, queremos mapas de tiles en formato ráster debemos utilizar TileMill,Figura I.58, herramienta que también permite añadir estilo por medio de CartoCSS.

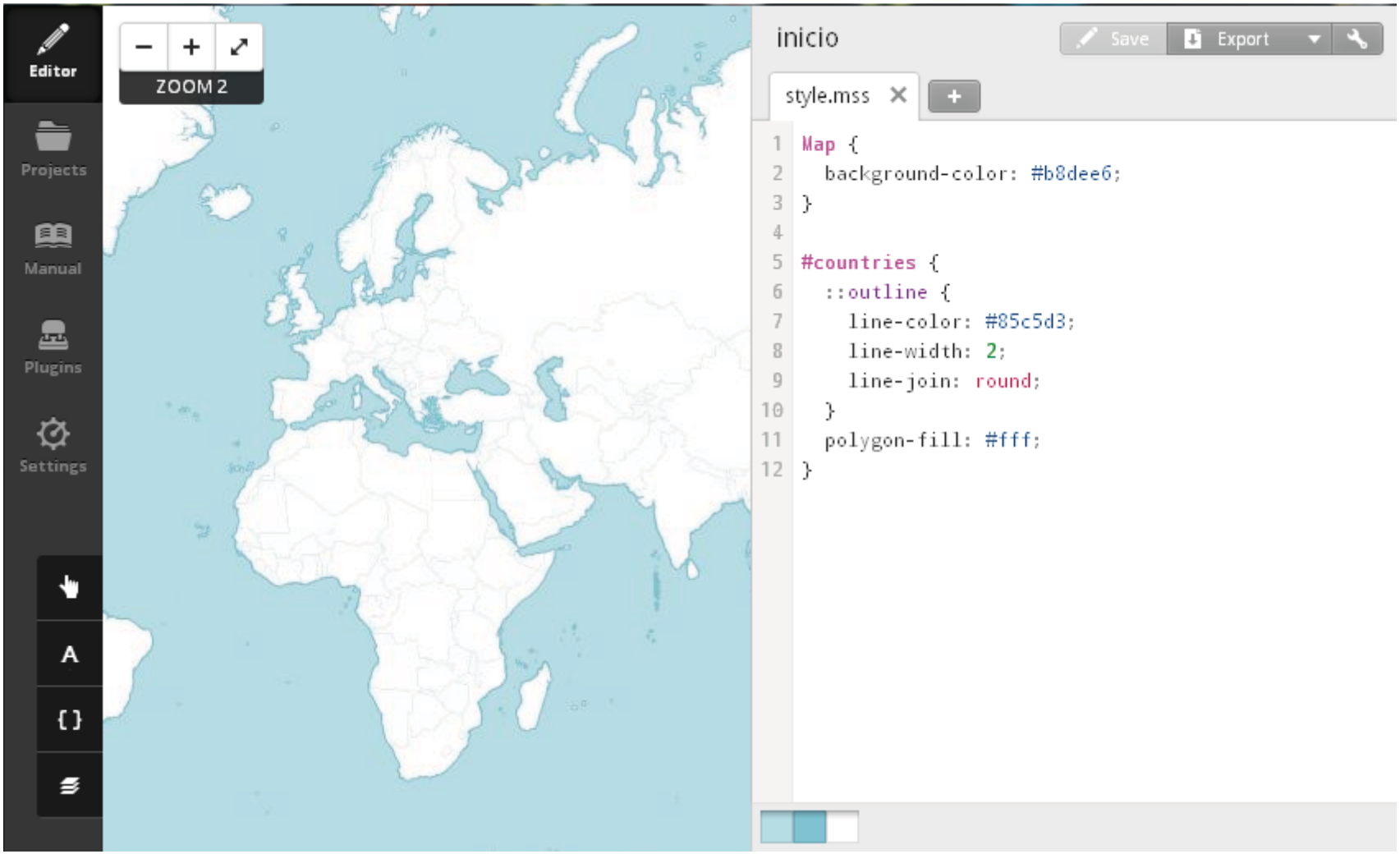


Figura I.58: Vista inicial de la herramienta TileMill, orientada al trabajo con mapas en formato ráster

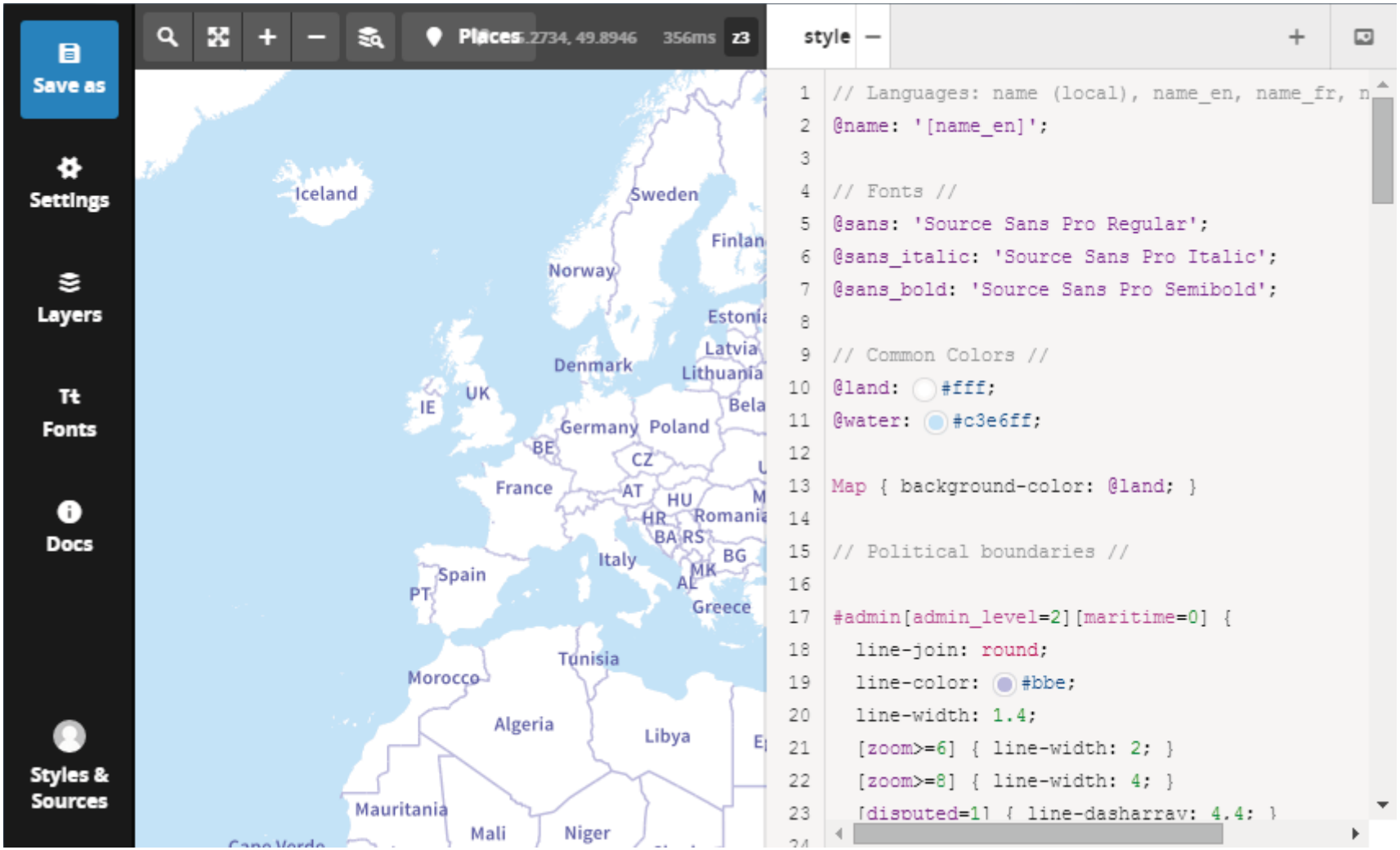


Figura I.59: Vista inicial de la herramienta Mapbox Studio, orientada al trabajo con mapas en formato vectorial

A raíz de que el trabajo con tiles vectoriales es relativamente nuevo, y por ende la utilización de Mapbox Studio, finalmente integraremos el software TileMill en el proceso que estamos llevando a cabo ya que es más intuitivo su manejo además de estar mejor documentado. Una de las desventajas que tiene la utilización de dicha herramienta es que casi está en desuso, lleva un tiempo sin ninguna actualización, debido a que Mapbox tiene como filosofía la utilización de mapas con tiles vectoriales. Una vez explicada la herramienta a usar para alojar los mapas podemos comenzar a desarrollar el proceso llevado cabo hasta tener los mapas en Mapbox.

La primera tarea a llevar a cabo es la de obtener el mapa de alguna de las fuentes mencionadas anteriormente. En nuestro caso, tal y como explicamos en su momento, las imágenes las obtendremos del CNIG cuando procedan. A continuación se explica el proceso seguido para cada tipo de mapa.

5.2 . Caso ortofotos

En este caso las ortografías ofrecidas por el CNIG estan organizadas en un sistema de cuadrículas creado a partir del Mapa Topográfico Nacional a escala 1:50,000 o MTN50. La isla de Gran Canaria viene organizada en seis cuadrículas enumeradas, Figura I.60.

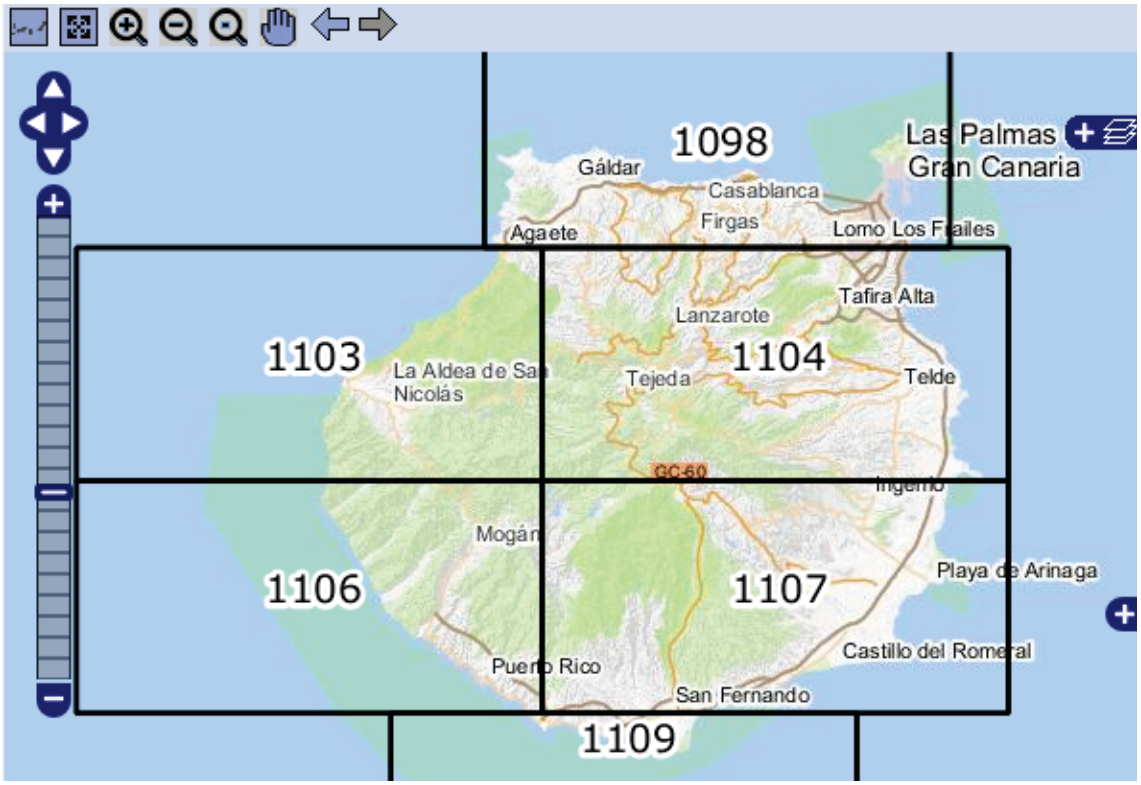


Figura I.60: Representación de la organización de las ortofotos de la isla de Gran Canaria por parte del visor del IGN

Estas imágenes las obtenemos en un formato y proyección distintos al que tenemos que utilizar por lo cual nos fuerza a realizar algunos cambios. Referente al formato, los mapas de cada cuadrícula nos son dados en el formato *ECW* (Enhanced Compressed Wavelet)[73]. Es un formato muy utilizado en geomática debido a sus ventajas de compresión de imagen además de que preserva la georreferenciación. Como nuestro formato es el *GeoTIFF*[74] tenemos que hacer uso del comando *gdal_translate* de GDAL para pasar de un formato a otro. En cuanto al sistema de referencia empleado en la cartografía del CNIG también reuqiere realizar cambios ya que el utilizado por el CNIG es el *EPSG:4083* o también llamado *REGCAN95 / UTM zone 28N* y el sistema a emplear por nuestra parte, el cual explicar más tarde su por qué, es el *EPSG:4326* o *WGS84*. Dicho cambio se lleva a cabo mediante GDAL al utilizar el comando *gdalwarp*.

Para obtener la ortofoto georreferenciada correspondiente de la isla de Gran Canaria seguiremos los siguientes pasos:

- Partiendo de las cuadrículas descargadas del CNIG, las tenemos que unir para crear la isla en su totalidad mediante el comando de GDAL *gdal_merge*. Este proceso puede tardar fácilmente varias horas, debido básicamente al tamaño de las ortofotos. En nuestro caso se realizó en aproximadamente 24 horas.

- El siguiente paso es el de cambiar el sistema de coordenadas de la ortofoto resultante por el que nos interesa a nosotros que es el WGS84. Esto lo hacemos con otro comando de GDAL llamado *gdalwarp*. Decir que este cambio de proyección de la ortofoto, al igual que con el anterior paso, se llevó a cabo en 1 un día, Figura I.61.

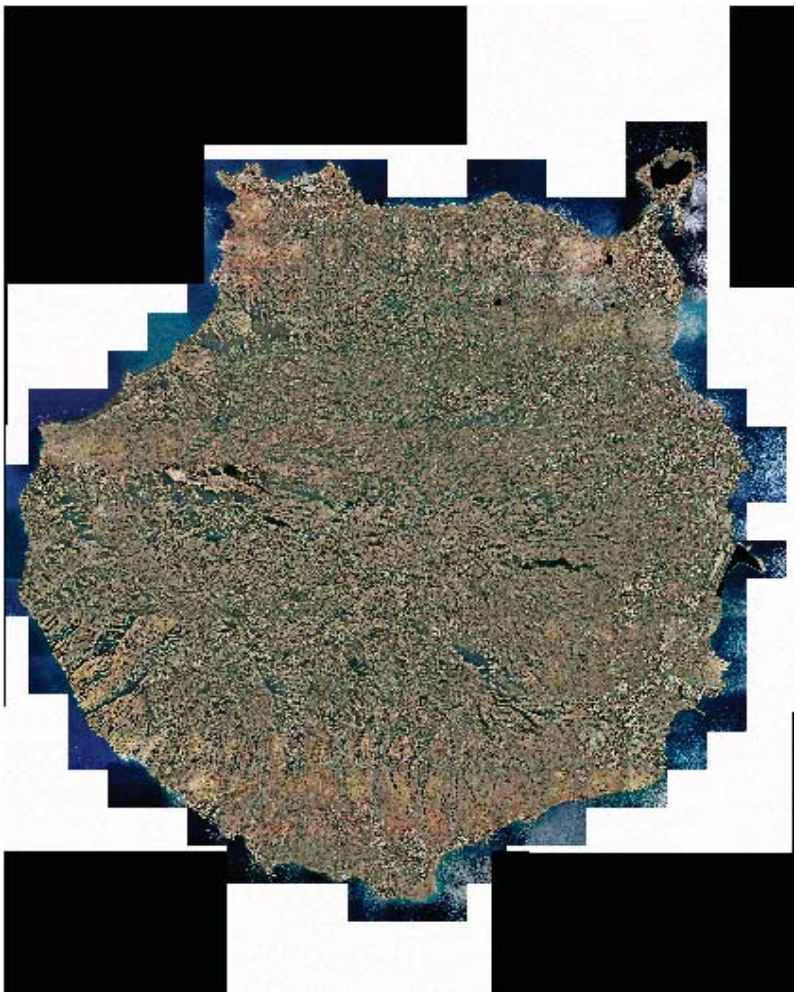


Figura I.61: Resultado de unir mediante la herramienta GDAL las diferentes cuadrículas que conforman la isla de Gran Canaria

Como se puede apreciar, la ortofoto resultante queda cuando menos poco vistosa. Para corregir de alguna manera este error tenemos que recortar la ortofoto para que solo quede bisible la isla.

Para llevar a cabo esta tarea tenemos que crear una especie de máscara que contenga la silueta de la isla de Gran Canaria. Nosotros creamos dicha máscara a partir de la cartografía del SIANE (Sistema de Información Geográfica del Atlas Nacional de España) en formato Shapefile(del cual se hablará en el siguiente capitulo), obtenida del Centro de Descargas del IGN. Una vez descargados estos archivos, utilizamos el que

tiene una escala de 1:3000000, el cual muestra las islas del archipiélago a partir de sus municipios. Como solo estamos interesados en Gran Canaria seleccionamos todos los archivos relacionados con los municipios de dicha isla y los movemos a una nueva carpeta. El siguiente paso conlleva crear la isla de Gran Canaria a partir de estos archivos mediante una de las opciones o herramientas que ofrece QGIS para el tratamiento de mapas vectoriales, que en nuestro caso se trata de combinar múltiples archivos con formato Shapefile en uno solo. Para ello tenemos que irnos en la herramienta QGIS a Vectorial → *Herramientas de gestión de datos* → *Combinar archivos shape en uno,* Figura I.62.

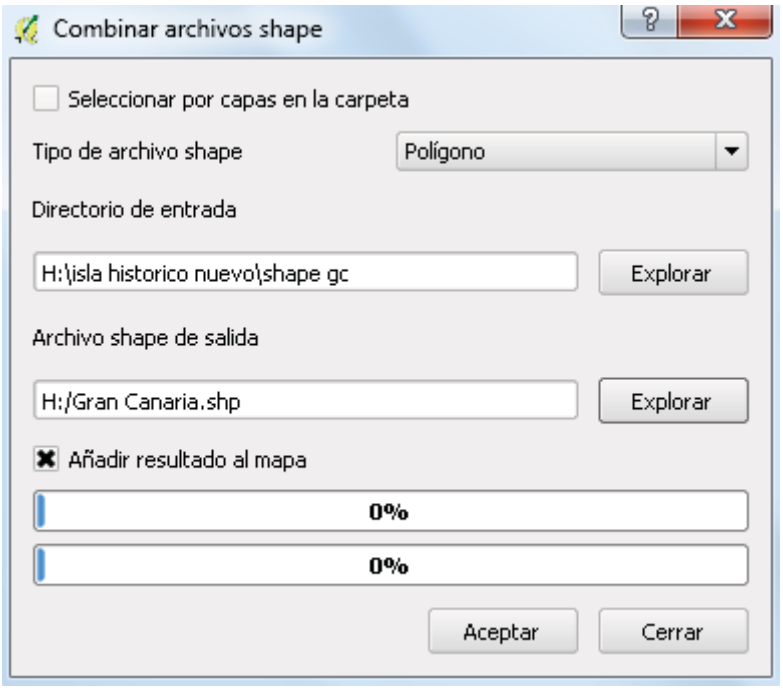


Figura I.62: Utilidad que permite la unión de varios archivos Shapefile

Como resultado obtenemos un mapa vectorial en formato Shapefile que solamente incluye a la isla de Gran Canaria, Figura I.63, dicho archivo será la máscara a utilizar en el siguiente paso.

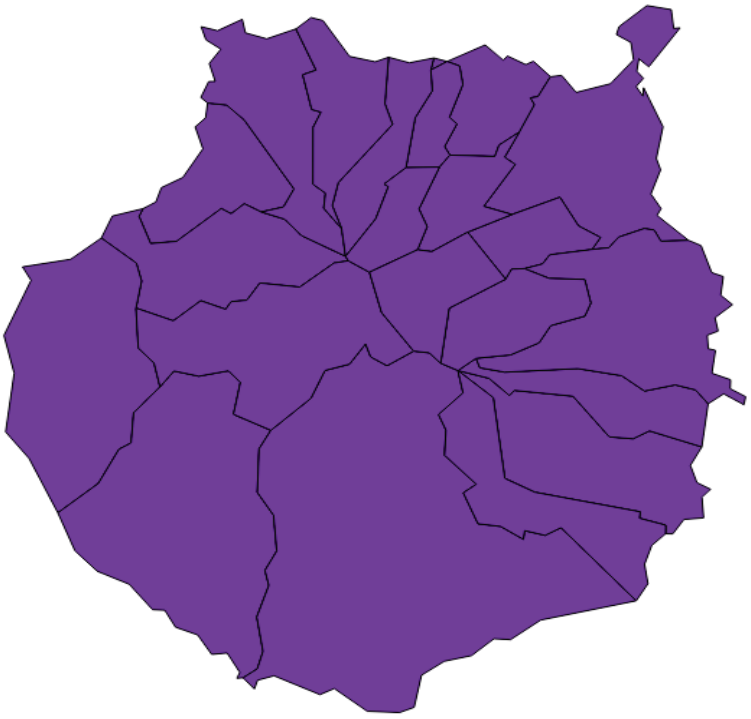


Figura I.63: Resultado de aplicar la utilidad de combinación de archivos Shapefile

Con la máscara en nuestras manos, el último paso es el de aplicar dicho archivo a la ortofoto anteriormente mostrada para obtener una nueva ortofoto en la cual solo se viese la isla de Gran Canaria. Para ello nos iremos nuevamente en QGIS a *Ráster → Extraccion → Clipper, Figura I.64*.

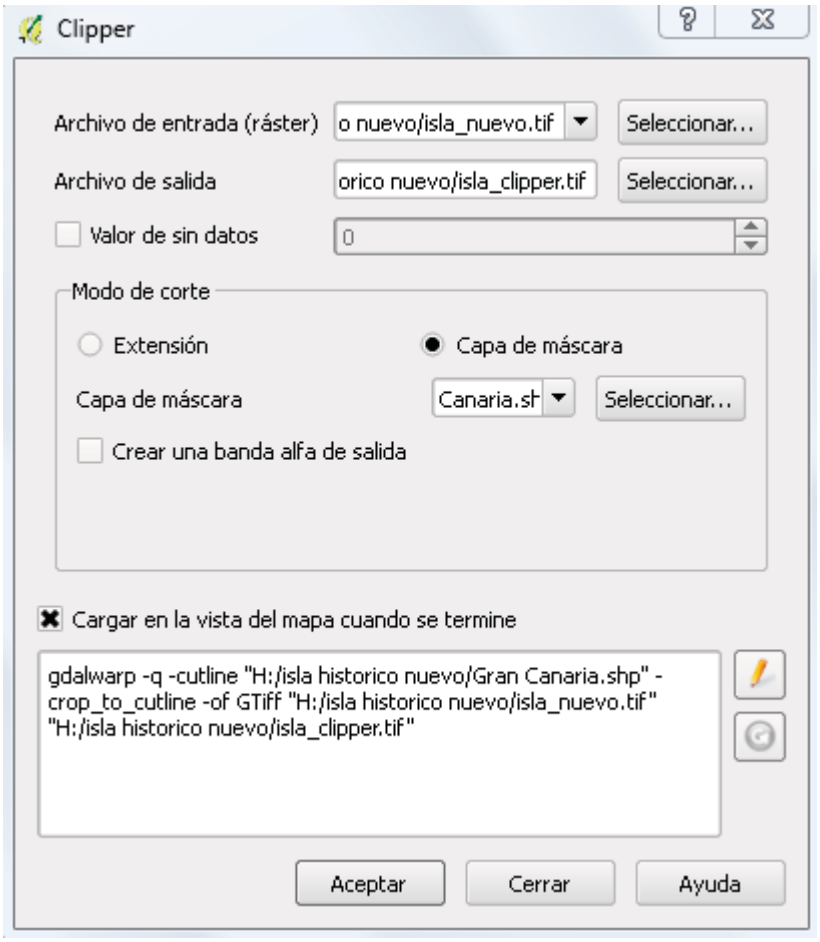


Figura I.64: Utilidad de QGIS que permite recortar un mapa en base a una máscara dada

Este último proceso también puede tardar varias horas ya que estamos trabajando con imágenes de más de 50 GB de tamaño. El resultado es el que vemos a continuación, Figura I.65.

Figura I.65: Resultado de aplicar la utilidad Clipper

5.3 . Caso líneas isométricas

En el caso de los mapas de líneas isométricas los obtenemos a partir del Mapa Topográfico Nacional a escala 1:25000, MTN25, que ofrece el CNIG. El sistema geodésico de referencia en el cual estan las cuadrículas es el *EPSG:32628*. En este caso las imágenes las podemos obtener tanto en *ECW* como en *TIFF+TFW*, donde el archivo con la extensión *TFW* almacena toda la información relacionada con la georreferenciación.

Su proceso de creación ha sido el mismo que con el de la ortofoto, solo que cambiando el tipo de mapas. En un primer paso lo que tenemos que hacer es combinar todas las cuadrículas descargadas del Centro de Descargas del IGN mediante el comando *gdal_merge, Figura I.66*.

Figura I.66: Representación del resultado de aplicar el comando gdal_merge al mapa de líneas isométricas de la isla de Gran Canaria

A continuación, nos hacemos cargo de eliminar toda la zona azul de la imagen mediante la máscara creada para el caso de las ortofotos, Figura I.67.

Figura I.67: Imagen del mapa de líneas isométricas de la isla de Gran Canaria luego de aplicar la utilidad llamada Clipper

5.4 . Caso geológicos

Para el caso de los mapas geológicos, el departamento de Geomática de la ULPGC y más concretamente Bárbara del Castillo-Olivares y Suárez nos proporcionaron dicho material. La información recibida constó de dos archivos por cada etapa de la Georruta, en total seis archivos. Para la *Etapa 1* tenemos un archivo con extensión *JPG*(el mapa en sí) y otro archivo con extensión *JGW,* Figura I.68. Este último archivo contiene toda la información acerca de la georreferenciación del mapa, además tiene que ser nombrado idénticamente que el mapa al que hace referencia, de otra forma el mapa no estaría georreferenciado.

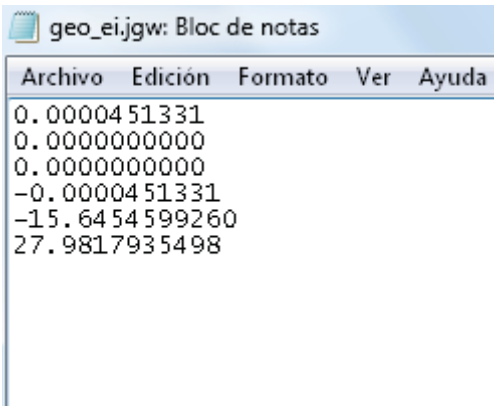


Figura I.68: Contenido del archivo con extensión JGW

El formato de este tipo de archivos suele seguir el siguiente patrón:

- La primera fila indica el tamaño de píxel en el eje x, comúnmente llamado Px.
- La segunda fila hace referencia al Parámetro de giro A, no se usa.
- La tercera fila, tampoco se usa, indica el Parámetro de giro B.
- La cuarta fila, da información acerca del tamaño de píxel en el eje Y, Py. Su valor es negativo ya que las imágenes van de arriba hacia abajo.
- La penúltima fila indica la coordenada X del centro de píxel superior izquierdo, Cx.
- La última fila hace referencia a la coordenada Y del centro de píxel superior izquierdo, Cy.

Cabe decir que esta información se refiere al sistema de coordenadas UTM (metros). Para el sistema de coordenadas geográficas (grados), basta con sustituir la información del eje X por la Longitud y la información del eje Y por la Latitud.

Luego de hacer el correspondiente cambio de sistema de coordenadas al *WGS84* mediante el comando *gdalwarp* tendremos listos los mapas geológicos de cada una de las etapas, Figura I.69.

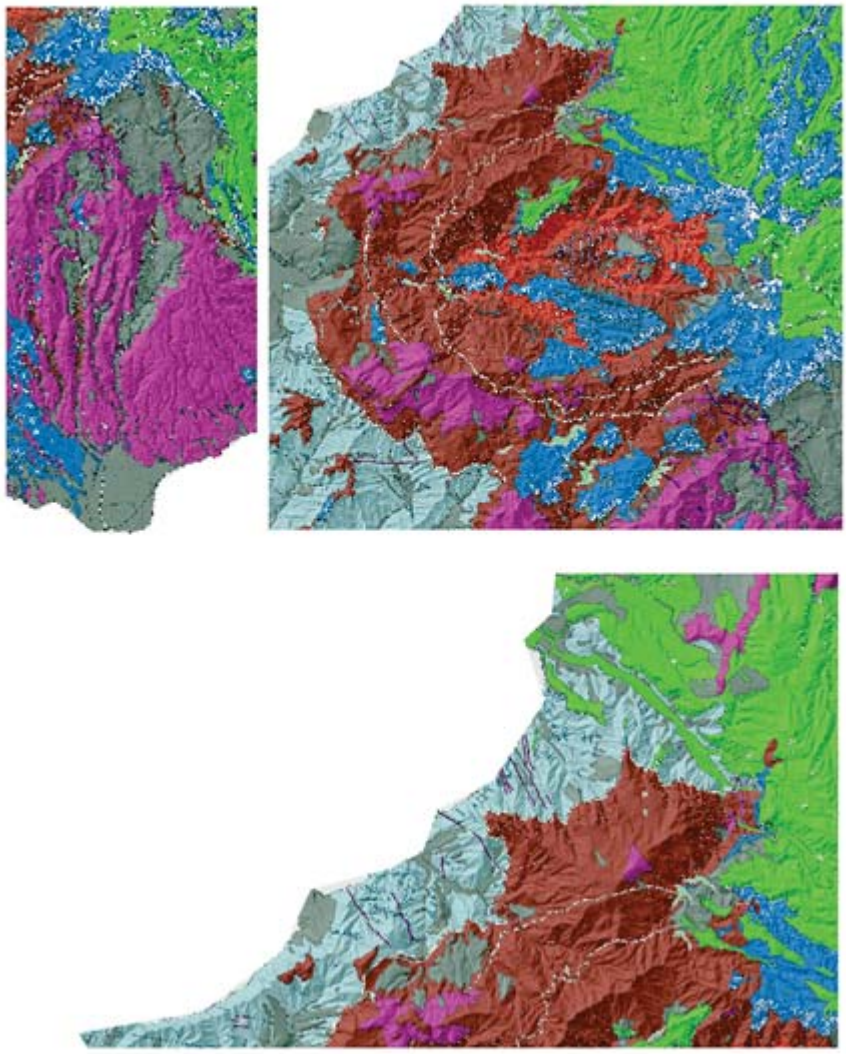


Figura I.69: Representación de los mapas geológicos correspondientes a la Georruta

Como solo tenemos imágenes de las etapas para los mapas geológicos, el mapa de la isla de Gran Canaria en su versión geológica será un poco diferente. Para ello lo que se lleva a cabo es poner otro de los mapas creados anteriormente, el mapa ortofoto de la isla entera, como una especie de capa base (con la opacidad a niveles bajos para dar importancia a los demás mapas) para luego ubicar como otra capa los distintos mapas geológicos de las etapas. Este diseño se realiza con el software TileMill(que se explica en detalle a continuación), Figura I.70.

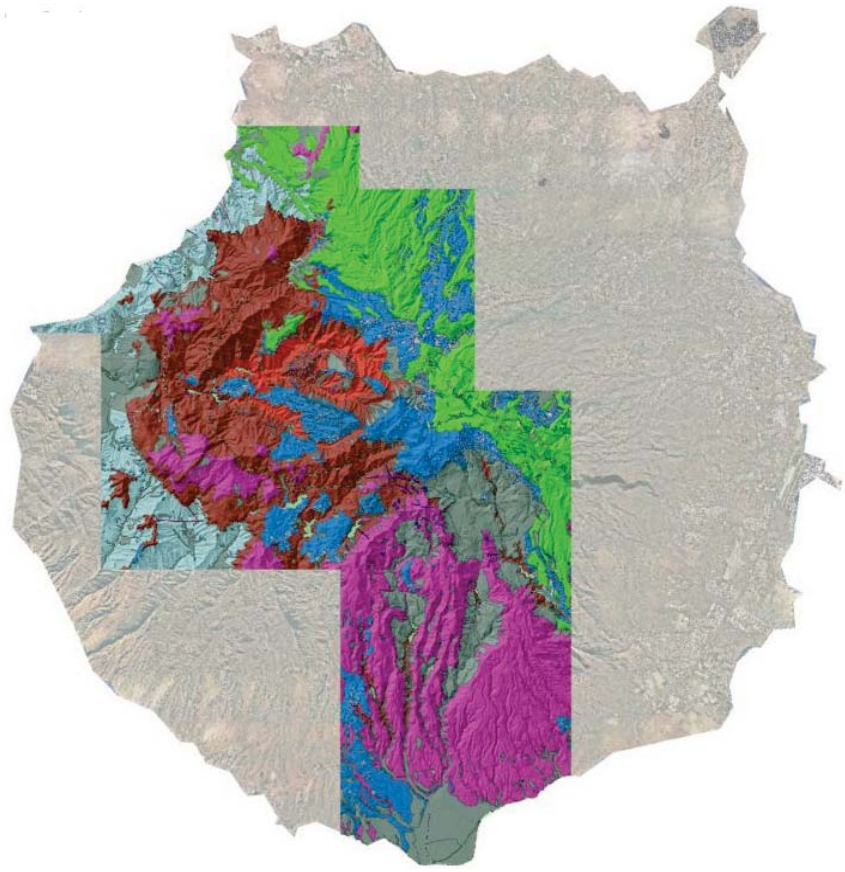


Figura I.70: Representación del mapa geológico correspondiente a la isla de Gran Canaria

5.5 . Caso tintas hipsométricas

Los mapas de tintas hipsométricas son aquellos mapas a los cuales se les asignan colores en función del relieve. Normalmente los colores claros representan terrenos con poca elevación mientras que los colores oscuros hacen referencia a terrenos con mucha elevación. Estos colores se asignan para determinados rangos de altura del terreno, un ejemplo de ello sería el de asignar un color verde claro al terreno que esté en el rango de los 0 y 100 metros y un color verde oscuro a terrenos que se encuentren entre los 100 y 300 metros.

Estos mapas, al igual que los mapas geológicos, nos han sido entregados por el departamento de Geomática de la ULPGC y más explícitamente por Bárbara del Castillo-Olivares y Suárez. La información recibida estaba estructurada de forma idéntica a la del caso de los mapas geológicos, dos archivos por cada *etapa de la Georruta* correspondiéndose uno de ellos al apartado de la georreferenciación.

El proceso hasta la creación de los mapas ha sido idénticamente el mismo que el que se ha hecho para los mapas geológicos, Figura I.71 y Figura I.72.

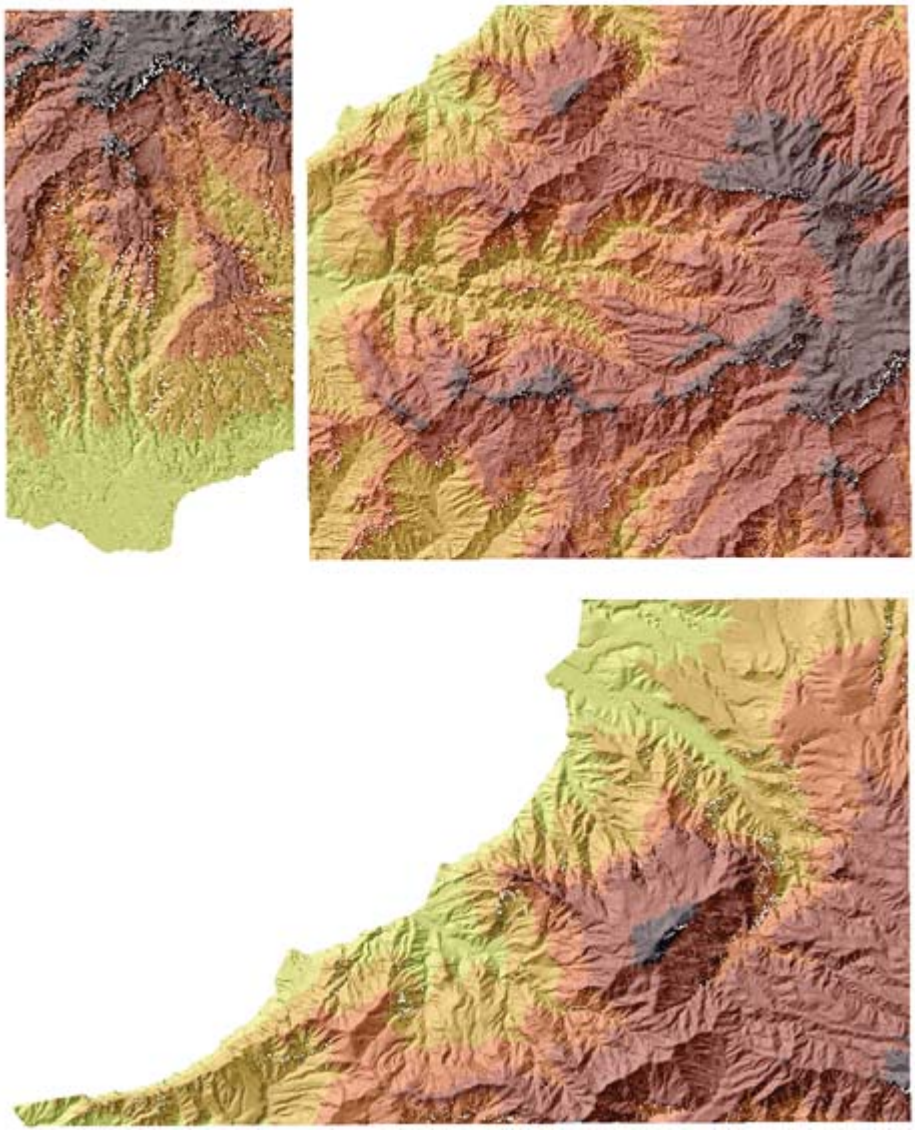


Figura I.71: Representación de los mapas de tintas hipsométricas correspondientes a la Georruta

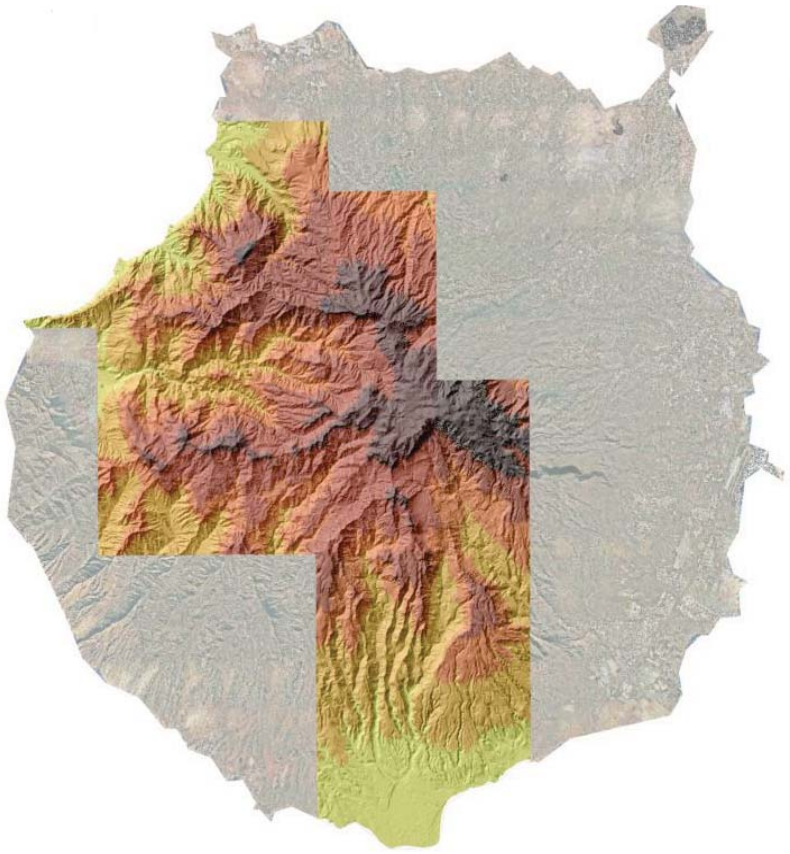


Figura I.72: Representación del mapa de tintas hipsométricas correspondiente a la isla de Gran Canaria

5.6 . TileMill

Una vez que tenemos todos los mapas en el formato y sistema geodésico de referencia elegido, el siguiente paso a llevar a cabo es subir todos estos mapas a Mapbox por medio de algún software que nos genere un archivo compatible para alojar en este servidor antes mencionado. Ese software es TileMill[75], del que ya se ha hablado anteriormente. A continuación se explica la forma de proceder con esta herramienta.

Lo primero a realizar es crear un proyecto en TileMill. Para ello, una vez abierto el software, hacemos clic en el botón "New Project" el cual nos lleva a una ventana en la cual introducimos información acerca del nuevo proyecto, Figura I.73.

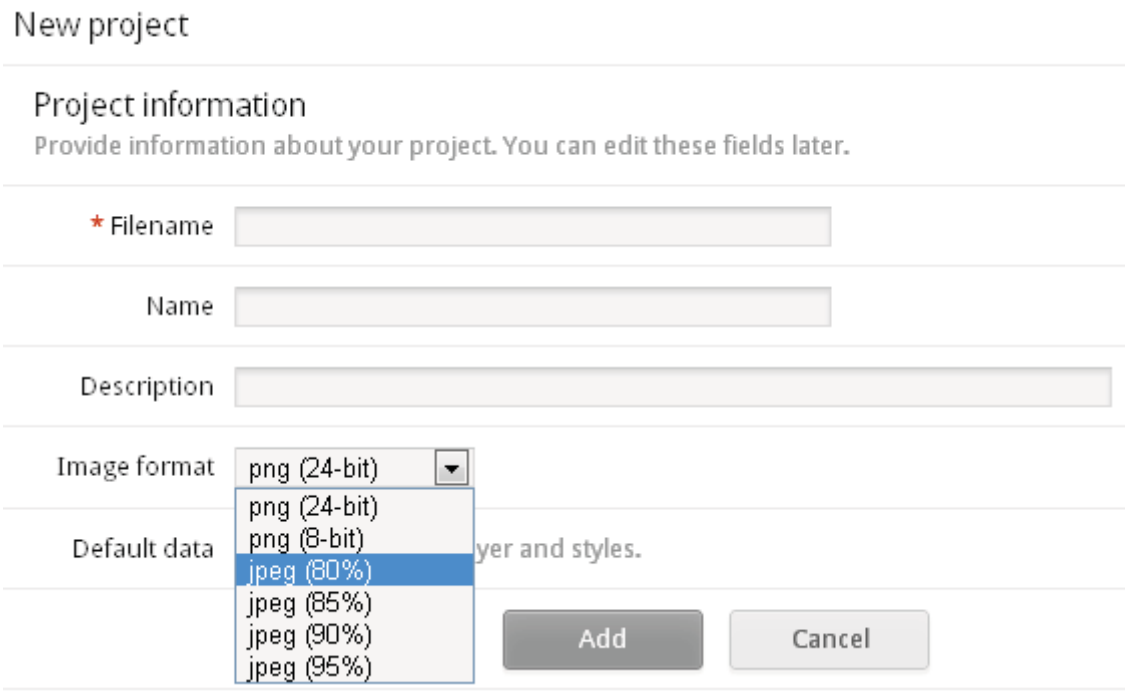


Figura I.73: Ventana para la creación de un proyecto en TileMill

Un parámetro de los que podemos modificar en esta ventana merece ser resaltado, se trata del formato de la imagen. En nuestro caso, la problemática relacionada con la cantidad de información que podemos alojar en servidores en la nube es un tema delicado debido a varios motivos. Uno de ellos es el que implica directamente al alojamiento de mapas en Mapbox, actualmente esta herramienta permite alojar en sus servidores una cantidad muy limitada de información de manera gratuita (alrededor de 100 Mb por cuenta), por ello tenemos que ser muy cuidadosos a la hora de elegir los datos que subiremos a Mapbox. Otros de los motivos que afecta directamente a la aplicación es la obligación a ser precavidos y muy minuciosos con los datos que contendrá la aplicación con el fin de que no ocupe mucho en los dispositivos móviles.

Teniendo en cuenta todo lo comentando anteriormente, de entre todas las opciones que nos pone a nuestra disposición el software nos decantamos por la opción

que preserva en un 80% la calidad de la imagen ya que luego de realizar diversos test con las otras opciones hemos llegado a la conclusión que dicha calidad es la que mejor se ajusta a nuestros objetivos debido principalmente a que ofrece la mejor relación entre calidad y peso de la imagen.

La vista que se tiene la primera vez que accedemos a nuestro proyecto es algo parecido a la Figura I.58, decimos parecido ya que la capa que aparece de base la podemos cambiar.

Esta imagen la podemos dividir en tres columnas. De izquierda a derecha se aprecia como en la primera columna tenemos un conjunto de opciones, la siguiente columna está destinada al mapa en sí mismo mientras que la última columna viene a ser como una hoja de estilos.

El siguiente paso a dar es añadir el mapa a nuestro proyecto, esto se realiza añadiéndolo como una capa a la ya existente. Para ello tenemos nos vamos a la primera columna y hacecemos click en el último botón, el cual nos abre una pestaña a través de la cual podemos añadir el mapa, Figura I.74.

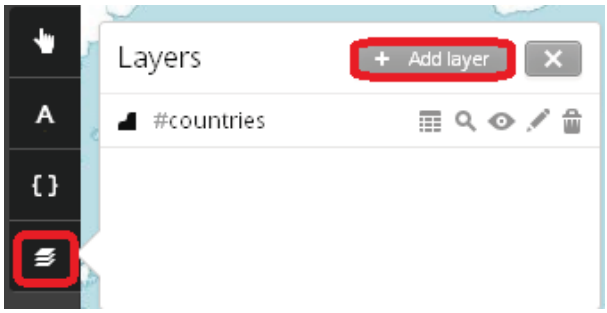


Figura I.74: Opción de TileMill para añadir nuevas capas

Cabe decir que TileMill, mediante pruebas hechas, reconoce automáticamente solo dos tipos de sistemas de referencia espaciales a pesar de que la propia herramienta nos asegura poder "autodetectar" la mayoría de ellos. Dichos sistemas de referencia son el *EPSG:4326* o *WGS84* y el *EPSG:900913*[76], Figura I.75. Sobre este último sistema de referencia cabe mencionar que lo podemos encontrar bajo diversos nombres tales

como *EPSG:3857*, *900913* que viene a corresponder con la palabra *GOOGLE* o el *EPSG:54004*.

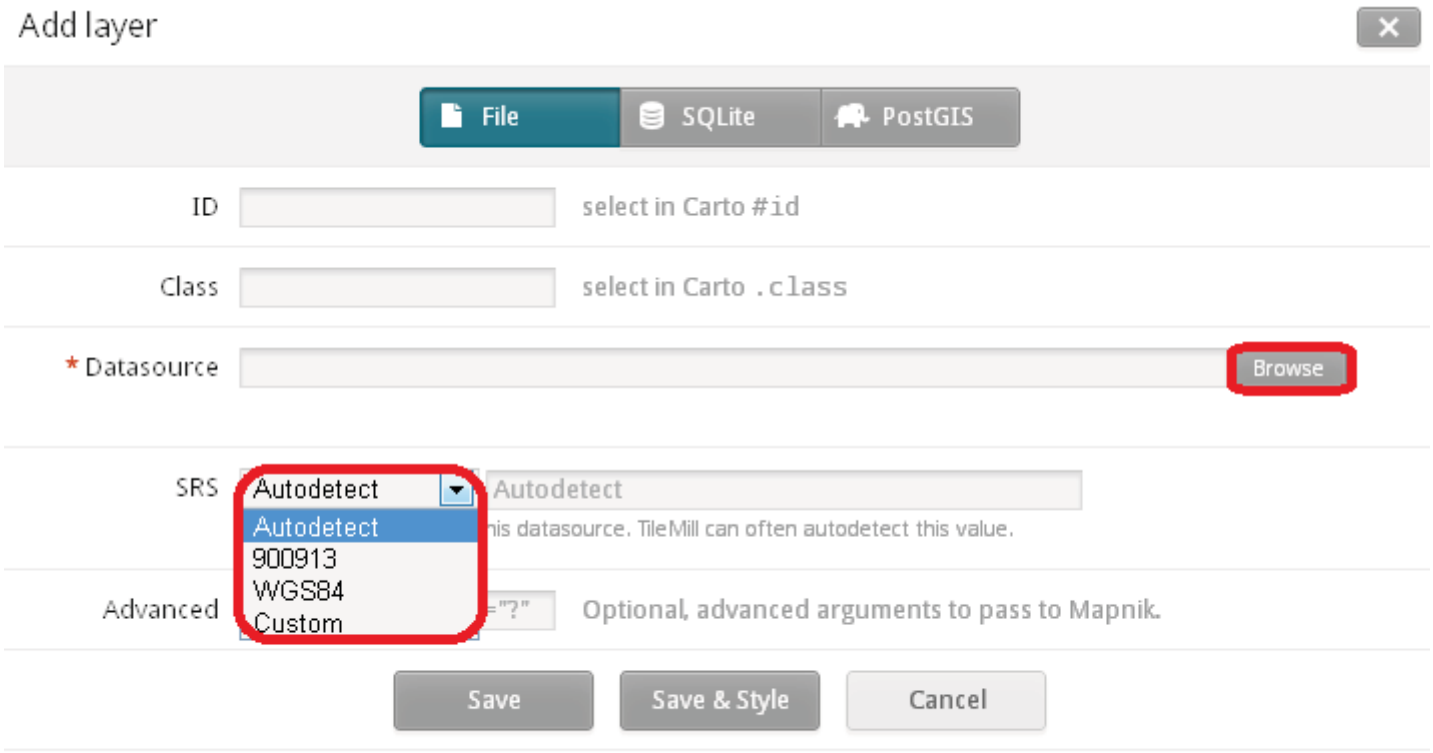


Figura I.75: Vista de los distintos sistemas de referencias aceptados por TileMill

Para terminar, elegimos la opción de guardado “Save & Style” con la que el software nos añade automáticamente a la hoja de estilo nuestra capa. Conviene mencionar que al hacer este paso de esta manera TileMill nos coloca la opción *raster-opacity* a su máximo valor por defecto, sin esta opción activada nuestra capa no sería visible.

Las capas en Tilemill pueden ser modificadas mediante este tipo de hoja de estilo bautizada como CartoCSS. Otro parámetro que añadimos a nuestros mapas por norma general es el de *raster-scaling*. Mediante esta opción se puede cambiar la resolución, y por consiguiente el tamaño de la capa. Para nuestro proyecto este parámetro lo pondremos siempre en *bilinear*, el cual nos ofrece un buen balance entre tamaño y resolución.

Algunos de los mapas creados con esta herramienta solo contienen una capa mientras que otros contienen varias, seguidamente se explica el por qué.

Para el caso del mapa con las ortofotos al tener todos los datos para su creación solo se tuvo que añadir como capa en TileMill el mapa realizado en QGIS, Figura I.76. Además se ha añadido como capa adicional en este caso el ruta Transgrancanaria.

En cuanto al mapa de líneas isométricas la mecánica fue exactamente la misma, esto es, obtener el mapa de la isla de Gran Canaria a partir de los datos recabados del IGN mediante el software QGIS, Figura I.77. Se ha añadido con fines demostrativos una capa adicional que representa los miradores que estan situados cerca de la Georruta Transgrancanaria.

Para los casos de los mapas geológicos y de tintas hipsométricas al no tener los mapas de toda la isla de Gran Canaria se tuvo que realizar unos ajustes. Concretamente se llevó a cabo la idea de sobreponer los mapas que teníamos de ambos casos sobre el mapa de ortofotos por cuestiones visuales, de esta manera el resultado final de ambos mapas es más ilustrativo. Con solo modificar el parámetro de opacidad de la capa de ortofotos a una valor inferior al que tenía en otros mapas se conseguía resaltar los mapas geológicos y de tintas hipsométricas, Figura I.78 y Figura I.79.

Figura I.76: Mapa de la isla con los datos de fotografía aérea.

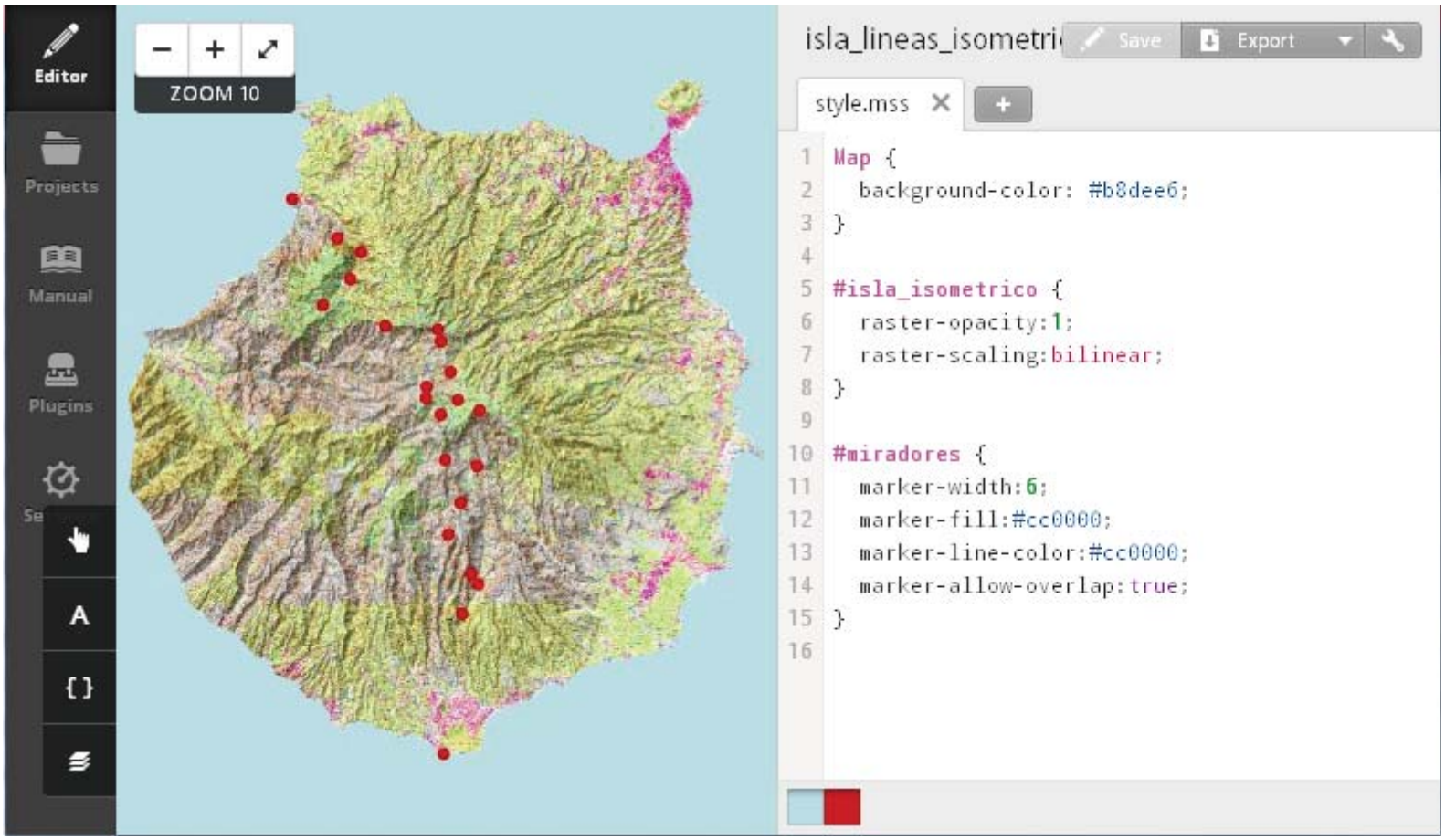


Figura I.77: Mapa de la isla de líneas isométricas con capa de miradores localizados en la Georruta.

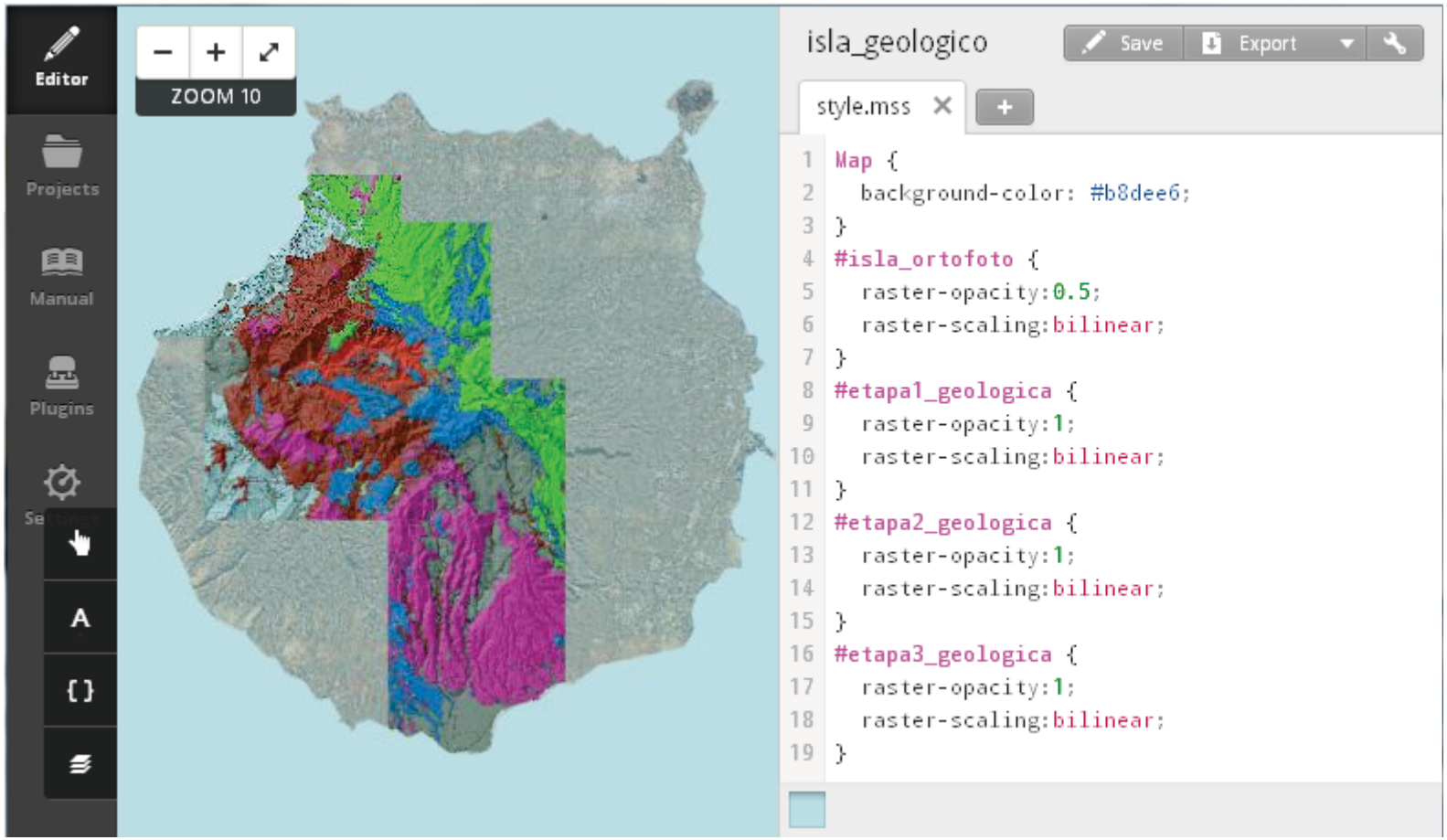


Figura I.78: Mapa de la isla en la versión geológica.

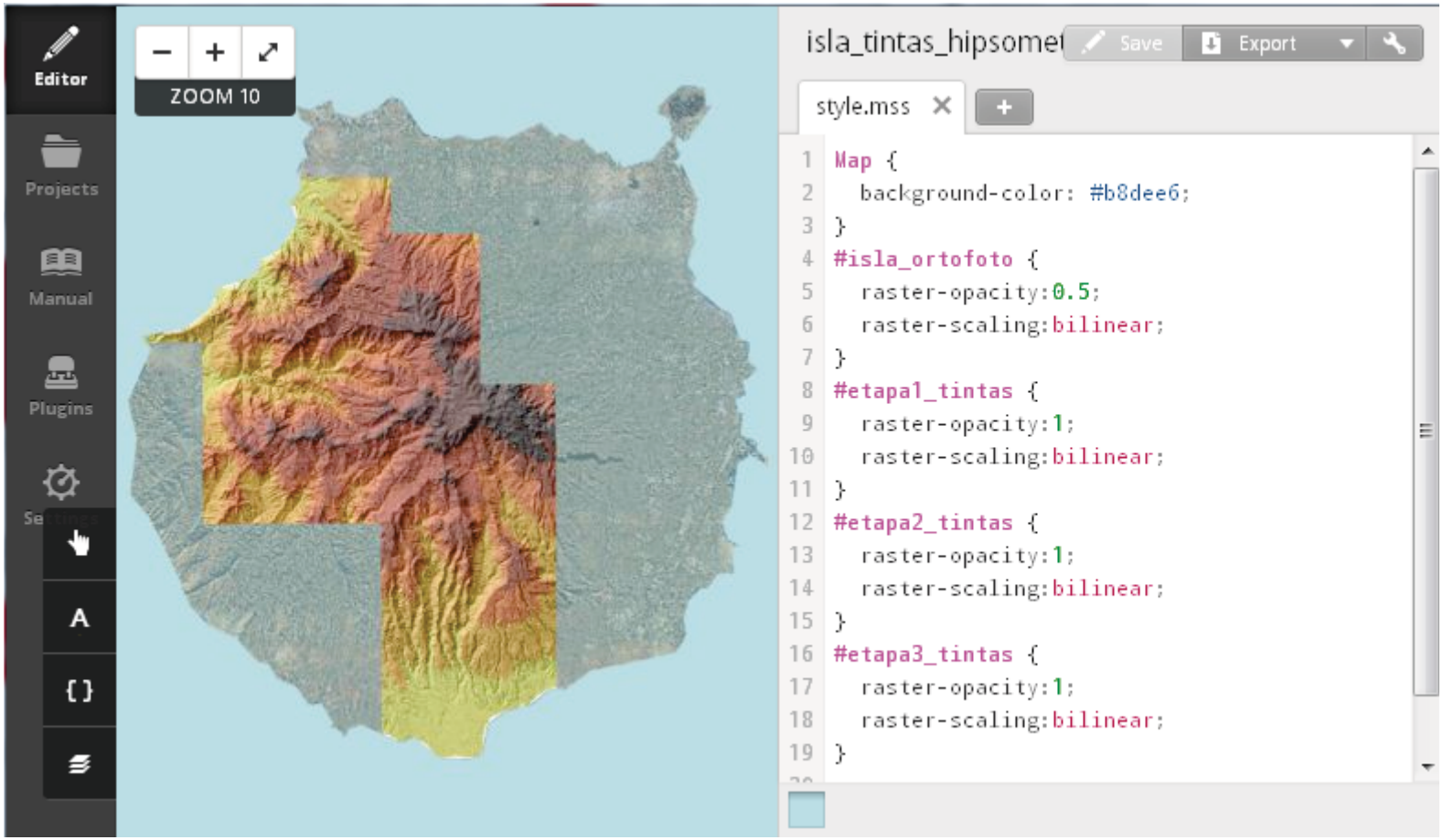


Figura I.79: Mapa de la isla en la versión de tintas hipsométricas.

El último paso a realizar con la herramienta TileMill es el de generar el fichero con extensión *MBTile*(MapBox tiles)[77]. Básicamente este archivo es una base de datos SQLite, esto nos da todo el poder y funcionalidad de esta archiconocida base datos. Algunas de las ventajas que ofrece es la de eliminar la duplicidad de tiles, esto se traduce en que para el caso de tener multitud de tiles iguales, áreas como océanos, solo se tiene que emparejar toda esa cantidad de tiles iguales a solo uno de ellos.

Para obtener el archivo con extensión *MBTile* tenemos que realizar lo siguiente. Primeramente hacemos click en la esquina superior derecha donde dice "Export", de todas las opciones que se despliegan a continuación elegimos la de *MBTile*. Esto nos lleva a una nueva ventana en la cual podemos ajustar diversos parámetros del mapa como los niveles de zoom disponibles, la extensión que tendrá o la posibilidad de asignar un punto que servirá como centro del mapa a la hora de visualizarlo, Figura I.80. Finalmente, una vez tenemos todas las opciones escogidas, hacemos click en el botón "Export" para que nos genere el fichero de extensión MBTile que necesitamos en MapBox.

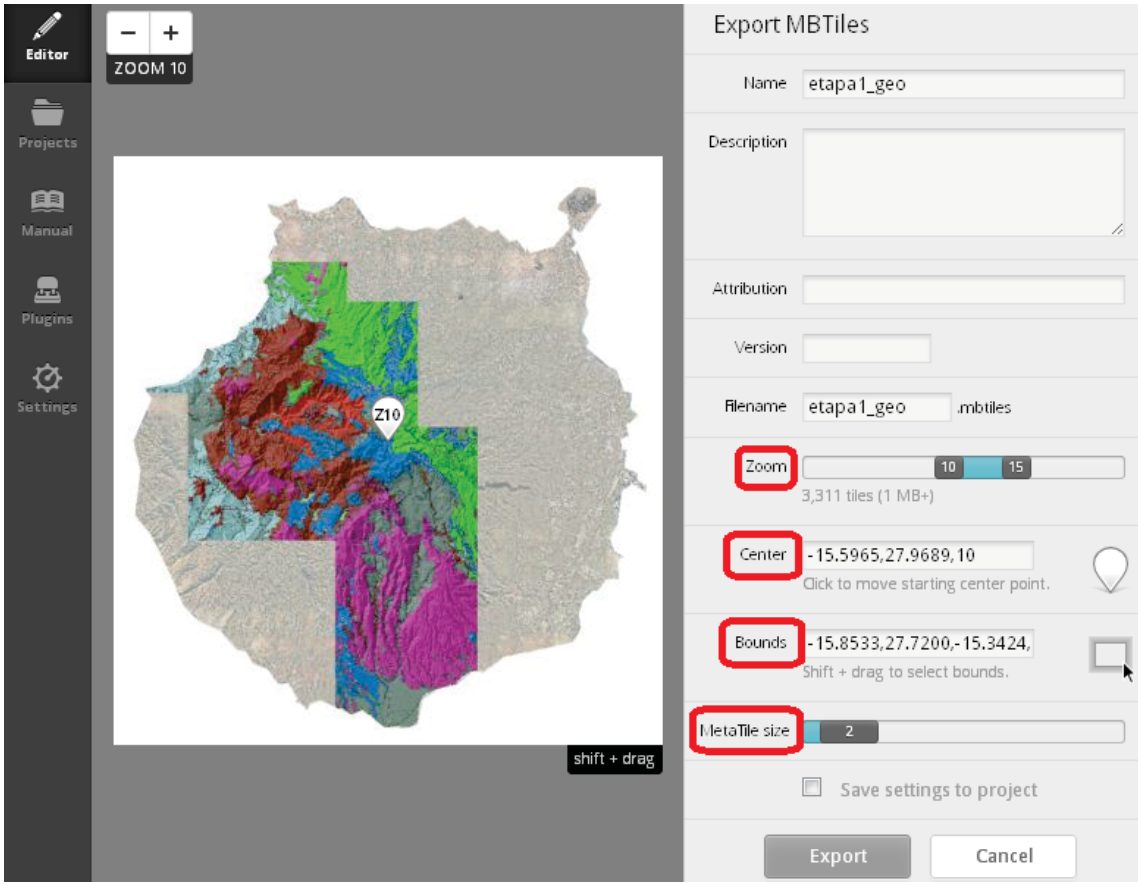


Figura I.80: Vista de la ventana Export MBTiles de TileMill.

Mención aparte merece otra opción que nos da TileMill, esta es la de asignar un tamaño al MetaTile. Básicamente, un MetaTile, Figura I.81, es el conjunto de varios tiles o mosaicos más un búfer. El valor usado por este software por defecto es 2, de esta manera un MetaTile quedaría conformado por la unión de 4 tiles de 256 píxeles cada uno más un búfer de 256 píxeles que englobarían a los 4 tiles. El búfer nunca se visualiza pero tiene como misión el permitir visualizar correctamente iconos y etiquetas, sin él estos atributos se verían algunas veces por duplicado en tiles adyacentes. El valor del búfer se puede cambiar mediante la hoja de estilos CartoCSS mientras que el tamaño en sí del MetaTile se puede ajustar en la ventana en la cual ajustamos los valores finales del mapa. Un tamaño de 1 para el MetaTile deshabilitaría el efecto que produce el MetaTile pero permitiría renderizar mucho más rápido los tiles mientras que un valor mayor a 2 para el MetaTile mejora el rendimiento en general del mapa y reduce el tiempo de generación del mapa debido a que mientras más grande sea el MetaTile menos objetos se tendrán que procesar.

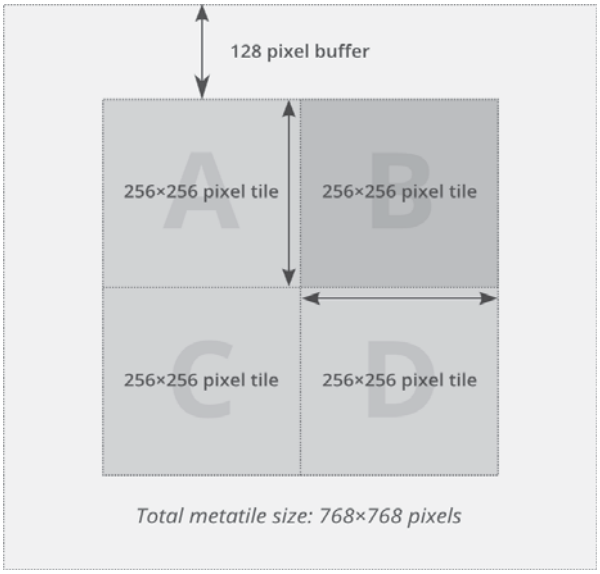


Figura I.81: Representación de un MetaTile

Con el fichero de extensión *MBTile* creado solo nos queda exportarlo hacia MapBox para tener los mapas totalmente listos para ser incluidos en la aplicación.

5.7 . Proceso de alojamamiento de los mapas en Mapbox

Para ello lo primero que tendremos que hacer es, luego de crearnos una cuenta en esta herramienta online, es alojar en dicha cuenta el archivo *MBTile* del mapa deseado. Desde la ventana principal hacemos click en la opción “Datasets” del menú de la columna izquierda, Figura I.82.

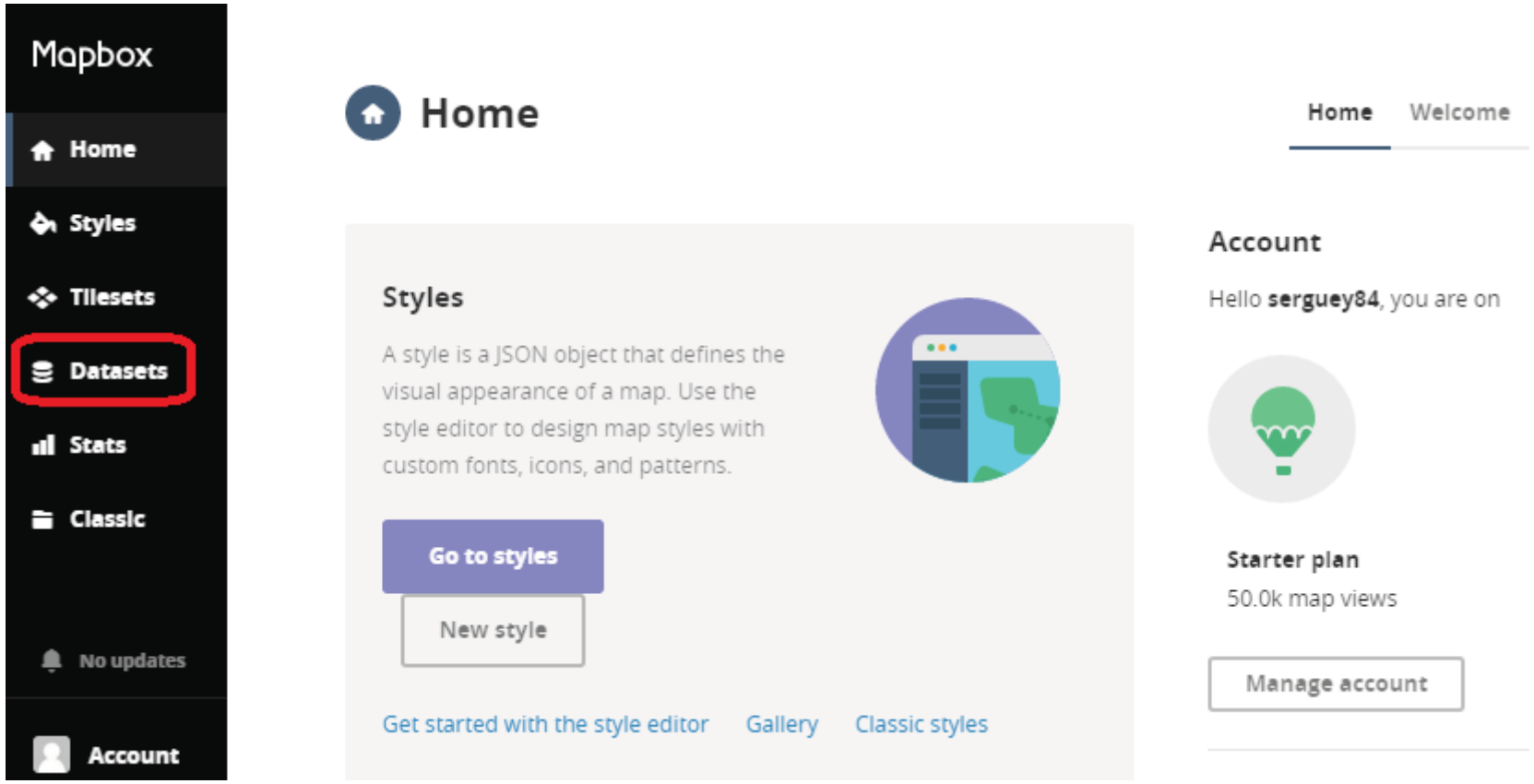


Figura I.82: Vista de la herramienta Mapbox en Internet

Esto nos lleva a una nueva ventana desde la cual elegimos el archivo que queramos alojar, Figura I.83.

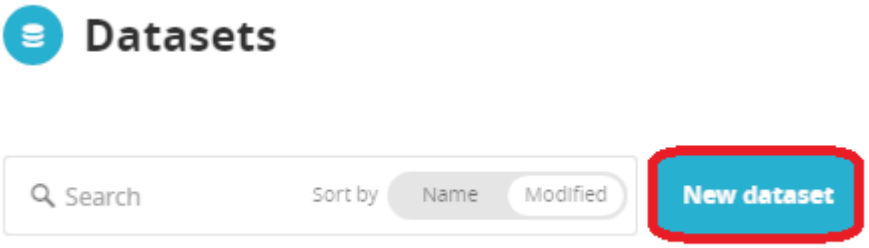


Figura I.83: Ventana de Mapbox desde la que se puede añadir información

Una vez que tenemos el archivo alojado en Mapbox, el siguiente paso es el de crear el mapa. Comenzaremos por seleccionar la opción llamada "Classic" en el menú de la izquierda. Esto lo debemos de realizar de esta manera ya que compañía ha hecho cambios bastantes significados en Mapbox y como todo el trabajo que hemos llevado a cabo se ha hecho con la versión anterior de Mapbox.

Dentro de *Mapbox Classic* tenemos a nuestra disposición todos los proyectos que hayamos realizado con la versión antigua de Mapbox. Para crearnos un nuevo proyecto tenemos que pinchar en la opción *New Mapbox Editor Project, Figura I.84*.

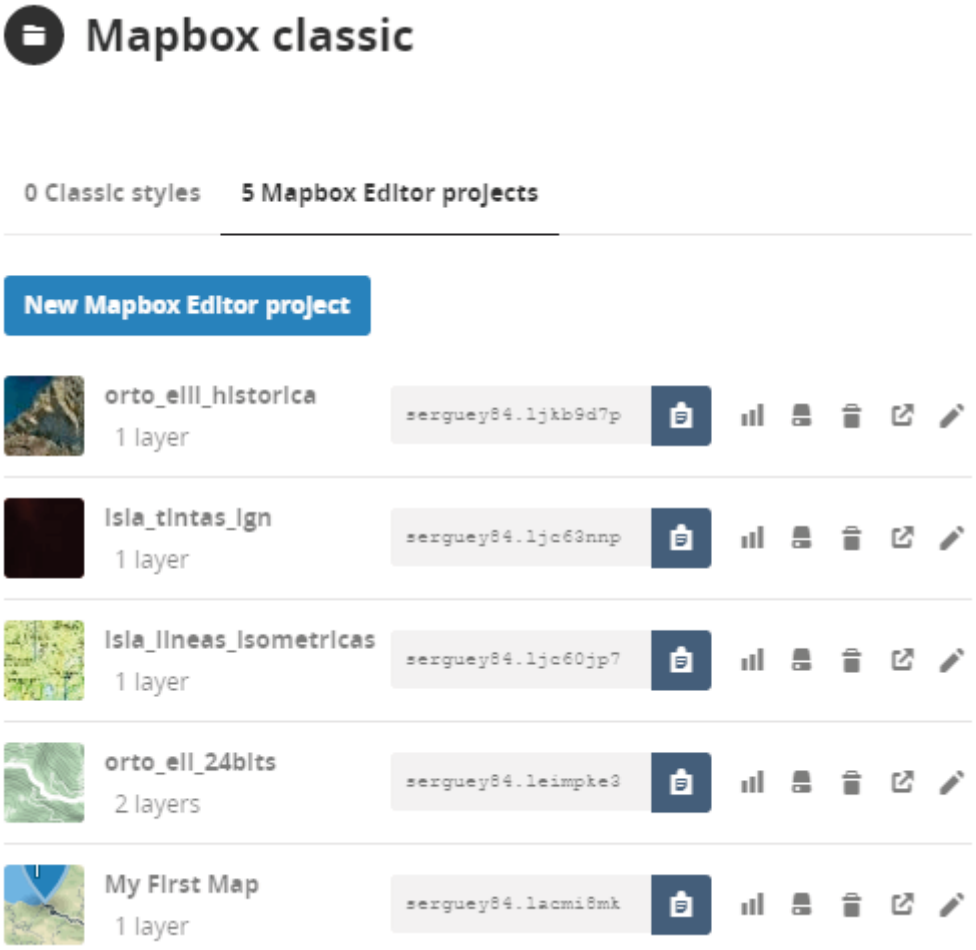


Figura I.84: Ventana desde donde se puede comenzar a crear un mapa con la información que se ha añadido

A continuación podemos elegir de entre una gran variedad de estilos, los mismos que aparecen en la Figura I.56, para aplicarlos a nuestro mapa si así fuera nuestro deseo, como hemos decidido solo mostrar nuestros mapas no elegiremos ninguno de los ofertados por la herramienta y nos decantamos por la opción *Transparent*.

Luego de este paso, para añadir el fichero *MBTile*, tenemos que seleccionar la opción *Data* de la Figura I.85. En la nueva ventana seleccionamos la opción *Menu* y luego *Add/remove* layers para incorporar el mapa que deseemos.

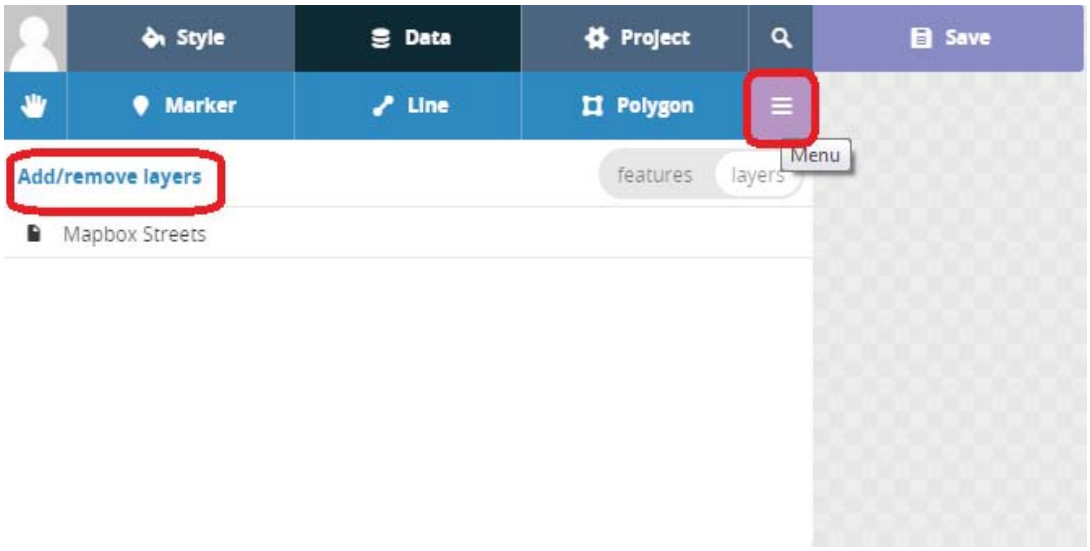


Figura I.85: Ventana desde la cual es posible añadir las capas que tendrán nuestro mapa

Antes de guardar nuestro proyecto podemos irnos a la pestaña *Project*, ubicada a la derecha de *Data*, y modificar algunos parámetros del proyecto como su nombre y añadir alguna información de interés acerca del mapa. De esta manera ya tendremos listo el mapa para poder incorporarlo a nuestra aplicación. Para incorporarlo a la aplicación solo necesitamos tener el *mapID* y el *token* de cada uno de ellos.

El *mapID* se corresponde con una forma de identificar los distintos proyectos o mapas que podamos tener bajo una misma cuenta de la herramienta Mapbox. Como tal, se asigna una completamente diferente a cada mapa de manera automática, Figura I.86.

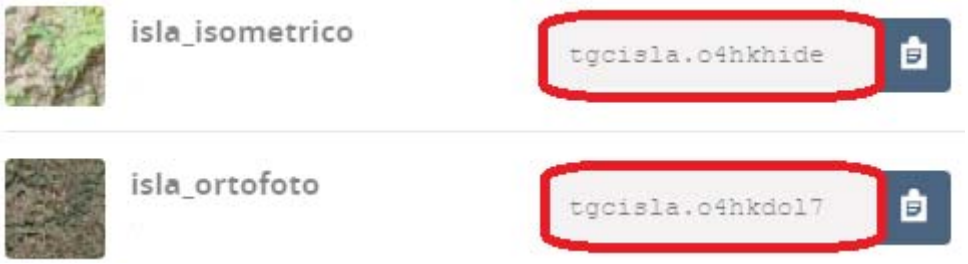


Figura I.86: Muestra de los *mapID* en Mapbox

Por otro lado el segundo dato, llamado comúnmente *token*, es el mismo para todos los proyectos que tengamos bajo una misma cuenta de Mapbox. Se genera automaticamente por la herramienta. Dicho parámetro puede crearse nuevamente con sólo hacer click en la opción que aparece en la parte superior derecha de la Figura I.87. Obviamente si realizamos cambios con el *token* tendremos que modificar el valor de dicho dato en la herramienta Parse, de lo contrario no podremos ver el mapa en cuestión en la aplicación.

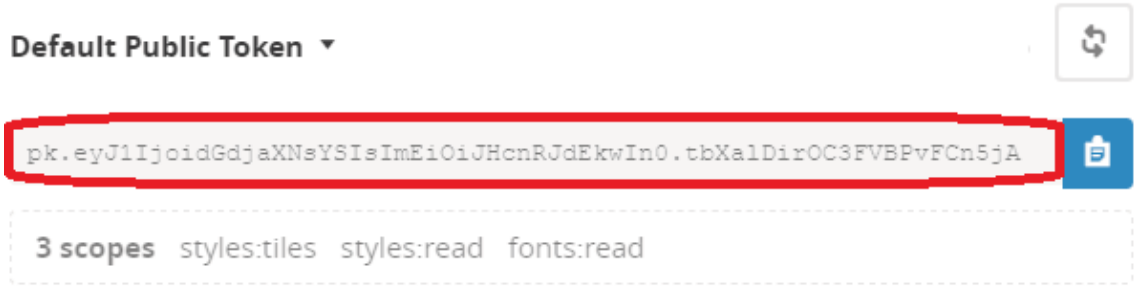


Figura I.87: Muesta del parámetro *token* en Mapbox

6 . APLICACIÓN RESULTANTE.

En este capítulo mostraremos, de una forma más gráfica, el resultado de todo el trabajo descrito anteriormente. Se plasmará como queda todo lo relacionado al apartado de las **Vistas**, en referencia al patrón MVP, además de comentar brevemente los **Presentadores** que han sido necesarios añadir para el buen funcionamiento final de la aplicación para dispositivos móviles.

Empezamos con lo primero que nos aparece, luego de la imagen introductoria, que nos muestra un mapa, una serie de botones en la parte inferior y un botón en la esquina superior derecha. Este último botón despliega un menú desde el cual tenemos acceso a distintas partes de la aplicación, Figura I.88.

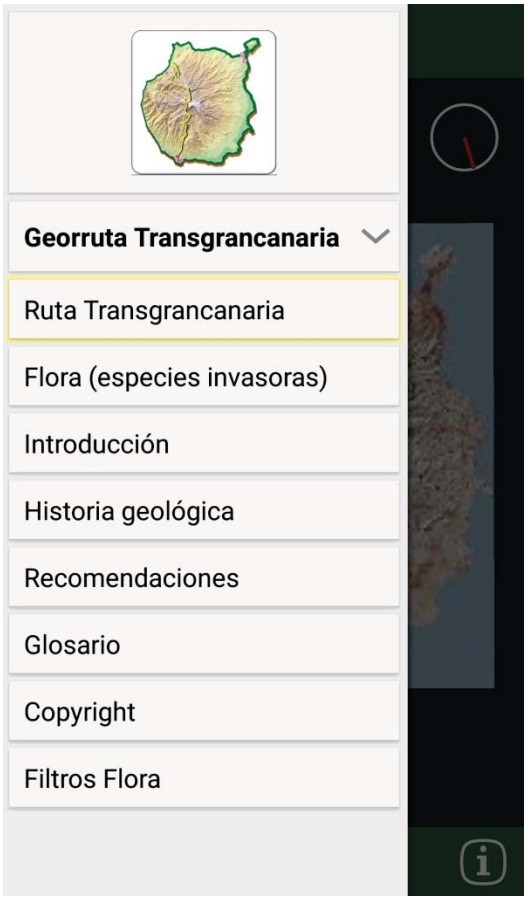


Figura I.88: Menú izquierdo de la aplicación

El mapa que se ve en la Figura I.89, al igual que los demás incluidos en la aplicación, se gestiona mediante un **Presentador** llamado ***g3MViewController*** que accede a la librería de G3M. El software permite interactuar con los mapas con los gestos típicos, amén de algún que otro exclusivo. Básicamente podemos rotar con dos dedos los mapas, también acercar o alejar el mapa (*zoom in* y *zoom out*). Para el caso de apreciar el mapa de forma tridimensional tenemos que abatir el mapa con tres dedos.

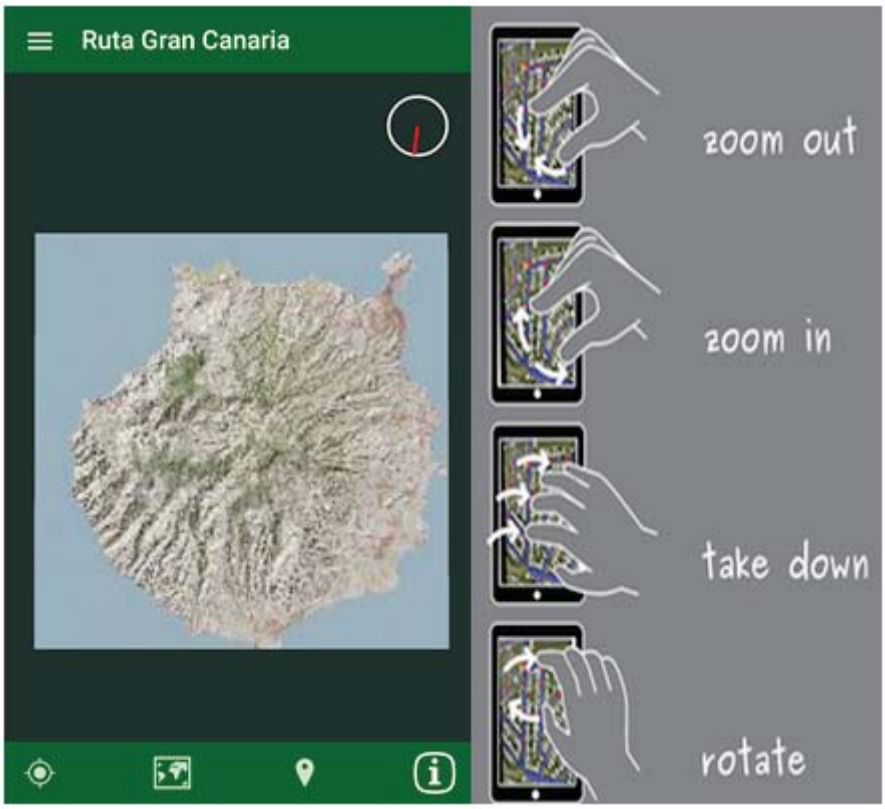


Figura I.89: Campo de actuación del presentador g3MViewController y gestos que se pueden realizar sobre los mapas, en este caso sobre el mapa que representa el Modelo de Terreno Lidar.

Además de el mapa mostrado anteriormente, la aplicación cuenta con otros cuatro mapas que han sido comentados previamente. El acceso a dichos mapas se realiza a traves del botón  . Los mapas que se pueden visualizar son el mapa de ortofoto que muestra la isla de Gran Canaria de una determinada altura, Figura I.90. También tenemos los mapas geológicos Figura I.91 y de tintas hipsométricas Figura I.92 encargados de mostrar la información acerca de historia geológica de la isla así como la del relieve respectivamente. Finalmente se puede apreciar el mapa de líneas isométricas correspondiente al Mapa Topográfico Nacional, Figura I.93.

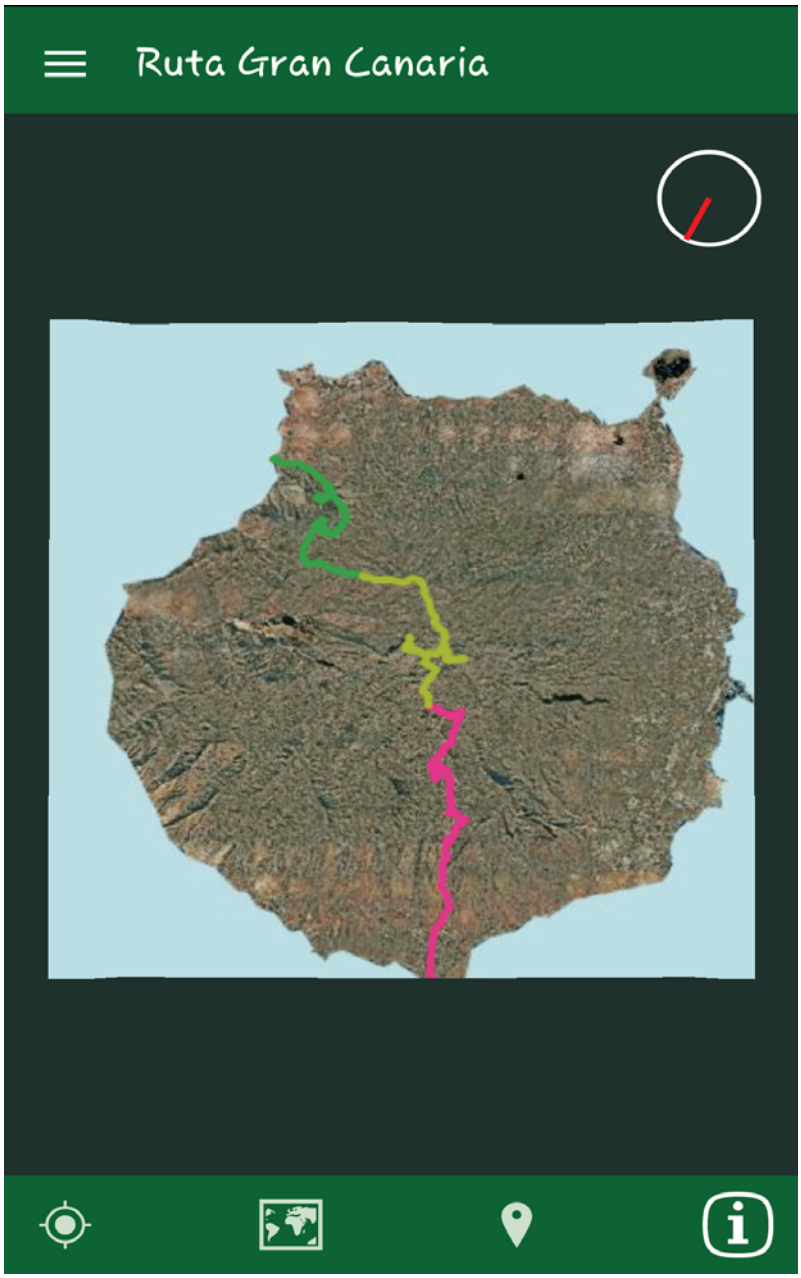


Figura I.90: Mapa de la isla de Gran Canaria en la versión de ortofoto.

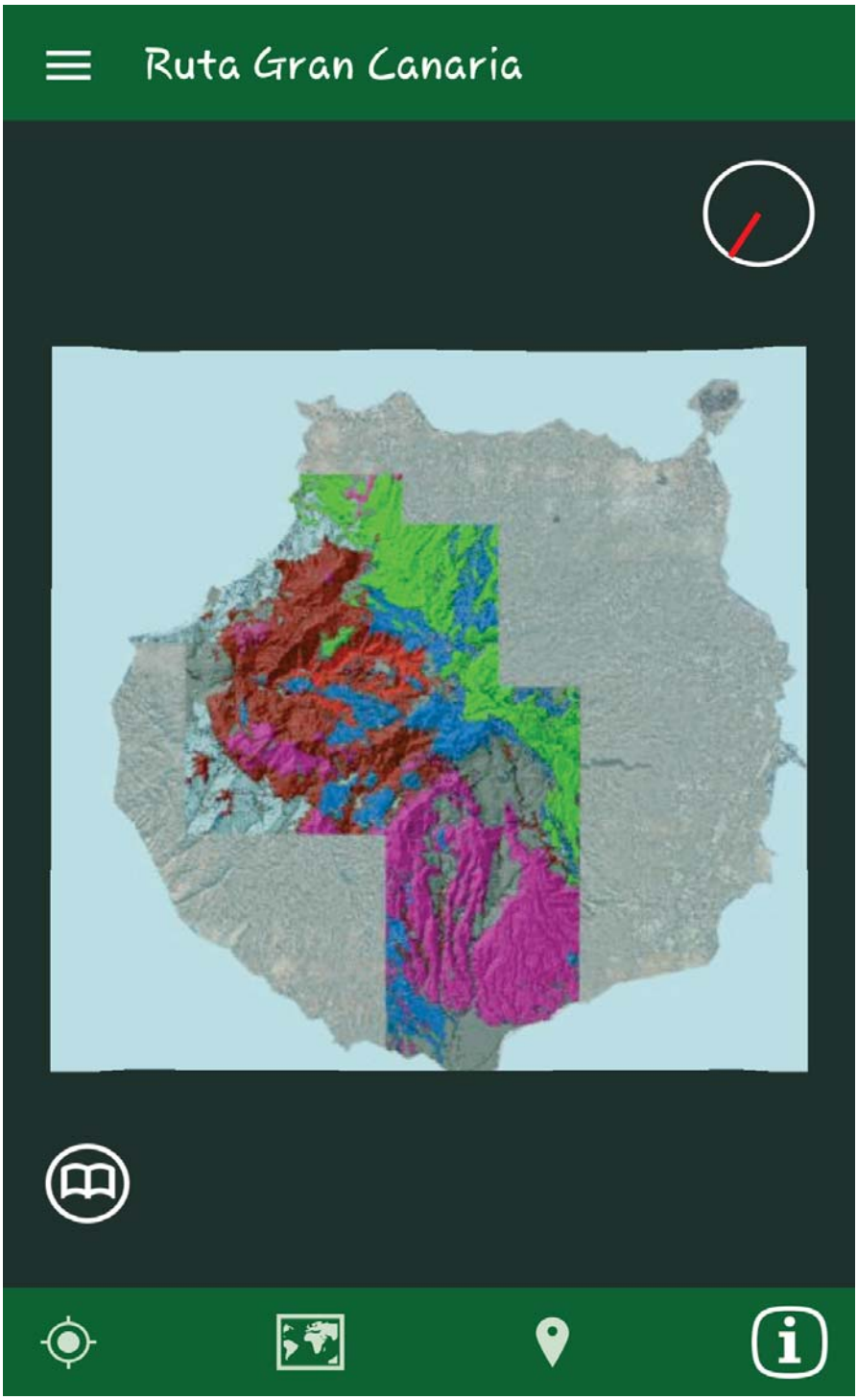


Figura I.91: Mapa de la isla de Gran Canaria en la versión geológica.

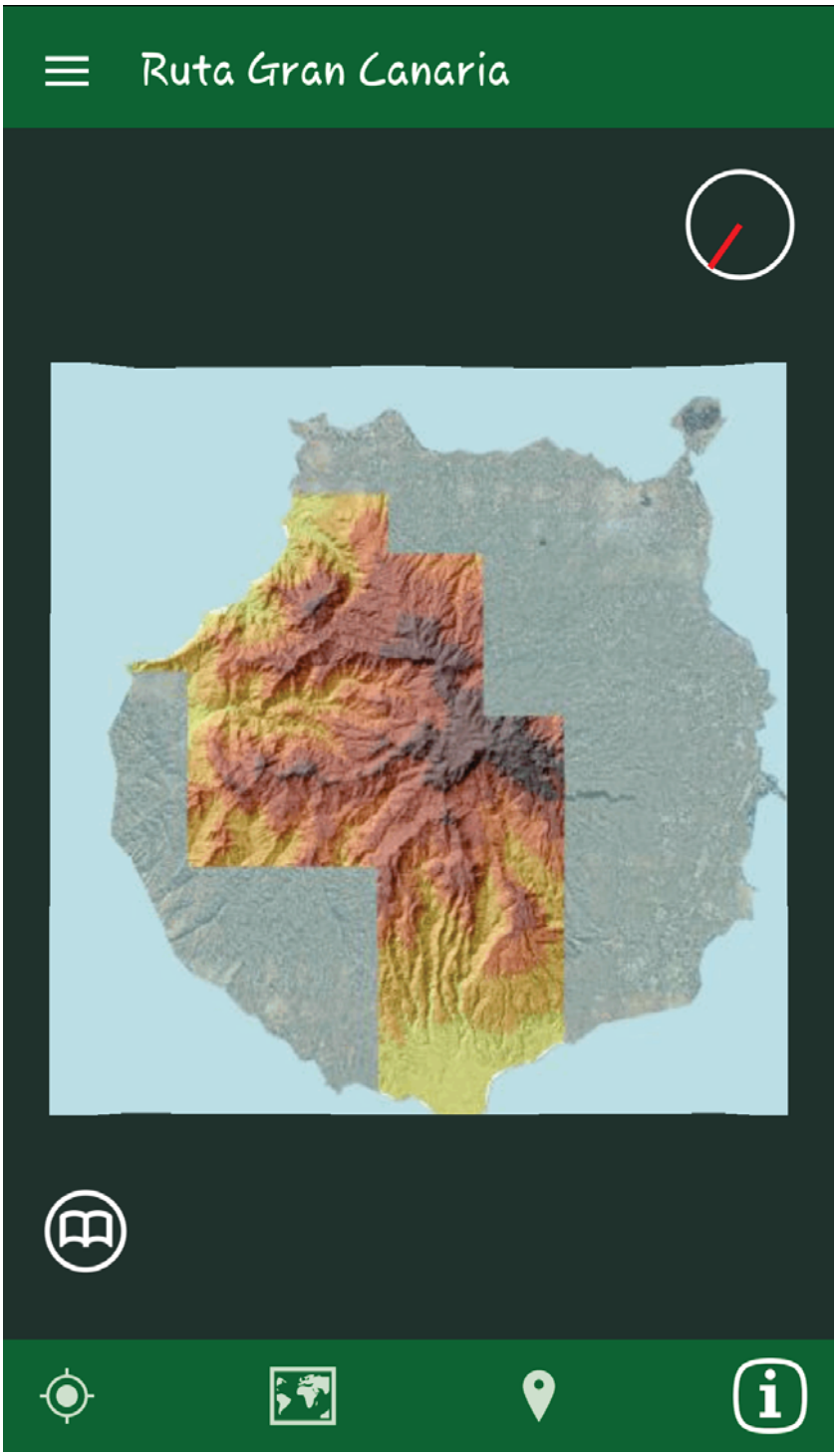


Figura I.92: Mapa de la isla de Gran Canaria en la versión de tintas hipsométricas.

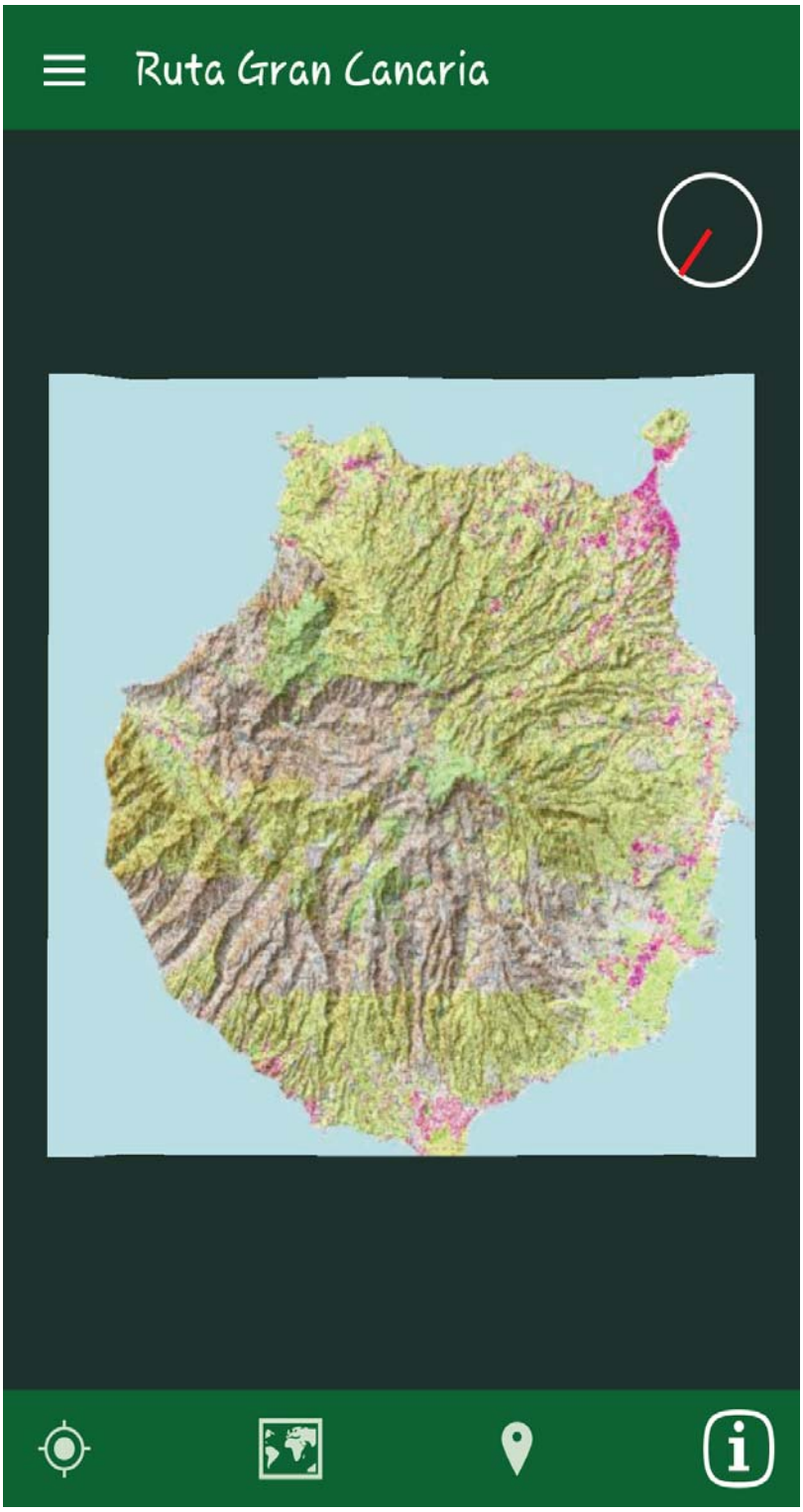


Figura I.93: Mapa de la isla de Gran Canaria en la versión de líneas isométricas.

Desde la **Vista** inicial de la aplicación podemos hacer clic en el botón que se encuentra en la esquina inferior derecha  . Con el **Presentador** ***infoStage*** accedemos a una nueva **Vista** desde la cual tenemos acceso a información sobre la Ruta Transgrancanaria, Figura I.94. Los datos que se muestran en esta **Vista** dan cuenta acerca de la distancia del recorrido así como del tiempo estimado en llevarlo a cabo. Otros datos de interés mostrados son la dificultad del trayecto además de un perfil topográfico para que el futuro usuario se haga a la idea del camino desde un punto de vista más práctico.

Otra forma de acceder a esta Vista es mediante el botón  . Nos lleva a un menú desde el cual vamos a poder incluir en nuestros mapas, entre otro tipo de información, las diferentes rutas incluidas en la aplicación.

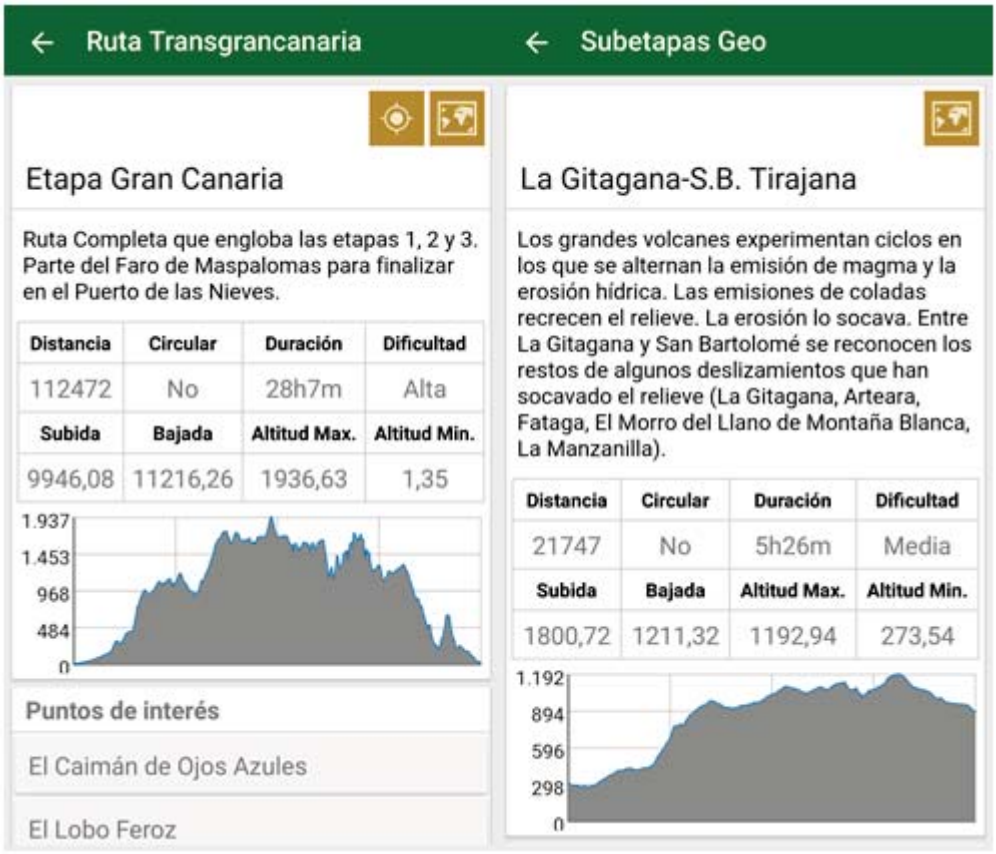


Figura I.94: Vista del presentador infoStage

Para tener acceso a los puntos de interés, miradores o relieves singulares incluidos en esta aplicación tenemos que echar mano del **Presentador *infoPOI***, Figura I.95. Este **Presentador** entra en acción de diversas maneras, de forma parecida al **Presentador *infostage***, aunque la mayoría de estos sitios de renombre se encuentran tras hacer clic en el botón  .

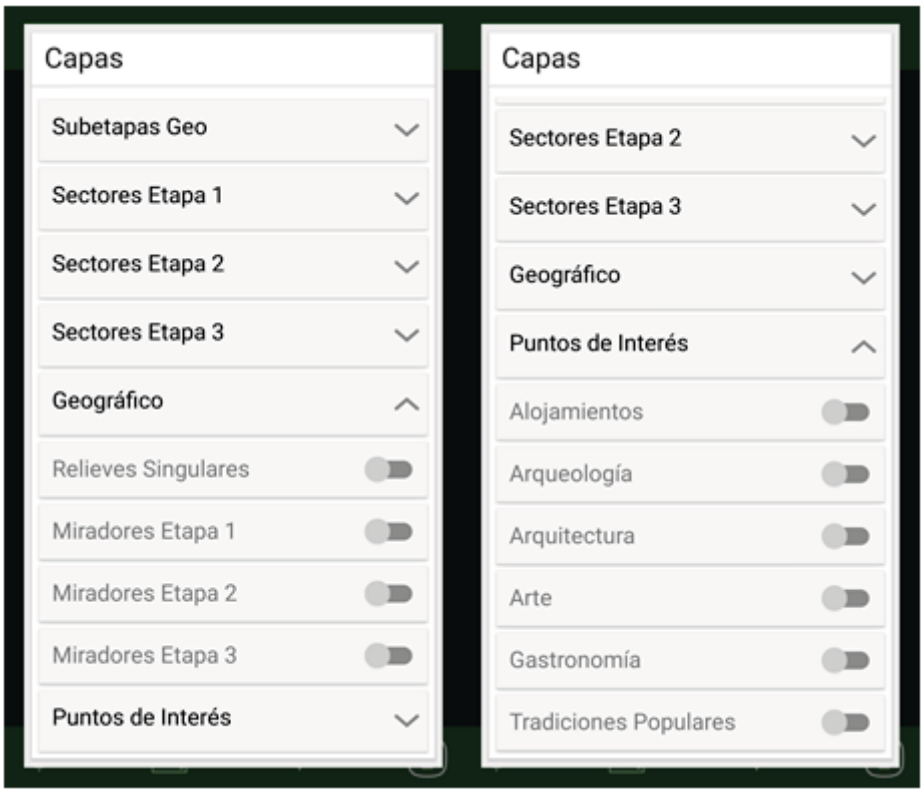


Figura I.95: Puntos de interés proporcionados por el presentador infoPOI

Estos enclaves de gran notoriedad en la isla los podemos ver de forma agrupada en función de ciertos criterios tomados tales como los miradores de una etapa en concreto o los alojamientos existentes en el trayecto de la Ruta Transgrancanaria al completo. Otra forma de visualizarlos es hacerlo por separado cuando los tengamos plasmados en cualquiera de los mapas que incluye esta aplicación, Figura I.96. La información que recaba el **Presentador *infoPOI*** del **Modelo** esta conformada en gran medida por una imagen del sitio en cuestión además de un texto que versa sobre dicho punto de interés en cuestión.

Figura I.96: Enclaves turísticos gestionados por el presentador infoPOI

La aplicación contiene un **Presentador**, llamado ***glossary***, encargado de mostrar información específica y detallada de todo lo relacionado con la Geología. Entra en acción al hacer clic en la opción Glosario, ubicada dentro del menú al que se llegue mediante el botón situado en la esquina superior izquierda de la aplicación, Figura I.97. Toda la información que se muestra en este apartado ha sido aportada por Jorge Yepes Temiño, coordinador del Departamento de Ingeniería Civil de la ULPGC.

Figura I.97: Información geológica gestionada por el presentador glossary

Con el **Prensetador** ***copyright*** mostraremos cuales han sido los softwares que han sido pilares fundamentales para la creación de la aplicación, Figura I.98. El primero de ellos la librería G3M, sin la cual no habríamos sido capaces de mostrar mapas en tres dimensiones. Luego tenemos el entorno de trabajo creado por el Instituto Universitario de Microelectrónica Aplicada de la ULPGC, clave en la organización y gestión de todos los datos necesarios para el buen funcionamiento de la aplicación.

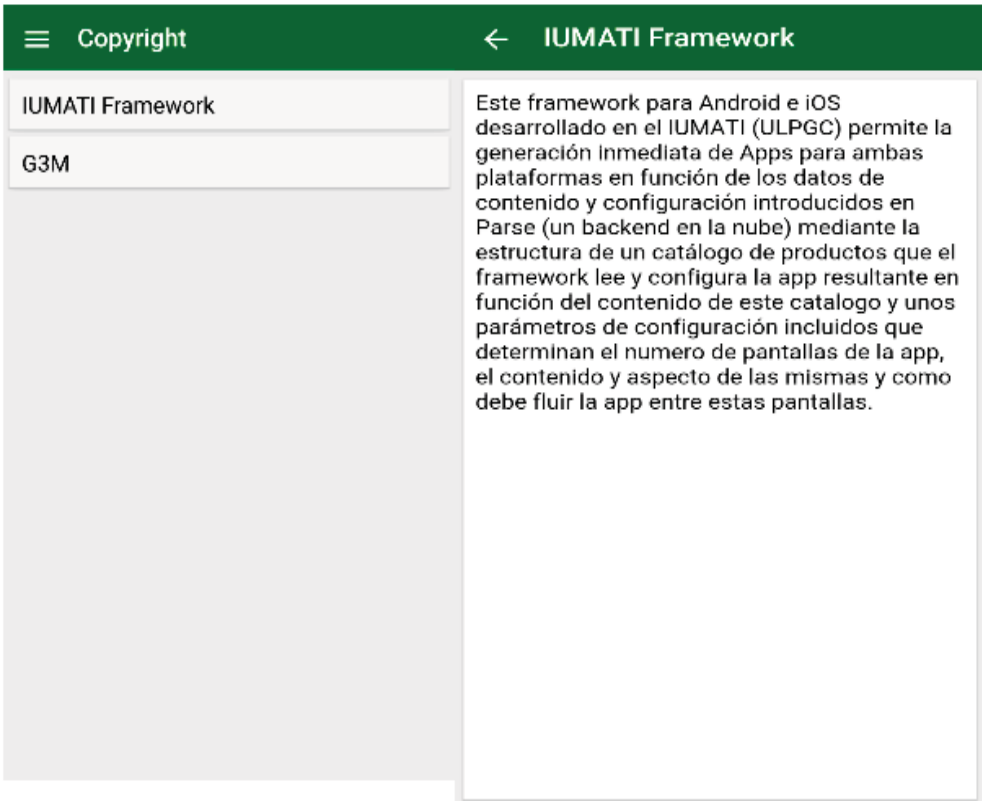


Figura I.98: Copyright

Finalmente tenemos un **Presentador**, cuyo nombre es ***credits***, encargado de mostrar todo el personal que de una forma u otra ha aportado su granito de arena a la confección de esta aplicación, Figura I.99.

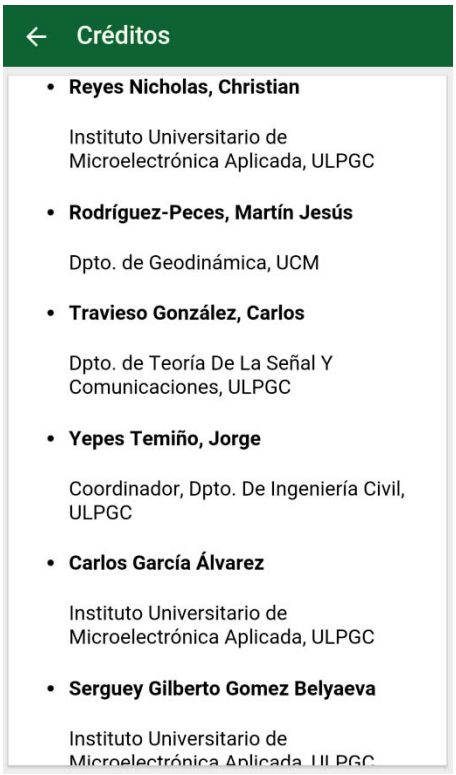


Figura I.99: Créditos

7 . CONCLUSIONES

Se ha formado parte en la creación de una aplicación para móviles inteligentes o smartphones mediante la inclusión de diversos tipos de mapas que dan la posibilidad al usuario de apreciar distintos aspectos de la isla de Gran Canaria.

Para ello se han estudiado algunas de las herramientas multi-plataformas más interesantes que tenemos en día para el desarrollo de aplicaciones.

Se ha hecho primeramente una explicación bastante amplia de Titanium Appcelerator y de PhoneGap.

Con respecto a Titanium Appcelerator se ha relatado en qué consiste, se ha revisado las librerías que ofrece al desarrollador, las ventajas de estas a la hora de crear interfaces gráficas para las distintas plataformas soportadas entre otras además de las diversas desventajas que presenta como el tener que lidiar con todos los SDK soportados todo el tiempo. Se ha mencionado en que consiste Alloy y los cambios que introduce con respecto a la forma estándar de proceder de esta herramienta.

Con PhoneGap se ha procedido de similar forma. Hemos hecho un repaso en cuanto al proceder con dicho software remarcando las similitudes presentadas con respecto al desarrollo de páginas web. También se ha dado buena cuenta acerca de las ventajas y desventajas que tiene este framework. Además se ha presentado la utilidad que facilita la compilación de las aplicaciones fruto del uso de este software llamada PhoneGap Build.

En base a los importantes inconvenientes que presentan ambas herramientas multi-plataforma se decidió optar por el Framework del IUMA.

Se ha repasado con bastante exactitud en qué consiste el novedoso entorno de trabajo creado por el departamento del IUMA, perteneciente a la ULPGC. Basado en el paradigma MVP (Modelo-Vista-Presentador) se ha ido desgranando paso a paso el contenido de estas tres partes.

Referente a los mapas, se ha hecho una amplia búsqueda de los mismos. Básicamente se ha partido desde cero en este apartado, recopilando información desde sitios contrastados y tomando contacto con todo lo referente al uso de este material geográfico en el ámbito informático. En general, se ha mostrado como se ha ido avanzando desde una idea algo rudimentaria hasta una opción bastante sólida.

Se ha hecho uso de la herramienta Mapbox para el almacenamiento de todo el material geográfico de la aplicación. Para ello se ha tenido que empapar de todo lo que concierne al tratamiento de mapas en formato ráster.

En el último capítulo se hace un repaso de cómo ha ido encajando todo lo referente al aporte por parte de este proyecto a la aplicación Georruta Transgrancanaria.

Como se puede observar se ha llevado un amplio trabajo de búsqueda de información acerca de tecnologías que están generando un gran impacto en cuanto a la realización de aplicaciones para móviles inteligentes. La valoración del proyectando sobre lo realizado es muy positiva ya que le ha permitido tomar contacto con herramientas de desarrollo de aplicaciones que se encuentran en gran efervescencia. Además ha tenido la oportunidad de tomar contacto con los Sistemas de Información Geográficos (SIG o GIS en inglés), muy necesarios para la inclusión de mapas en aplicaciones móviles.

Respecto a las mejoras futuras con respecto a lo visto en este proyecto se pueden hablar de varias.

No solo existen PhoneGap y Titanium Appcelerator en el mercado para el desarrollo de aplicaciones multi-plataformas. Existe un buen conjunto de ellas. Una de las que sobresale podría ser Xamarin[78], una herramienta que desarrolla aplicaciones desde la base de C#. El uso de dicha herramienta podría proporcionarnos alguna mejora con respecto a las otras dos antes mencionadas debido a que, entre otras razones, la comunidad de desarrolladores en este lenguaje es bastante numerosa y por lo tanto el avance en cuanto a resolución de errores sería mayor.

Otra de las mejoras versa acerca del tipo de mapa a utilizar. La aplicación resultante cuenta en su mayoría con mapas en formato ráster. Una mejora en este

apartado sería cambiarlos por los mapas en formato vectorial, de esta manera nos ahorraríamos bastante espacio en cuanto al almacenamiento. Cabe decir que actualmente software de tratamientos de mapas están poniendo mayor énfasis en este tipo de mapas, ejemplo de ello es Mapbox[79].

Referente al Framework del IUMA se pueden realizar diversos cambios que inducirían a mejoras sustanciales. Una de estas mejoras, y que se ya se está implementando por parte del grupo del IUMA, es la sustitución de la herramienta Parse por otra que permite la misma funcionalidad basándose en la base de datos MongoDB[80].

Con respecto al diseño de la aplicación se pueden realizar diversas mejoras que afectaran en positivo a la experiencia de uso de ésta por parte del usuario. Una de ellas podría ser la de incluir un apartado mediante el cual se pueda ir añadiendo más rutas a las ya añadidas para así enriquecer la aplicación.

El resto de mejoras futuras vendrá tanto de la revisión del estado del arte de todas las herramientas involucradas en el Framework del IUMA así como aquellas que tengan algo que ver con la gestión de la información geográfica.

8 . BIBLIOGRAFÍA

1: Hypertextual, Motorola DynaTAC: el primer teléfono móvil llegó al mercado hace 30 años, 2014, https://hipertextual.com/2014/03/motorola-dynatac-30-aniversario-venta

2: Retrocom, Retro Computing Reviewer.....BellSouth IBM SIMON, 2013, http://www.retrocom.com/bellsouth_ibm_simon.htm

3: Mathew Honan, Apple unveils iPhone, 2007, http://www.macworld.com/article/1054769/smartphones/iphone.html

4: Time, Lev Grossman, Invention Of the Year: The iPhone, 2007, http://content.time.com/time/specials/2007/article/0,28804,1677329_1678542,00.html

5: PLOS ONE, David A. Ellis, Heather Shaw, Lukasz Piwek, Beyond Self-Report: Tools to Compare Estimated and Real-World Smartphone Use, 2015, http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0139004

6: Canalys, Media alert: Smart phone shipments returned to growth in Q2 2016, 2106, https://www.canalys.com/newsroom/media-alert-smart-phone-shipments-returned-growth-q2-2016

7: Appcelerator Titanium, Sitio Web de Appcelerator Titanium, 2016, http://www.appcelerator.com/

8: PhoneGap, Sitio Web de PhoneGap, 2016, http://phonegap.com/

9: JavaScript, Sitio Web de JavaScript, 2016, https://www.javascript.com/

10: TechCrunch, Mark Hendrickson, Appcelerator Raises $4.1 Million for Open Source RIA Platform, 2008, https://techcrunch.com/2008/12/09/appcelerator-raises-41-million-for-open-source-ria-platform/

11: Apple, Programming with Objective-C, 2016, https://developer.apple.com/library/mac/documentation/Cocoa/Conceptual/ProgrammingWithObjectiveC/Introduction/Introduction.html/

12: Oracle, Java, 2016, https://academy.oracle.com/en/solutions-java.html/

13: GitHub, Sitio Web de GitHub, 2016, https://github.com/

14: Node.js, Sitio Web de la librería Node.js, 2016, https://nodejs.org/en/

15: Appcelerator Titanium, API de Titanium.Media, 2016, http://docs.appcelerator.com/platform/latest/#!/api/Titanium.Media

16: Appcelerator Titanium, API Titanium.Accelerometer, 2016, http://docs.appcelerator.com/platform/latest/#!/api/Titanium.Accelerometer

17: Appcelerator Titanium, API Titanium Analytics, 2016, http://docs.appcelerator.com/platform/latest/#!/api/Titanium.Analytics

18: Appcelerator Titanium, API Titanium UI, 2016, http://docs.appcelerator.com/platform/latest/#!/api/Titanium.UI

19: Appcelerator Titanium, Framework Alloy, 2016, http://docs.appcelerator.com/platform/latest/#!/api/Alloy

20: Backbone.js, Librería Backbone.js, 2016, http://backbonejs.org/

21: Underscore.js, Librería Underscore.js, 2016, http://underscorejs.org/

22: PhoneGap, Sitio Web de PhoneGap, 2016, http://phonegap.com/

23: John M. Wargo, PhoneGap Essentials: Building cross-platform mobile apps, 2012

24: Adobe, Adobe Announces Agreement to Acquire Nitobi, Creator of PhoneGap, 2011, http://www.adobe.com/aboutadobe/pressroom/pressreleases/201110/AdobeAcquiresNitobi.html

25: PhoneGap, PhoneGap, Cordova, and what's in a name?, 2012, http://phonegap.com/blog/2012/03/19/phonegap-cordova-and-what-e2-80-99s-in-a-name/

26: John M. Wargo, PhoneGap Essentials, 2012

27: PhoneGap, Web View en PhoneGap, 2016, http://docs.phonegap.com/develop/1-embed-webview/android/

28: PhoneGap, Sitio Web de PhoneGap Build, 2016, https://build.phonegap.com/

29: Eclipse, Sitio Web de Eclipse, 2016, http://www.eclipse.org/eclipse/

30: Android, Sitio Web del plugin ADT, 2016, https://developer.android.com/studio/tools/sdk/eclipse-adt.html

31: Apple, Sitio Web de Xcode, 2016, https://itunes.apple.com/es/app/xcode/id497799835?mt=12

32: Apple, Sitio Web del Apple Developer Program, 2016, https://developer.apple.com/programs/

33: Microsoft, Sitio Web de Visual Studio, 2016, https://www.visualstudio.com/

34: PhoneGap, Documentación de PhoneGap, 2016, http://docs.phonegap.com/

35: PhoneGap, Plugins en PhoneGap, 2016, http://docs.phonegap.com/references/plugin-apis/

36: PhoneGap, Plugin Device Orientation, 2013, https://github.com/apache/cordova-plugin-device-orientation

37: PhoneGap, Plugin Device Motion, 2013, https://github.com/apache/cordova-plugin-device-motion

38: Wizcorp, Facebook Plugin, 2013, https://github.com/Wizcorp/phonegap-facebook-plugin

39: Eddy Verbruggen, Calendar Plugin, , https://www.npmjs.com/package/cordova-plugin-calendar

40: PhoneGap/Cordova, Plataformas Soportadas, 2016, https://cordova.apache.org/docs/en/latest/guide/support/index.html

41: Mike Potel, MVP: Model-View-PresenterThe Taligent Programming Model for C++ and Java , 1996, http://www.wildcrest.com/Potel/Portfolio/mvp.pdf

42: Microsoft, Página web de ASP.NET, 2016, https://www.asp.net/

43: Mircrosoft, Página web de Silverlight, 2016, https://www.microsoft.com/silverlight/

44: Oracle, Sitio web de Java Swing, 2016, http://docs.oracle.com/javase/1.5.0/docs/guide/swing/

45: Google, Sitio web de Google Web Toolkit, 2016, http://www.gwtproject.org/

46: , MVP: Model-View-PresenterThe Taligent Programming Model for C++ and Java , , http://www.wildcrest.com/Potel/Portfolio/mvp.pdf

47: WordPress, Sitio Web del plugin Spider Catalog para WordPress, 2016, https://es.wordpress.org/plugins/catalog/

48: Web Dorado, Sitio web de la compañia creadora del plugin Spider Catalog, 2016, https://web-dorado.com/

49: Wordpress, Sitio Web de WordPress, 2016, https://es.wordpress.org/

50: Joomla, Sitio Web de Joomla, 2016, https://www.joomla.org/

51: Drupal, Sitio Web de Drupal, 2016, https://www.drupal.org/

52: Parse, Sitio Web de Parse, 2016, http://www.parse.com/

53: Xmind, Sitio Web de Xmind, 2016, http://www.xmind.net/

54: JSON, Especificación de JSON, 2016, http://www.json.org/json-es.html

55: IDE Canarias, FAQ_LIDAR, 2016, http://lidar.grafcan.es/lidar/FAQ_LIDAR.pdf

56: ESRI, ESRI Shapefile Technical Description, 1998, https://www.esri.com/library/whitepapers/pdfs/shapefile.pdf

57: Howard Butler, Martin Daly, Allan Doyle, Sean Gillies, Tim Schaub, Christopher Schmidt, The GeoJSON Format Specification, 2008, http://geojson.org/geojson-spec.html

58: Angel M. Felicísimo, Modelos Digitales del Terreno, 1994

59: Angel M. Felicísimo, Modelos Digitales del Terreno, 1994

60: Departamento de Ciencias Sociales, Geografía e Historia, ,

61: IGN, Centro de Descargas del IGN, 2016, http://centrodedescargas.cnig.es/CentroDescargas/index.jsp

62: ESRI, ESRI ASCII Raster format, 2016, http://resources.esri.com/help/9.3/arcgisdesktop/com/gp_toolref/spatial_analyst_tools/esri_ascii_raster_format.htm

63: IGN, Descripción del formato ASCII Grid, 2016

64: ESRI, Archivos ráster BIL, BIP y BSQ, 2016, http://desktop.arcgis.com/es/arcmap/10.3/manage-data/raster-and-images/bil-bip-and-bsq-raster-files.htm

65: Ministerio de Agricultura, Alimentación y Medio Ambiente, Sitio Web del Visor del SIGPAC, 2016, http://sigpac.mapa.es/fega/visor/

66: ESRI, Datums, 2016, http://desktop.arcgis.com/es/arcmap/10.3/guide-books/map-projections/datums.htm

67: Lt Col Richard K. Burkard, Geodesy For The Layman, 1984

68: IGN, Sistemas Geodésicos de Referencia, 2016, https://www.ign.es/ign/layoutIn/actividadesGeodesiaStmagd.do

69: QGIS, Sitio Web de QGIS, 2016, http://www.qgis.org/es/site/

70: IGN, Sitio Web del visor IBERPIX, 2016, http://www.ign.es/iberpix2/visor/

71: Mapbox, Sitio Web de Mapbox, 2016, https://www.mapbox.com/

72: Leaflet, Sitio Web de la librería Leaflet.js, 2016, http://leafletjs.com/

73: Hexagongeospatial, Sitio Web del formato ECW, 2016, http://www.hexagongeospatial.com/products/provider-suite/erdas-ecw-jp2-sdk

74: OSGEO, Sitio Web del formato GeoTIFF, 2016, https://trac.osgeo.org/geotiff/

75: Mapbox, Sitio Web de TileMill, 2016, https://www.mapbox.com/tilemill/

76: OSGEO, , 2016, https://trac.osgeo.org/openlayers/wiki/SphericalMercator

77: Mapbox, Sitio Web de MBTiles, 2016, https://www.mapbox.com/help/an-open-platform/#mbtiles

78: Xamarin, Sitio Web de Xamarin, 2016, https://www.xamarin.com/

79: Mapbox, Mapbox Vector Tiles, 2016, https://www.mapbox.com/vector-tiles/specification/

80: mongoDB, Sitio Web de mongoDB, 2016, https://www.mongodb.com/es

81: Glob3Mobile, Sitio web de G3M, , http://glob3mobile.com/

82: Glob3Mobile, Difusion de cambios, 2008, https://github.com/glob3mobile/g3m

83: Glob3Mobile, Notificaciones Push, 2008, https://github.com/glob3mobile/g3m

84: Glob3Mobile, Sitio Web de Mapboo, 2016, www.mapboo.com

85: GDAL, Sitio Web de GDAL, 2016, http://www.gdal.org/

86: GDAL, Servidor FTP, 2016, ftp://ftp.remotesensing.org/gdal

87: GDAL, Binarios de GDAL, 2016, https://trac.osgeo.org/gdal/wiki/DownloadingGdalBinaries

88: GDAL, Formatos ráster soportados por GDAL, 2016, http://www.gdal.org/formats_list.html

89: OSGEO, Especificación de TMS, 2012, https://wiki.osgeo.org/wiki/Tile_Map_Service_Specification

90: GDAL, Formatos vectoriales, 2016, http://www.gdal.org/ogr_formats.html

91: nextgis, Sitio web del plugin para QGIS llamado QTiles, 2014, https://github.com/nextgis/QTiles

II. PLIEGO DE CONDICIONES

1. PLIEGO DE CONDICIONES

A continuación se procederá a indicar los requisitos hardware de los equipos y las distintas herramientas software necesarias para poder llevar a cabo de manera satisfactoria el proyecto de fin de carrera.

1.1 . Requisitos hardware

Para llevar a buen puerto este proyecto de fin de carrera se utilizó los siguientes materiales:

- Microprocesador Intel Core i3-2350, con una frecuencia de reloj de 2.3 GHz.
- 4 Gb de memoria RAM.
- Sobre los 500 Gb de disco duro para, sobre todo, el tratamiento de mapas con los distintos SIG utilizados.
- Teléfono inteligente Samsung Galaxy S4 para probar la aplicación y los mapas.

1.2 . Requisitos software

Las herramientas que han sido utilizadas son las siguientes:

- Sistema operativo Windows 7 Home Premium de 64 bits.
- PhoneGap(JavaScript, HTML5, CSS)
- Appcelerator Studio
- QGIS Desktop 2.8.1

- Mapbox Studio 0.2.5
- Tilemill 0.10.1
- GDAL 1.10.1
- Java Development Kit 7
- Python 2.7, necesario para QGIS y GDAL
- Librería Node.js 0.12.7
- Librería G3M
- BonitaBPM 7.1.3
- Parse
- Wordpress
- Xmind
- Notepad++
- ERDAS 2011

III. PRESUPUESTO

1. PRESUPUESTO

Para la realización del presupuesto de este Proyecto de Fin de Carrera se han seguido las recomendaciones del Colegio Oficial de Ingenieros de Telecomunicación (COIT). Con el uso de ellas se obtiene la cuantificación económica según los baremos mínimos orientativos para trabajos profesionales para el año 2010. El presupuesto total se desglosa en distintos conceptos asociados al desarrollo del proyecto, los cuales se muestran seguidamente:

- Recursos materiales
- Trabajo tarificado por tiempo empleado
- Material fungible
- Costes de redacción del proyecto
- Derechos de visado del COIT
- Gastos de tramitación y envío
- Aplicación de impuestos

En los siguientes apartados se irán comentando cada uno de estos costes especificados.

1.1 . Recursos materiales

En este apartado se incluyen las herramientas hardware necesarias para llevar a buen puerto el proyecto así como aquellas herramientas software que se han utilizado tanto en el proyecto como en la redacción de esta memoria.

Los recursos materiales se clasifican en fungibles y amortizables. Fungible es todo aquel material propenso de gastarse con el uso, mientras que un material amortizable es aquel que posee un valor duradero durante varios años. La amortización se reparte entre todos los periodos en los cuales permanece el recurso material, en nuestro caso el coste de amortización se hará para un espacio de tiempo de tres años.

Se utiliza un método de amortización constante o lineal, aquí el material se supone que se deprecia de forma constante a lo largo de su vida útil. La cuota de amortización anual se calcula de la siguiente manera:

$$\textit{Cuota anual} = (\textit{Valor de adquisición} - \textit{Valor residual}) / \textit{Nº de años de vida útil} \quad (1.1)$$

De la fórmula anterior el factor "valor residual" equivale al valor que se presupone tendrá el material una vez pasada su vida útil.

1.1.1 . Costes de amortización de recursos hardware

Las herramientas amortizables de carácter hardware que han sido utilizadas a lo largo del proyecto son las siguientes:

- Ordenador portátil Asus X54H
- Smartphone Samsung Galaxy S4

Teniendo en cuenta que la duración de este proyecto fin de carrera es de aproximadamente un año y que, por otro lado, el cálculo del coste de amortización se constituye en un lapso de 3 años, los costes de amortización de los recursos usados se calcularán para el primer año. A continuación se muestran dichos costes.

Costes de amortización de recursos hardware			
Recurso	**Valor de adquisición**	**Valor residual**	**Cuota anual**
Ordenador portátil Asus	430,00 €	0,00 €	143,33 €
Smartphone Samsung Galaxy S4	600,00 €	120,00 €	160,00 €
		Total	303,33 €

Tabla III.4: Costes de amortización de recursos hardware

El coste total de estas herramientas hardware asciende a trescientos tres euros con treinta y tres céntimos de euro.

1.1.2 . Costes de amortización de recursos software

Las herramientas software amortizables que se han utilizado en el desarrollo del proyecto son las siguientes:

- Para la explicación de las herramientas de desarrollo multi-plataforma se han utilizado estas herramientas:
  - API de PhoneGap
  - API de Titanium
  - Appcelerator Studio

- Para el tratamiento de información geográfica se han utilizado las siguientes herramientas:

- QGIS
- GDAL
- G3M
- ERDAS 2011

▪ Para almacenar la información geográfica se ha hecho uso de estas utilidades:

- Mapbox

▪ Para el uso del Framework del IUMA se necesitó de estos software

- Xmind
- Parse
- Wordpress
- BonitaBPM

Todo el software mencionado se ha usado sin pagar nada. Alguno de ellos poseen la posibilidad de ofrecer algunas ventajas como más capacidad de almacenamiento o atención directa ante cualquier error que pudiera suceder durante su uso, no obstante a ello en este proyecto se ha sido capaz de ajustar nuestras necesidades para no tener que hacer desembolso alguno en cuanto a dichas herramientas.

Por otra parte, ha sido excluidos de las listas antes mencionadas el sistema operativo usado, en este caso Windows 7 Home Premium, por estar ya incluidos dentro del coste de hardware del equipo correspondiente.

Como conclusión, el total de las herramientas software empleadas asciende a la cantidad de *cero euros*.

1.1.3 . Costes de amortización de otros recursos de propiedad intelectual

Los recursos con propiedad intelectual que se utilizan el vigente proyecto de fin de carrera son los siguientes:

- Ortofotos históricas del PNOA (Plan Nacional de Ortografía Aérea).
- Archivos del MTN (Mapa Topográfico Nacional) en formato ráster y vectorial.
- Modelos digitales del terreno.
- Modelos digitales del terreno basados en la tecnología LIDAR.
- Modelo de elevación digital SRTM de la NASA.

El modelo de elevación digital procedente de la NASA es de acceso gratuito. Los demás recursos han sido obtenidos desde el sitio web del Instituto Geográfico Nacional, siendo estos de dominio público mientras se indiquen que proceden de ésta fuente.

Dicho esto, el coste total del resto de recursos de propiedad intelectual utilizados asciende a la cantidad de *cero euros*.

1.2 . Trabajo tarificado por tiempo empleado

Siguiendo las recomendaciones del COIT se obtiene una aproximación del importe de las horas empleadas en la realización del proyecto. Estos honorarios se calculan en base a la siguiente formula:

$$H=74,88*C_t*H_n+96,72*C_t*H_e \quad (1.2)$$

Dónde:

- H representa los honorarios totales por el tiempo dedicado.
- H_n son las horas normales trabajadas dentro de la jornada laboral.
- H_e son las horas especiales trabajadas.
- C_t representa un factor de corrección en función del número de horas trabajadas.

Teniendo en cuenta que este proyecto de fin de carrera ha sido hecho durante un periodo de 9 meses y siempre dentro de la jornada laboral normal, se obtiene un total de 1440 horas de trabajo(8 horas/día * 5 días/semana * 4 semanas/mes *9 meses). El COIT establece el valor del coeficiente según las horas trabajadas de acuerdo a la tabla siguiente.

Horas trabajadas	**Factor de corrección C_t**
Hasta 36 horas	1
Entre 36 y 72 horas	0,9
Entre 72 y 108 horas	0,8
Entre 108 y 144 horas	0,7
Entre 144 y 180 horas	0,65
Entre 180 y 360 horas	0,6
Entre 360 y 540 horas	0,55
Entre 540 y 720 horas	0,5
Entre 720 y 1080 horas	0,45
A partir de 1080 horas	0,4

Tabla III.5: Factor de corrección C_t según las horas trabajadas

Como el número de horas trabajadas es superior a 1080 horas, se aplica un factor de corrección de 0,4. Mediante la aplicación de la formula antes mencionada se calculan los consiguientes honorarios totales por las horas dedicadas al proyecto:

$$H=74,88*0,4*1440+96,72*0,4*0=43.130,88€ \quad (1.3)$$

El coste de trabajo tarificado por tiempo dedicado asciende a la cantidad de cuarenta y tres mil ciento treinta euros con ochenta y ocho céntimos de euro.

El tiempo total de trabajo dedicado a desarrollar este proyecto y sus costes se desglosa en la siguiente tabla.

Desglose de costes de trabajo tarificado por tiempo empleado			
Descripción	**Tiempo**	**Coste/Mes**	**Coste final**
Documentación	1 mes	4.792,32 €	4.792,32 €
Análisis	1 mes		4.792,32 €
Diseño	3 meses		14.376,96 €
Construcción	4 meses		19.169,28 €
		TOTAL	43.130,88 €

Tabla III.6: Desglose de costes de trabajo tarificado por tiempo empleado

1.3 . Material fungible

Los gastos del material utilizado así como los gastos para la edición de documentos, se muestran en la tabla que ascienden a la cantidad de *doscientos euros*.

Descripción	Coste
Papel, fotocopias, cartuchos de impresión, encuadernación, etc.	200,00 €

Tabla III.7: Costes de material fungible

1.4 . Costes de redacción del proyecto

El importe destinado a la redacción del proyecto se calcula en base a la siguiente formula:

$$R=0,07*P*C_h \quad (1.4)$$

En la cual:

- P es el presupuesto del proyecto obtenido.
- C_h es un coeficiente de ponderación en función del presupuesto.

El valor del coeficiente anterior viene dado por el OCIT, cuyo valor para proyectos presupuestados entre los 42.070,70€ y 63.106,05€ está fijado en 0,45. Dado que el valor del presupuesto hasta este momento es de 43.634,21€, el coste derivado de la redacción del proyecto es de:

$$R=0,07*43.634,21*0,45=1.374,48€ \quad (1.5)$$

Por lo tanto el importe final de redacción del proyecto asciende a la cantidad de mil trescientos setenta y cuatro euros con cuarenta y ocho céntimos de euro.

1.5 . Derechos de visado del COIT

Los gastos de visado del COIT se tarifican mediante la siguiente fórmula:

$$V=0,006*P*C_v \quad (1.6)$$

Dónde:

- P es el presupuesto del proyecto obtenido.
- C_v es un coeficiente reductor en función del presupuesto.

Hasta el momento el presupuesto de este proyecto asciende a la cifra de 45.008,89€. Este coste se obtiene mediante la suma de los costes de: recursos materiales, del trabajo tarificado, del material fungible y de la redacción del proyecto. Como el valor del coeficiente C_v para presupuestos de más de 30.050€ y menos de 90.150€ viene definido por el COIT con un valor de 0,9 el coste de los derechos de visado del proyecto se obtiene de la siguiente forma:

$$V=0,006*P*C_v=0,006*45.008,89*0,9=243,05€ \quad (1.7)$$

El coste de los derechos de visado del proyecto asciende a la cantidad de *doscientos cuarenta y tres euros con cinco céntimos de euro*.

1.6 . Gastos de tramitación y envío

Los gastos de tramitación y envío son fijos y se estipulan por el COIT en *seis euros con un céntimo de euro*.

1.7 . Impuestos aplicados

El importe final del presupuesto lleva implícito la aplicación del I.G.I.C. (Impuesto General Indirecto Canario). Para esta actividad económica el valor del I.G.I.C. graba el presupuesto con un 7%. El coste total del proyecto con el I.G.I.C incluido se desglosa a continuación.

Coste total del proyecto	
Descripción	**Total**
Recursos materiales: • Hardware: 303,33€ • Software: 0€	303,33 €
Trabajo tarificado	43.130,88 €
Material fungible	200,00 €
Redacción del proyecto	1.374,48 €
Derechos de visado	243,05 €
Tramitación y envío	6,01 €
Subtotal	45.257,75 €
I.G.I.C (7%)	2.262,89 €
Total	47.520,64 €

Tabla III.8: Coste total del proyecto

El importe final al que asciende el presupuesto de este proyecto es de *cuarenta y siete mil quinientos veinte euros con sesenta y cuatro céntimos de euro*.

Las Palmas de G.C., a ___ de Junio de 2017

Firma:

Serguey Gilberto Gómez Belyaeva

IV. ANEXOS

1. G3M

Glob3Mobile[81], G3M de aquí en adelante, es una herramienta multiplataforma de código abierto creada en el año 2010 por IGO SOFTWARE y la ULPGC con la ayuda del CDTI (Centro para el Desarrollo Tecnológico Industrial) y el Ministerio de Economía y Competitividad. Actualmente se puede hacer uso de esta herramienta tanto para los dispositivos móviles que tengan como sistema operativo iOS o Android, así como todos los entornos que soporten el lenguaje HTML5. Una de las razones por la que los decantamos por G3M es su capacidad de ofrecer mapas en tres dimensiones sin grandes requerimientos.

El funcionamiento de la API de esta herramienta es muy fácil e intuitivo, ya que por ejemplo, con solo varias líneas de código podemos tener a nuestra disposición un mapa tridimensional, Figura IV.100.

```
final G3MBuilder_Android builder = new G3MBuilder_Android(this);
_g3mWidget = builder.createWidget();
```

```
G3MBuilder_iOS builder([self G3MWidget]);
builder.initializeWidget();
```

```
final G3MBuilder_WebGL builder = new G3MBuilder_WebGL();
_widget = builder.createWidget();
```

Figura IV.100: Inicialización de G3M en Android, iOS y HTML5

1.1 . Características principales

Una de las características que llama la atención es que permite mostrar solo aquellas partes de las cuales estemos interesados en enseñar. Dicho de otro modo, con

G3M podemos delimitar sectores (llamados así por la propia herramienta) mediante la especificación de sus coordenadas geográficas. Además podremos cambiar el punto desde el cual se vea el mapa o inclinación con solo modificar varios parámetros.

Por otro lado, G3M permite el soporte en tiempo real por medio de la creación de un servidor de tal manera que cualquier cambio que hagamos en nuestro mapa se difundirá a los demás dispositivos conectados a dicho servidor, Figura IV.101. Asimismo podremos implementar lo que en el argot de los desarrolladores llaman "notificaciones push", mediante esta técnica podremos enviar información valiosa, como actualizaciones información acerca de los mapas, a todos los dispositivos conectados al servidor, Figura IV.102.

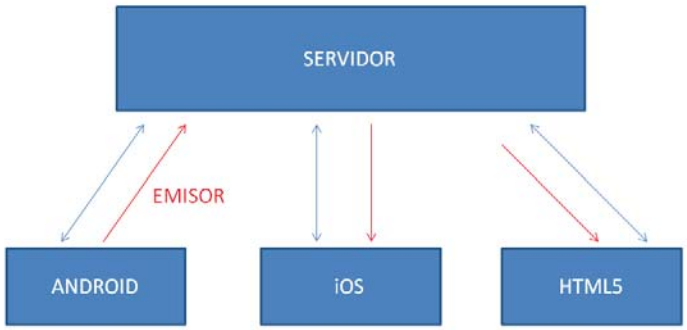


Figura IV.101: Difusión de cambios de una plataforma a las demás en G3M[82]

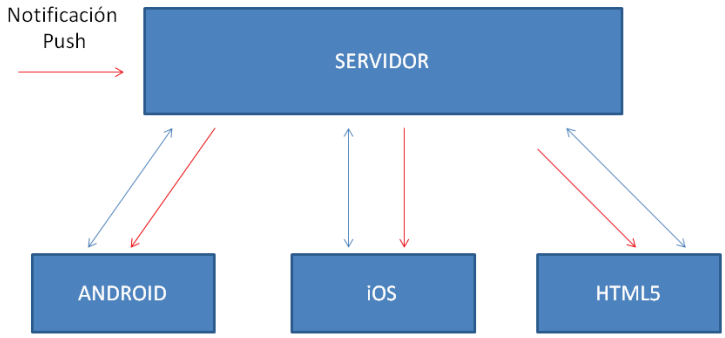


Figura IV.102: Notificaciones Push en G3M[83]

Otra de las virtudes que presenta esta herramienta es la de trabajar de manera offline ya que tiene implementada una memoria cache basada en SQLite. Esto nos permite poder guardar todos los datos e imágenes que aparezcan en la pantalla de nuestro dispositivo para poder utilizarlos sin ningún problema en caso que no tengamos acceso a Internet.

Refiriéndonos al apartado más técnico o específico de G3M, esta herramienta proporciona una gran versatilidad en cuanto al tratamiento con mapas ya que permite que estos vengan de distintas fuentes y formas, a continuación enumeramos algunos de ellos.

- WMS
- MapBox
- MapQuest
- BingMaps
- ArcGIS
- Raster
- Vectoriales

Permite solo dos tipos de proyecciones para los mapas. A primera vista podría ser un inconveniente pero con herramientas como GDAL, de la cual dedicaremos un apartado a hablar de ella dada su importancia para este proyecto, podemos convertir cualquier mapa con una proyección que no sea compatible con G3M a una que si lo sea. Estas dos proyecciones son la *EPSG:4326* y la *EPSG:3857*. La primera de ellas trata a la Tierra como un elipsoide mientras que la segunda la proyecta como una esfera.

Para el modelado de las alturas G3M usa los conocidos ficheros *bil* (Band Interleaved by Line), el cual no es más que un fichero ráster binario que contiene información acerca de las alturas. Dicho fichero puede tener variadas resoluciones, así cuanto mayor sea la resolución del fichero más información contendrá de las alturas por

metro cuadrado. Una vez más este fichero lo podemos obtener desde otros como pudieran ser aquellos con extensión *ASC*(ESRI ArcInfo ASCII Grid) vía GDAL.

G3M también ofrece la posibilidad de crear los mapas para las aplicaciones de una manera interactiva. Con Mapboo podremos crear mapas a través de Internet con sólo acceder a su página web[84] e ir siguiendo sencillos pasos. Una vez creado el mapa de esta manera podremos añadirlo sin ningún tipo de problema a nuestra aplicación.

2 . GDAL

GDAL[85] (Geospatial Data Abstraction Library) es una herramienta de libre acceso que nos permite trabajar con mapas, ya sea para crearlos, modificarlos o simplemente obtener información de ellos. Fue creada mediante la licencia del MIT por la fundación geoespacial de código abierto (Open Source Geospatial Foundation). Permite operar de manera precisa tanto con mapas ráster como con vectoriales, además funciona perfectamente en MacOS, Linux o Windows. Esta herramienta pone a disposición del investigador una potente variedad de comandos con los cuales poder modificar mapas o básicamente crearlos desde una imagen normal y corriente. Además de esto, la herramienta GDAL nos la podemos encontrar adaptada para diversos lenguajes de programación como C/C++, Python, Java, Perl o CSharp entre otros.

2.1 . Descarga e instalación

La obtención de GDAL se puede realizar mediante varias formas, una de ellas es a través de su página web, Figura IV.103. Desde aquí tenemos tres opciones desde las cuales poder obtener la herramienta.

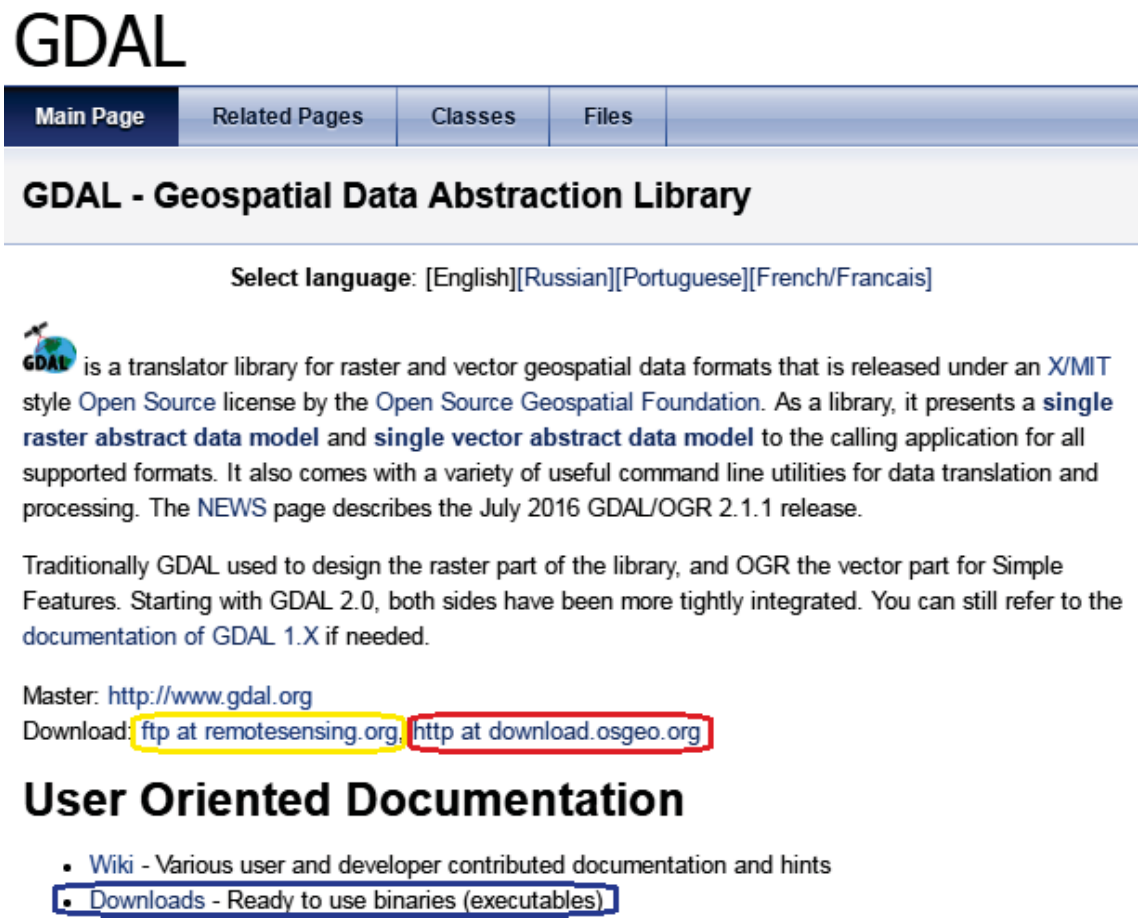


Figura IV.103: Sitio web de GDAL

La primera opción, marcada en amarillo, nos lleva a un servidor FTP [86] desde el cual podemos elegir la versión que queramos de la herramienta. La segunda opción, en rojo, nos conduce a una página desde la cual podemos descargar GDAL. La última opción nos lleva a una página[87] desde la cual se nos habilita la opción, además de poder instalar la herramienta para Windows y Linux, de instalar GDAL para MacOS X.

Otra manera de instalar esta herramienta es a través de terceros, o sea, mediante la instalación de otro software que ya venga con GDAL incluido. El software elegido ha sido ni más ni menos que QGIS, por varias razones. Una de estas razones es que este SIG es bastante potente y versátil, permitiéndonos utilizar una gran variedad de utilidades tanto para mapas vectoriales como para mapas ráster. Otra virtud de este software es que simple y llanamente es gratuito, lo cual hace realmente atractivo el querer trabajar con él. Una de las opciones que ofrece QGIS, que personalmente me

fascina, es la posibilidad de descargar desde el propio software "plugins" que no son más que pequeñas herramientas que no hacen más que darnos un plus más en cuanto a la utilización de QGIS.

Para poder obtener esta magnífica herramienta nos tenemos que ir a su sitio web desde la cual la podemos descargar sin ningún tipo de problemas. Actualmente QGIS se puede descargar para las siguientes plataformas, Figura IV.104:

- Windows
- Mac OS X
- Linux
- BSD
- Android

Desde la pestañas *Vectorial* y *Ráster* accedemos a todas las funciones de GDAL que ofrece la herramienta QGIS.

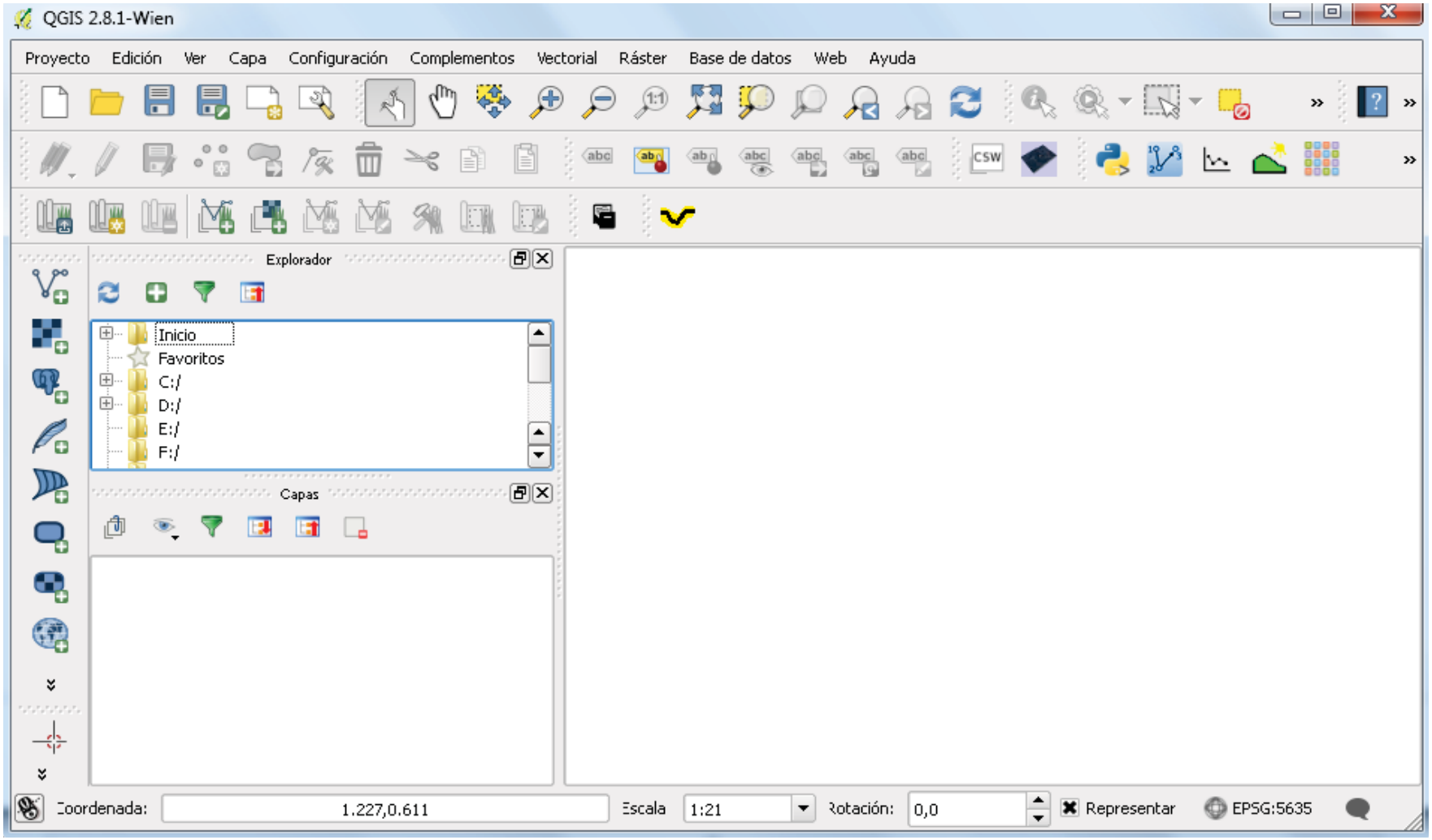


Figura IV.104: Vista estándar de QGIS

2.2 . Formas de uso

La librería GDAL se puede usar de múltiples formas, aunque en este tutorial nos centraremos en las dos maneras que hemos tenido para manejar de forma satisfactoria la herramienta.

2.2.1 . Uso de GDAL mediante la línea de comandos

La principal forma de utilizar esta herramienta es mediante la interfaz de línea de comandos. Para ello debemos poder acceder al terminal, que en Windows se realiza yendo a *Accesorios* → *Símbolo del sistema*. Seguidamente nos ubicamos en el directorio dentro del cual se encuentra el archivo a tratar y ya estaremos listos para aplicar los comandos de GDAL.

Esta biblioteca ofrece una gran variedad de comandos para el manejo tanto de mapas ráster o mapas vectoriales. Seguidamente comentamos algunos de los más influyentes y utilizados. Empezamos por aquellos que estan creados para el manejo de datos ráster.

2.2.1.1 . Comando gdalinfo

Este comando, nos proporciona información acerca del tamaño de la imagen, del formato en el cual esta la imagen. También nos dice el sistema de coordenadas en el cual se haya el fichero, además de proporcionarnos las coordenadas de las cuatro esquinas. Gracias a dicho comando podemos saber acerca de los GCP si es que hay, los GCP (Ground Control Points) son usados para georreferenciar una imagen. Para invocarla escribimos lo siguiente:

gdalinfo nombre_del_archivo (2.1)

El resultado es el siguiente, Figura IV.105 y Figura IV.106:

```
Driver: GTiff/GeoTIFF

Files: D:/PFC/isometricos/isla/isla_isometrico_4326_blanco.tif

Size is 52823, 52659

Coordinate System is:

GEOGCS["WGS 84",

    DATUM["WGS_1984",

        SPHEROID["WGS 84",6378137,298.257223563,

            AUTHORITY["EPSG","7030"]],

        AUTHORITY["EPSG","6326"]],

    PRIMEM["Greenwich",0],

    UNIT["degree",0.0174532925199433],

    AUTHORITY["EPSG","4326"]]
```

Figura IV.105: Resultado que ofrece gdalinfo, parte 1

```
Metadata:
  AREA_OR_POINT=Area
Image Structure Metadata:
  INTERLEAVE=PIXEL
Corner Coordinates:
Upper Left  ( -15.8449014,  28.1820035) ( 15d50'41.65"W, 28d10'55.21"N)
Lower Left  ( -15.8449014,  27.6790276) ( 15d50'41.65"W, 27d40'44.50"N)
Upper Right ( -15.3403590,  28.1820035) ( 15d20'25.29"W, 28d10'55.21"N)
Lower Right ( -15.3403590,  27.6790276) ( 15d20'25.29"W, 27d40'44.50"N)
Center      ( -15.5926302,  27.9305156) ( 15d35'33.47"W, 27d55'49.86"N)
Band 1 Block=52823x1 Type=Byte, ColorInterp=Red
  NoData Value=0
Band 2 Block=52823x1 Type=Byte, ColorInterp=Green
  NoData Value=0
Band 3 Block=52823x1 Type=Byte, ColorInterp=Blue
  NoData Value=0
```

Figura IV.106: Resultado que ofrece gdalinfo, parte 2

2.2.1.2 . Comando gdal_translate

El siguiente comando a comentar es *gdal_translate*. Su principal utilidad es el de permitir el cambio de formato (si no se especifica ninguno al realizar el cambio el formato final será *GeoTiff*) de un archivo ráster a otros formatos. La forma de utilizar este comando de una forma básica es de la siguiente manera:

gdal_translate ***-of*** *GTiff archivo_origen archivo_destino* (2.2)

Actualmente GDAL soporta el cambio de formatos mediante *gdal_translate* para casi 150 tipos de formatos. Algunos de los más relevantes son los siguientes[88]:

- ASCII Grid (AAIGrid)
- ERDAS Compressed Wavelets (*ECW*)
- ENVI .hdr Labelled Raster (*ENVI*), bil
- Graphics Interchange Format (*GIF*)
- TIFF/BigTIFF/GeoTIFF (*GeoTIFF*)
- JPEG JFIF (*JPEG*)
- Portable Network Graphics (*PNG*)

Otra de las funcionalidades que nos ofrece este comando de GDAL es el de poder seleccionar de una imagen dada un recuadro, dando para ello las coordenadas de las esquinas. Lo que deberíamos introducir en la consola sería lo siguiente:

gdal_translate ***-projwin*** *ulx uly lrx lry* ***-of*** *GTiff archivo_origen archivo_destino* (2.3)

Donde el sufijo ***–projwin*** es el que facilita la operación. Los datos siguientes indican las coordenadas a introducir y el orden en el cual se debe de hacer, donde:

- ulx: upper left x
- uly: upper left y
- lrx: lower right x
- lry: lower right y

Este comando permite georreferenciar una imagen, el inconveniente que puede tener es que para poder llevar a cabo esta tarea debemos estar en posesión de cierta información que en la mayoría de las veces es muy complicada de encontrar. Dicha información es la que se necesita para los llamados GCP (Ground Control Points), necesitamos saber exactamente las coordenadas de ciertos puntos (elegidos a nuestra conveniencia) para luego emparejarlos con puntos de la imagen a georreferenciar. A continuación haremos un ejemplo acerca de todo esto para que así quede más claro.

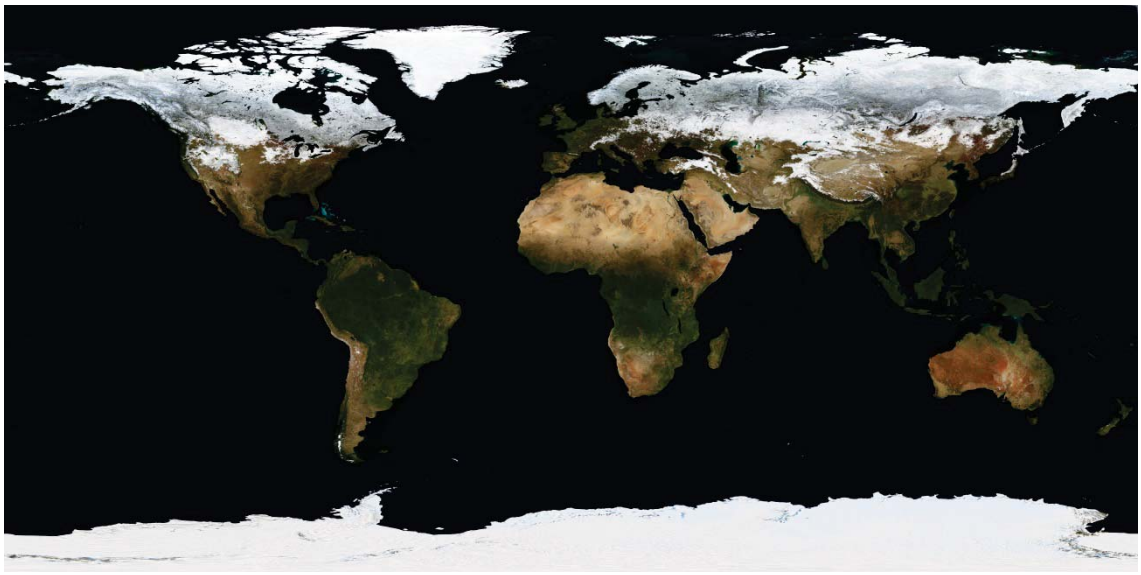


Figura IV.107: Fotografía obtenida gracias a la NASA

La imagen anterior es la que se va a georreferenciar. Es un archivo obtenido de la NASA que muestra todo el planeta, Figura IV.107. Su tamaño, en píxeles, es de 5400x2700. Los puntos que tomaremos para el georreferenciado son las esquinas ya que conocemos sus valores en píxeles y sus valores el sistema de proyección al cual trataremos de georreferenciar que será el *WGS84* o *EPSG:4326*.

1. GCP1: Esquina superior izquierda para el cual sus valores en píxeles son 0 0 y sus valores en el WGS84, -180^0 de longitud 90^0 de latitud.

2. GCP2: Corresponderá con la esquina superior derecha. En píxeles sus valores serán 5400 0 y 180^0 90^0 en grados.

3. GCP3: Esquina inferior derecha. En píxeles, 5400 2700. En grados 180^0 y -90^0.
4. GCP4: Esquina inferior izquierda. Sus valores serán 0 2700 en píxeles y -180^0 -90^0 en grados.

gdal_translate ***–a_srs*** EPSG:4326 ***–gcp*** 0 0 -180 90 ***–gcp*** 5400 0 180 90 ***–gcp*** 5400 2700 180 -90 ***–gcp*** 0 2700 -180 -90 *archivo_origen archivo_destino* (2.4)

Dónde:

- -a_srs: Permite asignar un sistema de coordenadas.
- -gcp: Puntos de controles. Asigna coordenadas a posiciones del archivo que se quiere georreferenciar.

De forma excepcional posibilita sobrescribir el sistema de coordenadas de un archivo por cualquier otro siempre y cuando esté soportado por GDAL.

gdal_translate ***–a_srs*** *srs_def archivo_origen archivo_destino* (2.5)

2.2.1.3 . Comando gdalwarp

Ahora pasaremos a comentar uno de los comandos más utilizados por GDAL, llamado *gdalwarp*. Permite pasar de un sistema de coordenadas a otro o asignar uno directamente, algo bastante útil en el ámbito de los Sistemas de Información Geográficos.

gdalwarp ***–s_srs*** *srs_def* ***–t_srs*** *srs_def archivo_origen archivo_destino* (2.6)

gdalwarp ***–t_srs*** *srs_def archivo_origen archivo_destino* (2.7)

Dónde:

- -s_srs: Sistema de coordenadas del archivo_origen

- -t_srs: Sistema de coordenadas del archivo_destino

También podemos algo parecido a lo que se puede hacer con *gdal_translate* en cuanto a recortar con una parte del archivo original. En este caso, en vez de pasarle unas coordenadas introducimos un archivo para que haga de mascara. De esta manera el archivo resultante tendrá la forma de la máscara introducida.

gdalwarp ***–cutline*** *máscara* ***-crop_to_cutline*** *archivo_origen archivo_destino* (2.8)

Dónde:

- -cutline: Indica el fichero que se va a usar como "molde".
- -crop_to_cutline: Se encargar de recortar el archivo_origen para que tenga la misma forma que la máscara.

2.2.1.4 . Comando gdal_merge.py

Este comando se suele utilizar principalmente para "combinar" varios archivos en uno. Es evidente la ventaja que tiene, ya que con esta utilidad podemos llegar a construir mapas más extensos desde la base de otros más pequeños.

gdal_merge.py ***-o*** *archivo_destino archivo_origen1 archivo_origen2* (2.9)

2.2.1.5 . Comando gdal2tiles.py

Con este comando podemos dividir un mapa cualquiera en pequeños mosaicos, agrupados por el nivel de zoom, de la imagen original. La forma de representar un mapa a partir de dichos mosaicos es un método bastante popular ya que permite renderizar un mapa en función del nivel de zoom que queramos y de la parte de dicho mapa que

estemos interesados en ver. Los mosaicos son generados siguiendo la especificación del Tile Map Services, TMS[89].

*gdal2tiles.py –**p** geodetic –**k** –**z** 1-8 mapa_original directorio_destino* (2.10)

Dónde:

- -p: Especifica el perfil de corte del mosaico. Puede ser geodetic, mercator o ráster.
- -k: Genera archivos en el formato *KML* usado por Google Earth.
- -z: Especifica el nivel o los niveles de zoom, el máximo es 21.

El resultado de ejecutar gdal2tiles.py es el siguiente, Figura IV.108:

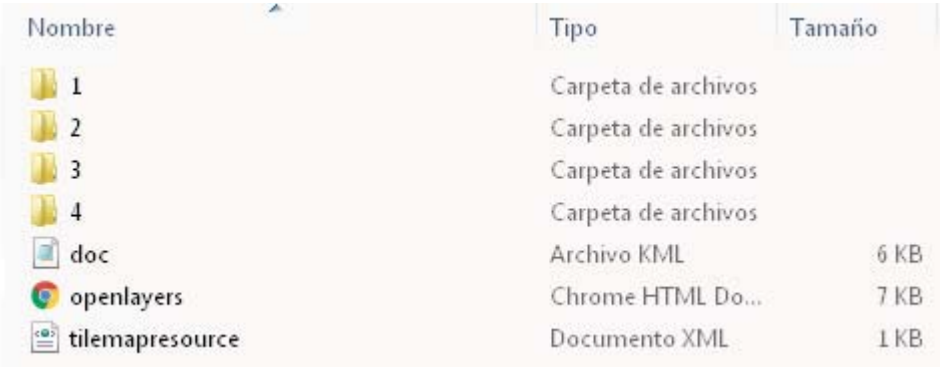


Figura IV.108: Contenido del directorio creado con el comando gdal2tiles

Para cada nivel de zoom se crea una carpeta. Además existen tres archivos más en este caso en concreto. El fichero con extensión *KML* se crea debido a que se ha incluido el parámetro –k, además, se crea un archivo con el misma extensión para cada mosaico. Con el archivo openlayers podremos visualizar el mapa en un navegador cualquiera. Por último, el archivo con extensión *XML* contiene información acerca de la especificación TMS.

2.2.1.6 . Comando gdal_contour

Con este comando podemos obtener las curvas de nivel de un mapa a partir de su modelo de elevación digital, o DEM en inglés. Estas llamadas curvas de nivel nos dan información acerca de la altura de una determinada zona geográfica. Cada curva de nivel representa puntos que se estan a la misma altitud. Con el *gdal_contour* podemos especificar cada cuantos metros dibujar una curva de nivel. El archivo que obtenemos esta en formato vectorial, Shape Figura IV.109.

*gdal_contour –**i** 10.0 archivo_DEM archivo_destino* (2.11)

Dónde:

- -i: Especifica cada cuántos metros se crea una nueva curva de nivel.

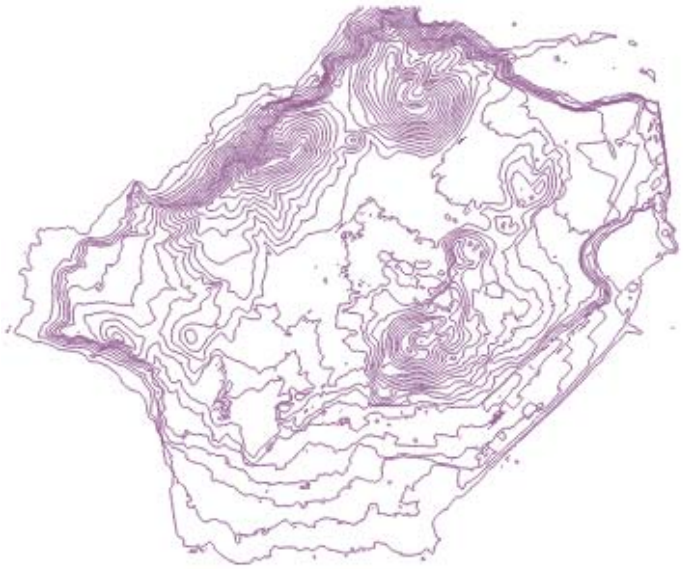


Figura IV.109: Curvas de nivel de La isleta

2.2.1.7 . Comando gdaldem

Es uno de los comandos de GDAL más poderosos ya que permite hacer bastantes operaciones con los modelos digitales de elevaciones. En este Anexo comentaremos algunos de los más relevantes.

Comenzamos con los llamados "mapas de sombras" o hillshade. Mediante este tipo de mapas simularemos un efecto tridimensional a partir de un DEM. Un ejemplo de utilización de este comando sería el siguiente, Figura IV.110:

gdaldem ***hillshade*** *archivo_DEM_origen archivo_destino* ***–z*** *1.0* ***–s*** *1.0* ***–az*** *315.0* ***–alt*** *45.0* (2.12)

Dónde:

- -z: Factor de exageración de las alturas.
- -s: Factor que representa la escala entre las unidades verticales y las horizontales.
- -az: Ángulo desde el cual el Sol ilumina.
- -alt: Altura a la que se encuentra el Sol respecto del horizonte.

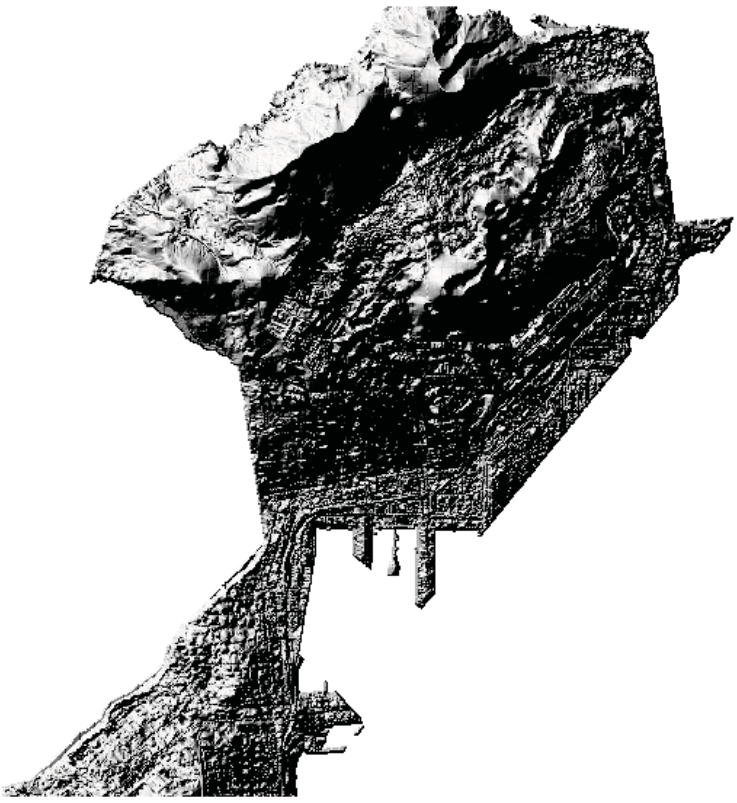


Figura IV.110: Mapa de sobras de La isleta

Los "mapas de relieve de color" o de tintas hipsométricas también se pueden obtener mediante este comando. Para crearlos, lo que se realiza básicamente es asignar colores a tramos de altitud. Así, en función de la altura el mapa posee un color u otro. Normalmente se suele asignar colores verdes para altitudes bajas, para alturas mayores se suele usar colores como el gris, el rojo o el blanco. Estos colores se guardan en ficheros de texto como el siguiente, Figura IV.111:

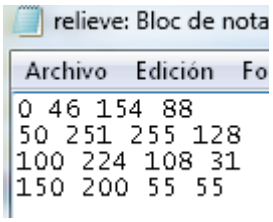


Figura IV.111: Definición de colores de mapa de relieve de colores

Para cada fila tendremos cuatro valores. El primero se corresponde con la altura mientras que los otros tres se corresponden con números que pueden ir desde el 0 hasta el 255 formando una tripleta de valores que hacen referencia a valores RGB. Con el archivo anterior tenemos colores verdes desde los cero metros hasta los 50 metros, desde ese valor hasta los 100 metros observamos un color amarillo. Hasta los 150 metros se ve un color anaranjado. A partir de aquí tenemos un color rojizo, Figura IV.112.

gdaldem ***color-relief*** *archivo_DEM_origen archivo_colores archivo_destino* (2.13)

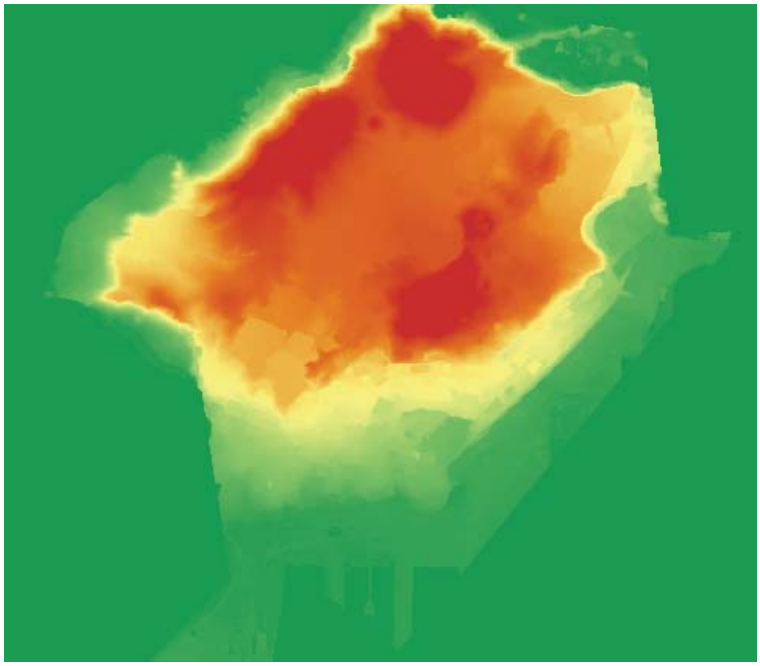


Figura IV.112: Mapa de relieve de colores de La isleta

Además de estos dos modos, *gdaldem* ofrece otros cinco modos que describimos brevemente.

Con *gdaldem slope* obtenemos un archivo con la información de la pendiente o inclinación, en grados, de cada píxel a partir de un archivo DEM.

*gdaldem **slope** archivo_DEM_origen archivo_pendientes* (2.14)

Con el modo *aspect* se crea un mapa de orientación. El archivo contiene valores entre los 0^0 y los 360^0 que representan el parámetro *azimuth* al que se encuentran las pendientes. Un valor de *azimuth* de 0^0 indica que la pendiente se encuentra de cara al Norte, 90^0 Este, 180^0 Sur y 270^0 Oeste, Figura IV.113.

*gdaldem **aspect** archivo_DEM_origen archivo_aspecto* (2.15)

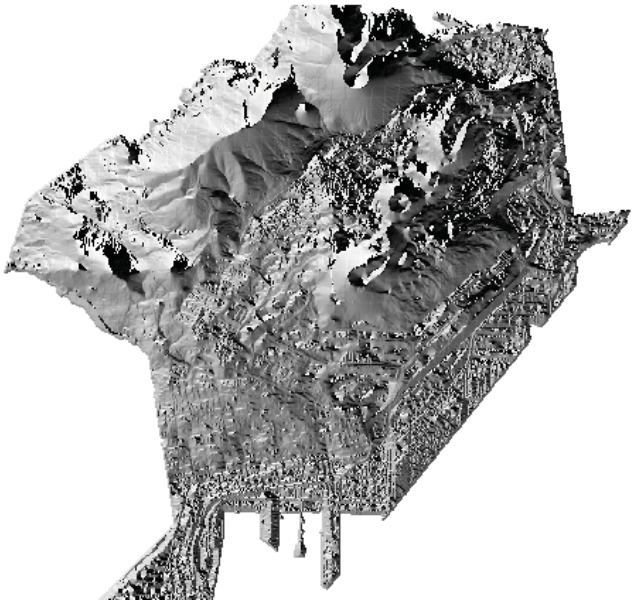


Figura IV.113: Mapa de orientación de La isleta

Con *gdaldem roughness* creamos un mapa que muestra la irregularidad del terreno dado el DEM, Figura IV.114.

*gdaldem **roughness** archivo_DEM_origen archivo_irregularidad* (2.16)

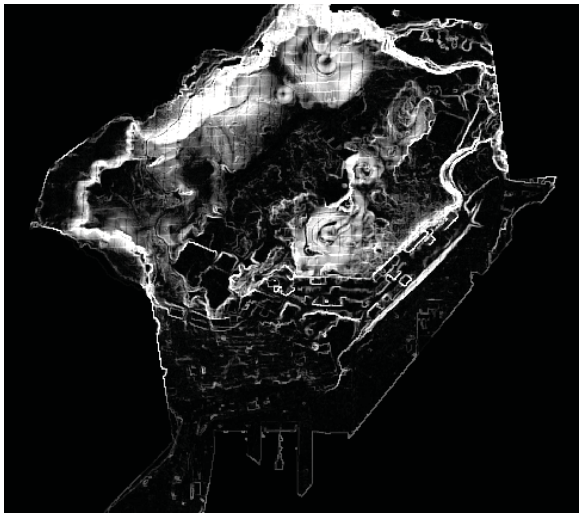


Figura IV.114: Mapa de irregularidad de La isleta

Finalmente con *gdaldem TRI* creamos un índice de robustez del terreno. Y con *gdaldem TPI* generamos un índice de posición topográfica.

2.2.1.8 . Comandos gdal_polygonize y gdal_rasterize

Para terminar con la parte de los comandos que tratan archivos en formato ráster por parte de GDAL hablaremos de *gdal_polygonize* y *gdal_rasterize*. Se puede decir que son comandos opuestos ya que el primero de ellos se ocupa de trasladar un fichero ráster en un vectorial, mientras que el segundo hace justo lo contrario, convierte un fichero vectorial en uno ráster.

gdal_polygonize.py ***–f*** *ESRI Shapefile archivo_origen_raster archivo_destino_vectorial* (2.17)

Dónde:

- -f: Especifica el formato vectorial que se va a usar, si no se especifica nada el formato de salida será GML.

gdal_rasterize ***—ts*** *3000 3000* ***–l*** *La_Isleta archivo_origen_vectorial archivo_destino_raster* (2.18)

Dónde:

- -ts: Tamaño de salida del ráster en píxeles.
- -l: Nombre de la capa.

Los comandos para el manejo de datos vectoriales con GDAL no son tan numerosos como los que existen para datos ráster. Seguidamente describiremos brevemente algunos de ellos.

2.2.1.9 . Comando ogrinfo

Comenzamos con *ogrinfo*. Permite obtener información del archivo tal y como se suele hacer con *gdalinfo*. Además proporciona diversos atributos que facilitan acceder a la información de manera más específica.

ogrinfo nombre_del_archivo (2.19)

2.2.1.10 . Comando ogr2ogr

Con *ogr2ogr* se realizan multitud de modificaciones a los ficheros vectoriales, aunque la principal utilidad es la de permitir el cambio entre diferentes formatos soportados por GDAL.

ogr2ogr ***–f*** *"formato_soportado" archivo_destino archivo_origen* (2.20)

A continuación, mostramos varios ejemplos de uso de este comando:

ogr2ogr ***–wrapdateline –t_srs*** *EPSG:4326* ***–clipdst*** *-5 40 15 55 archivo_4326.shp archivo_4083.shp* (2.21)

Aquí estamos reproyectando el archivo con ***–wrapdateline***, además de recortarlo con ***–clipdst***.

ogr2ogr Alberta.shp PG:dbname=canada ***–where*** *"province_name='Alberta'"* (2.22)

En el ejemplo anterior se extrae hacia un archivo *Shape* parte de un archivo *PostGis* mediante el atributo ***–where***.

Actualmente el comando ogr2ogr soporta gran cantidad de formatos, exactamente 84 formatos distintos. Algunos de los más importantes y notorios los listamos a continuación[90]:

- *ESRI Shapefile*
- *GeoJSON*
- *GML*
- *GMT*
- *GPX*
- *KML*
- *CartoDB*
- *PostgreeSQL/PostGIS*

2.2.1.11 . Comando ogrtindex

El comando *ogrtindex* nos da la opción de agrupar en un único archivo la información de varios archivos. El formato de salida por defecto es Shape.

ogrtindex streets.shp streets1/.shp streets2/*.shp* (2.23)

2.2.1.12 .

Finalmente, *ogrlineref* nos permite crear archivos que contengan referencias lineales de otro archivo dado.

ogrlineref ***–create –l*** *roads.shp* ***–o*** *parts.shp* ***–s*** *1000* (2.24)

Con este ejemplo creamos un archivo, llamado parts.shp que contiene los necesarios de las referencias lineales, en pasos de 1 kilómetro.

2.2.2 . Uso de GDAL en QGIS

Una vez explicado de manera concisa algunos de los comandos más utilizados en GDAL a través de la interfaz de línea de comandos, pasamos a comentar como se puede usar esta importante herramienta desde un punto de vista más visual con ayuda de QGIS.

Todas las acciones a realizar con archivos ráster se encuentran en la pestaña "Ráster" de QGIS. Asimismo, todo lo relacionado con ficheros vectoriales estan encuadrados en la pestaña "Vectorial".

2.2.2.1 . Comando gdalinfo

Empezamos, como hicimos anteriormente, con la utilidad de GDAL que proporciona información acerca de los archivos ráster *gdalinfo*. Para ello tenemos que ir a *Ráster* → *Miscelánea* → *Información.*

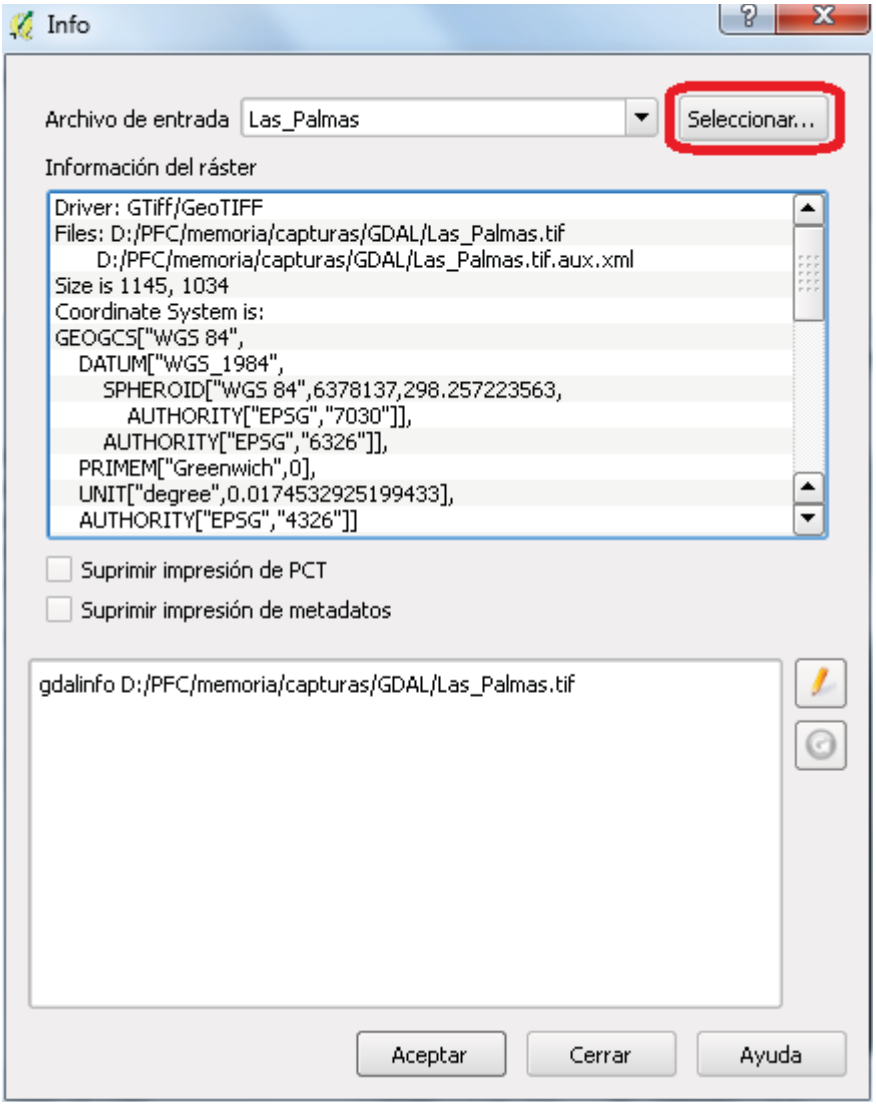


Figura IV.115: Uso de gdalinfo en QGIS

En la ventana que aparece a continuación solo tenemos que elegir el archivo del cual queramos obtener información y hacer clic en Aceptar, Figura IV.115.

2.2.2.2 . Comando gdal_translate

Con *gdal_translate* podemos realizar diversas operaciones como el cambio de formato o el recortar un mapa. Para el pasar de un formato a otro tenemos que ir a *Ráster* → *Conversión* → *Traducir.*

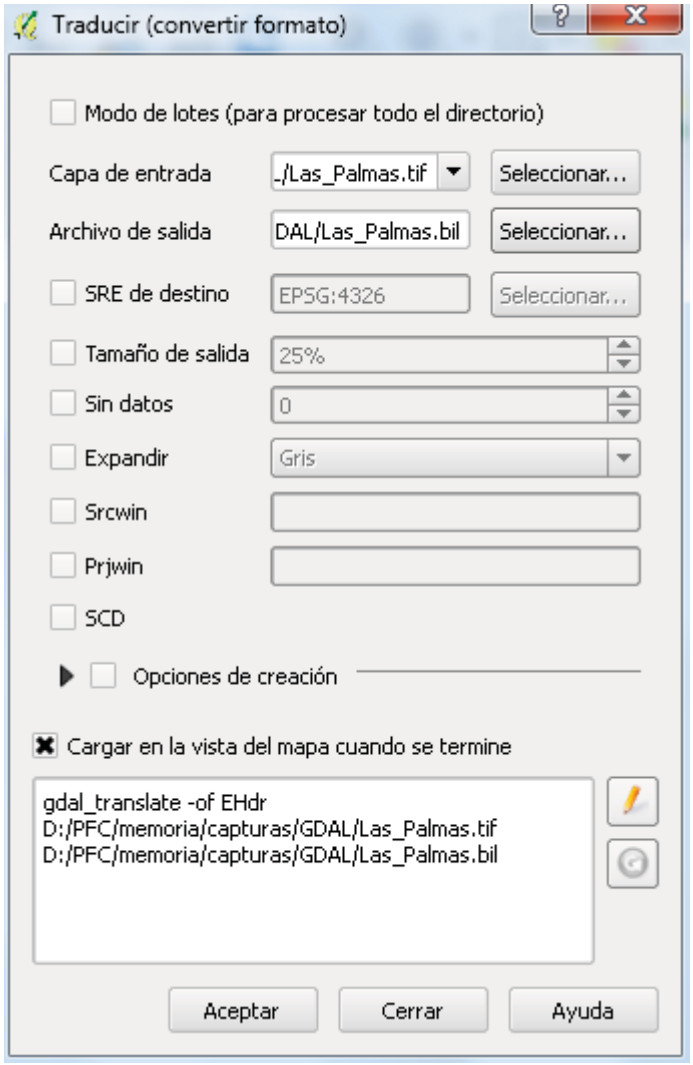


Figura IV.116: Uso de gdal_translate en QGIS

En esta ventana, Figura IV.116, tenemos a nuestra disposición diferentes opciones que nos da este comando para aplicar al mapa. Una de ellas es "Srcwin" que permite recortar el mapa original dando como datos de recorte píxeles. Con "Prjwin" podemos hacer lo mismo pero en este caso los datos que tenemos que introducir son coordenadas (su formato varia en dependencia del Sistema de Referencia con el que se trabaje).

Esto último se puede realizar de una manera más gráfica yendo a *Ráster* → *Extracción* → *Clipper*.

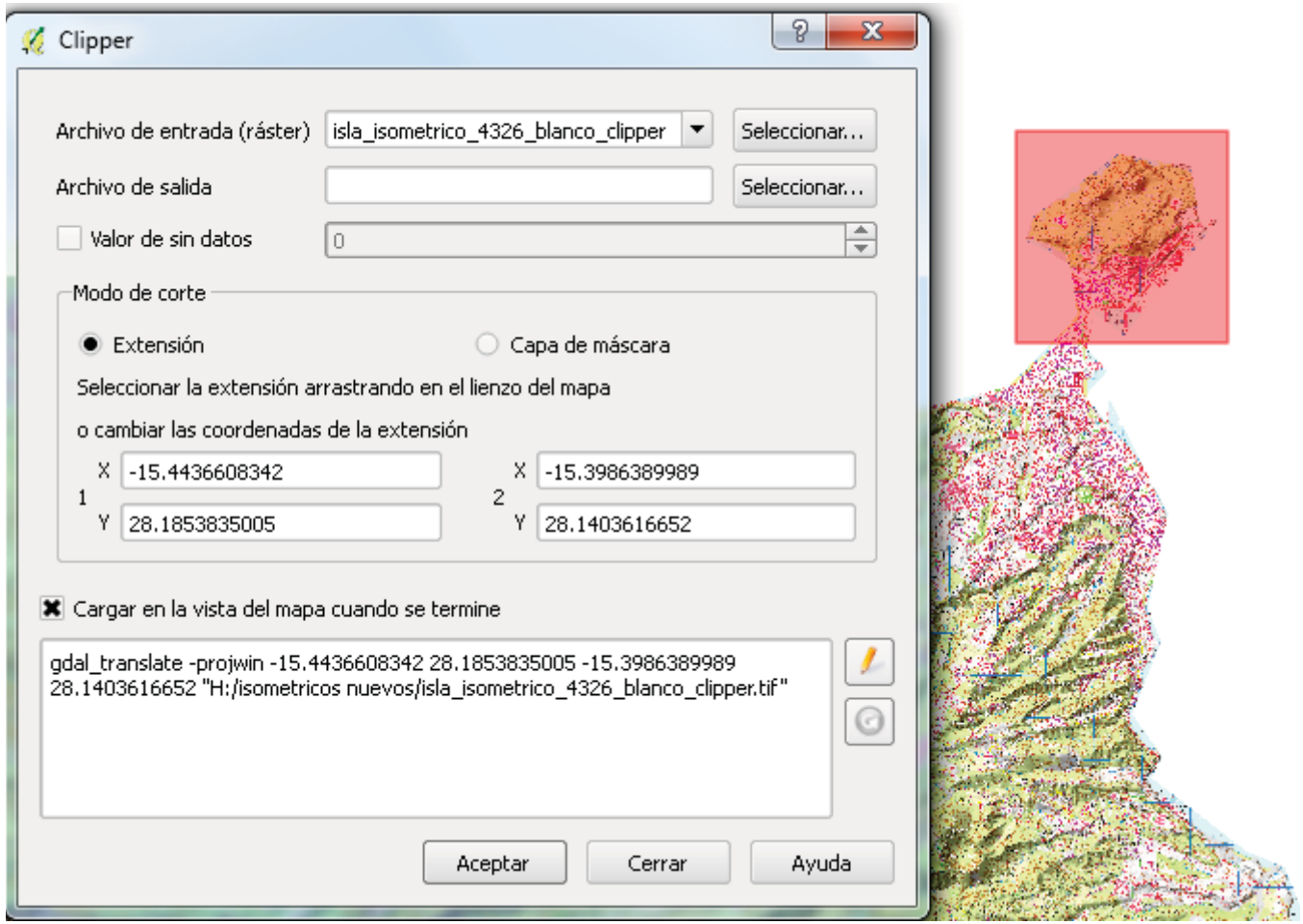


Figura IV.117: Recorte de imágenes con gdal_translate

El modo de corte a elegir es el llamado “Extensión”, Figura IV.117. La diferencia más sustanciosa con respecto al otro método radica en que podemos seleccionar, con esta forma, la subimagen deseada por medio del ratón. Esto nos da la ventaja de evitar lo engorroso y tedioso que resulta ser el poner las coordenadas una a una.

2.2.2.3 . Comando gdalwarp

Esta utilidad la podemos utilizar en QGIS para realizar múltiples tareas como cambiar la proyección de un determinado mapa o directamente asignarle una proyección. Otro uso es el de recortar un mapa en función de una capa o máscara que proporcionemos. Para el cambio de proyección tenemos que irnos a *Ráster* → *Proyecciones* → *Combar*.

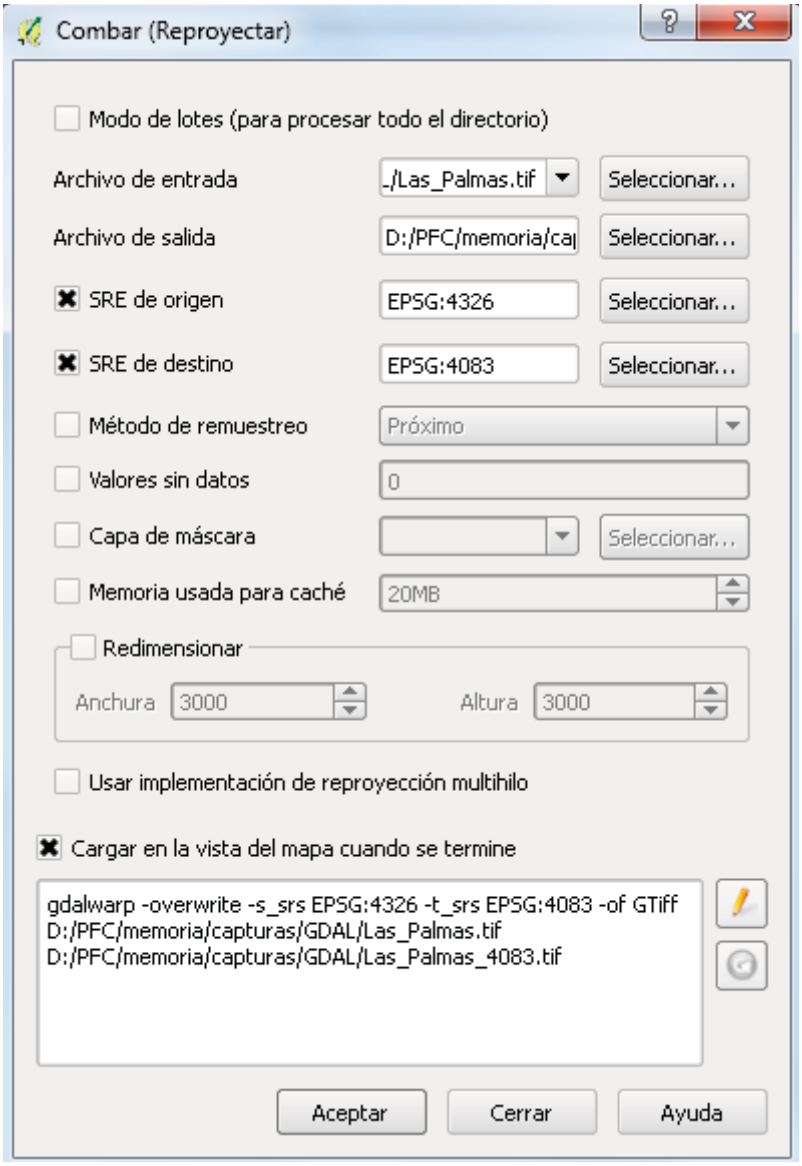


Figura IV.118: Uso de gdalwarp en QGIS

Los datos a introducir serían los archivos de entrada y de salida, además de sus Sistemas de Coordenadas, Figura IV.118. Si lo que queremos es, directamente, asignar un sistema de referencia a un archivo lo podemos hacer dirigiéndonos a *Ráster* → *Proyecciones* → *Asignar Proyección*.

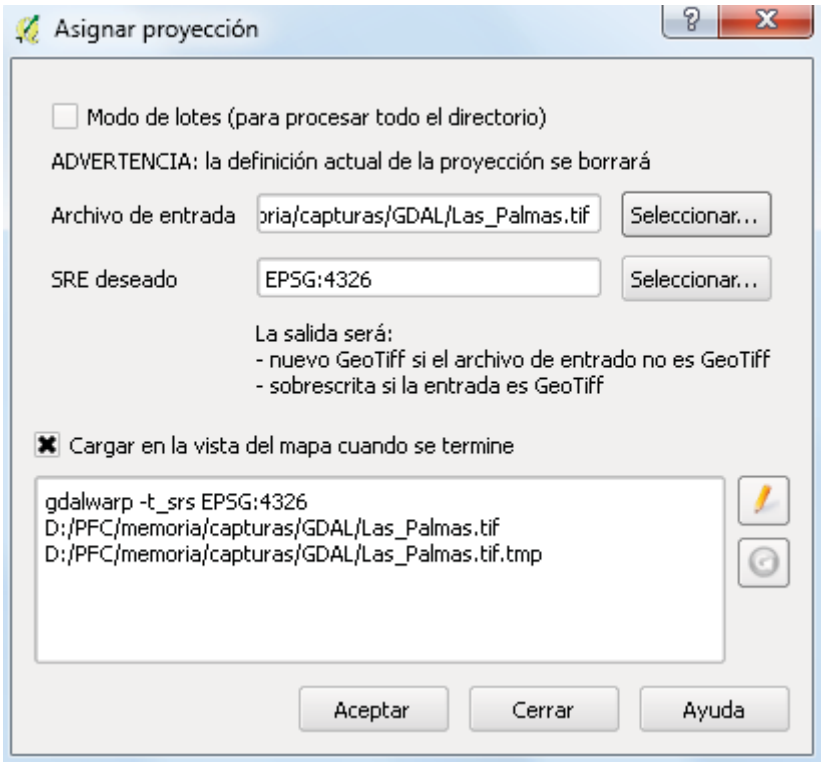


Figura IV.119: Proyecciones con gdalwarp

Los datos a introducir esta vez serían el archivo al que deseemos asignar una proyección específica y la misma proyección en sí, Figura IV.119. Finalmente, para obtener un mapa con una forma determinada lo haremos yendo a *Ráster* → *Extracción* → *Clipper*.

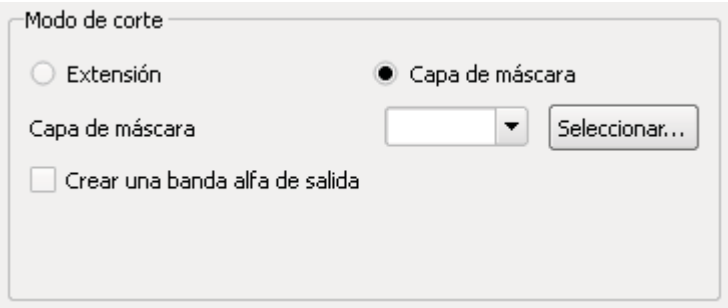


Figura IV.120: Ventana para aplicar una máscara a una imagen

La diferencia con respecto a lo que se hace con *gdal_translate* radica en el "Modo de corte", Figura IV.120. Aquí seleccionamos el "Modo de corte" por "Capa de máscara". De esta manera, en lugar de darle unas coordenadas los que proveemos es un archivo vectorial que tenga el tamaño y forma que deseemos para nuestro mapa.

2.2.2.4 . Comando gdal_merge.py

El uso de esta utilidad se lleva cabo en QGIS yendo a *Ráster* → *Miscelánea* → *Combinar*. Básicamente lo que tenemos que hacer es ir incluyendo los archivos que queramos combinar. Para ocasiones en las que se quieran combinar gran cantidad de archivos es de mayor utilidad usar una opción que aporta dicho comando, mediante el cual, colocando en una carpeta todos los archivos que deseemos combinar solo necesitaremos especificar la ruta de este directorio con *gdal_merge* solo tendríamos que nombrar el directorio, Figura IV.121.

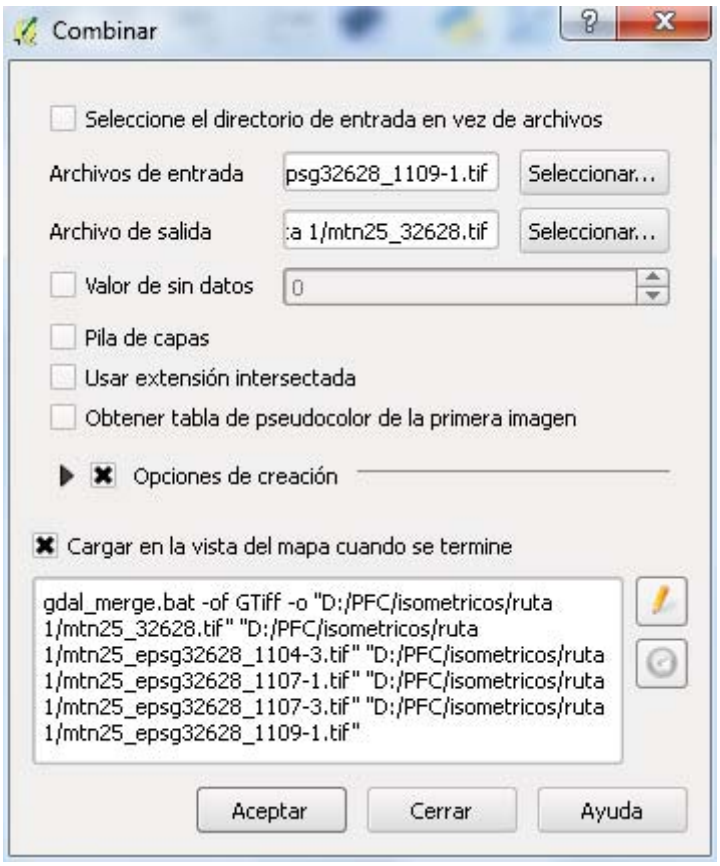


Figura IV.121: Uso de gdal_merge en QGIS

2.2.2.5 . Comando gdal2tiles.py

Este comando actualmente no esta disponible para su uso de con una interfaz gráfica tal y como se ha venido comentado con los demás comandos de GDAL. Lo más cercano que se puede estar de uso es mediante los "plugins" que descarguemos en QGIS, siendo el más fiel a su uso QTiles[91].

2.2.2.6 . Comando gdal_contour

El uso de *gdal_contour* es bastante sencillo, solo basta con ir a *Ráster* → *Extracción* → *Curvas de nivel*. En la ventana que aparecerá a continuación solo

tenemos que indicar el archivo del cual queremos hacer el estudio, así como el nombre el archivo que contendrá las curvas de nivel, Figura IV.122.

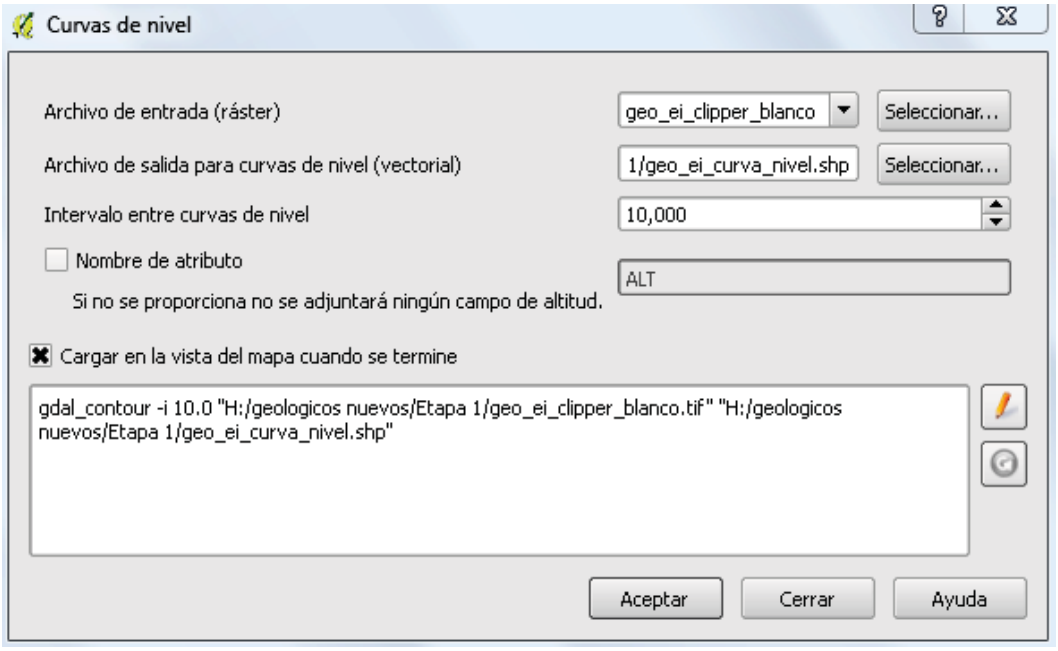


Figura IV.122: Uso de gdal_contour en QGIS

2.2.2.7. Comando gdaldem

Esta utilidad de GDAL es utilizada para el trabajo con Modelos Digitales de Terrenos, MDT Figura IV.123. Para usarlo basta con ir a *Ráster* → *Análisis* → *MDT*.

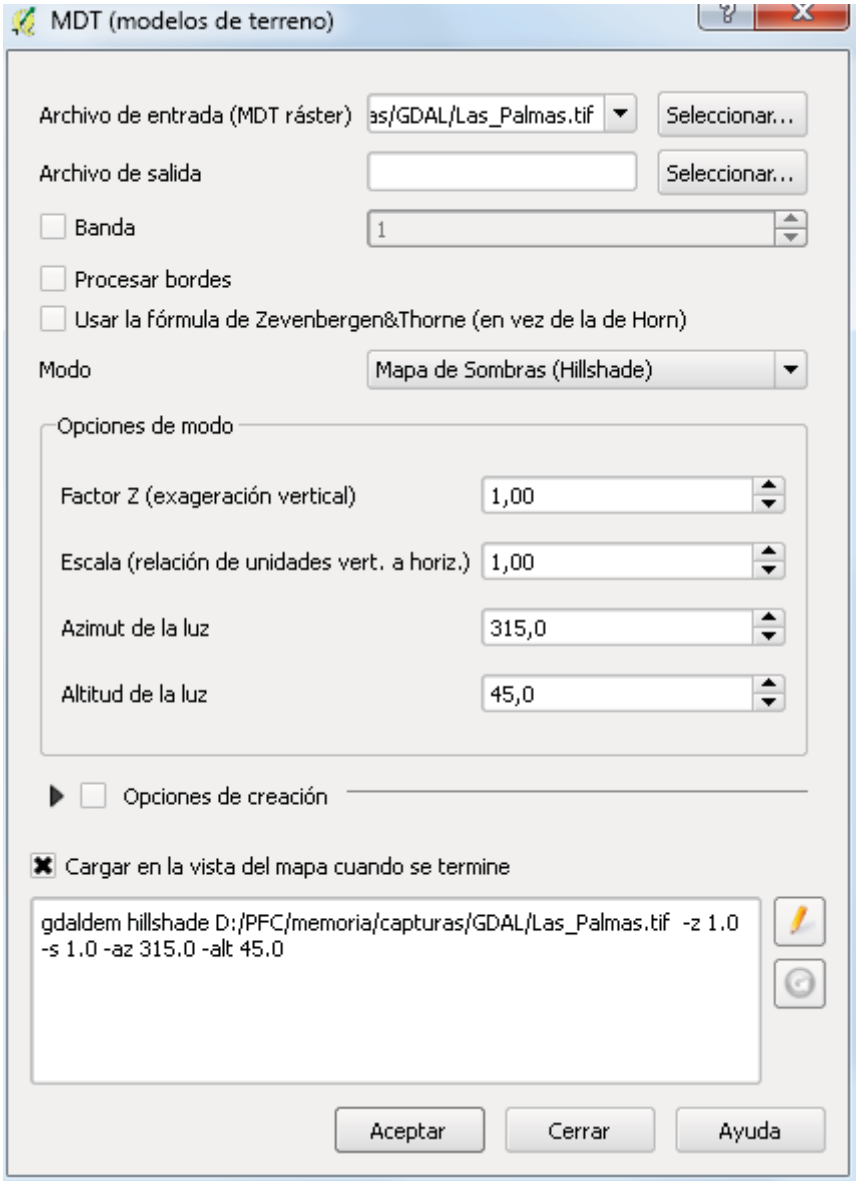


Figura IV.123: Uso de gdaldem en QGIS

A su vez, *gdaldem* engloba una serie de usos o modos que podemos ver en la opción "Modo", Figura IV.124. Como es lógico, las "Opciones de modo" irán cambiando en dependencia del uso que queramos hacer del comando.

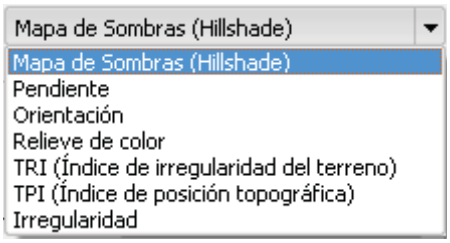


Figura IV.124: Variedad de modos de gdaldem

2.2.2.8 . Comandos gdal_polygonize y gdal_rasterize

Estas dos herramientas de GDAL realizan funciones opuestas. Mientras una se encarga de crear un fichero vectorial a partir de uno ráster (*gdal_polygonize*), con la otra y a partir de un archivo en cualquier formato vectorial soportado por GDAL podemos obtener uno ráster (*gdal_rasterize*).

Para usar *gdal_polygonize,* Figura IV.124, tenemos que ir a *Ráster* → *Conversión* → *Poligonizar*.

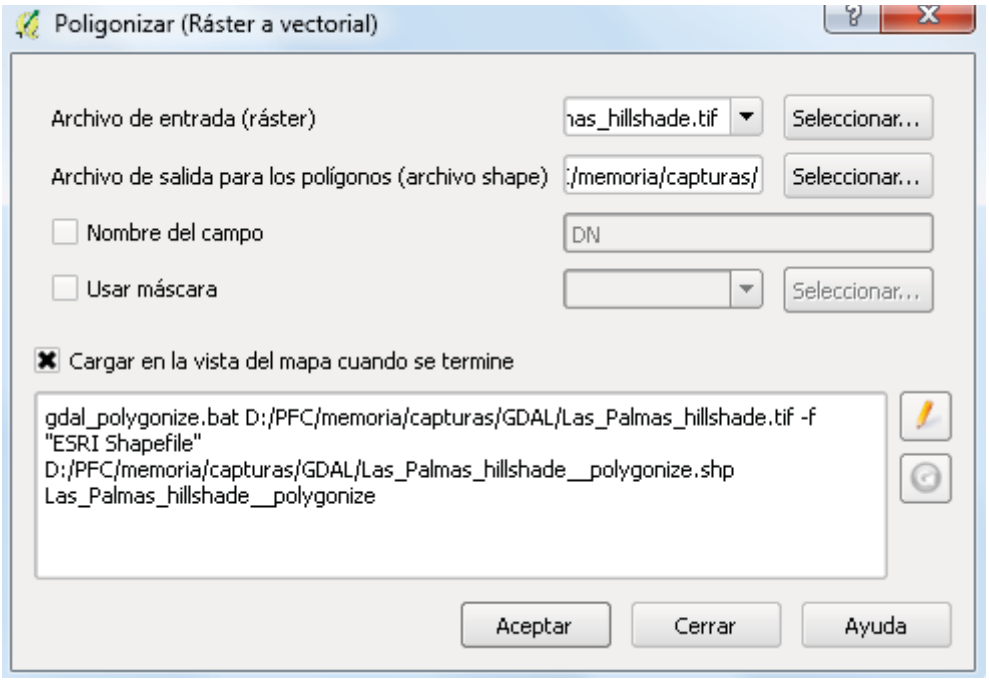


Figura IV.125: Ráster a vectorial con gdal_polygonize

Mientras que para hacer uso del comando *gdal_rasterize,* Figura IV.126, nos vamos a *Ráster* → *Conversión* → *Rasterizar*. Un atributo reseñable de este comando es el que te permite elegir el tamaño de salida del ráster en píxeles.

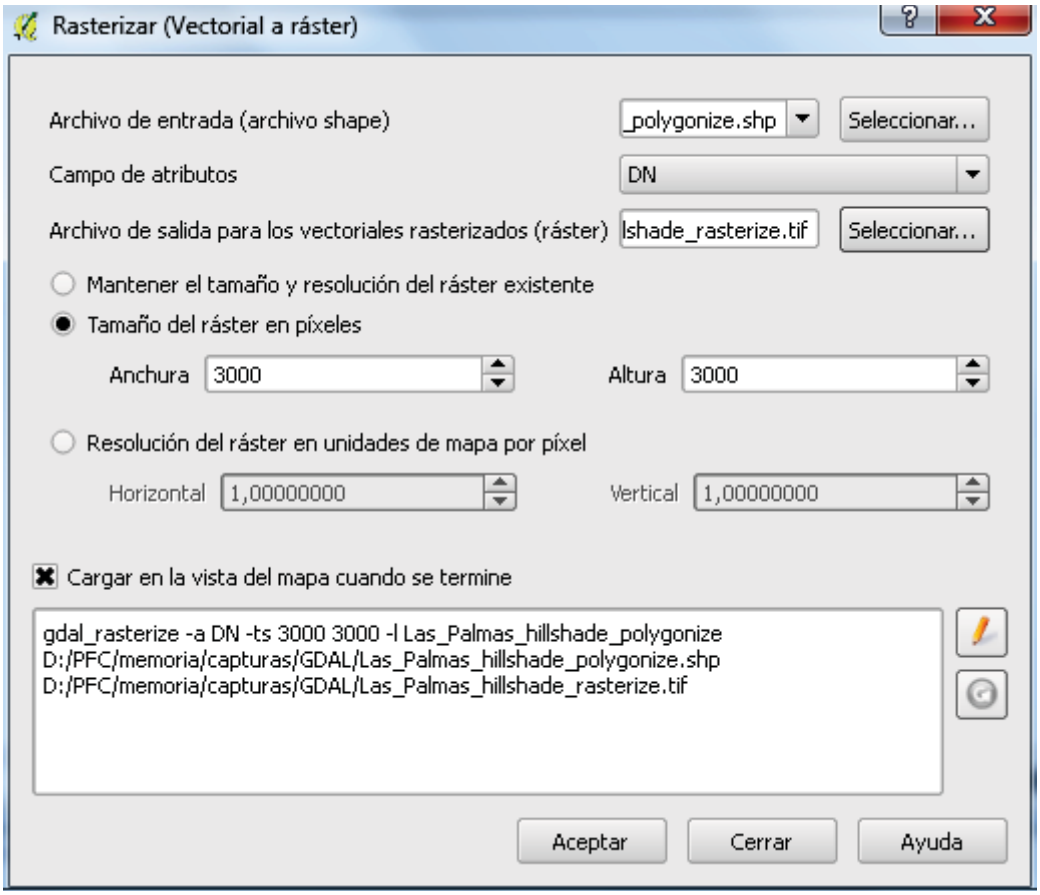


Figura IV.126: Vectorial a ráster con gdal_rasterize

Para el caso del tratamiento de ficheros en formato vectorial en QGIS el enfoque es algo distinto. La principal diferencia es que no se muestra que comando de GDAL/OGR se utiliza en cada momento. No obstante a ello, mostraremos algunas de las tareas que hemos llevado a cabo respecto al manejo de datos vectoriales en QGIS.

Para comenzar, podemos cambiar la proyección de un fichero con solo ir a *Vectorial* → *Herramientas de gestión de datos* → *Definir la proyección actual*. En la

ventana que aparecerá a continuación deberemos especificar el archivo al que se va a cambiar la proyección además de la proyección que elijamos, Figura IV.127.

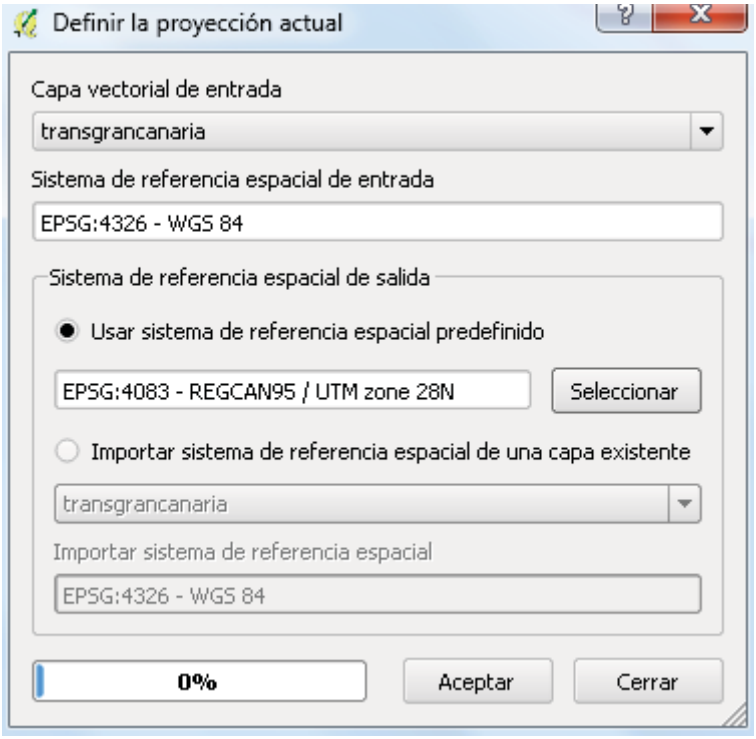


Figura IV.127: Cambio de proyección con formatos vectoriales

Otra funcionalidad muy útil es la de poder dividir un archivo en formato vectorial en dependencia de diversos parámetros que ya vengan identificados en el archivo original. Para ello tenemos que ir a *Vectorial* → *Herramientas de gestión de datos* → *Dividir capa vectorial*. El resultado final se almacena en un directorio de nuestra elección, siendo el formato de salida es *ESRI Shape,* Figura IV.128.

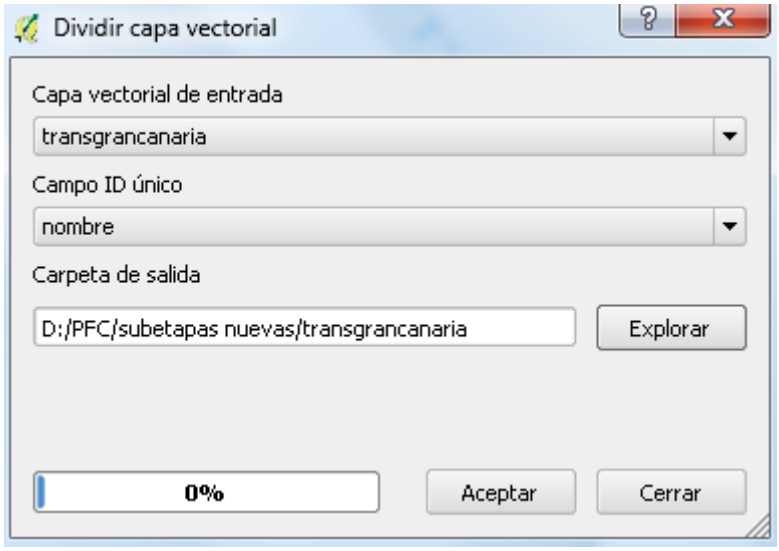


Figura IV.128: División por capas vectoriales