Escuela Politécnica Superior de Jaén

Grado en Ingeniería Geomática y Topográfica

UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior de Jaén

Trabajo Fin de Grado

IMPLEMENTACIÓN DE UN SISTEMA DE INFORMACIÓN GEOGRÁFICA PARA LA GESTIÓN TERRITORIAL DE UN MUNICIPIO

Alumno: Alberto Márquez Delay

Tutor: Prof. D. José Luis Mesa Mingorance
Prof. D. Antonio Garrido Almonacid
Dpto: Ingeniería Cartográfica, Geodésica y Fotogrametría

Septiembre, 2016

Universidad de Jaén
Escuela Politécnica Superior de Jaén
Departamento de Ingeniería Cartográfica, Geodésica y Fotogrametría

Don José Luis Mesa Mingorance y Don Antonio Garrido Almonacid, tutores del Trabajo Fin de Grado Titulado:

Implementación de un Sistema de Información Geográfica para la gestión territorial de un municipio, que presenta Alberto Márquez Delay, autorizan su presentación para defensa y evaluación en la Escuela Politécnica Superior de Jaén

Jaén, Septiembre de 2016

el alumno: Los tutores:

Alberto Márquez Delay

José Luis Mesa Mingorance

Antonio Garrido Almonacid

Índice

1. Objetivos

1.1. Objetivo principal

El objetivo principal del trabajo esta bastante claro en el título del mismo, queremos implementar un sistema de información geográfica para la gestión territorial del municipio de Alcalá de Guadaíra. Este puede ser el objetivo fundamental o principal del trabajo pero no podemos quedarnos solo con ese objetivo. Nos plantearemos una serie de objetivos secundarios que ayudarán a hacer que el trabajo sea mucho mas completo.

Está claro que como podemos observar nuestro fin será crear una aplicación para su utilización en una entidad local. Nos centraremos en crear una aplicación para el manejo de información catastral del municipio de Alcalá de Guadaíra y para completar esta base de datos catastral, añadiremos mas información, que será de interés para cada una de las parcelas.

Para esto último nos planteamos la creación de dos aplicaciones prácticas de nuestra base de datos geográfica. Serán dos aplicaciones, una distinta de la otra pero estrechamente relacionadas. Una primera consistirá en crear un script dentro de gvSIG, el cual podamos ejecutar y nos muestre una ventana emergente en la que podamos ver información de interés de la parcela seleccionada. El segundo script implementará la posibilidad de exportar a un fichero externo con extensión ODS, con la misma información que contendría el script anterior para poder imprimir el informe parcelario y facilitar la información a terceros que pudiesen estar interesado en la misma.

1.2. Objetivos secundarios

Para la realización de este trabajo nos plantearemos unos objetivos secundarios. Realizar todo el trabajo con software libre podría ser uno de estos objetivos. Pero no solo en la utilización de software especifico para información geográfica como son los Sistemas de Información Geográfica, si no que también nos planteamos utilizar un Sistema Operativo libre y programas de ofimática también libres. Todo esto pensando en la futura utilización del software en una entidad publica, abaratando así los costes y desviculandonos así de los software privativos dentro de la administración pública.

Por software libre entendemos el conjunto de software que por elección manifiesta de su autor, pueden ser copiado, estudiado, modificado, utilizado libremente con cualquier fin y redistribuido con o sin cambios o mejoras. Definición que encontramos en wikipedia.org.

Como Sistema Operativo utilizaremos Ubuntu 16.04, una versión LTS de larga duración y con soporte para 5 años. Es una versión muy estable de Linux y una distribución que cuenta con la mayor comunidad y por tanto con un respaldo para

resolver problemas mucho mayor que otras distribuciones conocidas de linux, como podrían ser Linux Mint o Debian. Esta última es una distribución equiparable en comunidad a Ubuntu pero la curva de aprendizaje es mas lenta.

El software que utilizaremos para nuestro sistemas de información geográfica será gvSIG, de origen valenciano. Desarrollado inicialmente por la Comunidad Valenciana y continuado por la Asociación gvSIG. Nos ayudará y nos dará la facilidad necesaria para poder programar utilidades añadidas para la gestión territorial del municipio. Esta será su baza fundamental para ser el pilar sobre el que programemos, ya que existe muchísima documentación sobre scripting en gvSIG. Nos ayudaremos en alguna ocasión también de QGIS, un Sistema de Información Geográfica de código libre y que fue uno de los primeros proyectos de la Fundación Espacial de Código Abierto (OSGeo). Este software lo utilizaremos en los casos en los que gvSIG no cubra las necesidades de procesamiento requeridos para nuestro trabajo.

Para el apartado de ofimática también nos proponemos utilizar software libre, LibreOffice será el utilizado por su mayor extensión en el mundo del software libre y por que es un software muy completo y amigable, parecido al paquete Office de la empresa Microsoft. Esto ultimo será de gran ayuda para hacer mas facil su aprendizaje.

2. Estudio y diagnóstico previo de la realidad territorial del municipio

El municipio que hemos seleccionado para la implementación de un Sistema de Información Geográfica es el municipio de Alcalá de Guadaíra, situado en la provincia de Sevilla, en la comunidad autónoma de Andalucía. Alcalá de Guadaíra cuenta con un censo poblacional de 74.845 habitantes (a 1 de Enero de 2015 según el Instituto de Estadística y Cartografía de Andalucía). Su extensión superficial es de 286 km², con una densidad de población de 261,70 hab/km². Su ubicación en coordenadas del sistema de referencia ETRS89 proyección UTM Huso 30 son 247501 metros en el eje de abscisas y 4135661 metros en el eje de ordenadas, con una altitud media sobre el nivel medio del mar en Alicante de 58 metros.

Ubicación de Alcalá de Guadaíra en España

Alcalá de Guadaíra forma parte del conjunto geográfico denominado Los Alcores, situado en una meseta que se levanta sobre el valle del Guadalquivir, al igual que los

municipios de Carmona, El Viso del Alcor y Mairena del Alcor.

Alcalá de Guadaíra es el tercer municipio con mayor población de la provincia de Sevilla, por detrás de la capital Sevilla y la población vecina de Dos Hermanas, es lógico pensar en la necesidad de implementar un sistema de información geográfica para la gestión territorial del municipio y con ello conseguir una mejora en todos los servicios públicos prestados por el ayuntamiento.

Ubicación de Alcalá de Guadaíra en la provincia de Sevilla

3. Estudio de la información territorial disponible del municipio

Tras realizar estudios de información geográfica existente en el municipio encontramos la primera causa de poder que empujaría aun mas a la necesidad de implementar un Sistema de Información Geográfica en este municipio, la información existente es pobre y muy dispersa.

La información encontrada referente al municipio de Alcalá de Guadaíra proviene de los Datos Espaciales de Referencia de Andalucía (DERA), la revisaremos ya que en algunos casos hemos encontrado falta de datos o datos mal posicionados. Debido a todo esto nos veremos forzados a hacer levantamiento de alguna información o modificación de parte de la información final que queremos en nuestra base de datos.

Las capas encontradas en el proceso de investigación son:

a) Capas de limites administrativos de la provincia de Sevilla y del municipio de Alcalá de Guadaíra. También de las secciones censales para tener información de la población lo mas precisa posible. Estos datos las obtenemos de DERA.

b) Capa de viales descargados del servicio web WFS de Cartociudad (Cartociudad).

c) Capa de manzanas de Alcalá de Guadaíra, también de Cartociudad.

d) Las capas de Farmacias, Parques de Bomberos, Policía, Museos, Ayuntamiento,

Bibliotecas, Equipo Educativo, Hospitales y Centros de Salud también los obtenemos de la web de DERA.

e) Capa de información de las vías de ferrocarriles (FFCC) obtenidas de los servicios de la web del DERA.

f) Descarga de un Modelo Digital del Terreno con una resolución de 5 metros del Instituto Geográfico Nacional (IGN MDT).

g) Capa de Hidrología descargada masivamente de DERA.

h) Descarga de información estadística sobre la distribución de la población según la sección censal de la provincia de Sevilla, descargada del Instituto Nacional de Estadística (INE).

i) Obtención de una imagen donde podemos localizar los puntos de Zonas Wi-Fi Libres situadas por el municipio. Solo hemos podido encontrar un folleto informativo en la web del ayuntamiento de Alcalá de Guadaíra (Ayunt. Alcalá de Guadaíra).

j) Información de las rutas de las cuatro lineas de autobús urbano que existen en Alcalá de Guadaíra. También obtenida de la web de la ayuntamiento de Alcalá de Guadaíra. Solo encontramo una lista de las calles por donde cada ruta hace sus paradas.

k) Y por ultimo descarga masiva de datos y cartografía vectorial en formato SHP de la sede electrónica del Catastro (Catastro).

4. Creación de la base de datos geográfica. Anexión de información de interés para el municipio

Para la realización de la base de datos geográfica hemos hecho un proceso de investigación a nivel municipal, provincial, autonómico y estatal. De hecho, la gran mayoría de la información geográfica que hemos podido encontrar es del Instituto de Estadística y Cartografía de Andalucía.

Para nuestra base de datos geográfica vamos a recabar una serie de capas de información relevantes para la gestión del municipio. Estas capas atenderán a necesidades básicas como pueden ser la educación o la sanidad. También incluiremos información catastral, que al final será sobre la que nos centremos en este trabajo, realizando un par de script ejecutables en gvSIG que nos mostrarán información de la parcela que deseemos o nos exportará dicha información a un fichero externo.

El software SIG elegido es gvSIG, principalmente por ser un *software libre* y también por la cantidad de documentación existente para ayudarnos y apoyarnos a la hora de programar nuestros propios script. El lenguaje que utiliza gvSIG para sus script es

Python, un lenguaje relativamente sencillo de aprender y fácil de usar para alguien familiarizado con el mundo de la programación y mas explicitamente con la programación orientada a objetos.

4.1. Lenguaje de programación Python

Python está administrado por la Python Software Foundation (Python Software Foundation), posee licencia de código abierto compatible con la licencia pública general de GNU a partir de la versión 2.1.1

Es un lenguaje de programación interpretado y de programación multiparadigma. Esto significa que mas que forzar a los programadores a adoptar un estilo particular de programación, permite varios estilos: programación orientada a objetos, programación imperativa y programación funcional. Otros paradigmas están soportados mediante el uso de extensiones.

Una característica importante de este lenguaje de programación es la resolución dinámica de nombre, es decir, lo que enlaza un método y un nombre de variable durante la ejecución del programa. Python es un lenguaje que puede incluirse en aplicaciones que necesiten una interfaz programable.

4.2. Selección de sistema de referencia

Una vez descargada toda la información detallada en el apartado anterior, nos disponemos a adecuarla a nuestras necesidades y exigencias, ya que para poder trabajar con toda la información en conjunto lo primordial será disponer de todas las capas de información en el mismo sistema de referencia, en nuestro caso se corresponde con el sistema geodésico de referencia ETRS89 en su Proyección UTM Huso 30.

Seleccionamos este sistema de referencia ya que en España oficialmente desde el 29 de agosto de 2007 el sistema geodésico de referencia oficial es el ETRS89 en su proyección UTM (husos 29, 30 y 31). Por lo tanto es lógico realizar nuestro trabajo dentro de este marco legal.

4.3. Procesos de adaptación de la información. Creación de Base de Datos

A continuación detallaremos qué procesos de modificación, eliminación,recorte, etc. han sufrido cada una de las capas de información que compondrá nuestra base de datos geográficos final.

En una primera etapa de trabajo nos vemos obligados a adaptar la información encontrada para hacerla uniforme, así facilitaremos el trabajo posterior con nuestra base de datos en posibles análisis o desempeños que pueda tener en un futuro.

4.3.1. Limites administrativos

Estas capas las descargamos de DERA masivamente y su sistema de referencia original es el ETRS89, por lo tanto no necesitaremos reproyectar la información. Sin embargo, al descargar masivamente obtenemos un fichero comprimido en el que está toda la información relevante en la división administrativas.

Para nuestro trabajo solo nos interesaremos de momento en tres capas: limites de términos municipales, capa de delimitación de las provincias andaluzas y la relativa a sección censal.

Para las tres capas deberemos realizar el mismo proceso aunque con pequeñas diferenciaciones que detallaremos a continuación:

a) Términos municipales. Capa con todos los términos municipales de Andalucía, por tanto deberemos seleccionar el que nos interesa, el municipio de Alcalá de Guadaíra. Para esto abrimos “Selección/Selección por Atributo”(Ilustración 4.1), estableceremos la condición de que “MUNICIPIO= Alcalá de Guadaíra”.

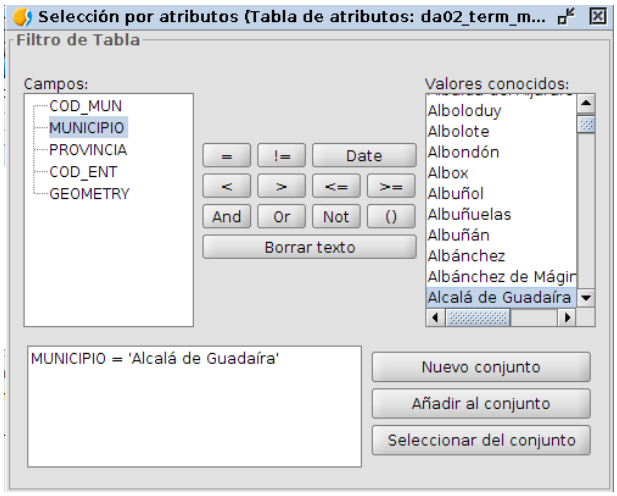


Ilustración 4.1

Como podemos ver esto nos seleccionará solo la geometría del municipio que nos interesa. A continuación deberemos guardar solo esta geometría del municipio en una capa. Para realizar esto ejecutamos “Capa/Exporta a” y exportamos a archivo SHP, solo las entidades seleccionadas y seleccionando una ruta donde guardar el fichero generado.

b) Provincias de Andalucía. Capa que contiene todas las provincias de Andalucía, por tanto tendremos que ejecutar el mismo proceso de “Selección/Selección por atributos”, solo que esta vez la condición para la búsqueda sera el “PROVINCIA = Sevilla”. Tras esto deberemos repetir también el mismo proceso para exportar sólo esta geometría a un archivo SHP.

c) Sección censal. Para esta capa, aprovechando que tenemos seleccionado de

antes en la capa de municipios el de Alcalá de Guadaíra, ejecutaremos una Intersección entre las capas Sección Censal y Municipios.

Como podemos ver en Ilustración 4.2 seleccionamos convenientemente las capas y deberemos chequear solo las geometría seleccionadas de la capa de municipios y por ultimo la ruta donde queremos guardar el archivo generado donde solo estarán las geometrías censales que nos interesan.

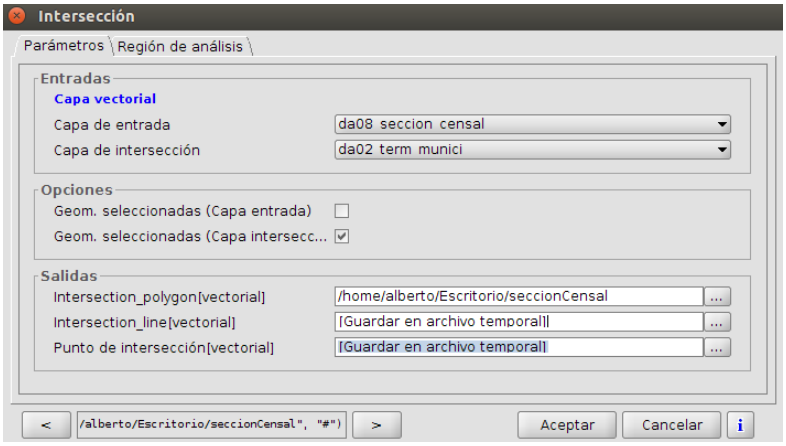


Ilustración 4.2

También habíamos descargado datos del Instituto Nacional de Estadística referentes a la población, y detallados por secciones censales, por lo tanto aprovechamos esto para ingresar en la tabla de atributos de la capa Sección Censal un campo con los valores censales a Enero de 2015 en sus respectivas secciones censales.

4.3.2. Capa de viales

Nuestra capa de viales, como comentamos anteriormente proviene de Cartociudad, el sistema de referencia que utiliza es el ETRS89, con coordenadas geográficas. Esto nos obligará a proyectar la capa de información.

Gracias a un geoproceso llamado Reproyección que se encuentra dentro de la caja de herramientas de gvSIG podremos obtener la capa final en el sistema de referencia exacto que nos interesa (Ilustración 4.3)

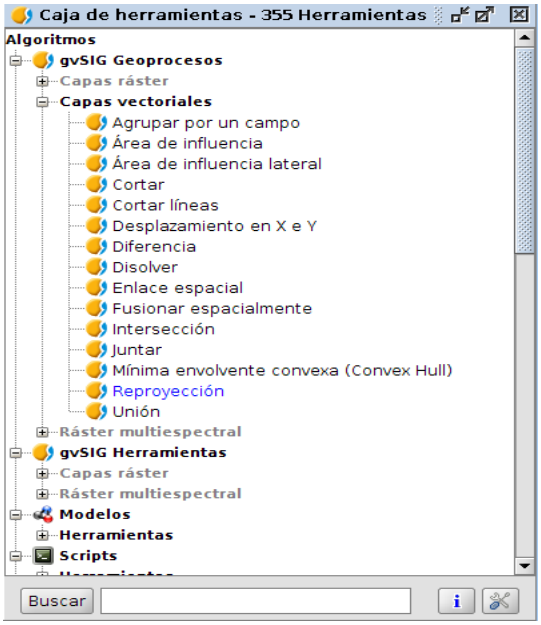


Ilustración 4.3

Como mostramos en la Ilustración 4.4 solo tenemos que seleccionar la capa que nos interese reproyectar, el sistema de referencia de destino que queremos y una ruta donde almacenar el archivo generado por el geoproceso. Con esto obtenemos la capa final en el sistema de referencia que queríamos.

4.3.3. Capa de manzanas

Como ya sabemos la información proveniente de Cartociudad, su sistema de referencia es el ETRS89 pero en coordenadas geográficas. Por lo tanto lo primero que debemos realizar es una Reproyección de los datos como explicamos anteriormente en la Ilustración 4.4 y una vez está en el sistema de referencia que precisamos podemos ejecutar una Intersección con la capa que límite el municipio de Alcalá de Guadaíra. Así obtendremos finalmente una capa de manzanas del municipio de estudio.

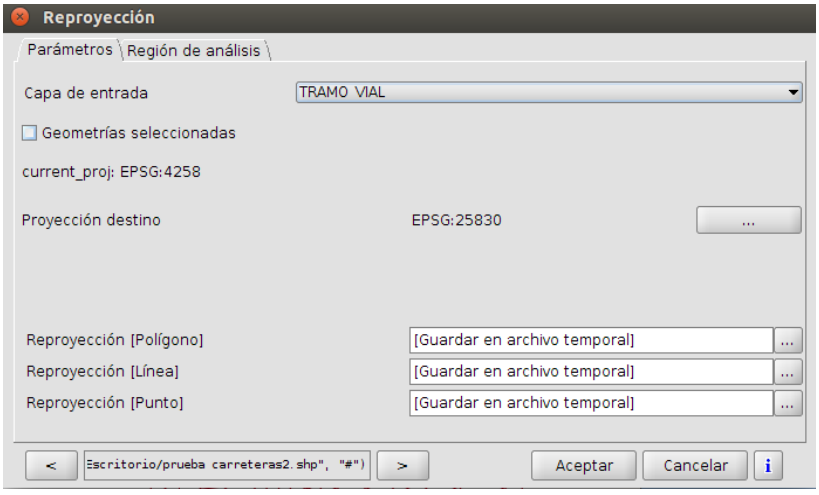


Ilustración 4.4

4.3.4. Capas Base del municipio

Las capas de Farmacias, Parques de Bomberos, Policía, Museos, Ayuntamiento, Bibliotecas, Equipo Educativo, Hospitales y Centros de Salud sufrirán exactamente el mismo proceso para homogeneizar la información. Se realizará una intersección según el limite administrativo de Alcalá de Guadaíra y una modificación/actualización de los datos geográficos según convenga en cada capa.

a) Capa de Farmacias. Para esta capa deberemos modificar la ubicación de algunas de las farmacias ya que no están bien localizadas. También deberemos añadir algunas que faltan. Las direcciones exactas de las farmacias que tuvimos que añadir las pudimos encontrar en la pagina web del ayuntamiento de Alcalá de Guadaíra.

b) Capa de Parques de Bomberos. Hay que actualizar la ubicación del parque de bomberos ya que los datos están desactualizados, la ubicación del parque de bomberos esta en el antiguo lugar donde se encontraba el parque.

c) Capa Policía. Añadimos a esta información la ubicación de la comisaria de policía local ya que solo esta ubicada en esta capa la comisaria de policía nacional. Añadiremos un campo, con nombre TIPO, en el que se definirá si la comisaria es nacional o local.

d) Capa Biblioteca. A esta capa de información tendremos que añadirle una biblioteca mas, esta biblioteca es nueva y por tanto no exite en la información descargada de DERA. También añadimos un campo NOMBRE_INS para añadir un nombre descriptivo de cada biblioteca.

e) Capa Equipo Educativo. En este caso, el Colegio de Educación Infantil y Primaria Oromana no existe en la base de datos, por lo tanto tendremos que añadirlo.

f) Capa Sanidad. Para esta capa nos encontramos que hay dos centros sanitario, uno de los centros de salud y el centro de consultas externas, que se encuentran mal ubicados. Estos dos centros tienen sus posiciones intercambiadas.

   En todas estas capas de información cuando hemos necesitado hacer levantamiento de algún detalle nuevo para agregar a nuestra base de datos lo hemos realizado digitalizando sobre una imagen raster del servicio WMS del Instituto Geográfico Nacional.

4.3.5. Capa Ferroviaria

La información ferroviaria como ya comentamos en un apartado anterior, la obtenemos de forma masiva de DERA. El sistema de referencia de estos datos es el ETRS89 con proyección UTM uso 30, por lo que no necesitaremos reproyectar la capa. Solo necesitaremos realizar una intersección entre la capa ferroviaria y la capa de limite

administrativo de Alcalá de Guadaíra.

4.3.6. Capa de Modelo Digital del Terreno (MDT)

Gracias al Instituto Geográfico Nacional (IGN) podemos disponer de información de bastante detalle respecto a los modelos digitales del terreno. Para nuestro caso vamos a descargar desde la web del IGN un MDT con un detalle (resolución) de 5 metros (Ilustración 4.5).

Ilustración 4.5

Realizaremos una búsqueda en el centro de descargas del IGN. Seleccionando el producto que queremos obtener, en nuestro caso MDT de 5 metros de detalle. Después seleccionamos Alcalá de Guadaíra como municipio de búsqueda. Esto nos reportara 5 archivos, uno en formato XML y cuatro en formato ASC ya que los datos provienen de metodología Lidar (Ilustración 4.6).

Ilustración 4.6

Descargamos todos los archivos. Como podemos leer en el archivo de metadatos, la información ya esta en el sistema de referencia que deseamos, así que no tendremos que reproyectar los datos.

Para este caso en especial, nos veremos obligados a utilizar QGIS, un software distinto a gvSIG pero con la misma filosofía, software libre. Utilizaremos este otro software porque para la unión de los cuatro ficheros es mas fácil realizar con QGIS. Cargamos los cuatro ficheros en QGIS y ahora deberemos combinarlos para solo tener un archivo de MDT. Esto lo realizaremos ejecutando Raster/Miscelánea/Combinar (Ilustración 4.7). Seleccionamos los archivos por parejas, por tanto tendremos que repetir el proceso una vez mas, obteniendo como salida dos archivos con extensión TIFF. Seleccionamos Valor de sin datos igual a cero.

Una vez generados los dos ficheros TIFF debemos combinarlos de nuevo con la misma herramienta. Obteniendo así un fichero final TIFF con la zona completa.

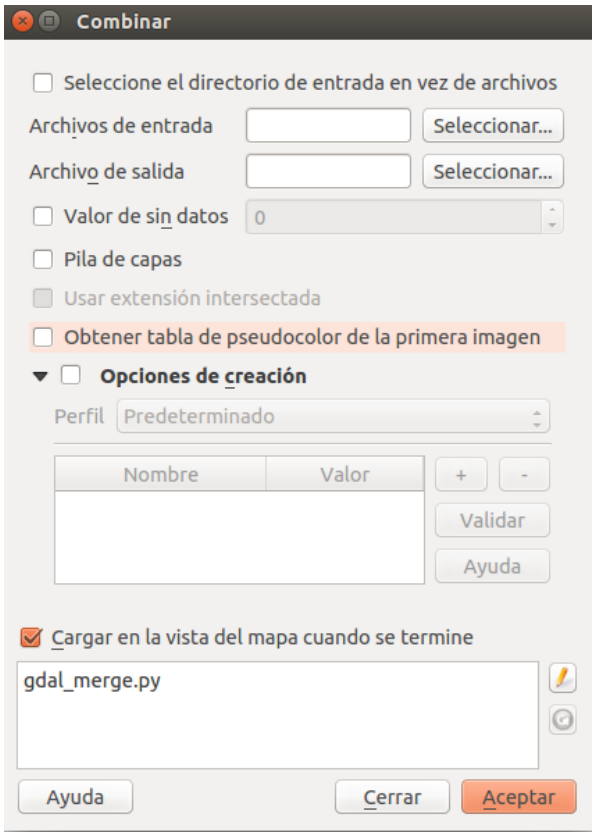


Ilustración 4.7

A continuación cargamos la capa final en gvSIG y recortamos para solo tener nuestra zona de interés, el municipio de Alcalá de Guadaíra. Ejecutamos el geoproceso Máscara por región de interés, utilizando como región de interés la capa de limite administrativo del municipio. Así llegamos a nuestra capa final de MDT de Alcalá de Guadaíra con una resolución de 5 metros.

Para ayudar a una mejor visualización del MDT podríamos hacer un mapa de sombras

o HillShade. Añadiendo este mapa de sombras de fondo y adjudicando al MDT un nivel de transparencia conseguiremos realzar el relieve del MDT.

4.3.7. Capa de ríos

Para la capa de ríos solo tendremos que ejecutar una intersección de la información descargada de DERA utilizando como capa de recorte el limite administrativo del municipio(Ilustración 4.2). Con esto ya tenemos la red hidrológica de Alcalá de Guadaíra y en el sistema de referencia ETRS89 en la proyección UTM huso 30.

4.3.8. Capa Paradas de Autobuses Urbanos

Al respecto de los servicios de transporte urbanos la información georeferenciada es inexsitente. Solo encontramos en la pagina web del ayuntamiento de Alcalá de Guadaíra un listado de las calles en las que cada una de las rutas efectua una parada. En la gran mayoría de los casos para poder ubicar bien paradas de autobús de las distintas líneas existentes es necesario un levantamiento in situ, es decir, un traslado hasta la zona donde se encuentra la parada y unicarla correctamente.

Para realizar la recolección de paradas utilizaremos una app para móvil llamada Vespucci((Vespucci). Esta aplicación utiliza la información cartográfica proveniente de OpenStreetMap por lo que tendremos que tener en cuenta el sistema de referencia que utiliza.

Para ubicar las paradas solo deberemos cerciorarnos que estamos correctamente situados sobre el punto que queremos guardar, dejamos el dedo pulsado un par de segundos sobre la pantalla del móvil en el punto correcto y nos encontraremos con el punto situado pero vacío de información. Ahora tendremos que añadirle información adicional para enriquecer el dato de posición. Esto lo haremos mediante pares de

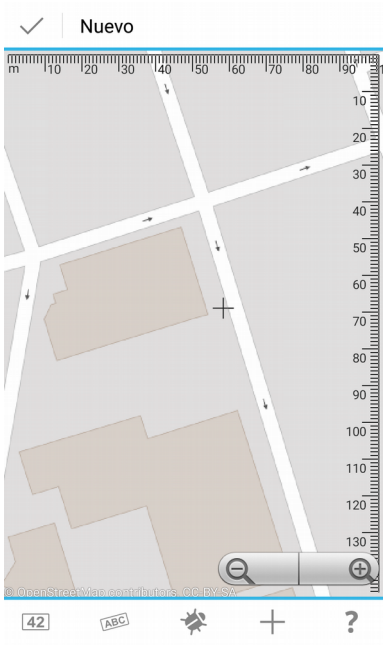


Ilustración 4.8

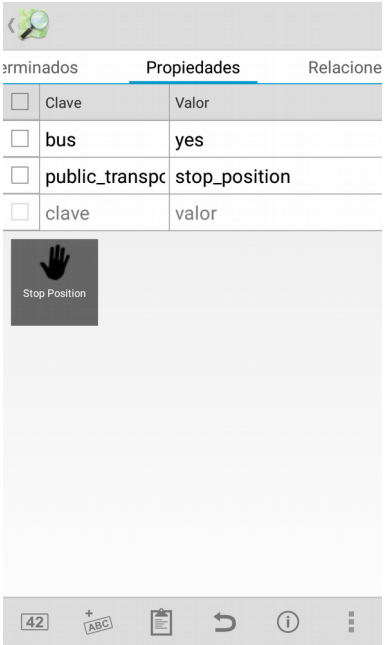


Ilustración 4.9

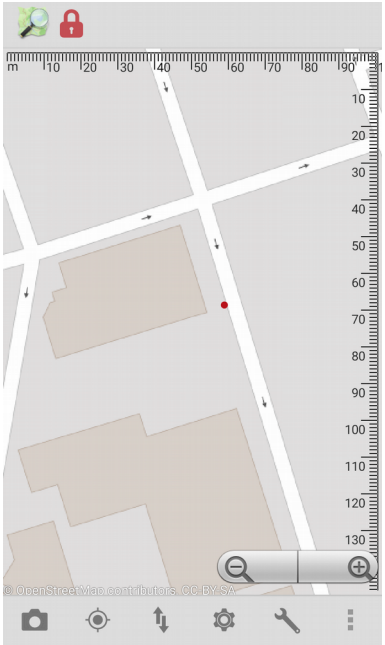


Ilustración 4.10

"clave" y "valor", para nuestro caso las dos claves utilizadas serán *public_transport y bus*, y sus valores respectivamente serán *stop_position y yes*. Esto lo repetiremos para toda las paradas de las cuatro lineas de autobuses urbanos del municipio.

Una vez que ya tenemos todos los puntos, gracias a esta app podemos obtener las coordenadas geográficas de las paradas de autobús de la localidad. Crearemos una capa de puntos nueva y añadiremos todas las paradas, completando la información en la tabla de atributos de la capa con 4 campos que llamaremos Linea A, Linea B, Linea C y Linea D. Como mostramos en la Ilustración 4.11 dándole valor 1 si pertenece la parada a la Linea o un 0 si no pertenece a esa ruta.

Tabla de atributos: ParadasBusUrbano

	Linea A	Linea B	Linea C	Linea D
1	1	0	0	0
2	1	0	0	0
3	1	0	0	0
4	1	0	0	0
5	1	0	0	0
6	1	1	1	0
7	1	1	1	0

Ilustración 4.11

4.3.9. Capa Ruta de Autobuses Urbanos

Para la capa de las rutas de los autobuses, también deberemos crear una capa vectorial totalmente nueva. Pero en este caso disponemos de una lista de calles por las que pasa cada linea.

Así, digitalizaremos las cuatro rutas existentes de autobuses urbanos apoyándonos sobre nuestra capa de viales que previamente ya teníamos terminada.

4.3.10. Capa Zonas Wi-Fi

En el caso de las zonas Wi-Fi repartidas por todo el municipio, vamos a ayudarnos de una imagen en la que se observa la localización de los mismos. Esta imagen la importaremos a gvSIG y la georeferenciamos gracias a la herramienta Transformaciones Geográficas/Georreferenciación. Utilizaremos una transformación afín con lo cual con un total de 5 puntos en la imagen a georreferenciar y la cartografía de apoyo para la georreferenciación tendremos suficiente para la precisión que necesitamos manejar.

Una vez georreferenciada, la imagen nos ayudara para utilizarla en la digitalización de las ubicaciones de las zonas Wi-Fi. Esto lo llevaremos a cabo creando una nueva capa de tipo multi-punto.

Para esta capa solo añadiremos un campo, llamado *descripción* donde introduciremos el nombre descriptivo de la zona en la que se encuentra el punto Wi-Fi.

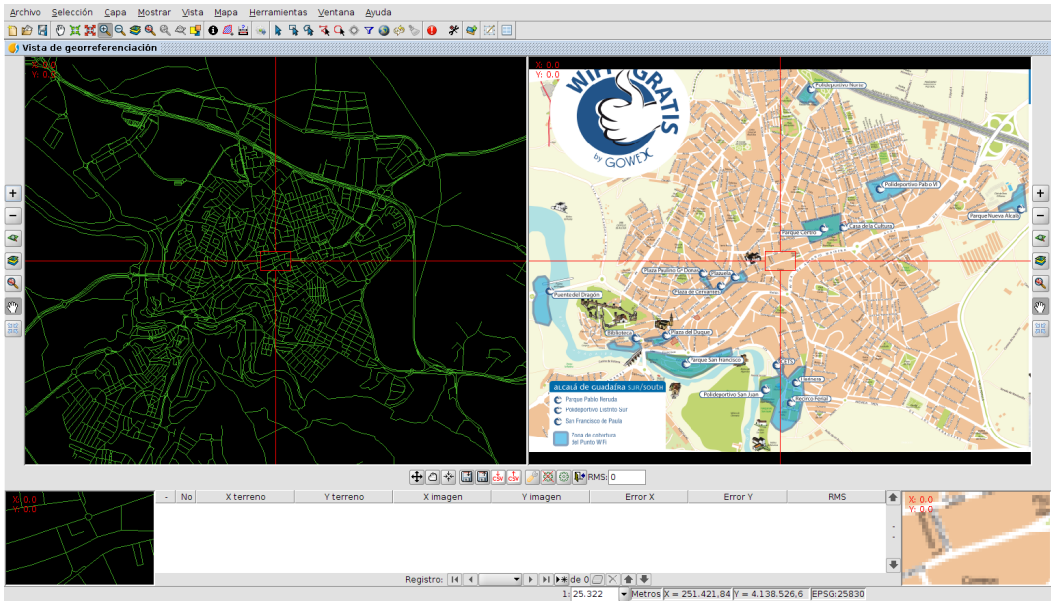


Ilustración 4.12

4.3.11. Capa Parcelas

La capa Parcela es una capa de información vectorial en formato SHP descargada de la sede electrónica de catastro. Donde existe un apartado donde podemos descargar masivamente información por municipios enteros, lo cual nos ayudara muchísimo para nuestro fin.

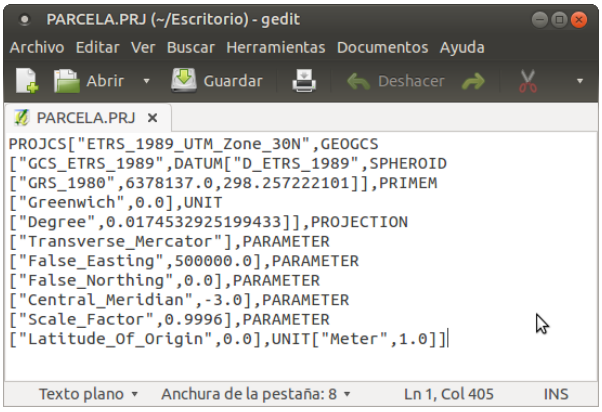


Ilustración 4.13

Una vez descargada la información catastral del municipio solo deberemos añadir a la tabla de atributos la información que nos pueda interesar para nuestro trabajo. Como ya sabemos, la información descargada esta en nuestro sistema de referencia, esto lo sabemos porque catastro ya esta trabajando en el sistema de referencia ETRS89 y porque también podemos abrir el fichero PRJ de la capa y ver la información que contiene respecto al sistema de referencia de la capa. Como observamos en la Ilustración 4.13 encontramos que el fuchero de parcelas esta en el sistema de referencia ETRS89, utilizando la proyección UTM y el Huso 30 Norte.

A la tabla de atributos de la capa le añadiremos los siguientes campos con el fin de

completar una información que podría ser valiosa para la gestión municipal.

a) NOMPROPIE y DNI. Campo que contendrá el nombre completo del dueño catastral de la parcela a la que hace referencia y el DNI de dicho dueño respectivamente.

b) DIRECCION y PROVINCIA. Dirección postal de la parcela y la provincia donde se encuentra.

c) INTECULTU. Este campo será de tipo booleano, siendo *true* en caso de ser una parcela dentro de un Bien de Interés Cultural o por el contrario será *false.*

d) FECHA_ITV. De tipo texto de tamaño 8, donde se representa la fecha en formato AAAAMMDD, donde AAAA es el año, MM es el mes y DD es el día de la realización de la Inspección Técnica de la Vivienda. Si no existiese fecha de ITV el valor de este campo sera cero.

e) MINUSVAL. También es de tipo booleano, aquí se reflejara si la parcela esta adaptada para minusválidos o si por el contrario existe alguna barrera arquitectónica parra minusválidos.

f) ASCENSOR. De nuevo un campo booleano para mostrarnos si la parcela cuenta con ascensor o no.

g) PLANTAS y VIVIENDAS. Estos dos campos son ambos enteros, que describirán el numero de plantas de que dispone la parcela y el numero de viviendas totales dentro de la parcela catastral. En el caso en el que no se tuviesen datos de plantas ni de viviendas dentro de la parcela, los valores por defecto serán 999 y 9999 respectivamente.

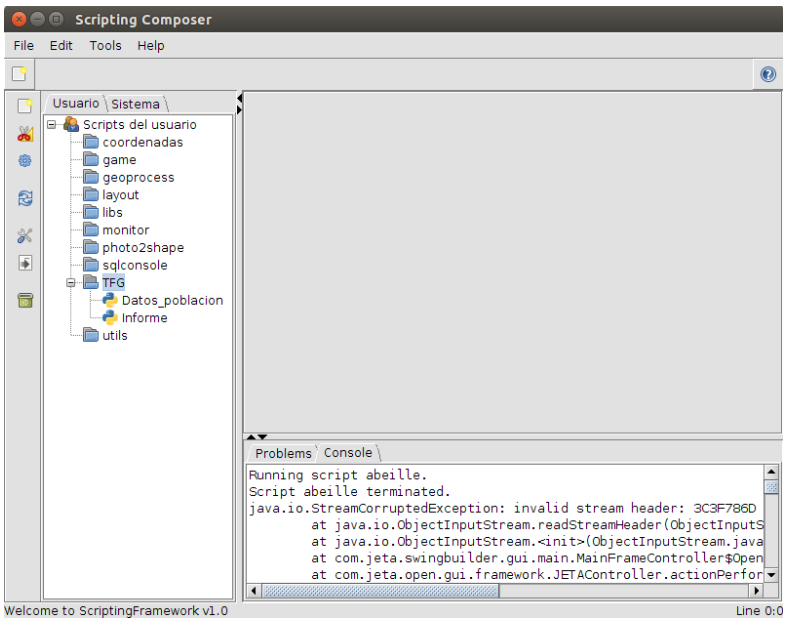


Ilustración 5.1

5. Casos de uso de la base de datos geográfica

Decidimos que con todas las capas que disponemos hasta el momento serán suficientes para tener una base de datos geográfica bastante completa del municipio de

Alcalá de Guadaíra. Una vez ya tenemos esta base de datos completa debemos seguir con los objetivos del trabajo. Nos planteamos implementar en gvSIG dos aplicaciones que servirán para mejorar la atención al ciudadano por parte de las instituciones administrativas del municipio. Para esto realizaremos en un primer lugar un script que nos detallara por pantalla información relevante a la parcela que queramos del municipio. En un segundo script esta misma información se exportara a un archivo externo para su posterior impresión si fuese necesario.

El lenguaje de programación de los script será Python, el utilizado por el software gvSIG parar crear sus añadidos. Es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis que favorece un código legible. Soporta orientación a objetos, programación imperativa y programación funcional.

En primer lugar explicaremos cómo empezar a programar en gvSIG. Para ello deberemos ejecutar Herramientas/Scripting/Scripting Composer, donde llegaremos a nuestra interfaz de programación de gvSIG (Ilustración 5.1).

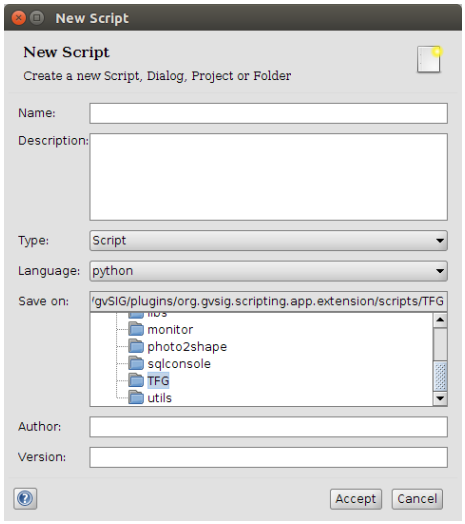


Ilustración 5.2

En Scripting Composer crearemos una carpeta que llamaremos TFG donde guardaremos los script que creemos. Para crear un script nuevo ejecutaremos File/New donde nos saldrá una interfaz (Ilustración 5.2) parar rellenar algunos datos respecto al nuevo script que vamos a crear. Aquí podremos darle nombre y seleccionar el repositorio donde queremos guardarlo. Con esto creamos un script nuevo y vacío, solo contendrá el código mostrado a continuación para obtener por consola la salida “hola mundo”:

```
from gvsig import *

def main(*args):

  print "hola mundo"
```

```
pass
```

Partiendo de este código base ya podemos entrar a describir con todo detalle qué hemos hecho en nuestros script .

5.1. Script para mostrar información de parcelas

En este script queremos facilitar la visualización de información referente a las parcelas del municipio. Información como nombre del propietario, dirección, parcela, polígono, etc e información de interés como tipo de uso, si es un bien de interés cultural, numero de viviendas, etc. Esto lo haremos ayudándonos de una interfaz que nos mostrará la información de la parcela seleccionada.

5.1.1. Creación de interfaz

Para este primer script en el que queremos mostrar por pantalla, dentro de gvSIG, una ventana que nos muestre toda la información de que disponemos de la parcela que tengamos seleccionada. En primer lugar deberemos crear la interfaz que realizara esto, nos ayudaremos de una pequeña aplicación, Abeille que nos facilitara el trabajo y desde la cual podremos exportar la interfaz a archivo XML que mas tarde podrá reconocer gvSIG en la programación.

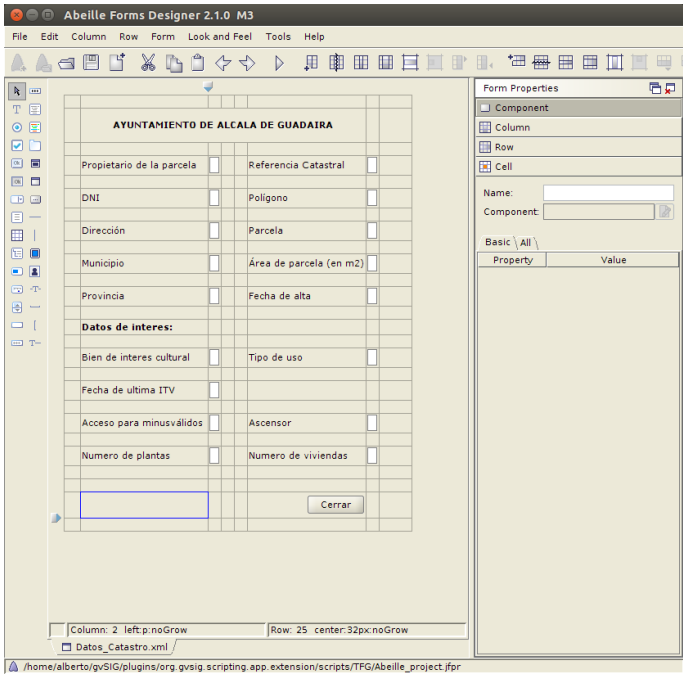


Ilustración 5.3

Abeille es muy fácil de utilizar, solo tendremos que ir colocando JLabel, para introducir etiquetas descriptivas de texto o *JTextField* si lo que queremos es poner un recuadro en el que mas tarde, mediante la programación en Python, introduciremos los valores especificos de la parcela seleccionada. Estos últimos deberemos ponerle un nombre descriptivo y único, nombre de evariable, ya que será el nombre que utilizaremos en nuestro script para acceder a ellos y añadirles contenido.

Una vez hemos añadido todo lo necesario para mostrar la información que queremos desde gvSIG para terminar añadiremos un botón, de nombre btnCerra, que nos ayudara para cerrar la ventana cuando la estemos ejecutando en gvSIG. Esta función de cerrar también la añadiremos en la programación de nuestro script.

Para finalizar ejecutamos Guardar Como, lo guardamos en formato XML y en la ruta donde estará nuestro script que accederá a él.

5.1.2. Descripción del código

Una vez creada la interfaz que nos mostrara la información, podemos decir que tenemos el esqueleto del complemento creado, solo tendremos que rellenar los huecos con información en función de la parcela seleccionada. Para esto crearemos nuestro código en lenguaje Python. Como ya explicamos anteriormente como crear un script nuevo, ahora vamos a explicar directamente como implementar nuestra idea en este lenguaje, aprovechando las librerías ya existentes en gvSIG o importando las librerías que necesitemos.

En primer lugar vamos a crear en nuestro script una función que nos ayudará posteriormente con el problema que queremos solucionar. La primera función nos ayudará a extraer la fecha bien formateada de cualquier campo que contenga una fecha, ya que sera un dato de ocho dígitos (AAAAMMDD) en formato texto del cual queremos extraer año, mes y día por separado. El código sera el siguiente:

```
def extraerFecha(fechaalta):
 #El dato de la fecha en str
 lista = []
 for n in fechaalta:
  n=int(n)
  lista.append(n)
 year = str(lista[0]) + str(lista[1]) + str(lista[2]) +
str(lista[3])
 year = int(year)
 month = str(lista[4]) + str(lista[5])
 month = int(month)
 day = str(lista[6]) + str(lista[7])
 day = int(day)
 date = datetime(year, month, day, 0,0,0)
 formato1 = '%d-%b-%Y'
 return date.strftime(formato1).
```

Esto nos devolverá la fecha formateada, un ejemplo podría ser *15-jul-2009*, donde el dato original era 20090715.

Ahora ya podemos crear nuestra clase principal, pero antes deberemos importar todas las librerías necesarias para la implementación de nuestro código.

```
from gvsig import *
```

```
from commonsdialog import msgbox
import sys
from geom import *
from java.io import File
from com.sun.jimi.core import Jimi
from org.gvsig.app import ApplicationLocator
import time
from datetime import datetime, date, timedelta

execfile( script.getResource("../libs/import_utils.py").getAbsol
utePath() )

import_from_module("../libs.formpanel","ProgressBarWithTaskStat
us,FormPanel")
```

Una vez importado todo lo necesario vamos a describir el código implementado y a continuación mostraremos el código. Creamos una clase llamada *DatosCatastro* en la que inicializamos el método *mostrarDatos* e importamos el archivo XML que representa nuestra interfaz con el usuario final.

```
class DatosCatastro(FormPanel):

 def __init__(self):

FormPanel.__init__(self,script.getResource("Datos_Catastro.xml"
))
  self.mostrarDatos()
```

A continuación creamos una función que se ejecutará cuando hagamos click en el botón de cerrar ventana que creamos antes.

```
def btnCerrar_click(self,*args):
  self.hide()
```

Ahora creamos el método *mostrarDatos* en el que encontraremos el grueso de nuestro código. Para empezar deberemos comprobar que hay alguna geometría seleccionada, si es positivo entonces seguirá avanzando en nuestro código.

Guardaremos la geometría seleccionada en una variable y realizaremos una iteración sobre ella para poder obtener todos los valores de la tabla de atributos de la aprcela seleccionada.

```
 def mostrarDatos(self):

  selection = currentLayer().getSelection()
  count = selection.getCount()

  if currentLayer() == None:
  msgbox ('Seleccione la capa de informacion', 'Error
selecionando capa',2)
```

```
  else:
   if count < 1:
    msgbox ('No hay seleccionada ninguna entidad', 'Error de
seleccion',2)
   elif count > 1:
      msgbox ('Tiene seleccionada mas de una entidad', 'Error de
seleccion',2)
   else:
    for select in selection:
     self.refCat.setText(str(select['REFCAT']))
     self.propParcela.setText(str(select['NOMPROPIE']))
     self.dni.setText(str(select['DNI']))
     self.poligono.setText(str(select['MASA']))
     self.direccion.setText(str(select['DIRECCION']))
     self.parcela.setText(str(select['PARCELA']))

     # Municipio al que pertenece la parcela
     if select['MUNICIPIO']== 4:
      municipio = 'ALCALA DE GUADAIRA'
      self.municipio.setText(municipio)

     # Área en metros cuadrados de la parcela según catastro
     area = select['AREA']
     self.areaParcela.setText(str(area))

     # Provincia en la que se encuentra la parcela
     self.provincia.setText(str(select['PROVINCIA']))

     # Fecha de alta de la parcela en Catastro
     fecha = str(select['FECHAALTA'])
     date = extraerFecha(fecha)
     self.fechaAlta.setText(date)

     # Parcela de interés cultural Si/No
     if select['INTECULTU']== True:
      self.interesCultural.setText('Si')
     elif select['INTECULTU']== False:
      self.interesCultural.setText('No')
     else:
      self.interesCultural.setText('No especificado')

     # Uso de la parcela
     uso = select['TIPO']
     if uso ==  'U':
      self.tipoUso.setText("Urbano")
     elif uso == '':
      self.tipoUso.setText("Comercial")
     else:
```

```
 self.tipoUso.setText("No especificado")

# Fecha ITV
fecha = select['FECHA_ITV']
if fecha == 0: self.itv.setText('Sin revision')
else:
 fecha = str(fecha)
 date = extraerFecha(fecha)
 self.fechaAlta.setText(date)

#Acceso para minusválidos
if select['MINUSVAL'] == True:
 self.accesoMinusvalido.setText('Si')
elif select['MINUSVAL'] == False:
 self.accesoMinusvalido.setText('No')
else:
 self.accesoMinusvalido.setText('Sin especificar')

# Ascensor
if select['ASCENSOR'] == True:
 self.ascensor.setText('Si')
elif select['ASCENSOR'] == False:
 self.ascensor.setText('No')
else:
 self.ascensor.setText('Sin especificar')

# Numero de plantas en la parcela
if select['PLANTAS'] != 999:
 self.numeroPlantas.setText(str(select['PLANTAS']))
else:
 self.numeroPlantas.setText('Sin especificar')

# Numero de viviendas en la parcela
if select['VIVIENDAS'] != 9999:
 self.numeroViviendas.setText(str(select['VIVIENDAS']))
else:
 self.numeroViviendas.setText('Sin especificar')
```

Como podemos observar, en algunos de los casos necesitaremos realizar algún procesamiento para poder seleccionar y adjudicar el valor que les corresponda correctamente. En el caso del numero de plantas que hay en la parcela, si el numero existente en la base de datos geográfica es de 999 sabremos que el valor realmente no esta especificado. Para el numero de viviendas en la parcela nos encontramos con el mismo problema, solo que en este caso el valor que nos daría como resultado “Sin especificar” es 9999.

Para los campos de minusválidos, ascensor y bien de interés cultural encontramos

que son valores booleanos, es decir o positivos o negativos. Por lo tanto debemos “transformar” true o false en un SI o un NO respectivamente.

En el caso del municipio, en nuestra base de datos solo existen datos de Alcalá de Guadaíra, por lo tanto podemos comprender que donde en el campo de municipio es igual 4 sería equivalente a “Alcalá de Guadaíra”.

Para el campo de fecha de alta en el registro de catastro utilizamos la función de extracción de fecha, para así poner un formato de fecha mas facil de leer para el usuario final.

En el uso de la parcela hemos hecho el mismo proceso que en el anterior, solo que en este caso el valor que nos encontramos en nuestra base de datos para este campo es U, donde éste valor se corresponde con urbano.

El resto de campos no necesitan de un proceso previo así que solo los extraemos de nuestra base de datos y los adjudicamos en sus lugares correspondientes en nuestra interfaz, siempre convirtiéndolos a tipo texto.

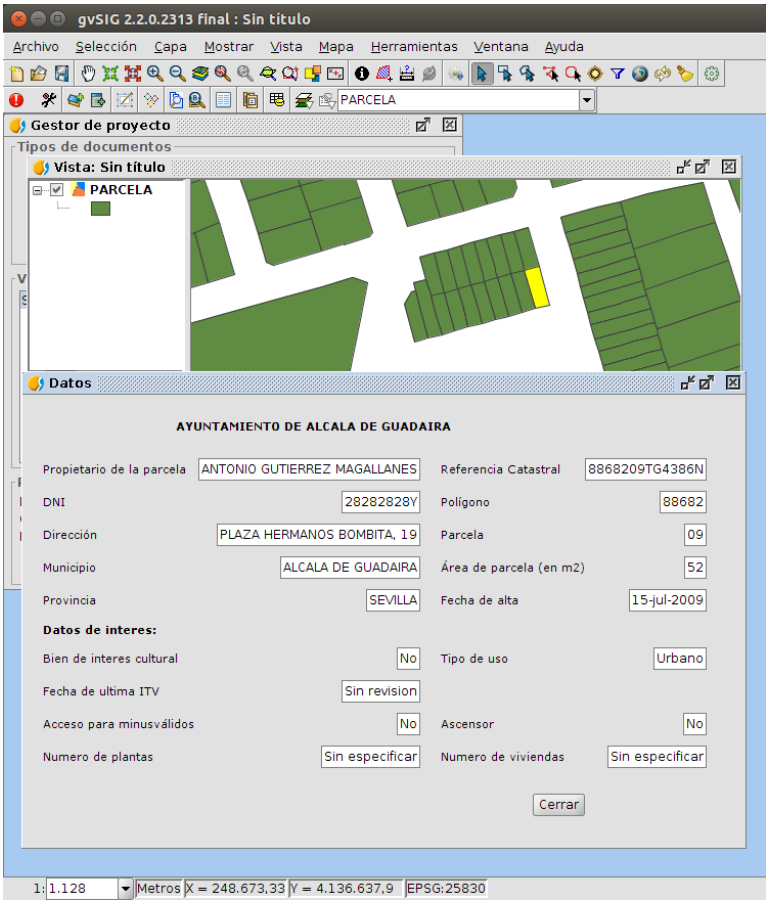


Ilustración 5.4

Para finalizar nuestro script solo nos quedaría implementar el método main que llamará a nuestra clase y método correspondientes, ejecutando todo nuestro código. Obteniendo el resultado observado en la Ilustración 5.4. Para que se ejecute nuestro

script deberá haber solo un objeto geométrico seleccionado, esto lo solucionamos con una serie de IF anidados.

```
def main(*args):
 datos = DatosCatastro()
 if currentLayer() != None:
  if currentLayer().getSelection().getCount()==1:
   datos.showTool("Datos")
```

5.1.3. Conclusión

En una primera impresión, se nos plantean diferentes mejoras o futuras versiones de nuestro código en el que nuestra interfaz no solo mostrase la información de la parcela seleccionada, sino que de alguna manera pudiésemos modificar dicha información y salvaguardarla en nuestra base de datos geográfica de una forma mas intuitiva y gráfica para el usuario final, que no tiene por qué ser un experto en sistemas de información geográfica, y al que modificar los valores de la tabla de atributos de la capa desde dicha interfaz siempre le resultará mucho mas fácil que la forma tradicional de hacerlo en este tipo de software tan especifico.

Como conclusión y analisis del script creado, cumple las expectativas que nos planteamos en un principio, mostrar por pantalla a la misma vez que ejecutamos gvSIG una serie de datos de interés para la administracion local. Con esto facilitamos la visualización de la información a alguien no especializado en este tipo de software, pudiendo acceder mas fácilmente a la base de datos geográfica.

5.2. Script para generar informe parcelario

Con este segundo script queremos conseguir exportar a un fichero externo de tipo ODS toda la información que mostrábamos antes por pantalla mediante una interfaz. De este modo lo primero que deberemos tener en cuenta son las librerías que nos harán falta y de donde debemos descargarlas. Así, para poder exportar dicho tipo de archivo necesitaremos una librería que se llama JOpenDocument (JOpenDocument) en su versión 1.3, recomendada por ser estable.

Antes que nada debemos construirnos una plantilla de hoja de calculo sobre la que exportaremos mas tarde la información de una forma especifica para preservar el formato de la plantilla con la información ya añadida.

5.2.1. Creación de plantilla para informe

Creando la plantilla no encontramos ningún tipo de restricción. Con esto queremos decir que podemos colocar los datos como mas nos guste, siempre tratando de respetar de alguna manera la estética o formato que siguen los documentos oficiales dentro del ayuntamiento de Alcalá de Guadaíra. Como mostramos en la Ilustración 7.1,

así nos quedaría finalmente la plantilla que mas tarde utilizaremos como base para todos nuestros informes parcelarios.

5.2.2. Descripción del código

Al igual que en script anterior aquí también deberemos de hacer uso de funciones auxiliares como son centerView y de nuevo extraerFecha. Como en el caso anterior extraerFecha realizara la función de extraer y formatear el dígito en formato texto que encontramos en los distintos campos de fechas (AAAAMMDD). El método centerView realizará un centrado de la vista sobre la geometría seleccionada.

```
def extraerFecha(fecha):
 #El dato de la fecha en str
 lista = []
 for n in fecha:
  n=int(n)
  lista.append(n)
 year = str(lista[0]) + str(lista[1]) + str(lista[2]) +
str(lista[3])
 year = int(year)
 month = str(lista[4]) + str(lista[5])
 month = int(month)
 day = str(lista[6]) + str(lista[7])
 day = int(day)
 date = datetime(year, month, day, 0,0,0)
 formato1 = '%d-%b-%Y'
 return date.strftime(formato1)


def centerView(geomi):
 envelope = geomi.getEnvelope()

currentView().getMapContext().getViewPort().setEnvelope(envelop
e)
 time.sleep(2)
```

Como podemos ver el código es exactamente el mismo que en el script anterior ya que buscamos en el exactamente la misma funcionalidad.

En el caso que nos ocupa ahora no hemos creado una clase para exportar el informe, si no que hemos escrito todo el código sobre el método main. Antes de empezar a explicar detenidamente el código de exportación debemos prestar atención sobre las librerías que usaremos durante todo nuestro código.

```
from gvsig import *
import sys
import time
from java.io import File
from commonsdialog import *
from geom import *
```

```
from java.io import File
from org.jopendocument.dom.template import JavaScriptFileTemplate
from org.jdom import Namespace
from com.sun.jimi.core import Jimi
from org.gvsig.app import ApplicationLocator
from datetime import datetime, date, timedelta
from org.jopendocument.model import OpenDocument
from org.jopendocument.dom.spreadsheet import SpreadSheet
from org.jopendocument.dom import OOUtils
```

Una vez importadas todas las librerías ya si podemos comenzar a describir cómo funciona nuestro código. En primer lugar debemos almacenar las direcciones de dónde se encuentra nuestra plantilla y dónde queremos almacenar nuestro informe final. También tenemos que señalar el repositorio donde querremos guardar las imágenes generadas para acompañar a nuestro informe. Estas deberán estar en la misma dirección que el fichero de salida, pero cada imagen con un nombre descriptivo sobre ella, ya que una de las imágenes (img.pgn) sera una captura de la vista de gvSIG a la escala que tengamos dicha vista cuando ejecutamos el script y la otra (envolvente.png) sera una imagen capturada igual que la anterior pero esta vez forzada a una escala de 1:10000 y centrada sobre nuestro objeto seleccionado.

```
  #Archivos
  pathTemplate =
r"/home/alberto/Documentos/Informes_gvSIG/plantilla_personal_in
formes.ods"
  pathOutput =
r"/home/alberto/Documentos/Informes_gvSIG/resultado/fillingTest
1.ods"

  #Imágenes
  pathEnvelope =
r"/home/alberto/Documentos/Informes_gvSIG/resultado/envolvente.
png"
  pathImageOut =
r"/home/alberto/Documentos/Informes_gvSIG/resultado/img.png"
```

A continuación inicializaremos una serie de variables que nos ayudaran en el manejo de la vista de gvSIG y almacenaremos en otra variable la escala de la vista en el momento de ejecutar el script. Después de esto forzaremos que la vista del documento este a escala 1:10000 , tomaremos y almacenaremos la imagen capturada bajo el nombre de envolvente en la dirección que antes estimamos oportuna. Para seguir tenemos que crear la hoja de calculo a partir de la plantilla que creamos antes y seleccionar la hoja adecuada donde queremos escribir.

```
  #Inicio
  print "Informe parcelario"
  application = ApplicationLocator.getManager()
```

```
layer = currentLayer()
docvista = currentView()
docwin = application.getDocumentWindow(docvista())

#Coger escala
gsv = currentView().getMapContext().getScaleView()

#Envolvente
currentView().getMapContext().setScaleView(10000)
time.sleep(2)
img = docwin.getMapControl().getImage()
Jimi.putImage(img,pathEnvelope)

#Acceso al libro y numero de hoja
file = File(pathTemplate)
sheet = SpreadSheet.createFromFile(file).getSheet(0)
```

Ya podemos empezar a introducir información de la parcela en nuestro informe. Vamos a empezar por establecer en el documento la fecha en la que realizamos la exportación del fichero.

```
#Establecer fecha actual a la castilla A8
fecha = datetime.today()
formatoFecha = '%d-%b-%Y'
sheet.getCellAt("B7").setValue(fecha.strftime(formatoFecha))
```

Para poder extraer la información de la base de datos geográfica tenemos que guardar la selección de la geometría e iterar gracias a un FOR, accediendo así fácilmente a toda la información de la parcela seleccionada. Así asignaremos todos los valores de los campos que nos interesen y recordando en qué celda de nuestra hoja de calculo debemos introducirlos, para que quede perfectamente sincronizado con nuestra plantilla.

```
layer = currentLayer()
for f in layer.getSelection():
  print f.REFCAT, f.geometry()
  #Centrar vista y poner escala
  geomf = f.geometry()
  centerView(geomf)
  currentView().getMapContext().setScaleView(gsv)
  time.sleep(2)

  #Imagen
  img = docwin.getMapControl().getImage()
  Jimi.putImage(img,pathImageOut)

  values = f.getValues()
  #Nombre del propietario de la parcela
  sheet.getCellAt("A14").setValue(values['NOMPROPIE'])
```

```
    #DNI del propietario de la parcela
    sheet.getCellAt("A16").setValue(values['DNI'])

    #Dirección de la parcela
    sheet.getCellAt("A18").setValue(values['DIRECCION'])

    #Municipio de la parcela
    if values['MUNICIPIO']==4:
     sheet.getCellAt("A20").setValue("ALCALA DE GUADAIRA")

    #Provincia de la parcela
    sheet.getCellAt("A22").setValue(values['PROVINCIA'])

    #Referencia catastral de la parcela
    sheet.getCellAt("A24").setValue(values['REFCAT'])

    #Polígono en el que se encuentra la parcela
    sheet.getCellAt("A26").setValue(values['MASA'])

    #Parcela dentro del polígono
    sheet.getCellAt("A28").setValue(values['PARCELA'])

    #Área en metros cuadrados de la parcela
    area = values['AREA']
    sheet.getCellAt("A30").setValue(area)

    #fecha de alta en catastro

sheet.getCellAt("B30").setValue(extraerFecha(str(f.FECHAALTA)))

    #Bien de interés cultural
    if values['INTECULTU']== True:
      sheet.getCellAt("A37").setValue('Si')
    elif values['INTECULTU']== False:
      sheet.getCellAt("A37").setValue('No')
    else:
      sheet.getCellAt("A37").setValue('No especificado')

    #Tipo de uso
    if values['TIPO'] == "U":
     sheet.getCellAt("B37").setValue(str("Urbano"))

    #Fecha ITV
    fecha = values['FECHA_ITV']
    if fecha == 0:
      sheet.getCellAt("A39").setValue("Sin especificar")
    else:
```

```
    fecha = str(fecha)
    date = extraerFecha(fecha)
    sheet.getCellAt("A39").setValue(date)

  #Ascensor
  if values['ASCENSOR'] == True:
    sheet.getCellAt("B39").setValue('Si')
  elif values['ASCENSOR'] == False:
    sheet.getCellAt("B39").setValue('No')
  else:
    sheet.getCellAt("B39").setValue('Sin especificar')

  #Numero de plantas de la parcela
  if values['PLANTAS'] != 999:
    sheet.getCellAt("A41").setValue(str(values['PLANTAS']))
  else:
    sheet.getCellAt("A41").setValue('Sin especificar')

  #Numero de viviendas en la parcela
  if values['VIVIENDAS'] != 9999:
    sheet.getCellAt("B41").setValue(str(values['VIVIENDAS']))
  else:
    sheet.getCellAt("B41").setValue('Sin especificar')

  #Acceso para minusválidos
  if values['MINUSVAL'] == True:
    sheet.getCellAt("A43").setValue('Si')
  elif values['MINUSVAL'] == False:
    sheet.getCellAt("A43").setValue('No')
  else:
    sheet.getCellAt("A43").setValue('Sin especificar')
```

Como en el script anterior, hay datos que podemos coger directamente de nuestra base de datos y hay otros que debemos procesar antes para dar una salida mas correcta o mas conveniente. En el caso de las fechas, como ya sabemos el dato de fecha en nuestra base de datos es de tipo texto con ocho dígitos. Para extraer la fecha de este texto creamos la función extraerFecha y así ponerlo en un formato mas legible.

También tenemos varios casos en los que las opciones son *true* o *false*, como son los campos Bien de Interés Cultural, Ascensor y Minusválidos, donde el valor *true* representa que SI existe esa propiedad y por el contrario si el valor es *false* no existe la propiedad.

Para el numero de viviendas y de plantas sucede algo parecido, debemos procesar los datos antes de posicionarlos en nuestro informe parcelario. Si el valor de numero de viviendas es igual a 9999, quiere decir que no está especificado el número de viviendas dentro de la parcela. En el caso del número de plantas, el valor de numero de plantas

no especificado es 999.

En el tipo de uso solo encontramos un valor al respecto en nuestra base de datos geográfica, U que corresponderá con un uso Urbano.

Por ultimo solo nos quedará guardar el fichero que estamos creando en el directorio que especificamos al principio y abrir el fichero.

```
#Guardamos fichero
outputFile = File(pathOutput)
msgbox ('Informe exportado con exito', 'Info',1)
OOUtils.open(sheet.getSpreadSheet().saveAs(outputFile))
```

Una vez lo abramos el documento podemos modificar alguna cosa si fuese necesario o añadir mas información. Con las imagenes nos encontramos un problema, al estar referenciada y no incluida dentro del documento hay que modificar la conexión entre el marco de imagen y el fichero de imagen que queremos que se vea en el informe. Para realizar esto debemos ejecutar Editar/Enlaces… y modificar el enlace al archivo de imagen PNG y seleccionar la imagen, de las dos que hemos exportado con el código, que mas convenga. Haciendo esto el informe quedará completo.

Cuando ya tengamos el documento listo gracias a LibreOffice podemos generar un fichero de tipo PDF(Ilustración 7.2) para dar al usuario final o si requiere que se imprima también lo podrá imprimir sin perder ni modificar nada del informe.

5.2.3. Conclusión

Con este código hemos conseguido lo que nos proponíamos en un principio, crear un algoritmo que nos exportase toda la información que queramos a un archivo tipo ODS. Durante la creación ya encontrar algunas mejoras a este respecto. Una mejora clara y evidente sería que la imagen que introducimos en el informe no se referenciase, si no que estuviese ya embebida en el archivo, evitando asi que haya que modificar a posteriori el archivo para que la imagen sea visible en el informe. Para encontrar una solución a este tema habría que estudiar mas a fondo las librerías de JOpenDocument y encontrar, si existe, una posibilidad de introducir directamente la imagen en el documento, sabiendo incluso la escala final de la imagen. A este respecto, en nuestro informe no podemos saber la escala final de la imagen ya que al introducirla en el informe ésta sufre deformaciones y escalado por parte de LibreOffice para adaptarla al hueco que le hemos dejado evitando asi que podamos conocer la escala real de la imagen en el informe parcelario

6. Conclusiones generales

A la vista de los objetivos que nos planteamos en nuestro inicio en este proyecto, podemos decir que estos objetivos están cubiertos de una forma bastante satisfactoria. En lo que comprende a la utilización de software libre hemos encontrado sobre todo

problemas en el aprendizaje del manejo de los software, pero estos se solventaron de manera eficiente ya que la curva de aprendizaje de estos es baja.

En el caso de la programación o scripting en gvSIG hubo algunos problemas que pudimos solucionar gracias a la gran cantidad de documentación existente en la red y a la ayuda mas especifica de Oscar Martínez (gvSIG Blog), perteneciente a la Asociación gvSIG (Asociación gvSIG). Quien nos ayudo con algunos problemas a la hora de implementar y depurar bien el código.

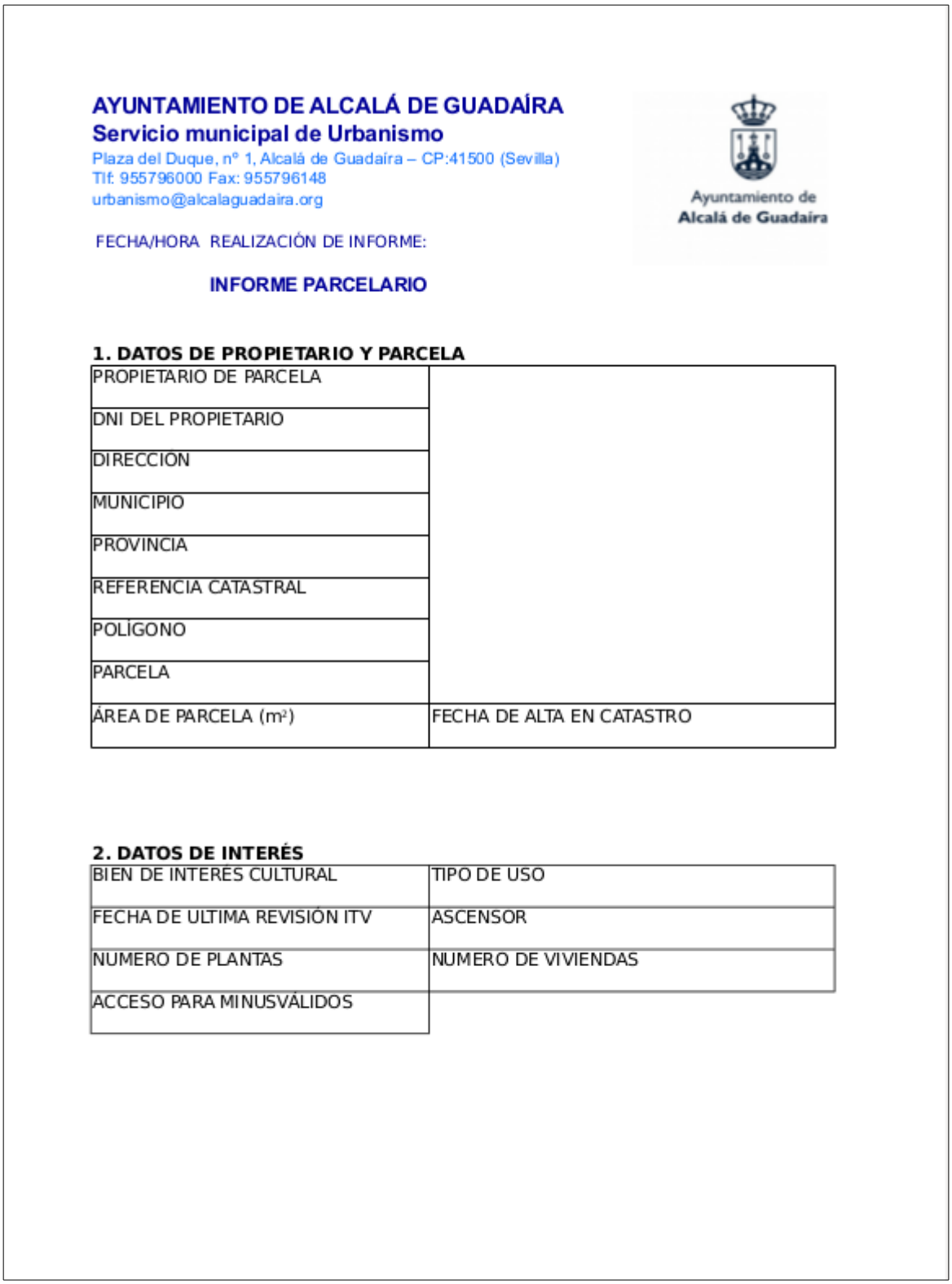
Analizando el resultado del objetivo principal de nuestro trabajo, implementar un sistema de información geográfica para la gestión territorial de un municipio y mas exactamente los script implementados en el entorno de gvSIG, podemos contrastar con un nivel óptimo que el los scripts creados ayudarian y facilitarian el trabajo en las administraciones publicas locales. Cierto es que en las infraestructuras del ayuntamiento de Alcalá de Guadaíra no cuentan con personal cualificado para manejar Sistemas de Información Geográfica de forma eficiente dando mas a relucir la necesidad de adaptarles las funcionalidades de estos software tan desconocidos, para así evitar un retraso en el uso de esta información georeferenciada o un desaprovechamiento de los datos disponible para realizar futuros análisis o donde apoyarse para tomar futuras decisiones en el ambito local.

Es relevante que en un municipio como Alcalá de Guadaíra, considerado como el tercer municipio con mas población de la provincia de Sevilla, por detrás de la capital Sevilla y la población vecina Dos Hermanas, y siendo una comunidad innovadora contando con uno de los parques empresariales mas grandes de España, no cuente prácticamente con ningún tipo de Sistema de Información Georeferenciada o alguna implementación en busca de una Ciudad Inteligente o de crear una ciudad mucho mas eficiente. Pequeños municipio como por ejemplo la localidad de Alcira en Valencia con aproximadamente 44000 habitantes tienen la consideración de SmartCity, beneficiándose de la tecnología y obteniendo una eficiencia en la utilización de los recursos del municipio mucho mayores de los que Alcalá de Guadaíra obtienen.

Finalmente podemos concluir que este trabajo ha cumplido con el objetivo principal que se planteabla en un principio y tambien ha conseguido alcanzar los objetivos secundarios o transversales que se plantearon. Demostrando así que el software libre dentro de las administraciones debería plantearse como una opción muy viable.

7. Archivos adjuntos

7.1. Plantilla de informe parcelario



AYUNTAMIENTO DE ALCALÁ DE GUADAÍRA
Servicio municipal de Urbanismo
Plaza del Duque, nº 1, Alcalá de Guadaíra – CP:41500 (Sevilla)
Tlf: 955796000 Fax: 955796148
urbanismo@alcalaguadaira.org

Ayuntamiento de Alcalá de Guadaíra

FECHA/HORA REALIZACIÓN DE INFORME:

INFORME PARCELARIO

1. DATOS DE PROPIETARIO Y PARCELA

PROPIETARIO DE PARCELA	
DNI DEL PROPIETARIO	
DIRECCIÓN	
MUNICIPIO	
PROVINCIA	
REFERENCIA CATASTRAL	
POLÍGONO	
PARCELA	
ÁREA DE PARCELA (m²)	FECHA DE ALTA EN CATASTRO

2. DATOS DE INTERÉS

BIEN DE INTERÉS CULTURAL	TIPO DE USO
FECHA DE ULTIMA REVISIÓN ITV	ASCENSOR
NUMERO DE PLANTAS	NUMERO DE VIVIENDAS
ACCESO PARA MINUSVÁLIDOS	

Ilustración 7.1

7.2. Salida de informe parcelario

AYUNTAMIENTO DE ALCALÁ DE GUADAÍRA
Servicio municipal de Urbanismo
Plaza del Duque, nº 1, Alcalá de Guadaíra – CP:41500 (Sevilla)
Tlf: 955796000 Fax: 955796148
urbanismo@alcalaguadaira.org

Ayuntamiento de Alcalá de Guadaíra

FECHA/HORA REALIZACIÓN DE INFORME: **01-jun-2016**

INFORME PARCELARIO

1. DATOS DE PROPIETARIO Y PARCELA

PROPIETARIO DE PARCELA **ANTONIO GUTIERREZ MAGALLANES**	
DNI DEL PROPIETARIO **28282828Y**	
DIRECCIÓN **PLAZA HERMANOS BOMBITA, 19**	
MUNICIPIO **ALCALA DE GUADAIRA**	
PROVINCIA **SEVILLA**	
REFERENCIA CATASTRAL **8868209TG4386N**	
POLÍGONO **88682**	
PARCELA **09**	
ÁREA DE PARCELA (m²) **52**	FECHA DE ALTA EN CATASTRO **15-jul-2009**

2. DATOS DE INTERÉS

BIEN DE INTERES CULTURAL **No**	TIPO DE USO **Urbano**
FECHA DE ULTIMA REVISIÓN ITV **Sin especificar**	ASCENSOR **No**
NUMERO DE PLANTAS **Sin especificar**	NUMERO DE VIVIENDAS **Sin especificar**
ACCESO PARA MINUSVÁLIDOS **No**	

Ilustración 7.2

8. Bibliografía

OSGeo: , Open Source Geospatial Foundatioon, , http://www.osgeo.org/

DERA: Instituto de Estadistica y Cartografia de Andalucía, Datos Espaciales de Referencia de Andalucía, ultima actualizacion 23/06/2015, http://www.juntadeandalucia.es/institutodeestadisticaycartografia/DERA/

Cartociudad: Cartociudad, , , http://www.cartociudad.es/portal/web/guest/directorio-de-servicios

IGN MDT: Instituto Geográfico Nacional, Modelo digital de elevaciones del Instituto Geográfico Nacional, , http://www.ign.es/ign/layoutIn/modeloDigitalTerreno.do

INE: , Instituto Nacional de Estadistica, , http://www.ine.es

Ayunt. Alcalá de Guadaíra: , WEB oficial del Ayuntamiento de Alcalá de Guadaíra, , http://www.ciudadalcala.org

Catastro: , Sede electronica de Catastro, , http://www.sedecatastro.gob.es

Python Software Foundation: , Python, , http://www.python.org/

Vespucci: Marcus Wolschon, Vespucci app, , https://play.google.com/store/apps/details?id=de.blau.android

JOpenDocument: , Librería JOpenDocument 1.3 , , http://jopendocument.org/downloads.html

gvSIG Blog: Oscar Martínez, gvSIG Blog, , https://blog.gvsig.org/

Asociación gvSIG: , Asociación gvSIG, , http://www.gvsig.com/es/asociacion-gvsig