-

Universidad Politécnica de Valencia

Departamento de Sistemas Informáticos y Computación

y

Departamento de Ingeniería Cartográfica, Geodesia y Fotogrametría

Tesina de Máster en

Ingeniería de Software, Métodos Formales y Sistemas de Información

Extensión de gvSIG Desktop 2.0 para detección automática de cambios a partir de una secuencia multitemporal de imágenes satélite

Valencia, junio 2012

Autor: **Sergio Izquierdo Núñez**

Directores: **Vicente Pelechano Ferragud**

Ignacio Brodín

-

Agradecimientos

A mi director de tesina Vicente Pelechano por guiarme y aconsejarme en el desarrollo de esta tesina.

A mi mentor en la empresa Prodevelop s.l. Ignacio Brodín, por todo lo que me ha enseñado sobre el desarrollo de software, por su paciencia, su apoyo y su amistad.

A mis compañeros y amigos que me han acompañado en el seminario de Teledetección, por todos los buenos momentos que hemos pasado juntos: Jose Luis, Jaime y Txomin.

A los miembros del Grupo de investigación de Cartografía GeoAmbiental y Teledetección por su apoyo y amistad: Luis Ángel, Josep, Maru, Alfonso y Jorge.

A todos mis compañeros de Máster, muy en especial a Marta González (también compañera de Geodesia y Cartografía) y a Rafael Esteller.

A todo el equipo de desarrolladores del proyecto gvSIG por su atención y sus respuestas a mis preguntas en las listas de desarrolladores.

A todas las personas que durante este tiempo se han interesado por la evolución de esta tesina.

A mis padres y mis hermanos con especial cariño.

Resumen

El seguimiento de la evolución del territorio es muy importante para la correcta gestión de los diferentes entornos, tales como agrícola, urbano, forestal, etc. Por otra parte, en los últimos años, gracias a la creciente disponibilidad de imágenes satélites comerciales con una resolución espacial muy alta y la cobertura periódica (1C/1D IRS, Ikonos, QuickBird, etc.) existe una amplia gama de aplicación y técnicas para la gestión y seguimiento en el campo de la detección de cambios en el medio ambiente.

La detección de cambios en una secuencia multi-temporal de imágenes de satélite es una de las aplicaciones más importantes de la teledetección. La detección de cambios es la identificación de diferencias en el estado de un fenómeno a través de la comparación de imágenes de diferentes observaciones en estaciones diferentes. La comparación multitemporal de imágenes se ha utilizado principalmente para la detección de cambios en la cobertura terrestre, para seguir la evolución de las áreas forestales, áreas quemadas, los desastres naturales, los recursos naturales, el crecimiento urbano, etc.

Uno de los algoritmos más útiles para la detección de cambios a partir de imágenes es algoritmo de Vector de Cambio. En este trabajo se lleva a cabo un estudio sobre el algoritmo de Vector de Cambios y el desarrollo de una extensión de gvSIG Desktop Version2.0 (software para la gestión de información geográfica con precisión cartográfica, distribuido bajo la licencia GNU GPL v2), que permita el cálculo automático de cambios a partir de una secuencia multi-temporal de imágenes Landsat, mediante la aplicación del algoritmo de Vector de Cambios.

GvSIG es conocida por tener una interfaz amigable, siendo capaz de acceder a los formatos más comunes, tanto vectoriales como raster. Cuenta con una amplia gama de herramientas para trabajar con información geográfica (herramientas de consulta, la creación de diseño, geoprocesamiento, redes, etc). GvSIG está desarrollado en lenguaje de programación Java, y está disponible para Linux, Windows y Mac OS X.

-

Abstract

Monitoring the territory is very important for the proper management of different environments, such as agricultural, urban, forest, etc..

Moreover, in recent years, thanks to the increasing availability of commercial satellite images with very high spatial resolution and the periodic coverage (1C/1D IRS, Ikonos, QuickBird, etc..) enables a wide range of application and techniques for detecting changes in environmental management and monitoring fields.

Change detection from multi-temporal sequence of satellite images is one of the most important applications in remote sensing. Change detection is the identification of differences in the state of a phenomen on through the comparison of images from different observations in diferents seasons. The multi-temporal images comparison has been used primarily for detecting changes in the land cover,to follow the evolution of forest areas, burned areas, natural disasters, natural resources, urban growth, etc.

One of the most useful algorithms for changes detection from images is Vector Change Algorythm .

This paper conducted a research study about the Vector Change Algorythm to develop a authomatic Change Detection Tool. A project extending gvSIG Desktop version2.0 (software for managing geographic information with cartographi caccuracy, distributed under the GNU GPL v2) that enables the automatic calculation from amulti-temporal sequence of Landsat images, the changes by applying the Vector Change Algorythm.

GvSIG is known for having a user-friendly interface, being able to access the most common formats, both vector and raster ones. It features a wide range of tools for working with geographic-like information (query tools, layout creation, geoprocessing, networks, etc.), which turns gvSIG into the ideal tool for users working in the land realm. GvSIG is developed using Java, and is available for Linux, Windows and Mac OS X platforms.

-

ÍNDICE DE CONTENIDO

CAPÍTULO : INTRODUCCIÓN....11
1.1. Antecedentes....12
1.1.1. Imágenes de sensores remotos en estudio medioambientales....12
1.1.2. Detección de cambios a partir de una secuencia multitemporal de imágenes satélite....13
1.1.3. Algoritmos de detección de cambios....13
1.1.4. Método de vector de cambios....14
1.1.5. gvSIG Desktop 2.0....15
1.1.6. gvSIG y Extensiones....17
CAPÍTULO II : OBJETIVOS....18
2.1. Objetivos....18
2.2. Estructura de la Tesina....19
CAPÍTULO III : DATOS....21
3.1. Imágenes multiespectrales....21
3.2. Escenarios de interés medioambiental....23
3.2.1. Incendios forestales....24
3.2.2. Deforestación y grandes transformaciones agrícolas....24
3.2.3. Zonas urbanizadas....24
3.2.4. Dinámica de zonas acuáticas....24
3.3. Pre-procesado de las imágenes....24
3.4. Selección de las zonas de estudio en los escenarios de interés medioambiental....25
3.4.1. Zona de evaluación de escenario de incendios....25
3.4.2. Zona de evaluación de escenario agrícola....26
3.4.3. Zona de evaluación de escenario urbano....28
3.4.4. Zona de evaluación de escenario húmedo....29
3.4.5. Programas informáticos empleados....30
CAPÍTULO IV : METODOLOGÍA DE DESARROLLO....32
4.1. Entorno de desarrollo....32
4.2. Maven....33
4.2.1. Introducción....33
4.2.2. Plugins de maven....34
4.2.3. Repositorios local y remotos....34
4.2.3.1. Repositorio local....34
4.2.3.2. Repositorios remotos....35
4.2.4. Objetivos habituales....35
4.2.5. Recursos de maven disponibles....36
4.3. Creación de una extensión para gvSIG....36
4.3.1. Cosas a tener en cuenta antes de desarrollar un plugin....36
4.3.2. Estructura de un proyecto en gvSIG....37
4.3.2.1. Maven y eclipse....37
4.3.2.2. Trabajar en eclipse con maven....38
4.3.2.3. Desarrollar una extensión para gvSIG....38
4.3.3. Creando nuestro proyecto....40
4.3.3.1. Cómo montar un workspace de gvSIG para Eclipse....44
4.3.3.2. Integración con eclipse....49
4.3.3.3. Preparar el workspace para trabajar con maven....49

4.3.4. Compilar un grupo de proyectos....51
4.3.5. Compilar un gvSIG completo....52
4.3.6. Arrancar gvSIG....53
4.3.7. Trabajando con un proyecto gvSIG....54
4.3.7.1. Compilar mi proyecto....54
4.3.7.2. Desplegar mi proyecto....54
4.3.7.3. Publicar mi proyecto....55
4.3.7.4. Generación de documentación e informes....55
4.3.7.5. Cómo añadir o actualizar una dependencia....56
4.3.7.6. Cómo subir una libraría al repositorio maven de gvSIG....56
CAPÍTULO V : RESULTADOS....59
5.1. Resultados obtenidos en la experimentación del algoritmo de Vector de Cambios....59
5.1.1. Análisis de vector de cambios en zona de escenario incendiado....60
5.1.2. Análisis de vector de cambios en zonas de escenario agrícola....60
5.1.3. Análisis de vector de cambios en zona de escenario urbano....63
5.1.4. Análisis de vector de cambios en zona de escenario de humedal....65
5.2. Resultado del desarrollo de la extensión....66
5.2.1. Proceso de aplicación del algoritmo....66
5.2.2. Ejemplo de aplicación....70
CAPÍTULO VI : CONCLUSIONES....73
CAPÍTULO VII : FUTUROS TRABAJOS....75
CAPÍTULO VIII : REFERENCIAS....76

-

ÍNDICE DE FIGURAS

Figura 1: Recorte de imagen Landsat en falso color infrarrojo (IR,R,G) del escenario de incendios de 1999.....25
Figura 2: Recorte de imagen Landsat en falso color infrarrojo (IR,R,G) del escenario de incendios de 2002.....26
Figura 3: Recorte de imagen Landsat en falso color infrarrojo (IR,R,G) de la zona 1 del escenario agrícola de 1999.....27
Figura 4: Recorte de imagen Landsat en falso color infrarrojo (IR,R,G) de la zona 1 del escenario agrícola de 2002.....27
Figura 5: Recorte de imagen Landsat en falso color infrarrojo (IR,R,G) de la zona 2 del escenario agrícola de 1999.....28
Figura 6: Recorte de imagen Landsat en falso color infrarrojo (IR,R,G) de la zona 2 del escenario agrícola de 2002.....28
Figura 7: Recorte de imagen Landsat en falso color infrarrojo (IR,R,G) de la zona del escenario urbano de 1999.....29
Figura 8: Recorte de imagen Landsat en falso color infrarrojo (IR,R,G) de la zona del escenario urbano de 2002.....29
Figura 9: Recorte de imagen Landsat en falso color infrarrojo (IR,R,G) de la zona del escenario húmedo de 1999.....30
Figura 10: Recorte de imagen Landsat en falso color infrarrojo (IR,R,G) de la zona del escenario húmedo de 2002.....30
Figura 11: Entrada de menú: Create a Plugin.....41
Figura 12: Menú Create project.....41
Figura 13: Menu Configure eclipse workspace.....42
Figura 14: Importar proyectos.....43
Figura 15: Diagrama de dispersión de vector de cambios en zonas incendiadas.....60
Figura 16: Diagrama de dispersión de vector de cambios en zona1 del escenario agrícola.....61
Figura 17: Diagrama de dispersión de vector de cambios en zona 2 del escenario agrícola.....62
Figura 18: Diagrama de dispersión de vector de cambios en la zona urbana.....64
Figura 19: Diagrama de dispersión de vector de cambios en la zona de humedal.....65
Figura 20: Interfaz gráfica de usuario principal.....67
Figura 21: Interfaz gráfica de usuario II.....68
Figura 22: JOptionPanel de informe estadístico.....69
Figura 23: Imagen landsat y poligonos de zona de estudio.....70
Figura 24: Mascara resultante del proceso de detección de cambios.....71
Figura 25: Shape de poligonos resultante de la vectorización.....71
Figura 26: Mascara resultante del proceso sin filtrar, resultado con filtro 3x3, resultado con filtro 5x5 y resultado con filtro 7x7, respectívamente.....72
Figura 27: Informe estadístico.....72

ÍNDICE DE TABLAS

Tabla 1: Imágenes Landsat disponibles.....23
Tabla 2: Plataformas soportadas.....51
Tabla 3: Estadísticas básicas del vector de cambios de zonas incendiadas de zonas de cambio.....60
Tabla 4: Estadísticas básicas del vector de cambios de la zona 1 agrícola de zonas de cambio.....61
Tabla 5: Estadísticas básicas del vector de cambios de la zona 2 agrícola de zonas de cambio.....63
Tabla 6: Estadísticas básicas del vector de cambios de la zona 2 agrícola de zonas de no cambio.. .63
Tabla 7: Estadísticas básicas del vector de cambios urbanos de zonas de cambio.....64
Tabla 8: Estadísticas básicas del vector de cambios urbanos de zonas de no cambio.....65
Tabla 9: Estadísticas básicas del vector de cambios en humedales de las zonas de cambio.....66

-

CAPÍTULO I

INTRODUCCIÓN

Las técnicas de **detección de cambios** permiten la identificación de diferencias en el estado de objetos o fenómenos a través del tiempo. En percepción remota esta operación involucra la capacidad de cuantificar los cambios ambientales usando datos radiométricos de imágenes multitemporales. La disponibilidad de datos periódicos obtenida con los satélites de observación de la tierra ha hecho posible que la detección de cambios ambientales sea una de las mayores aplicaciones de la percepción remota. El formato digital de este tipo de datos ha facilitado el manejo de series temporales de imágenes con ayuda del computador. En los últimos años se ha desarrollado una amplia variedad de procedimientos digitales para detección de cambios. Cabe destacar que las técnicas de **teledetección** tienen su mayor aplicación en el campo de los **Sistemas de Información Geográfica**.

Un **Sistema de Información Geográfica** (**SIG** o **GIS**, en su acrónimo inglés Geographic Information System) es una integración organizada de hardware, software y datos geográficos diseñada para capturar, almacenar, manipular, analizar y desplegar en todas sus formas la información geográficamente referenciada con el fin de **resolver problemas complejos de planificación y gestión geográfica**. También puede definirse como un modelo de una parte de la realidad referido a un sistema de coordenadas terrestre y construido para satisfacer unas necesidades concretas de información. En el sentido más estricto, es cualquier sistema de información capaz de integrar, almacenar, editar, analizar, compartir y mostrar la información geográficamente referenciada. En un sentido más genérico, los SIG son herramientas que permiten a los usuarios

crear consultas interactivas, analizar la información espacial, editar datos, mapas y presentar los resultados de todas estas operaciones.

La tecnología de los Sistemas de Información Geográfica puede ser utilizada para investigaciones científicas, la gestión de los recursos, gestión de activos, la arqueología, la evaluación del impacto ambiental, la planificación urbana, la cartografía, la sociología, la geografía histórica, el marketing, la logística por nombrar unos pocos. Por ejemplo, un SIG podría permitir a los grupos de emergencia calcular fácilmente los tiempos de respuesta en caso de un desastre natural, el SIG puede ser usado para encontrar los humedales que necesitan protección contra la contaminación, o pueden ser utilizados por una empresa para ubicar un nuevo negocio y aprovechar las ventajas de una zona de mercado con escasa competencia.

Actualmente existen numerosos software GIS de escritorio, entre los cuales gvSIG es la apuesta del gobierno valenciano (España) para ofrecer un SIG de escritorio multiplataforma, modular y potente pero de fácil uso para todo tipo de profesionales, desde expertos en cartografía a usuarios noveles que buscan un visualizador ágil. Con multitud de extensiones, dispone de herramientas de edición cartográfica, análisis raster y vectorial, creación de metadatos, publicación de cartografía en servidores de mapas, etcétera.

A pesar de que existen también en el mercado numerosas soluciones software de teledetección (como por ejemplo **Envi** o **Erdas**), con el peso del tiempo los **SIG** de escritorio desarrollan cada vez más nuevas funcionalidades en el campo de la teledetección.

1.1. Antecedentes

1.1.1. Imágenes de sensores remotos en estudio medioambientales

Desde la aparición de imágenes captadas por sensores remotos, éstas se han utilizado para diversos estudios medioambientales. Actualmente, existen varios dispositivos por satélite de captación de imágenes además de varios proyectos de adquisición de datos por sensores remotos como son el Plan Nacional de Ortofotografía Aérea (PNOA) y el Plan Nacional de Teledetección (PNT). Ambos planes permiten la adquisición gratuita de imágenes multiespectrales. El inconveniente de estos programas es que la amplitud temporal de sus imágenes es escasa si se compara con otros sensores satelitales como el Landsat, cuyo registro temporal comenzó en la década de los 70 y actualmente sigue en funcionamiento. Las imágenes Landsat son actualmente de libre adquisición y descarga a través del USGS (U.S. Geological Survey) y recientemente a través de la ESA (European Space Agency).

Para el actual proyecto se ha considerado el uso de las imágenes Landsat por las razones expuestas anteriormente, pudiendo emplearlas para el estudio de zonas concretas en varias épocas diferentes. De la misma forma, los datos Landsat han permitido una monitorización global y continua de alteraciones antrópicas y otras relativas a usos del suelo o medioambientales.

El potencial medioambiental del sensor Landsat se ha testeado en trabajos como las dinámicas de cambio en entornos forestales para el cálculo de estimaciones de deforestación tropical. Los cambios en estos entornos han sido objetivo de estudio debido a que éstos representan la mayor parte de la biomasa terrestre y contienen elevados niveles de biodiversidad. Además, Landsat permite la monitorización y seguimiento de estas zonas mediante métodos sistemáticos y objetivos (Duveiller et al., 2008; Phua et al., 2008). Así mismo, trabajos medioambientales de conservación

-

han sido desarrollados utilizando Landsat sobre áreas de humedales (McAlister y Mahaxay, 2009), consiguiendo precisiones aceptables para el trabajo.

También existen otros trabajos se han centrado en el estudio de modificaciones antrópicas del territorio como crecimientos de áreas metropolitanas de grandes ciudades. Los trabajos se desarrollaron sobre una misma escena Landsat con diferentes épocas obteniendo resultados precisos y económicos, permitiendo cuantificar y analizar los cambios de crecimiento (Yuan et al., 2005; Diermayer y Hostert, 2007).

Landsat genera un amplio abanico de posibilidades en estudios medioambientales a través de sus imágenes que, actualmente, se siguen desarrollando gracias a la monitorización global y continua de la superficie terrestre y a su amplia disposición temporal. Además, nuevos sensores Landsat se están desarrollando, lo que aportará una continuidad y estabilidad de los datos (Loveland et al., 2008).

1.1.2. Detección de cambios a partir de una secuencia multitemporal de imágenes satélite

La detección de cambios a partir de una secuencia multi-temporal de imágenes de satélite es unas de las aplicaciones más importantes en teledetección. Como es bien sabido, este proceso requiere de una adecuada corrección radiométrica y geométrica de las imágenes, de manera que los cambios detectados sólo sean atribuibles a verdaderas modificaciones del paisaje.

La comparación de imágenes multi-temporales se ha venido empleando fundamentalmente en la detección de cambios en la cubierta terrestre, para seguir la evolución de áreas forestales, superficies quemadas, desastres naturales, recursos naturales, crecimiento urbano, etc.

Las premisas básicas para usar datos de percepción remota para la detección de cambios es que: las alteraciones en el fenómeno estudiado producen cambios proporcionales en los valores numéricos de las imágenes. A la vez que un aspecto fundamental es que estos cambios son mayores que los producidos por otros factores sin importancia. Estos otros factores incluyen diferencias en las condiciones atmosféricas, ángulo de inclinación solar y humedad del suelo entre muchos otros.

El efecto de algunos de estos factores puede ser minimizado seleccionando las imágenes adecuadas. Por ejemplo, el uso de imágenes de la misma época reduce las diferencias en el ángulo de iluminación y además elimina las diferencias estacionales en áreas vegetadas.

1.1.3. Algoritmos de detección de cambios

Los métodos clásicos empleados para la detección de cambios con carácter multiespectral a nivel de píxel utilizan generalmente imágenes obtenidas por sensores satelitales o aerotransportados. En la bibliografía se encuentran muchos métodos de detección de cambios orientados a píxel. Algunos de los métodos más utilizados se comentan a continuación (Lunetta et al, 1999):

Diferencias de imágenes: Consiste en la sustracción de una segunda imagen de una época posterior sobre una imagen de una época anterior píxel por píxel. Es un método simple y de fácil interpretación. Este método requiere una selección de umbrales para la detección de los cambios. Además de esto, el método de diferencias necesita un ajuste radiométrico preciso y sus resultados no aportan información del propio cambio. No obstante, el coste de procesamiento es muy bajo permitiendo procesar grandes imágenes sin ningún tipo de inconveniente. Este método engloba los métodos de diferencias de imágenes, diferencia de ratios entre imágenes - como el método de diferencias NDVI -, etc.

Método de comparación post-clasificación: Consiste en la clasificación por separado de dos imágenes de épocas diferentes, lo que conlleva los correspondientes estudios por separado y definición de clases, selección de bandas para la correcta clasificación, etc. Entre imágenes no es necesario un buen ajuste radiométrico – si recomendable – pero en cambio, si que necesita un buen registro entre imágenes. El método se ve limitado por varios factores como es la selección de las clases y la propagación de los errores en ambas clasificaciones. Sin embargo, es el método más extendido para el mantenimiento de cartografía y bases de datos.

Clasificación multitemporal directa: Este método consiste en una clasificación de dos imágenes obtenidas en dos épocas diferentes realizando una clasificación que muestra los cambios directamente en una imagen final clasificada. Para ello hay que realizar una superposición de imágenes y una selección de clases que indiquen en cambio producido entre las dos fechas. Para este método es importante tanto un buen ajuste como una buena georreferenciación.

Análisis de componentes principales: la ventaja de los componentes principales reside en que, la transformación en componentes principales sintetiza la información en una única dimensión, reduciendo un elevado número de variables a la menor cantidad de información posible. Los nuevos componentes generados son una combinación lineal de las variables originales. De esta forma, a la hora de realizar un estudio multitemporal entre dos imágenes,

Los primeros componentes principales representan la zona invariante. Estas zonas asumen una gran correlación entre las imágenes de las diferentes épocas, siendo los píxeles menor correlados en los que se presentan los cambios.

Vector de cambios: Calcula el módulo y dirección del vector que une los valores de los píxeles en el espacio definido por las bandas espectrales en dos épocas diferentes. La magnitud, calculada como la distancia euclídea entre los valores de dos épocas diferentes de una misma variable, indica la magnitud del cambio producido, mientras que la dirección hace referencia al tipo de cambio producido. A partir de los componentes resultantes de este cálculo, es posible generar una imagen en el espacio de color HSI (Hue, Saturation, Intensity), en el que el módulo del vector resultantes es la intensidad (I) y el tono (H) se refiere a la dirección. La imagen se transforma al espacio color RGB (Red Green Blue) para facilitar su interpretación.

Los métodos expuestos son los más comunes en el proceso de detección por píxel y que actualmente se siguen utilizando en infinidad de trabajos. A parte de estos métodos, se pueden encontrar otros nuevos basados en modelos estadísticos y algoritmos genéticos. En este trabajo el método que se empleará sera el método de **Vector de Cambios**.

1.1.4. Método de vector de cambios

El método del vector de cambios consiste en el cálculo del módulo y la dirección del vector que une los valores de los píxeles, en el espacio definido por dos bandas espectrales, en dos fechas diferentes. Se utilizaron dos bandas del sensor, la banda del rojo (b3) e infrarrojo cercano (b4), de las dos imágenes ajustadas tanto geométrica como radiométricamente. El módulo hace referencia a la magnitud o importancia del cambio. Un módulo alto supone un cambio significativo. La dirección se refiere al tipo de cambio. El módulo y la dirección se obtienen aplicando píxel a píxel las expresiones:

-

$\Delta x = im2\ b3 - im1\ b3$
$\Delta y = im2\ b4 - im1\ b4$
módulo = $\Delta x^2 + \Delta y^2$
dirección = $\arctan(\Delta x / \Delta y)$

Una vez finalizado el algoritmo, se obtienen como resultado dos imágenes, una cuyo valores de pixel son los módulos y otra cuyos valores de pixel son las orientaciones. Las zonas de cambio poseen unos valores de módulo y orientación comprendidos en unos umbrales determinados. Cada tipo de cambio (incendio, agua, cultivo,...) responde a unos umbrales determinados, los cuales se han obtenido, en este trabajo, para cada escenario en la fase de investigación.

1.1.5. gvSIG Desktop 2.0

gvSIG Desktop es un programa informático para el manejo de información geográfica con precisión cartográfica que se distribuye bajo licencia GNU GPL v2.. Es un Sistema de Información Geográfica (SIG), esto es, una aplicación de escritorio diseñada para capturar, almacenar, manipular, analizar y desplegar en todas sus formas, la información geográficamente referenciada con el fin de resolver problemas complejos de planificación y gestión. Se caracteriza por disponer de una interfaz amigable, siendo capaz de acceder a los formatos más comunes, tanto vectoriales como ráster y cuenta con un amplio número de herramientas para trabajar con información de naturaleza geográfica (herramientas de consulta, creación de mapas, geoprocesamiento, redes, etc.) que lo convierten en una herramienta ideal para usuarios que trabajen con la componente territorial.

En gvSIG Desktop encontramos las herramientas propias de un completo cliente SIG de escritorio, entre otras:

- **Formatos soportados**: Puede consultar los formatos soportados en la última versión aquí.
- **Navegación**: zooms, desplazamiento, gestión de encuadres, localizador.
- **Consulta**: información, medir distancias, medir áreas, hiperenlace.
- **Selección**: por punto, por rectángulo, por polígono, por polilínea, por círculo, por área de influencia, por capa, por atributos, invertir selección, borrar selección.
- **Búsqueda**: por atributo, por coordenadas.
- **Geoprocesos**: área de influencia, recortar, disolver, juntar, envolvente convexa, intersección, diferencia, unión, enlace espacial, translación 2D, reproyección, geoprocesos Sextante.
- **Edición gráfica**: añadir capa de eventos, snapping, rejilla, flatness, pila de comandos, deshacer/rehacer, copiar, simetría, rotar, escalar, desplazar, editar vértice, polígono interno, matriz, explotar, unir, partir, autocompletar polígono, insertar punto, multipunto, línea, arco, polilínea, polígono, rectángulo, cuadrado, círculo, elipse.
- **Edición alfanumérica**: modificar estructura tabla, editar registros, calculadora de campos.
- **Servicio de catálogo y nomenclátor**.
- **Representación vectorial**: símbolo único, cantidades (densidad de puntos, intervalos, símbolos graduados, símbolos proporcionales), categorías (expresiones, valores únicos), múltiples atributos, guardar/recuperar leyenda, editor de símbolos, niveles de simbología, bibliotecas de símbolos.
- **Representación raster**: brillo, contraste, realce, transparencia por píxel, opacidad, tablas de color, gradientes.
- **Etiquetado**: etiquetado estático, etiquetado avanzado, etiquetado individual.

- **Tablas**: estadísticas, filtros, orden ascendente/descendente, enlazar, unir, mover selección, exportar, importar campos, codificación, normalización.
- **Constructor de mapas**: composición de página, inserción de elementos cartográficos (Vista, leyenda, escala, símbolo de norte, cajetín, imagen, texto, gráfico), herramientas de maquetación (alinear, agrupar/desagrupar, ordenar, enmarcar, tamaño y posición), grid, plantillas.
- **Impresión**: impresión, exportación a PDF, a Postcript, a formato de imagen.
- **Redes**: topología de red, gestor de paradas, costes de giro, camino mínimo, conectividad, árbol de recubrimiento mínimo, matriz orígenes-destinos, evento más cercano,área de servicio.
- **Raster y teledetección**: estadísticas, filtrado, histograma, rango de escalas, realce, salvar a raster, vectorización, regiones de interés, componentes generales, georreferenciación, geolocalización, clasificación supervisada, cálculo de bandas, perfiles de imagen, árboles dedecisión, componentes principales, tasselep cap, fusión de imágenes, diagramas de dispersión, mosaicos.
- **Publicación**: WMS, WFS, WCS de MapServer, WFS de Geoserver.
- **3D y animación**: Vista 3D plana y esférica, capas 3D, simbología 3D, extrusión, edición de objetos 3D, encuadres 3D, animación 2D y 3D, visualización estéreo (anaglifo, horizontal split).
- **Topología**: construcción topológica, edición topológica, generalizar, suavizar, invertir sentido de líneas, convertir capa de líneas/polígonos a puntos, convertir capa de polígonos a líneas, triangulación de Delaunay/Poligonación de Thiessen, build, clean, correcciones topológicas en modo Batch.
- **Otros**: gestión de Sistemas de Referencia Coordenados, exportar/importar WMC, scripting, gestión de traducciones.

gvSIG Desktop se caracteriza por:

- Integrar en una Vista tanto datos locales (ficheros, bases de datos) como remotos a través de estándares OGC.
- Está diseñado para ser fácilmente extensible, permitiendo una mejora continua de la aplicación, así como su uso para desarrollar soluciones a medida.
- Es software libre, con licencia GNU/GPL, lo que permite su libre uso, distribución, estudio y mejora.
- Está disponible en diversos idiomas: español, inglés UK, inglés USA, francés, alemán, italiano, portugués, portugués-brasileño, ruso, chino, serbio, swahili, turco, checo, polaco, rumano, griego, euskera, valenciano, gallego.
- Está desarrollado con Java y está disponible para plataformas Linux, Windows y Mac OS X.

-

1.1.6. gvSIG y Extensiones

gvSIG contiene un amplio abanico de herramientas en las extensiones Sextante y Remote Sensing útiles para el tratamiento de datos ráster enfocado tanto para el pre-procesado como para los propios procesos de detección de cambios. Entre estas herramientas se encuentra la georreferenciación de imágenes, calculadora ráster y los procesos de clasificación.
En general, los procesos de georreferenciación y las clasificaciones no tienen por qué funcionar de manera diferente (con menor precisión), que las herramientas de ENVI o Erdas, ya que gvSIG utiliza transformaciones afines o polinómicas al igual que los otros programas. Los métodos de clasificación supervisada son también los mismos.

CAPÍTULO II

OBJETIVOS

Este trabajo tiene su origen en la necesidad de obtener un método que permita la detección automática de cambios a partir de una secuencia multitemporal de imágenes satélite, sin necesidad de realizar un trabajo de campo. Dicha necesidad es planteada por el grupo de investigación **CGAT** de la UPV (Cartografía MedioAmbiental y Teledetección) junto con la empresa **Prodevelop s.l.**, quienes participan en el programa de investigación y desarrollo tecnológico del proyecto **ENACDES** (Environmnet Authomatic Change Detection System).

El proyecto ENACDES tiene como objetivo fundamental la investigación industrial de tecnologías de integración de fuentes de datos geoespaciales estandarizados, algoritmos de detección automática de cambios en imágenes de satélite y tecnologías que hagan uso de los mismos para la obtención de parámetros significativos para la activación y gestión de alertas en tiempo real ante el riesgo de desastres medioambientales.

2.1. Objetivos

El objetivo general de esta tesina es desarrollar una herramienta de detección automática de cambios a partir de una secuencia multitemporal de imágenes satélite. El fin es disponer de novedosos mecanismos de predicción de riesgos naturales y alertas, mediante la observación remota de información de la tierra (imágenes de teledetección) y la aplicación de nuevos algoritmos matemáticos a partir de esta información.

Para afrontar el objetivo general, el trabajo se ha dividido mediante el planeamiento de los siguientes objetivos:

-

1. Investigar en el desarrollo de una metodología específica para la detección automática de cambios en los fenómenos geográficos con el paso del tiempo a partir de la comparación de distintas fuentes de información geográfica, tales como imágenes de satélite –de distinta resolución--, modelos digitales de elevaciones, o incluso, bases de datos espaciales para las detección de riesgos medioambientales.

2. Experimentar mediante ensayos la aplicación del algoritmo de Vector de Cambios en distintos ámbitos de uso (incendios, agua, cultivo y urbanización) analizando el comportamiento en cada uno de ellos.
3. Desarrollar una extensión aplicable dentro de un sistema de información geográfica (gvSIG) de Open Source para la detección automática de cambios.

2.2. Estructura de la Tesina

Esta tesina está dividida en 8 capítulos:

En el **Capítulo I** se realiza introducción sobre el campo de la teledetección, la importancia de la detección de cambios y la forma en la que los programas SIG cada vez incorporan mas este tipo de herramientas. Posteriormente se realiza una visión completa del estado del arte de la aplicación de imágenes satélite en el estudio medioambiental, la técnica de detección de cambios, los distintos algoritmos de detección de cambios (incluyendo el algoritmos de Vector de Cambios, que será el algoritmo objeto de estudio) y una visión sobre software SIG en la cual se desarrollará la herramienta.

En el **Capítulo II** se define la motivación del estudio, se establecen los objetivos y la estructura de la Tesina.

En el **Capítulo III** describe los datos necesarios para el estudio. En el se presentan los datos de partida, así como los escenarios establecidos (incendios, agua, cultivo y urbanización) para el estudio del algoritmo de detección de cambios. El estudio consiste en distintos ensayos de un algoritmo seleccionado para la detección automática de cambios. Se trata de obtener conclusiones, a partir de experimentos, que expliquen el comportamiento de las imágenes ante el algoritmo ,dependiendo del ámbito de uso o escenario.

En el **Capítulo IV** se describe el entorno de desarrollo y se intenta explicar la metodología de trabajo a la hora de desarrollar nuevas funcionalidades para gvSIG Desktop. En este capítulo se tratan temas como la estructura de un proyecto de gvSIG, el uso de la herramienta Maven y los repositorios, etc.

En el **Capítulo V** se recogen y analizan críticamente los resultados obtenido de los ensayos realizados con el algoritmo de Vector de Cambios con la ayuda de tablas e imágenes. Estos resultados se muestran para cada escenario establecido en el capítulo III. Posteriormente se muestra el resultado final del desarrollo de la herramienta, explicando su uso con una pequeña guía de usuario con ayuda de ilustraciones.

En el **Capítulo VI** se presentan las conclusiones extraídas tanto del proceso de investigación como del proceso de desarrollo.

Los **Capítulos VII y VIII** dan a la definición de las futuras líneas de trabajo y a las referencias bibliográficas, respectivamente.

-

CAPÍTULO III

DATOS

A continuación se presentan los tipos de datos utilizados para los ensayos, los distintos escenarios o ámbitos de uso de interés medioambiental y la selección de las zonas de estudio de cada escenario, sobre las cuales se aplica el algoritmo de detección de cambios.

3.1. Imágenes multiespectrales

Los ensayos se han realizado utilizando una base de datos de imágenes multiespectrales de media resolución de los sensores Landsat TM y ETM+. Como es bien sabido ser trata de imágenes con una resolución espacial en las bandas que cubren la región óptica del espectro electromagnético de 30 m y en el caso de las ETM+ además presentan una banda pancromática con 15 m de resolución espacial.

Se han empleado imágenes de distintas escenas de la Comunidad Valenciana correspondientes a una amplia serie de años, las cuales han sido pre-procesadas geométricamente por el USGS. También se han realizado algunas pruebas sobre las imágenes suministradas por el PNT. Las primeras, como se ha comentado, son de libre adquisición por parte de cualquier usuario que se identifique en el servidor del USGS o de la ESA, y las segundas están habilitadas para ser empleadas por parte de la administración y los centros de investigación españoles tras un permiso que otorgan los servicios cartográficos oficiales en cada comunidad autónoma española (en el caso valenciano el Institut Cartogràfic Valencià). La serie norteamericana cubre con bastante detalle los periodos 1984-1987,

1990, 1999-2003, 2007-2011. La base de datos del PNT cubre con detalle el periodo 2004-2006 y 2008-2011. Técnicamente, la principal diferencia es que las imágenes tratadas por la NASA para el USGS están remuestreadas mediante el procedimiento de convolución cúbica mientras que las tratadas por EURIMAGE para el PNT español lo están por el procedimiento del vecino más próximo. El número total de imágenes disponibles para un mismo lugar disponible a partir de estas dos fuentes de datos gratuitas puede oscilar entre 40 y 100 pero con lapsos de tiempo significativamente largos –sobre todo los años 90—sin prácticamente información disponible. Para el trabajo las pruebas que se han hecho, principalmente, con imágenes Landsat TM y ETM+ procedentes del USGS. En la se adjunta las imágenes disponibles para el periodo 1984-2007 dentro del territorio valenciano. La Comunidad Valenciana queda cubierta completamente con cinco escenas Landsat (198-32; 198-33; 199-32; 199-33; 199-34).

Nº	Sensor	Zona		ID	Fecha	Nubes (%)	Calidad			Sun azimuth	Sun elevation
		Path	Row				Calidad de la adquisición*¹	Calidad de la imagen VCID1*²	Calidad de la imagen VCID 2*²		
1	Landsat 4-5 TM	198	32	LT5198032198 4146XXX02	25-05-84	10	9	–	–	121.7525368	59.8924399
2	Landsat 4-5 TM	198	32	LT5198032198 4178XXX06	26-06-84	0	9	–	–	116.6389549	60.6198903
3	Landsat 4-5 TM	198	32	LT5198032198 4274XXX04	30-09-84	0	9	–	–	146.0977155	41.0899449
4	Landsat 4-5 TM	198	32	LT5198032198 5084XXX04	25-03-85	10	7	–	–	138.3265562	43.5096227
5	Landsat 4-5 TM	198	32	LT5198032198 5100XXX04	10-04-85	0	7	–	–	135.0730264	49.3327781
6	Landsat 4-5 TM	198	32	LT5198032198 7106XXX03	16-04-87	0	9	–	–	131.4581206	50.1625620
7	Landsat 4-5 TM	198	32	LT5198032198 7122XXX02	02-05-87	0	7	–	–	127.3023254	54.8669244
8	Landsat 4-5 TM	198	32	LT5198032198 7250XXX02	07-09-87	0	9	–	–	136.3784377	47.7401339
9	Landsat 4-5 TM	198	32	LT5198032198 7266XXX02	23-09-87	0	9	–	–	142.5756891	43.1527732
10	Landsat 4-5 TM	198	32	LT5198032200 2179MTI00	28-06-02	0	9	–	–	119.0386931	61.7644672
11	Landsat 4-5 TM	198	32	LT5198032200 3182MTI01	01-07-03	0	9	–	–	119.1766658	61.6068988
12	Landsat 4-5 TM	198	32	LT5198032200 3214XXX01	02-08-03	0	9	–	–	125.7201666 7	57.43379976
13	Landsat 4-5 TM	198	32	LT5198032200 3230MTI01	18-08-03	0	9	–	–	131.7137824 7	54.16262228
14	Landsat 4-5 TM	198	32	LT5198032200 6302MPS00	29-10-06	0	9	–	–	159.6277996 8	33.84649647
15	Landsat 4-5 TM	198	32	LT5198032200 6318MPS00	14-11-06	0	9	–	–	160.9798512 3	29.21748039
16	Landsat 4-5 TM	198	32	LT5198032200 7129MPS00	09-05-07	0	9	–	–	136.6216086 8	61.01020156
17	Landsat 4-5 TM	198	32	LT5198032200 7209MPS00	28-07-07	0	9	–	–	130.1760866 9	60.86162888
18	Landsat 4-5 TM	198	32	LT5198032200 7225MPS00	13-08-07	0	9	–	–	135.6211352 3	57.56197449
1	L 7 SLC-on(1999-2003)	198	32	LE7198032199 9195EDC00	14-07-99	0	–	9	9	126.5312653	62.9383774
2	L 7 SLC-on(1999-2003)	198	32	LE7198032199 9259EDC00	16-09-99	0.01	–	9	9	148.4954529	48.0968437
3	L 7 SLC-on(1999-2003)	198	32	LE7198032200 0022EDC01	22-01-00	0.53	–	9	9	154.2490387	25.6742401
4	L 7 SLC-	198	32	LE7198032200	13-05-	0	–	9	9	134.168121	61.6679688

-

	on(1999-2003)			0134EDC00	00					3	
5	L 7 SLC-on(1999-2003)	198	32	LE7198032200 0278EDC00	04-10-00	9.23	–	9	9	153.480377 2	41.6581345
6	L 7 SLC-on(1999-2003)	198	32	LE7198032200 0326EDC00	21-11-00	0.44	–	9	9	160.050888 1	27.1296501
7	L 7 SLC-on(1999-2003)	198	32	LE7198032200 1008EDC01	08-01-01	8.42	–	9	9	155.715011 6	23.5841885
8	L 7 SLC-on(1999-2003)	198	32	LE7198032200 1072EDC00	13-03-01	6.34	–	9	9	146.197509 8	41.2731705
9	L 7 SLC-on(1999-2003)	198	32	LE7198032200 1104EDC03	14-04-01	0.03	–	9	9	140.815261 8	53.2742767
10	L 7 SLC-on(1999-2003)	198	32	LE7198032200 1152EDC00	01-06-01	19.6 4	–	9	9	127.753959 7	63.9200172
11	L 7 SLC-on(1999-2003)	198	32	LE7198032200 1168EDC00	17-06-01	23.0 1	–	9	9	124.506744 4	64.3862457
12	L 7 SLC-on(1999-2003)	198	32	LE7198032200 1280EDC00	07-10-01	20.3 4	–	9	9	153.513702 4	40.540863
13	L 7 SLC-on(1999-2003)	198	32	LE7198032200 1312EDC00	08-11-01	0.26	–	9	9	159.014892 6	30.3800755
14	L 7 SLC-on(1999-2003)	198	32	LE7198032200 2011AGS00	11-01-02	11.6 6	–	9	9	154.894088 7	23.7398262
15	L 7 SLC-on(1999-2003)	198	32	LE7198032200 2091EDC00	01-04-02	0.9	–	9	9	142.857101 4	48.330162
16	L 7 SLC-on(1999-2003)	198	32	LE7198032200 2107EDC00	17-04-02	0.47	–	9	9	139.760543 8	54.0386963
17	L 7 SLC-on(1999-2003)	198	32	LE7198032200 2139EDC00	19-05-02	0	–	9	9	131.092697 1	62.1686325
18	L 7 SLC-on(1999-2003)	198	32	LE7198032200 2187EDC00	06-07-02	20.8 7	–	9	9	123.720756 5	63.0892334
19	L 7 SLC-on(1999-2003)	198	32	LE7198032200 2299EDC00	26-10-02	0.02	–	9	9	157.324996 9	34.3327103
20	L 7 SLC-on(1999-2003)	198	32	LE7198032200 3014EDC00	14-01-03	0.64	–	9	9	154.401702 9	24.0603809
21	L 7 SLC-on(1999-2003)	198	32	LE7198032200 3142ASN00Índice de figuras	22-05-03	24.3 8	–	9	9	130.244186 4	62.6043053

Tabla 1: Imágenes Landsat disponibles.

3.2. Escenarios de interés medioambiental

Las aplicaciones de la primera fase del proyecto se orientan a la detección de cambios en tres tipos genéricos de coberturas: vegetación, áreas urbanas y zonas acuáticas, por lo que se plantean cuatro grupos de aplicaciones:

3.2.1. Incendios forestales

Como es bien sabido es uno de los principales riesgos de carácter medioambiental que afecta a la cuenca mediterránea española debido a sus condiciones climáticas, el tipo de estructura de las cubiertas forestales y la dinámica social. Su detección, obviamente plantea una disminución brusca en la cubierta vegetal viva y una señal oscura en el terreno. Dada la frecuencia de imágenes Landsat disponibles está herramienta tiene una aplicabilidad directa y permitirá cartografiar y cuantificar la superficie de las áreas incendiadas.

3.2.2. Deforestación y grandes transformaciones agrícolas

Es un tipo de cambio que también se manifiesta en la desaparición brusca de biomasa pero, con una clara diferencia respecto a los espacios quemados dado que aquí suele haber una señal de reflectancia muy alta. Es interesante para poder determinar alteraciones significativas en las masas forestales o en grandes estructuras agrarias (ampliación o desaparición de manchas de cítricos, etc.).

3.2.3. Zonas urbanizadas

Tras el proceso de deforestación muchas veces se ha verificado un proceso de urbanización de los espacios. Las áreas urbanas -con una elevada complejidad interna y entre las distintas modalidades de urbanización- sin embargo presentan algunos rasgos en su respuesta espectral que permite hacer un buen acercamiento a su detección. Este tipo de análisis puede ser especialmente interesante para cartografiar la profunda alteración paisajística y medioambiental habida durante los últimos quince años.

3.2.4. Dinámica de zonas acuáticas

La localización de las áreas con agua resulta especialmente sencilla siempre que se disponga señal en la banda del infrarrojo próximo o medio, como es el caso de las imágenes Landsat. Por ello este tipo de análisis es uno de los que han de introducir. Su aplicación podrá ayudar a detectar cambios –con las restricciones asociadas al nivel escalar de la dimensión del pixel-en la grandes acumulaciones de agua como los embalses o lagos naturales o incluso los grandes cambios en las áreas de playa (si bien estos suelen presentar dimensiones inferiores al tamaño de pixel del Landsat)-. Dada la elevada cantidad de imágenes disponibles también sería posible utilizarlas para detectar y cartografiar áreas inundadas.

3.3. Pre-procesado de las imágenes

Debido al pre-procesado que reciben las Ladnsat, no se ha considerado necesaria la aplicación de una nueva georreferenciación pero si un ajuste radiométrico entre las escenas de las diferentes épocas. Las épocas seleccionadas para el ensayo son las escenas del sensor Landsat 7, zona 199-33

-

(sur de la provincia de Valencia) correspondientes a los años 1999, 2000, 2001 y 2002.

El ajuste radiométrico realizado a las imágenes se ha basado en el método de ajuste por regiones pseudo-invariantes. Este consiste en la selección de regiones comunes e invariantes sobre las imágenes de las diferentes épocas. Una vez realizada la selección, se modifican los valores de los niveles digitales (ND) banda a banda con el ajuste de media y desviación calculado a partir de las regiones seleccionas. Este proceso es relativo, por lo que los ajustes se realizan con respecto a una imagen que se toma como referencia.
Una vez aplicado al ajuste a las imágenes, el siguiente paso a seguir fue el de la aplicación de los métodos de cambios a las imágenes.

3.4. Selección de las zonas de estudio en los escenarios de interés medioambiental

Los estudios se centraron en zonas concretas donde se encontraban los diferentes escenarios de estudio para el proyecto. De esta forma en los resultados obtenidos se puede encontrar la relación existente entre aquellas zonas variantes en el tiempo y aquellas que no han cambiado.
En los siguientes apartados se muestran las que se corresponden con las zonas de evaluación seleccionadas y sus características dentro de los escenarios del proyecto.

3.4.1. Zona de evaluación de escenario de incendios

En la figura 1 y en la figura 2 se muestra la zona seleccionada para el estudio del comportamiento del escenario de incendios. La figura 1es una composición en falso color infrarrojo, con el que se resalta la vegetación, en el año 1999. En la figura 2 se puede ver la misma zona de estudio en el año 2002 tras un incendio forestal en el que apenas ha comenzado a crecer la vegetación de nuevo.

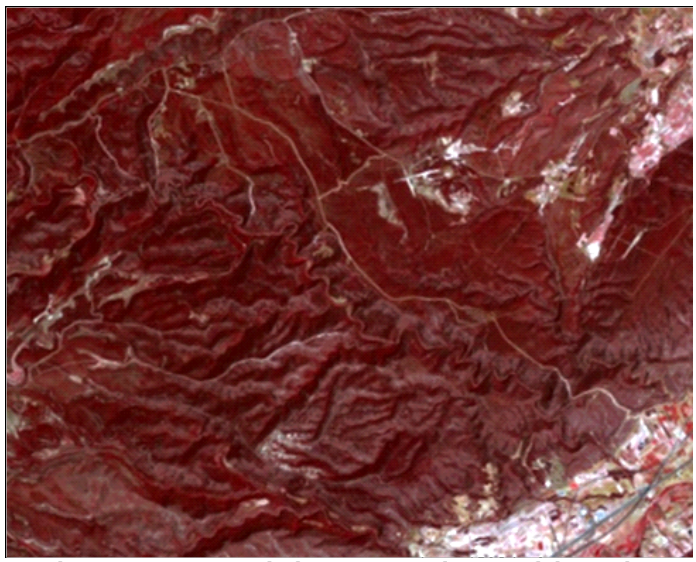


Figura 1: Recorte de imagen Landsat en falso color infrarrojo (IR,R,G) del escenario de incendios de 1999.

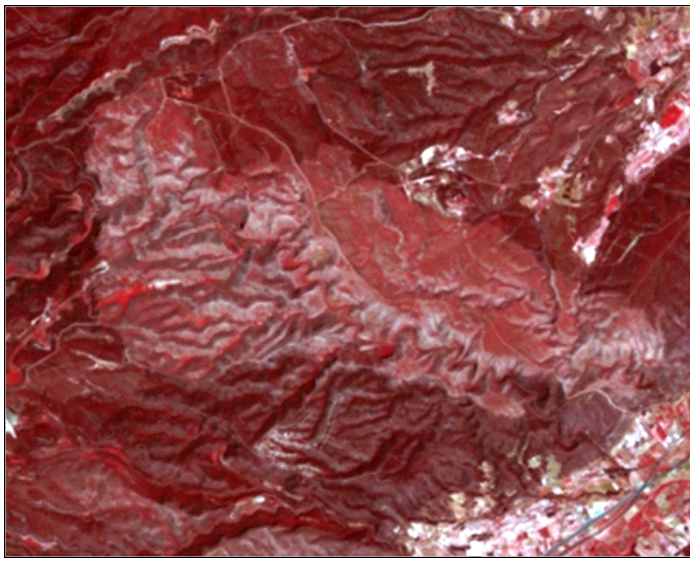


Figura 2: Recorte de imagen Landsat en falso color infrarrojo (IR,R,G) del escenario de incendios de 2002.

Estas zonas se caracterizan por tener un cambio brusco en los valores de vegetación fácilmente identificable a simple vista pero de difícil selección debido a que pueden encontrarse zonas en las que se produzca este decrecimiento de las vegetación de manera antrópica, esto es, una posible tala forestal o la construcción de urbanizaciones en zonas montañosas.

3.4.2. Zona de evaluación de escenario agrícola

Para este escenario se han considerado dos tipos de zonas con diferentes características. La primera zona se muestra en la figura 3, correspondiente a los cultivos situados en la zona norte de la ciudad de Valencia. Esta zona presenta cierta dificultad debido a que la resolución espacial Landsat (30 metros/píxel) es superior al tamaño de las parcelas de estudio. Esto implica que, en un mismo píxel están representadas varias parcelas con diferentes características, lo que dificulta la detección de los cambios a esta escala determinada. En la figura 4 se ve la misma zona de estudio correspondiente a la época de 2002.

-

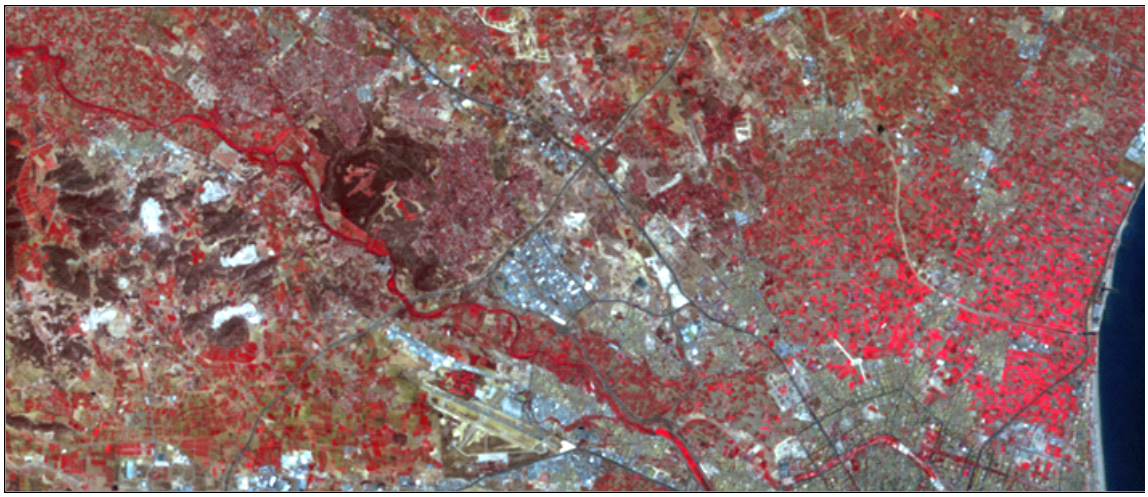


Figura 3: Recorte de imagen Landsat en falso color infrarrojo (IR,R,G) de la zona 1 del escenario agrícola de 1999.

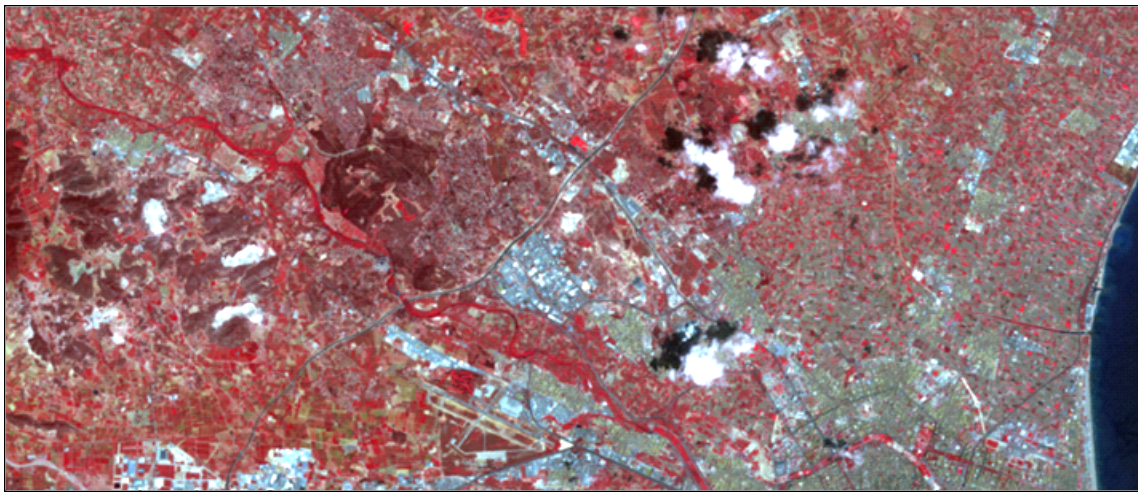


Figura 4: Recorte de imagen Landsat en falso color infrarrojo (IR,R,G) de la zona 1 del escenario agrícola de 2002.

La segunda zona de estudio agrícola (figura 5) se corresponde con un área de cultivos de gran extensión, lo que facilita la detección de los cambios debido a que una parcela con una misma clase de cultivo se ve representada por varios píxeles. En la figura 6 se observa la misma zona agrícola de la época 2002.

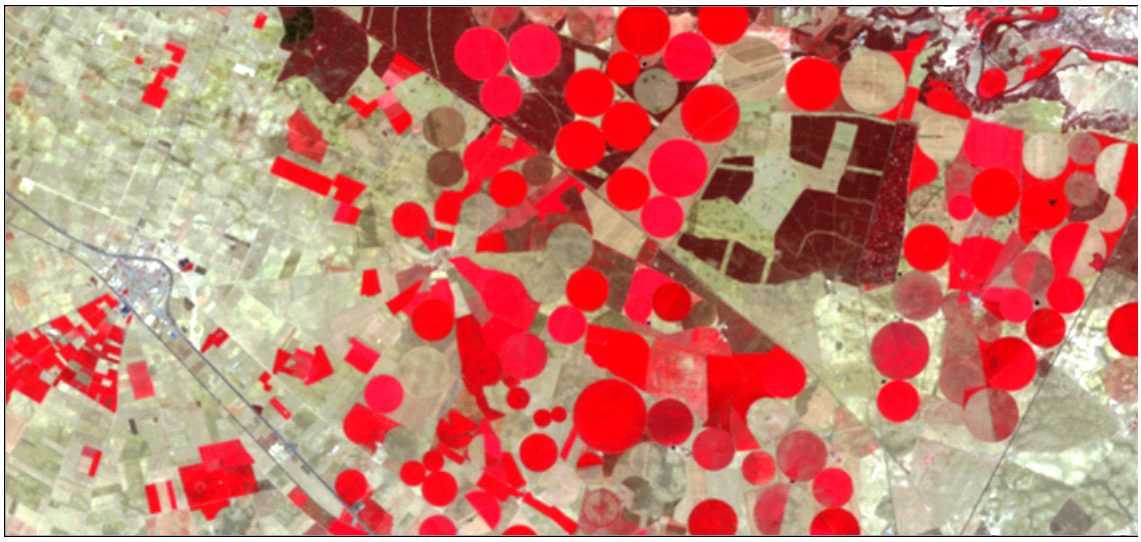


Figura 5: Recorte de imagen Landsat en falso color infrarrojo (IR,R,G) de la zona 2 del escenario agrícola de 1999.

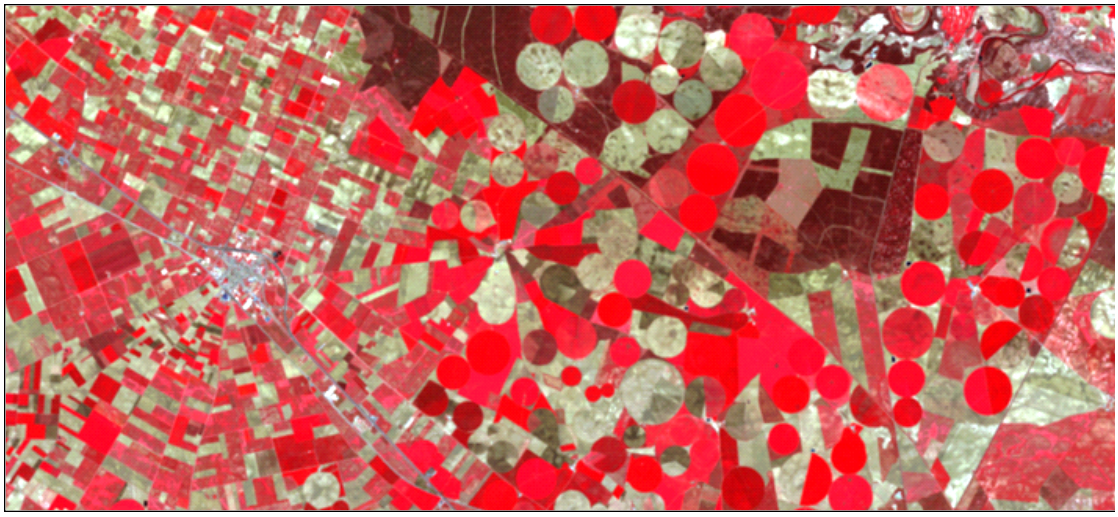


Figura 6: Recorte de imagen Landsat en falso color infrarrojo (IR,R,G) de la zona 2 del escenario agrícola de 2002.

3.4.3. Zona de evaluación de escenario urbano

La zona urbana seleccionada es la zona de la ciudad de Valencia. En esta se plantea el mismo problema que con la zona 1 del escenario agrícola (figura 7), ya que varios componentes urbanos (edificios, calles, etc.) se sitúan en un mismo píxel, por lo que los cambios singulares como puede ser un único edificio, o un cambio de una parcela agrícola a edificio en el contorno de la ciudad, es detectado con dificultad. La Figura 7 y la Figura 8 se centran en la zona urbana de estudio escogida en las épocas 1999 y 2002 respectivamente.

-

Figura 7: Recorte de imagen Landsat en falso color infrarrojo (IR,R,G) de la zona del escenario urbano de 1999.

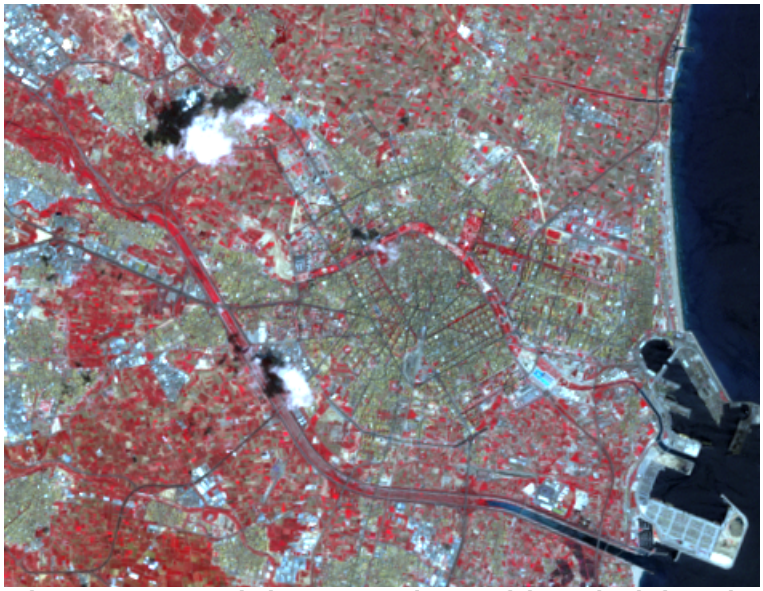


Figura 8: Recorte de imagen Landsat en falso color infrarrojo (IR,R,G) de la zona del escenario urbano de 2002.

3.4.4. Zona de evaluación de escenario húmedo

Como última zona de estudio, se ha escogido una pequeña área de la Albufera de Valencia, en la que se pueden encontrar anualmente cambios notables debido a sus características naturales y a su explotación agrícola.

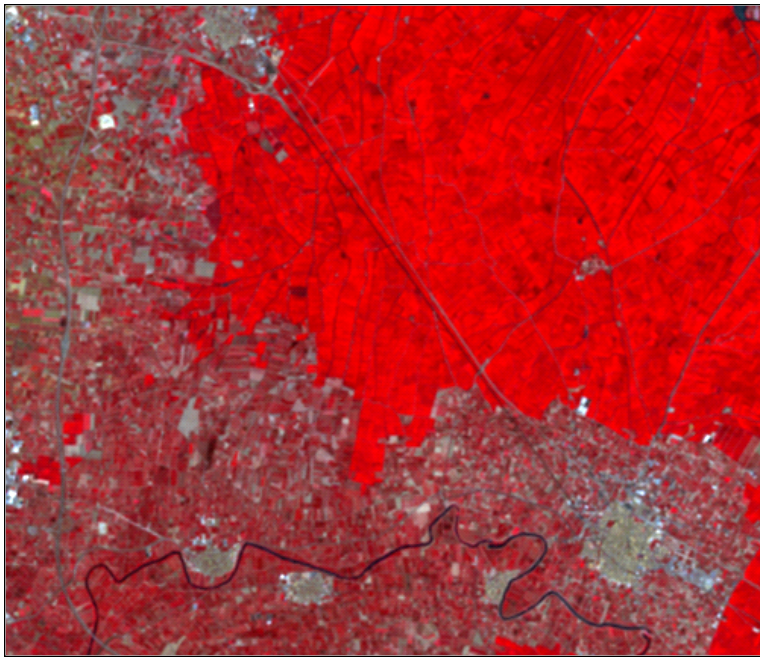


Figura 9: Recorte de imagen Landsat en falso color infrarrojo (IR,R,G) de la zona del escenario húmedo de 1999.

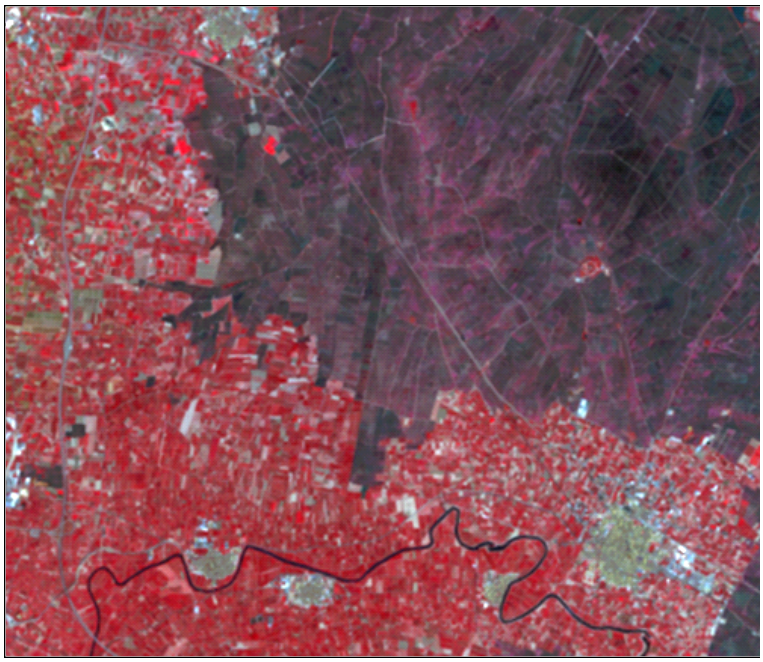


Figura 10: Recorte de imagen Landsat en falso color infrarrojo (IR,R,G) de la zona del escenario húmedo de 2002.

3.4.5. Programas informáticos empleados

La programación específica de los métodos de detección de cambios y cualquier procesado digital

-

de las imágenes para la fase de preparación de datos y ensayos se ha realizado a través de desarrollos propios de software, combinando el lenguaje de programación IDL (Interactive Data Languaje) con ENVI. Ambos son propiedad de la compañía ITT Visual Information Solutions.

La manipulación de ficheros vectoriales shapefile y el proceso de edición y maquetación cartográfica se ha realizado con el mismo gvSIG Desktop.

CAPÍTULO IV

METODOLOGÍA DE DESARROLLO

En esta sección se describe el entorno de desarrollo utilizado y se intenta explicar la metodología de trabajo a la hora de desarrollar nuevas funcionalidades para gvSIG Desktop 2.0. En este capítulo se tratan temas como la estructura de un proyecto de gvSIG Desktop 2.0, el uso de la herramienta Maven y los repositorios, etc.

4.1. Entorno de desarrollo

Para el desarrollo de la extensión de gvSIG Desktop para la detección de cambios se utilizó la IDE **Eclipse** en su versión **Indigo**, instalada en una máquina con sistema operativo **Ubuntu 12.04 LTS 32 bit** y utilizando una **jdk 1.5** (es necerario un versión de jdk 1.5 o superior para desarrollar e gvSIG). El desarrollo se realizó sobre un versión en desarrollo de gvSIG Desktop 2.0 (v 2.0.0-2047), la cual se puede encontrar y descargar de forma libre en el repositorio remoto de desarrollo http://devel.gvsig.org/svn/gvsig-desktop/.

Es importante destacar que se ha hecho uso de la herramienta Maven, para la gestión y construcción de proyectos software (de la cual se hablará mas adelante).

Otra herramienta utilizada es **Apache Ant**. Ant es una herramienta usada en programación para la realización de tareas mecánicas y repetitivas, normalmente durante la fase de compilación y construcción (build).

Para el control de versiones se ha hecho uso de **Subversion.** Subversion es un sistema de control de

-

versiones diseñado específicamente para reemplazar al popular CVS. Es software libre bajo una licencia de tipo Apache/BSD y se le conoce también como *svn* por ser el nombre de la herramienta utilizada en la línea de comando.

Una característica importante de Subversion es que, a diferencia de CVS, los demás archivos con versionamiento no tienen cada uno un número de revisión independiente, en cambio, todo el repositorio tiene un único número de versión que identifica un estado común de todos los archivos del repositorio en un instante determinado del repositorio que se esta trabajando.

4.2. Maven

4.2.1. Introducción

Maven es una herramienta para la gestión y construcción de proyectos software, sobretodo para aquellos basados en lenguaje Java. Se trata de una herramienta similar, en cuanto a su objetivo, a *make* y *ant*, aunque con una orientación mayor a la configuración por convenio.
Actualmente es un proyecto de nivel superior de la Apache Software Foundation, y lo podemos encontrar en la dirección: http://maven.apache.org.
Los objetivos principales de maven son:

- Facilitar el proceso de compilación y construcción del proyecto.
- Generar documentación de calidad sobre el proyecto.
- Proporcionar guías de desarrollo basadas en buenas prácticas.
- Facilitar la migración transparente a nuevas funcionalidades de maven.

La configuración de un proyecto con maven se realiza a través del archivo *pom.xml*, en el que se describen las propiedades principales del proyecto, sus dependencias con otros módulos y componentes externos, así como la configuración de maven a aplicar sobre él mismo. En la web de maven tenemos una guía de referencia del formato del archivo pom.xml
Siguiendo la filosofía de configuración por convención, maven define una estructura de proyecto por defecto de maven:

```
project
|-- pom.xml
`-- src
    |-- main
    |   |-- java
    |   `-- resources
    |
    `-- test
        |-- java
        `-- resources
```

Si seguimos esta estructura (aunque si es distinta, se puede configurar), con una configuración básica podremos realizar las tareas típicas de cualquier proyecto Java, ya predefinidas en maven:

- Compilación de código.
- Empaquetado en archivos *.jar*, *.war* ó *.ear*.
- Generación de javadoc y otros informes.
- Ejecución de tests unitarios.

- etc.

Cada tarea viene determinada por un *objetivo* de maven, cuyo nombre se le pasa como parámetro. Ej:

```
mvn install
```

Existen dos versiones principales de maven, la 1.0.* y la 2.*. Esta última es la que se utiliza para desarrollo en gvSIG.

4.2.2. Plugins de maven

La funcionalidad que proporciona maven está compuesta por una serie de plugins que podemos registrar y configurar a través del archivo *pom.xml* para cada proyecto.

Los más usados habitualmente están registrados y configurados por defecto. Si queremos usar alguno más a nuestro proyecto, o bien adaptar la configuración de alguno de los ya existentes, podemos acceder a la página de plugins de maven.

Para cada plugin encontraremos una guía de uso y documentación sobre los distintos parámetros que acepta cada uno de ellos.

En la página de plugins de maven podremos encontrar también otros repositorios de plugins disponibles, siendo los principales los existentes en *codehaus.org* y *code.google.com*.

4.2.3. Repositorios local y remotos

Maven gestiona dos tipos de repositorios de depedencias:

4.2.3.1. Repositorio local

Ubicación en disco local donde maven guarda las dependencias que se va descargando, para no tener que descargarlas cada vez. La estructura se corresponde con una carpeta por *groupId* (si éste tiene puntos, se creará una subcarpeta por cada elemento), una carpeta con el nombre del *artifactId* y, finalmente, una carpeta con la versión.

Dentro de la carpeta de la versión ya se encuentran los archivos de las dependencias descargadas. Por ejemplo, la siguiente dependencia en un *pom.xml*:

```
<dependency>
    <groupId>org.gvsig</groupId>
        <artifactId>org.gvsig.tools</artifactId>
        <version>2.0-SNAPSHOT</version>
</dependency>
```

Estará ubicada, dentro de nuestro repositorio local de maven, en el path:

```
MAVEN_REPO/org/gvsig/org.gvsig.tools/2.0-SNAPSHOT/org.gvsig.tools-2.0-
SNAPSHOT.jar
```

Los archivos que maven descarga en el repositorio local pueden ser de tipo:

- pom.xml: con la configuración y las dependencias de la propia dependencia. Así maven, de forma transitiva, se descargará todo el árbol de dependencias que necesitemos en un

-

momento dado, sin tener que especificar las dependencias de una dependencia de nuestro proyecto.
- archivo .jar con clases compiladas.
- archivo .jar con el código fuente.
- archivo .jar con el javadoc.
- etc.

Cuando especificamos una dependencia en nuestro proyecto, maven se descarga por defecto sólo el archivo *.jar* con las clases compiladas. Podemos indicarle que se descargue también el *jar* con los fuentes y/o con el javadoc.

4.2.3.2. Repositorios remotos

Los repositorios remotos son servidores de dependencias a los que maven va a buscar cuando no encuentra una dependencia en nuestro repositorio local.

Se puede acceder a estos repositorios a través de protocolos habituales como: HTTP/S, FTP, SSH, etc.

El propio proyecto maven tiene un repositorio de dependencias, accesible por HTTP, que viene configurado por defecto. En nuestros archivos *pom.xml* podemos añadir otros repositorios externos, o bien de nuestro propio proyecto. Por ejemplo, en gvSIG se ha creado un repositorio propio, que se registra por defecto en los proyecto de gvSIG, además de los repositorios oficiales.

El repositorio de gvSIG está accesible actualmente por HTTP en modo lectura, y por SCP en modo escritura. En este repositorio se pueden encontrar principalmente los archivos generados por los propios proyectos de gvSIG, así como alguna dependencia externa que no está disponible en el repositorio oficial.

4.2.4. Objetivos habituales

En el Maven Getting Started Guide tenemos un resumen con las opciones que se suelen emplear habitualmente, entre las que destacamos:

- Compilar un proyecto:

  ```
  mvn compile
  ```

- Generar los artefactos (habitualmente archivos jar) e instalarlos en el repositorio local de maven para que estén disponibles para el resto de proyectos. Esto implica a su vez la compilación, generación de archivos .jar, lanzar los tests

  ```
  mvn install
  ```

- Subir un jar (o jars) generado al repositorio de maven, para publicar un binario para el resto de proyectos:

  ```
  mvn deploy
  ```

  Esta opción la usaremos cuando hayamos terminado una versión de un proyecto, que queramos publicar para que esté accesible para el resto de desarrolladores.

- Borrar todo lo que se genera con maven, dentro del proyecto:

```
mvn clean
```

- Lanzar los tests unitarios:

  ```
  mvn test
  ```

  NOTA: hay que tener en cuenta que la máquina virtual a través de la cual invocamos a maven es la que lanzará los tests unitarios. Esto implica que, por lo general, tendremos los mismos requisitos que gvSIG en sí, como por ejemplo tener las JAI instaladas. Esto hay que tenerlo en cuenta siempre, ya que incluso para generar un build se van a lanzar los tests unitarios.

4.2.5. Recursos de maven disponibles

- La **web del proyecto Maven**: es el lugar dónde podemos obtener las últimas versiones de maven, documentación general y de los principales plugins, etc.
- El **buscador de dependencias** del repositorio del proyecto maven: nos permite buscar artefactos que queramos incluir como dependencias, que se encuentren dentro del repositorio del proyecto maven.
- El **repositorio de plugins de Codehaus**: el proyecto *Mojo* en codehaus desarrolla plugins adicionales para maven.
- El libro **Better builds with maven**: sus autores son algunos de los principales desarrolladores de maven, y podemos descargarlo sin coste, previo registro.
- **Maven: The Definitive Guide**: otro libro sobre maven, con licencia Commons, que podemos leer online o descargar en PDF.

4.3. Creación de una extensión para gvSIG

4.3.1. Cosas a tener en cuenta antes de desarrollar un plugin.

Para desarrollar un proyecto para gvSIG 2.0 se deben tener en cuenta varias cosas si se quiere que éste tenga el respaldo oficial de gvSIG y pase a formar parte de la distribución oficial de gvSIG.

- Se usará maven como herramienta de contrucción del proyecto.

  Maven define un nombre de grupo, *groupId* y de identificador, *artifactId*, de nuestro proyecto. Se usará como *groupId org.gvsig* y como *artifactId* el *groupId* seguido de un identificador de nuestro proyecto.

  Por ejemplo si se esta creando un plugin para gvSIG de topografía los paquetes del proyecto debe comenzar todos por *org.gvsig.topography*.

  El *artifactId* se usa para:

  - Identificar el paquete raíz de nuestro proyecto java. Todos nuestros paquetes deben comenzar por el *artifactId*.
  - El nombre de nuestro proyecto de eclipse. Los proyectos de eclipse de nuestro desarrollo se nombran igual que el *artifactId*.
  - Los jars generados por nuestro proyecto deben comenzar por el *artifactId* seguidos de la versión del proyecto y el *clasificador* de maven.

-

- Debe haber una separación estricta entre API e implementación.

  Parece tema obvio, pero es muy frecuente que no exista esta separacion, y cuando existe no suele ser tan estricta como se requiere desde gvSIG.

  Los principales motivos de esto responden a temas como:

  - ¿Cómo se puede realizar valoraciones del alcance de una modificación dentro del código? ¿Cuál sería el impacto en la aplicación gvSIG si se modifica esto?

    Si se tienen definidos los APIs de forma estricta facilita enormemente el dar respuesta a esta pregunta.

  - ¿Cómo se puede documentar correctamente el uso de las librerías?. Si tenemos separada la implementación del API, es fácil centrarse en documentar solo el API sin que se ande mezclando éste con la implementación.

  - ¿Cómo se puede realizar test automatizados de nuestra librería?. Si se dispone de un API claro y bien delimitado, podemos centrarnos en preparar test automatizados que comprueben las especificaciones del API de forma que podamos verificar fácilmente que un cambio en la implementación afecta o no a los consumidores del API.

- Deberá existir un sistema de test automatizados que verifiquen el funcionamiento del API de nuestro proyecto. Hasta la fecha se está usando *junit* para realizar estas tareas.

- Los APIs deben estar documentados y esta documentación subida a un sitio público.

  Para la documentación de los APIs se usan los javadocs sobre los interfaces y clases del API. La documentación del API debe ser completa y en inglés. Para generar y desplegar éste se aprovecha la utilidad de maven para la generación de *sites*.

- Deben estar identificadas las dependencias del proyecto con otras librerías.

  Esto se realizará a través de maven manteniendo actualizado *ManageDependencies* en el proyecto maven raíz.

- Debe adjuntarse al proyecto un plan de pruebas. Este plan de pruebas funciona a modo de una batería de pruebas de aceptación que el proyecto debe pasar tras cada construcción de un instalable de gvSIG para una versión final.

- Se debe confeccionar en ReST la documentación de usuario del proyecto.

- Deben desplegarse los jars del proyecto en un repositorio de maven, y seguir una política de versionado acorde a los cambios que se realicen en el proyecto. gvSIG dispone de un repositorio de maven propio en el que desplegar las librerías, pero puede utilizarse otro si se considera adecuado.

4.3.2. Estructura de un proyecto en gvSIG

4.3.2.1. Maven y eclipse

Una de las características más interesantes de maven es su capacidad de trabajo con proyectos

multimódulo y las facilidades que ofrece a la hora de ejecutar objetivos predefinidos para realizar ciertas tareas claramente definidas, como la compilación del código y su empaquetado.

En su contra, la versión actual de Eclipse no incorpora los mecanismos necesarios para poder soportar los proyectos multimódulo, por lo que la integración con los mecanismos de maven resultan complejos y requieren el uso de determinados mecanismos y estrategias para conseguir simular dichos comportamientos.

Maven utiliza un Project Object Model (POM) para describir el proyecto de software a construir, la estructura del proyecto y sus submódulos, sus dependencias de otros módulos y componentes externos y el orden de construcción de los elementos. Haciendo uso de estas características definidas, se puede llegar a conseguir que eclipse emplee la información que en los POMs se detalla, para simular un comportamiento multimódulo.

4.3.2.2. Trabajar en eclipse con maven

A la hora de trabajar con eclipse, tal y como se mencionaba anteriormente, hay que tener en cuenta unas pautas que permiten hacer uso de las funcionalidades que maven ofrece.

Los nombres aplicados a los directorios no son casuales, y es que a través de ellos, se pretende simular la estructura jerárquica del proyecto. Para ello, el proyecto padre tiene el nombre del *artifact_id* y, a partir de él, sus submódulos completan su nombre con el del padre (es decir, si se tiene el proyecto padre *org.gvsig.ejemplo*, el submódulo *Ejemplo1* debe estar contenido en el directorio '*org.gvsig.ejemplo*/*org.gvsig.ejemplo.ejemplo1*', el *Ejemplo2* en '*org.gvsig.ejemplo*/*org.gvsig.ejemplo.ejemplo2*', si el *Ejemplo1* tuviera a su vez el submódulo *Ejemplo1a* estaría en '*org.gvsig.ejemplo*/*org.gvsig.ejemplo.ejemplo1*/*org.gvsig.ejemplo.ejemplo1a*', y así sucesivamente). Esto da pie a una estructura del proyecto en forma de árbol:

- *org.gvsig.ejemplo*
    - *org.gvsig.ejemplo.ejemplo1*
        - *org.gvsig.ejemplo.ejemplo1.ejemplo1a*
    - *org.gvsig.ejemplo.ejemplo2*

Los nodos padre no son tomados como proyectos Java por eclipse, ya que simplemente son los contenedores de los submódulos. Estos proyectos contendrán el fichero POM con su definición y la descripción de los submódulos que lo conforman. De esta forma, se puede lanzar órdenes a módulo padre, para que se ejecute en él y en todos sus descendientes de manera automática (install, compile, etc.)

Por su parte, los nodos hijos (u hojas del árbol) sí que son proyectos Java, con su estructura habitual en proyectos de este tipo. Estos nodos pueden aprovechar la configuración del padre (dependencias, propiedades, atributos, ...) si especifica en su POM quién es su padre, simplificando su configuración.

4.3.2.3. Desarrollar una extensión para gvSIG

Cuando se va a desarrolla un proyecto para gvSIG se debe tener una separación lo más estricta posible entre el API y la implementación de ese API. Para llevar esto a cabo, y apoyánse en la facilidad de maven para gestionar proyectos multimódulo se ha confeccionado una plantilla base

-

para un proyecto de gvSIG. Esta plantilla sirve para orientar sobre cómo organizar el proyecto, separando de forma estricta en varios proyectos de eclipse y distintos jars la lógica de nuestra extensión, del interface de usuario y a su vez, el API de la lógica de su implementación y el API del interface de usuario de su implementación.

La estructura base de un proyecto de será:

- *library*
  - *tags*
  - *branches*
  - *trunk*
    - *org.gvsig.example*
      - *org.gvsig.example.lib*
        - *org.gvsig.example.lib.api*
        - *org.gvsig.example.lib.spi*
        - *org.gvsig.example.lib.impl*
      - *org.gvsig.example.prov*
        - *org.gvsig.example.prov.provider1*
      - *org.gvsig.example.swing*
        - *org.gvsig.example.swing.api*
        - *org.gvsig.example.swing.impl*
      - *org.gvsig.example.main*
- *extensión*
  - *tags*
  - *branches*
  - *trunk*
    - *org.gvsig.example.app*
      - *org.gvsig.example.app.extensión*

Lo primero que se observa es que el proyecto está orientado como un proyecto de *SVN multimódulo*. Por un lado se tiene el desarrollo de la parte de librería y por otro, el de la extensión o extensiones para gvSIG. Esto se ha organizado así ya que normalmente los cambios en la lógica de nuestra extensión suelen llevar un ciclo de vida claramente separados de la integración de estos en gvSIG. Una vez desarrollada la extensión es fácil que de una versión a otra de gvSIG se tenga que hacer alguna modificación en la parte de nuestro código que contiene la integración con gvSIG mientras que la parte de lógica no sufra cambios a no ser que se traten de correcciones de bugs. Para conseguir esta independencia, el repositorio del SVN se separa en dos grandes grupos:

- *library*

  Contiene la parte de lógica e interface de usuario que son independientes de gvSIG como aplicación. No quiere decir que no dependan de algunas librerías de gvSIG como la de acceso a datos, geometrías o mapcontext. SI necesitan depender de ellas puede hacerlo.

- *extensión*.

  Aquí está la parte del código que tiene dependencias con la aplicación gvSIG, andami y otras extensiones de gvSIG. Aquí estarán las clases *Extension* así como la configuración de

nuestro *plugin* para andami.

Vamos a ver ahora que nos encontramos dentro de *library*. Aquí residirán

- *org.gvsig.example.lib*

  Nos proporciona el API, SPI y la implementación de ese API.

- *org.gvsig.example.prov*

  En caso de que la librería necesite trabajar con algún proveedor de servicios aquí estarán las distintas implementaciones de estos proveedores de servicios.

- *org.gvsig.example.swing*

  En caso de que el proyecto suministre un interface de usuario para la parte de lógica, este se ubicará aquí. La parte de *org.gvsig.example.lib* no debe contener dependencias con el interface de usuario.

- *org.gvsig.example.main*

  Se trata de un módulo destinado a proveer una aplicación de prueba que permita arracar el interface de usuario y la lógica de la librería sin tener que arrancar un gvSIG para probarlo.

Y por último nos encontramos con la separación en algunos de estos módulos entre API, SPI e implementación. Esta separación podemos encontrarla en *org.gvsig.example.lib* y *org.gvsig.example.swing*.

En cada uno de los distintos niveles de carpetas se encuentran ficheros *pom.xml*. Así se tiene un *pom* en *org.gvsig.example* que nos permite hacer operaciones con todos los submódulos de nuestro proyecto, así como contener la configuración general del proyecto. Por ejemplo, se puede construir los proyectos de eclipse de cada uno de nuestros módulos directamente, atacando a este *pom*. A su vez, *org.gvsig.example.lib* y *org.gvsig.example.swing* tienen sus propios ficheros *pom* con su configuración específica. Por último, cada uno de los proyectos *finales* tienen su propio fichero *pom*. Cada uno de estos *pom* estará configurado indicando como *pom* padre el de la capeta superior para heredar de él la configuración común.

4.3.3. Creando nuestro proyecto

Para poder trabajar con nuestro proyecto se necesita un gvSIG 2.0 instalado. Además debe estar instalado el complemento *Development project wizard*, disponible a través del gestor de complementos (*Herramientas > Admnistrador de complementos*).

Esta extensión ofrece la opción de *Crear plugin*. Mediante esta utilidad, se permite la creación de proyectos maven, siguiendo la estructura de proyecto antes descrita, personalizada en función de las opciones que seleccionemos para trabajar con esta instalación de gvSIG.

Esta utilidad se puede encontrar en el menú *Herramienta* en la opción *Desarrollo*, tal y como se muestra a continuación:

-

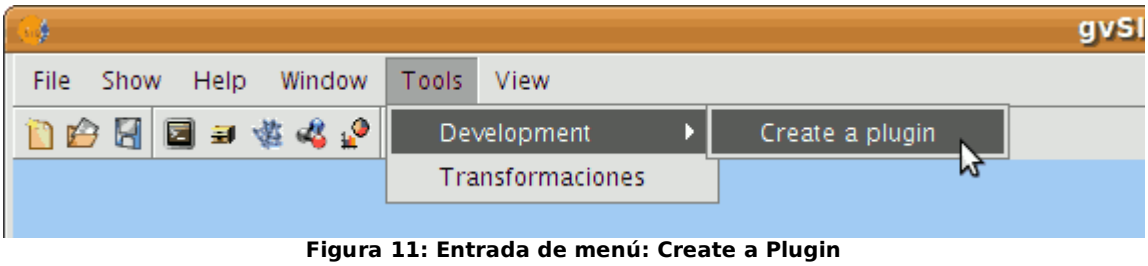


Figura 11: Entrada de menú: Create a Plugin

Tras acceder a la opción *Crear plugin*, la extensión presenta un asistente. Este asistente nos guia para crear una plantilla de proyecto maven asociada a esa instancia de gvSIG. Como aspectos más interesantes, destacar que el asistente requiere que le introduzcamos el nombre de nuestro proyecto, el *group_id*(por defecto *org.gvsig*) y la ubicación donde se generará el plugin.

También requiere que seleccionemos la plantilla a emplear, pudiendo elegir entre dos tipos (espacial y no espacial).

Por último, se debe indicar si se desea que el plugin genere la extensión para gvSIG o no.

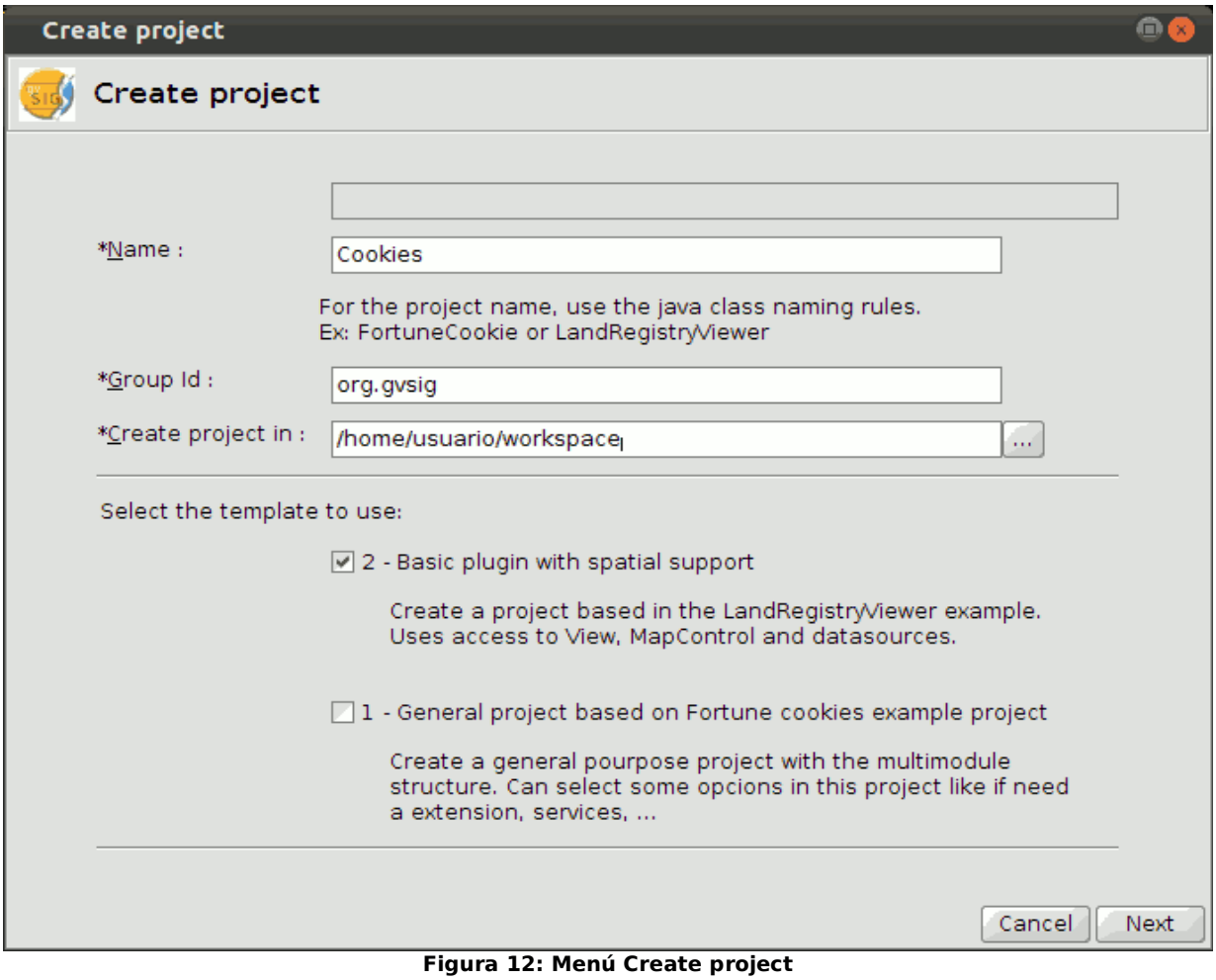


Figura 12: Menú Create project

Como último paso en la generación de nuestro plugin, la herramienta realiza un *prepare_workspace* para dejar nuestro proyecto listo. Realiza las tareas de maven '*mvn configure-eclipse-workspace*','*mvn install*' y '*mvn eclipse:eclipse*'.

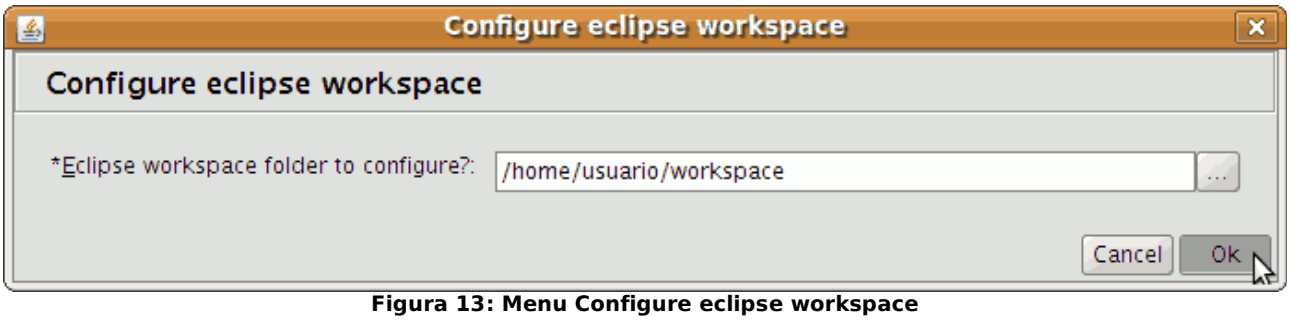


Figura 13: Menu Configure eclipse workspace

Una vez creado el proyecto, se importa la carpeta en la que hemos creado el proyecto desde Eclipse. Para poder trabajar con ella, se debe importar tanto el proyecto padre (*org.gvsig.<nombre_del_proyecto>*) como todos los submódulos (*org.gvsig.<nombre_del_proyecto>.lib.api*, *org.gvsig.<nombre_del_proyecto>.lib.impl*, *org.gvsig.<nombre_del_proyecto>.main* y los demás en función de la variante escogida).

-

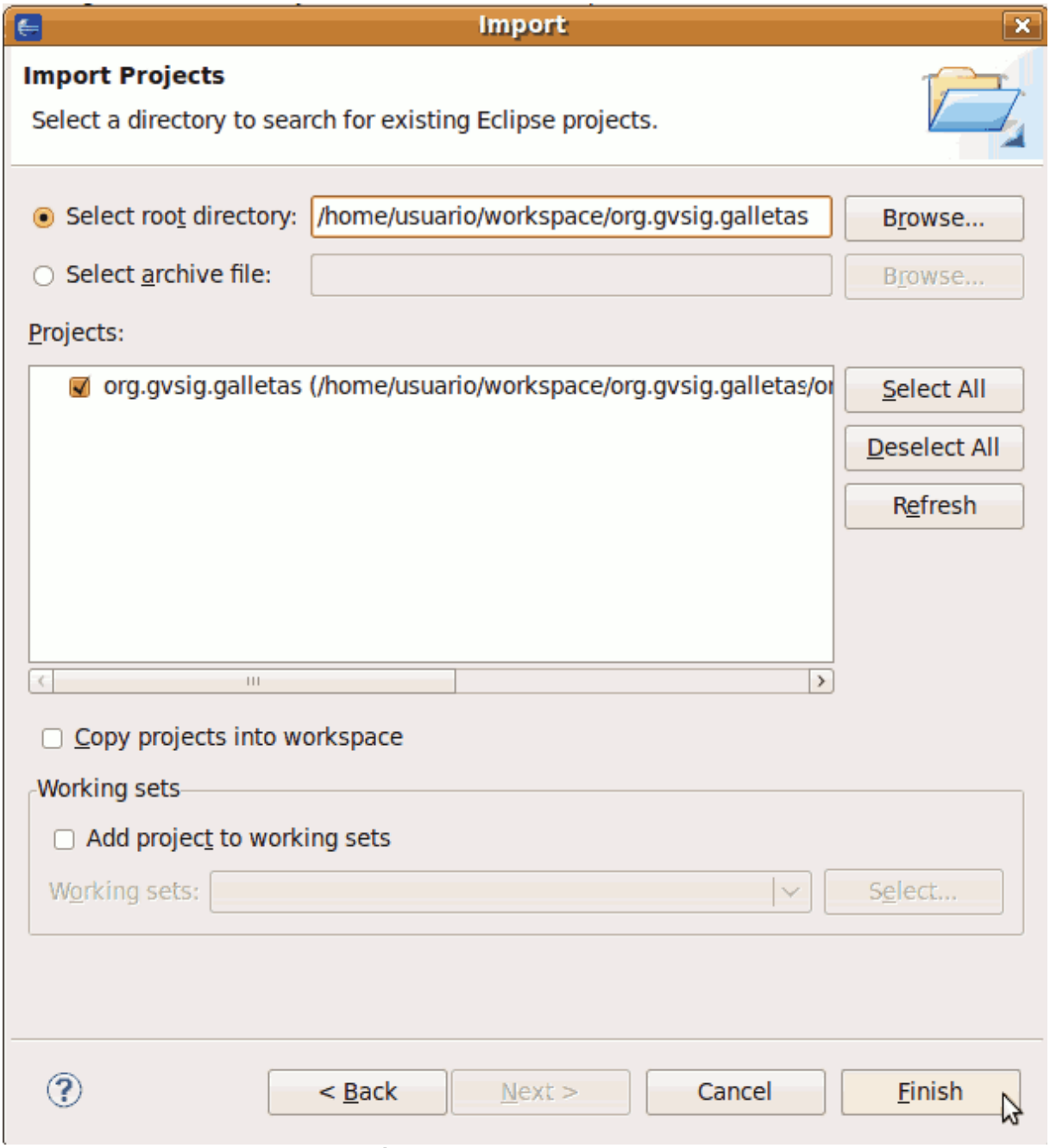


Figura 14: Importar proyectos

Una vez el proyecto haya sido importado en eclipse, se puede empezar a crear el plugin y a arrancarlo normalmente, utilizando el fichero *build.xml* del proyecto, y utilizando el target *mvn-install*.

Para terminar, una vez desarrollado el plugin, se utiliza la opción *Empaquetar plugin* de gvSIG. Una utilidad para generar binarios de nuestro plugin y que nos permite distribuirlo como un plugin independiente o incluirlo en una instalable de gvSIG ya existente.

4.3.3.1. Cómo montar un workspace de gvSIG para Eclipse

Los pasos a realizar para montar un workspace de Eclipse para trabajar con gvSIG son los siguientes:

1. Arrancamos Eclipse y creamos un nuevo workspace para gvSIG 2.0.
2. Establecemos el encoding usado en gvSIG (ISO-8859-1):

   ```
   Window > Preferences: General > Workspace > Text file encoding = ISO-8859-1
   ```

3. Establecemos la compatibilidad con Java 1.5 / 5.0 en general para todos los proyectos (aquellos proyectos que son compatibles con Java ME CDC tienen establecida a compatilidad con Java 1.4):

   ```
   Window > Preferences > Java > Compiler > Compiler Compliance Level = 1.5 (ó 5.0).
   ```

4. En eclipse, registramos el repositorio de subversion sobre el que vamos a trabajar.

   Podemos acceder de dos formas al repositorio subversion de gvSIG:

   - Acceso anónimo de sólo lectura.
   - Acceso de lectura y escritura, para lo cuál necesitaremos un usuario en la infraestructura de gvSIG.

   La url de acceso es: https://devel.gvsig.org/svn/gvsig-desktop.

   Como eclipse no lleva soporte por defecto para subversion, deberemos tener instalado alguno de los plugins existentes para ello:

   - Subclipse.
   - Subversive.

   es importante tener en cuenta la versión o configuración del plugin de soporte para subversion, ya que debemos user la misma versión de subversion desde cliente en todos los casos: linea de órdenes, eclipse, etc. Esto se debe a que, con cada versión mayor de subversion, el formato local de información cambia y no es compatible hacia atrás.

   Es decir, si usamos un cliente de subversion 1.6 para hacer un checkout o un update, si a continuación usamos otro cliente de una versión anterior (1.5. 1.4, etc.), éste no funcionará correctamente.

   Si accedemos a Internet a través de un proxy, deberemos configurar antes el acceso en Eclipse. Para ello accedemos a la opción:

   ```
   Window > Preferences > General > Network Connection
   ```

   En esta ventana de preferencias podemos configurar la conexión con el proxy de nuestra red. Dependiendo de nuestro caso, podemos elegir la opción *System proxy configuration* o bien la opción *Manual proxy configuration* y definir los valores de conexión manualmente.

   Para registrar el repositorio de subversion elegido, deberemos abrir la *Perspectiva de exploración de respositorios de subversion*. Sino la tenemos disponible podemos abrirla

-

desde:

```
Window > Open Perspective > Other : SVN Respository Exploring
```

Una vez abierta la perspectiva, desde la vista de repositorios SVN podemos añadir el repositorio de gvSIG.

A partir de este momento se hará referencia a algunas ubicaciones de proyectos dentro del repositorio de subversion de gvSIG. Dicho repositorio está organizado con la estructura habitual de subversion:

- trunk: última versión en desarrollo.
- tags: versiones cerradas y builds, de los que se genera un tag.
- branches: ramas de desarrollo.

A partir de ahora, en los paths de subversion indicaremos [REPOSITORY]/[VERSION] en vez de indicar el repositorio y si es *trunk*, *tags/v2_0_0_Build_2007*, etc. Hay que tener en cuenta que, en el momento de escribir este documento, la versión 2.0 se está desarrollando en la rama *branches/v2_0_0_prep*, que es de dónde habrá que descargar los proyectos en estos momentos, aunque en un futuro pasará al *trunk*.

5. Descargar el proyecto *build*.

   Para ello haremos un *checkout* desde el repositorio de fuentes. Desde la vista de repositorios SVN, navegaremos a:

   ```
   [VERSION] (ej: branches/v2_0_0_prep)
   ```

   Desde aquí, seleccionaremos la carpeta *build* y haremos un checkout, pulsando el botón izquierdo sobre la carpeta y seleccionando la opción correspondiente. En el diálogo de checkout podemos dejar las opciones tal cuál nos aparecen y pulsamos *Finish*.

   Una vez haya terminado, si vamos a la perspectiva *Java* o *Resource*, ya tendremos el proyecto disponible.

   Como alternativa, si queremos hacerlo desde la consola, hay que lanzar lo siguiente:

   ```
   svn co https://devel.gvsig.org/svn/gvsig-
   desktop/branches/v2_0_0_prep/build
   ```

   Igual que en el caso de eclipse, si accedemos a Internet a través de un proxy, deberemos configurar el cliente de subversion para que se conecte al servidor a través del proxy. Podemos encontrar las instrucciones de configuración en el FAQ de la web oficial de subversion, o bien en el apartado sobre configuración de servidores en el libro de subversion.

   Una vez descargado el proyecto *build* desde consola, procederemos a importarlo en Eclipse.

6. Abriremos la vista de Ant. Para ello usaremos la opción de menú:

   ```
   Window > Show view > ant
   ```

7. Instalaremos la configuración básica de maven para gvSIG.

   El proyecto *build* contiene la configuración base de maven para todos los proyectos de gvSIG en un *pom.xml* general y otro específico por tipo de proyecto:

- librería: build/libraries-pom/pom.xml
- librería nativa: build/libraries-jni-pom/pom.xml
- extensión: build/extension-pom/pom.xml

La primera vez, y luego cada vez que hayan cambios en alguno de estos archivos, tendremos que instalar estos archivos en nuestro repositorio local de maven. Podemos hacerlo de dos formas:

- Eclipse: Añadiremos a la vista de Ant que hemos abierto, el archivo *build.xml* que tenemos dentro del proyecto *build*. Entonces nos aparecerán una serie de objetivos de ant, entre los cuáles está *mvn-install*. Lo lanzamos con doble click o seleccionando el objetivo indicado y pulsando en el icono con la flecha.
- Consola: desde la carpeta build, lanzaremos:

  ```
  mvn install
  ```

8. Configuraremos eclipse para que cargue correctamente los proyectos generados desde maven.

   Para ello, desde la vista de ant, invocaremos al objetivo: *mvn-configure-eclipse-workspace*, que nos creará automáticamente una variable de entorno en eclipse que se emplea para hacer referencia a los jars gestionados de las dependencias gestionadas a través de maven. Nos preguntará por la ubicación del workspace, que generalmente será la que nos propone.

   Alternativamente, desde consola podemos usar maven de la siguiente forma:

   ```
   mvn -Declipse.workspace=<path-to-eclipse-workspace> eclipse:add-maven-repo
   ```

   Entonces cerraremos Eclipse y lo volveremos a abrir.

9. Validaremos que la variable M2_REPO está definida correctamente en eclipse, a través del menú:

   ```
   Window > Preferences : Java > Build Path > Classpath Variables
   ```

   El valor de la variable debe ser algo como lo siguiente (cambiando [USER_HOME] por el directorio del usuario, según el Sistema operativo):

   ```
   M2_REPO = [USER_HOME]/.m2/repository
   ```

10. Añadiremos a la vista de Ant el archivo *build.xml* del grupo de proyectos con el que vamos a trabajar. Más adelante se explica en detalle qué son los grupos de proyectos, por ahora lo podemos tratar como listas de proyectos (librerías y extensiones) que están dentro de la carpeta *build/projects*.

    Si queremos descargar los proyectos de un gvSIG completo, añadiremos a la vista de ant el archivo:

    ```
    build/projects/gvsig-standard/build.xml
    ```

11. Procederemos a descargar todos los proyectos.

    Para ello, desde la vista de Ant, invocaremos al objetivo: *svn.checkout.all*, que nos abrirá un formulario que nos permitirá seleccionar:

-

- La URL del servidor de subversion, según queramos el acceso público de sólo lectura, o el que requiere usuario y contraseña con acceso de lectura y escritura.
- La versión de SVNKit a emplear. Hay que comprobar que es la misma que la empleada por el plugin de soporte a subversion que tengamos instalado en eclipse. Igual que en casos anteriores, si accedemos a Internet a través de un proxy, habrá que configurar el acceso para la librería SVNKIT. Justamente la librería lee la misma configuración que tiene el cliente de subversion nativo, cuya configuración se indica en el punto *Descargar el proyecto build*. Podemos encontrar más información en la propia guía de usuario de SVNKIT
- El usuario y la clave para acceder a subversion, si procede.
- Si queremos que se cree la configuración de proyecto de eclipse automáticamente, una vez terminada la descarga. Si desactivamos esta opción, deberemos generar nosotros los proyectos de eclipse, bien desde la opción *mvn eclipse* desde el menú de *External tools*, o bien desde consola con la orden: *mvn -P eclipse-project*.

De forma alternativa, desde consola podemos invocar a este objetivo de ant, o también podemos hacer el checkout de cada uno de los proyectos, bien desde eclipse, bien desde la consola. En este último caso, tendremos que generar también nosotros la configuración de proyecto de eclipse.

12. Importaremos los proyectos en eclipse. Una vez haya terminado la instrucción anterior, podemos importar los proyectos en eclipse desde el menú:

    ```
    File > Import > General > Existing projects into workspace
    ```

    Pulsaremos en *Select root directory* y seleccionamos el directorio de nuestro workspace. Nos aparecerá la lista de proyectos a importar y aceptamos, tras lo cuál Eclipse procederá a compilar los proyectos importados.

    al importar los proyectos, compruebe que la opción *Copy projects into workspace* esté desactivada, ya que lo que queremos es que se usen los mismos proyectos descargados directamente.

13. El plugin de eclipse para maven no relaciona los proyectos entre sí hasta que están importados dentro del workspace de eclipse, por lo tanto tendremos que regenerar la configuración de eclipse de los proyectos.

    Para ello, desde la vista de Ant, invocaremos al objetivo: *mvn-eclipse-eclipse*. Una vez haya terminado, seleccionaremos todos los proyectos y usaremos la opción de refrescar (F5 o botón derecho y la opción *Refresh*).

14. Para terminar, el plugin de eclipse para maven genera configuración de proyecto de eclipse sólo para aquellos proyectos que son java, es decir, que tienen código fuente java. Dado que existen proyectos multimódulo, esto significa que sólo se habrá generado configuración de proyecto eclipse para los submódulos hijo finales. Por ejemplo, el proyecto *org.gvsig.annotation* tiene la siguiente estructura:

    ```
    org.gvsig.annotation
        org.gvsig.annotation.lib
            org.gvsig.annotation.lib.api
            org.gvsig.annotation.lib.impl
        org.gvsig.annotation.main
        org.gvsig.annotation.swing
    ```

```
org.gvsig.annotation.swing.api
org.gvsig.annotation.swing.impl
```

Si miramos en los proyectos importados actualmente, veremos que tenemos sólo los siguientes:

- org.gvsig.annotation.lib.api
- org.gvsig.annotation.lib.impl
- org.gvsig.annotation.main
- org.gvsig.annotation.swing.api
- org.gvsig.annotation.swing.impl

Necesitaremos tener, al menos, el proyecto padre raíz para poder tener todas las actualizaciones cuando nos sincronicemos desde subversion, además de permitirnos realizar acciones en todos los submódulos de un mismo proyecto.

Para ello no hay otro remedio que realizar el siguiente proceso para cada uno de estos proyectos padre:

- Seleccionar en el menú de eclipse *File* > *New* > *Project...*.
- En la ventana que aparece seleccionaremos la opción *General* > *Project* y pulsaremos *Next*
- En la siguiente ventana, rellenaremos los siguientes campos:
    - *Project name*: El nombre del proyecto padre a importar, ej: *org.gvsig.annotation*.
    - *Use default location*: Marcaremos esta opción

  Finalmente pulsaremos *Finish* y ya tendremos el proyecto en eclipse. Se trata de un proyecto no java, por lo que no editaremos los fuentes java desde el mismo, sino desde el subproyecto hijo correspondiente. Estos proyectos sólo los usaremos para realizar acciones de maven sobre todos los hijos, o para sincronizar con subversion.

Los proyectos a importar de esta forma son:

- org.gvsig.annotation
- org.gvsig.annotation.app
- org.gvsig.app.document.layout.app
- org.gvsig.app.document.table.app
- org.gvsig.exportto
- org.gvsig.exportto.app
- org.gvsig.geometrymeasurement.app
- org.gvsig.hyperlink.app
- org.gvsig.installer
- org.gvsig.installer.app
- org.gvsig.newlayer
- org.gvsig.newlayer.app
- org.gvsig.personaldb
- org.gvsig.selectiontools.app

-

- org.gvsig.symbology
- org.gvsig.symbology.app

Este paso final es ciertamente farragoso pero no hemos encontrado otra forma más automática de hacerlo. En su momento probamos a subir a subversion los .project de estos proyectos padre, pero entonces eclipse, al hacer la importación de todos los proyectos inicial, sólo incluía los de los proyectos padre. Entonces tocaba ir subproyecto por subproyecto importándolo, y es todavía peor que esta opción. Existe la opción de usar el plugin de maven para eclipse, pero eso nos da algunos otros problemas que tenemos pendiente resolver.

4.3.3.2. Integración con eclipse

En el momento de preparar esta guía, Eclipse no lleva soporte de forma oficial para maven. Existen algunos plugins disponibles que permiten integrar maven en eclipse, aunque parece que todavía no son completamente funcionales. Uno de ellos (Q4E), de hecho, está en proceso de ser incorporado a eclipse de forma oficial, bajo el nombre Eclipse IAM, pero todavía no ha sido liberado.

Por no ligarnos a un plugin no oficial de eclipse, por ahora se va a emplear una integración básica de eclipse con maven, que permita en cierta forma compartir la configuración de proyecto de maven con eclipse, así como invocar a maven desde el propio eclipse.

4.3.3.3. Preparar el workspace para trabajar con maven

si nos hemos descargado el workspace siguiendo las instrucciones del documento *Cómo montar un workspace con Eclipse* (ver documentos relacionados), ya tendremos el workspace preparado.

Para que el eclipse sea capaz de encontrar el repositorio de maven es necesario configurar la variable en el workspace. Esto lo podemos hacer de forma sencilla empleando el plugin *eclipse* de maven, con el objetivo:

```
mvn configure eclipse workspace
```

Al hacerlo desde eclipse, ya tomará por defecto el valor del path del workspace sobre el que estamos trabajando.

Si queremos lanzarlo desde consola, la instrucción es:

```
mvn eclipse:add-maven-repo -Declipse.workspace=WORKSPACE_PATH
```

En 'WORKSPACE_PATH' especificaremos la ruta absoluta al workspace.

Generación de proyectos de eclipse desde maven

los proyectos generados desde eclipse con los objetivos *create library* y *create extension* ya generan también automáticamente el proyecto de eclipse.

Si tenemos un proyecto configurado con maven, no es necesario crear manualmente el proyecto de eclipse, ya que maven puede generarlo por nosotros. Para ello bastará con usar el siguiente objetivo:

```
mvn eclipse
```

Desde consola, la instrucción equivalente es:

```
mvn -P eclipse-project
```

La forma habitual de generar un proyecto de eclipse desde maven es emplear la instrucción *mvn eclipse:eclipse*. Sin embargo, para algunos proyectos, como los que generan varios jars en los que se incluyen clases de tipo *Library*, el plugin de maven que genera los proyectos de eclipse no es capaz de generar toda la configuración necesaria, por lo que se ha creado un perfil desde el que se invoca al plugin de eclipse y, además, añade el resto de configuración necesaria. Si en algún caso se creara el proyecto de eclipse sin usar el perfil *eclipse-project*, el único problema es que los tests unitarios que empleen el mecanismo de inicialización automática de librerías no funcionaran si los lanzamos desde eclipse.

Esto generará los archivos de eclipse necesarios, que nos permitirán importar el proyecto desde eclipse, con todas las dependencias, directorios de fuentes, etc. ya definidos.

Si ya existe el proyecto de eclipse, el objetivo anterior no lo sobreescribe, para ello tenemos que emplear antes el objetivo:

```
mvn eclipse clean
```

O desde consola:

```
mvn eclipse:clean
```

En los proyectos de gvSIG, está configurado de forma que, además de descargar y enlazar las dependencias binarias (jars con clases) en el proyecto de eclipse, maven intentará descargar también los jars con los fuentes y los javadocs correspondientes, enlazándolos a cada jar de binarios. Esto nos permitirá ver fácilmente la documentación y el código fuente de las dependencias de nuestros proyectos.

Por otro lado, si generamos los proyectos de eclipse para un grupo de proyectos, maven nos enlazará las dependencias entre los proyectos del grupo a nivel de proyecto, en vez de a nivel de archivo jar.

Esto permite que los proyectos de eclipse no se suban al repositorio de fuentes de gvSIG, ya que se pueden generar fácilmente para un proyecto o conjunto de proyectos.

Cómo invocar maven desde eclipse

Para facilitar el uso de maven desde eclipse, se han preparado una serie de lanzadores con los objetivos más usados en los proyectos con maven.

Dichos lanzadores están definidos dentro del proyecto *build*, por lo que una vez incluido este proyecto en nuestro workspace de eclipse, los tendremos disponibles automáticamente.

Para lanzar cualquiera de ellos deberemos tener abierto el proyecto sobre el que queremos trabajar, y seleccionar el lanzador correspondiente desde la opción de *Herramientas externas (External Tools)*.

Errores habituales

- **Eclipse no detecta una dependencia en el repositorio de maven, aunque el jar correspondiente sí que existe en disco**

  Por alguna razón, parece que Eclipse no se da cuenta de cambios en librerías de dependencias que están puestas a través de una variable de entorno. Incluso refrescando y

-

recompilando el proyecto a veces no detecta el jar correspondiente.

Una forma que suele funcionar consiste en:

1. Cerrar el proyecto.
2. Abrir el proyecto.
3. Recompilar el proyecto.

platform	distribution	compiler	arch	libraryType	Comentarios
linux	all	gcc4	i386	dynamic	Versión de 32 bits (equivalente en general a una Ubuntu-10.04)
win	nt	vs8	i386	dynamic	Versión de 32 bits, compatible con windows XP, Vista y 7.
mac	*10.5*	*gcc4*	*universal*	*dynamic*	**Plataforma todavía no disponible**. Está en proceso de preparación y pueden variar alguno de los parámetros de la plataforma.

Tabla 2: Plataformas soportadas.

4.3.4. Compilar un grupo de proyectos

Maven permite trabajar con lo que llama multimódulos o una estructura multiproyecto. Esto nos permite lanzar una orden de maven sobre un grupo o jerarquía de proyectos, desde una única llamada a éste. Para ello tenemos un *pom.xml* que hace referencia a un conjunto de proyectos de maven.

La estructura habitual consiste en tener una organización en árbol de proyectos, pero también podemos tener una estructura plana, en el que el proyecto que contiene el *pom.xml* en el que se hace referencia al resto de proyectos se encuentra a la misma altura que estos.

En gvSIG la estructura de proyectos que hay en el repositorio de fuentes no está pensada para ser descargada tal cuál, sino que se debe descargar proyecto a proyecto, con lo que tenemos una lista de proyectos todos dentro del mismo directorio.

Aunque en un futuro se podría migrar a una estructura multimódulo de proyectos, por ahora la vamos a simular mediante grupos de proyectos dentro del directorio *build* de gvSIG. Dentro de este, en el subdirectorio *projects* se han creado una serie de directorios, cada uno de los cuáles hace referencia a un grupo de proyectos:

```
build
`-projects
  |-gvsig-base
  |-gvsig-cdc-compat
  |-gvsig-coverage
  |-gvsig-coverage-base
  ...
```

Si accedemos a uno de estos directorios, vemos que contienen un archivo *pom.xml*, el cuál hace referencia a un conjunto concreto de proyectos. Así, si desde aquí usamos maven, la orden se lanzará automáticamente para cada uno de los proyectos referenciados.

Este mecanismo nos permite, de forma cómoda, realizar operaciones de compilación e instalación sobre un conjunto de proyectos, o incluso generar un build de gvSIG completo.

El criterio para definir los grupos ha sido el de unir proyectos sobre una misma funcionalidad o cuyo desarrollo se realice de forma conjunta. Además, aprovechando que un *pom.xml* puede hacer referencia, tanto a proyectos sueltos, como a otros conjuntos de proyectos, se ha definido un grupo para la generación de un build completo con la unión de otros grupos.

Por ejemplo, si estamos desarrollando sobre GPE, actualmente está dividido en los siguientes proyectos:

- libGPE
- libGPE-XML
- libGPE-GML
- libGPE-KML
- extGPE-gvSIG

Si queremos compilarlos todos a la vez, bastará con ir al directorio *gvsig-gpe* y realizar la orden:

```
mvn compile
```

Esto realizará la compilación de los distintos proyectos de GPE, uno tras otro. Si, en alguno de ellos, se produce un error de compilación, el proceso se detendrá y no seguirá en los siguientes.

Dentro de cada uno de estos directorios de grupos de proyectos, tenemos también un script de utilidad que nos permite hacer un checkout de forma rápida de los proyectos del grupo. Por ejemplo, si queremos desarrollar sobre GPE, bastará con hacer un checkout inicial del directorio *build* y, a continuación, desde el subdirectorio *projects/gvsig-gpe* llamar al script *svnco.sh*, que se encargará de hacer un checkout de los proyectos de GPE.

4.3.5. Compilar un gvSIG completo

Aprovechando el mecanismo anterior de los grupos de proyectos, se han definido dos grupos que permiten trabajar con todos los proyectos de un build de gvSIG:

- gvsig-base: conjunto de proyectos básicos para tener un build de gvSIG mínimo con soporte vectorial sobre algunos formatos de archivo básicos. Se puede emplear durante el desarrollo de nuevas extensiones que no dependan de otros proyectos, sin cargar otras extensiones.
- gvsig-standard: conjunto de proyectos de un build estándar de gvSIG. Incluirá habitualmente el conjunto de proyectos que conformarán un build oficial de gvSIG.

A partir de ahora consideraremos que vamos a trabajar con la versión del build estándar.

Para compilar todo gvSIG usaremos maven, bien desde eclipse, bien desde consola. La forma habitual será mediante el objetivo *install*:

- Eclipse: si no lo tenemos aún, añadiremos a la vista de Ant el archivo *build/projects/gvsig-standard/build.xml* y lanzaremos el objetivo *mvn-install*. De la misma forma, tenemos otros objetivos de uso habitual accesibles con este mecanismo, con su equivalencia en maven, como por ejemplo:
    - mvn-clean
    - mvn-install

-

- mvn-install-without-tests: equivalente a *mvn -Dmaven.test.skip=true install*
- mvn-reinstall: equivalente a *mvn clean install*
- mvn-reinstall-without-tests: equivalente a *mvn -Dmaven.test.skip=true clean install*

Al menos la primera vez deberían compilarse y lanzarse los tests unitarios en los proyectos que los tengan activados. El resto de veces, se puede hacer la compilación sin lanzarlos mediante el objetivo *mvn-install-without-tests*.

- Consola:

```
mvn install
```

Esto compila todos los proyectos e instala todas las extensiones y sus archivos en la ubicación correspondiente.

4.3.6. Arrancar gvSIG

Los builds de gvSIG desde maven están configurados para ser generados en el directorio *build/product*, en vez de dentro del proyecto de *_fwAndami* como se hacía con ant.

Podemos lanzar gvSIG de dos formas distintas:

- Eclipse: automáticamente, al haber incluido el proyecto *build* se nos habrán configurado una serie de *configuraciones de arranque* de utilidad. Para arrancar gvSIG seleccionaremos:

  ```
  Run > Run Configurations > Java Application
  ```

  De la lista de configuraciones, tendremos al menos las siguientes 3:

  - gvSIG Linux
  - gvSIG Mac
  - gvSIG Windows

  Se selecciona la que corresponda al sistema operativo. Esta configuración se puede usar, tanto para ejecución normal, como para ejecución de gvSIG en depuración.

  Estos launchers están configurados incluyendo en su classpath la lista the archivos *.jar* disponibles en la carpeta *build/product/lib* de forma individual, los cuáles se corresponden con el propio jar the andami más todas sus dependencias. Dado que dichos archivos se generan y copian a través de maven, algunas veces eclipse no se da cuenta de los cambios en dicha carpeta.

  Algo que pasa algunas veces es que eclipse cree que no hay nada en la carpeta *build/product/lib*, y sin embargo sí que están los archivos. Entonces, al usar uno de los launchers eclipse nos da un error, algo como:

  ```
  The archive: /build/product/lib/castor-0.9.5.3.jar which is referenced by
  the classpath, does not exist
  ```

  Para solventarlo basta con ir a la carpeta *build/product/lib* con el package explorer, project explorer o navigator de eclipse y refrescarlo. Entonces ya aparecen los archivos jar necesarios y se puede lanzar el launcher de nuevo.

- Consola: dentro del directorio *build/product* tenemos ya un *gvSIG.sh*, o un *gvSIG.bat* según corresponda, para arrancar el build de gvSIG que hayamos generado.

4.3.7. Trabajando con un proyecto gvSIG

4.3.7.1. Compilar mi proyecto

Para compilar un proyecto desde maven, basta con invocar el objetivo:

```
mvn compile
```

Desde eclipse, se ha creado un lanzador en la opción de *External Tools* que permite compilar desde maven fácilmente, con el nombre: *mvn compile*.

Antes de usar este lanzador, deberemos seleccionar o estar editando algun archivo del proyecto que vamos a compilar, ya que se emplea la variable *project_loc* para invocar a maven sobre un proyecto.

Hay que tener en cuenta que el proyecto de eclipse está configurado de forma que las clases compiladas se generan en el mismo directorio que lo hace maven, así que, por lo general, al invocar este objetivo sobre un proyecto que tengamos abierto en eclipse, ya esté todo compilado y no haga nada.

Además de compilar, si queremos lanzar los tests unitarios de nuestro proyecto desde maven, emplearemos el objetivo:

```
mvn test
```

Finalmente, si queremos que el código que compilemos en un proyecto, esté disponible para otros proyectos que dependendan del mismo, deberemos generar el o los archivos .jar correspondientes, e instalarlos en el repositorio local de maven. Para ello emplearemos el objetivo:

```
mvn install
```

El objetivo *install*, además de compilar, lanza los tests unitarios del proyecto. Si queremos hacer la instalación sin lanzarlos, podemos usar el objetivo:

```
mvn install (no tests)
```

El equivalente desde consola es:

```
mvn -Dmaven.test.skip=true install
```

Los proyectos de gvSIG están configurados para generar, además del archivo .jar con las clases compiladas, otro archivo .jar con los fuentes. Esto nos será útil al enlazar las dependencias en maven y su uso desde los proyectos de eclipse.

Por otro lado, si queremos hacer una instalación y compilación desde cero, podemos usar el objetivo:

```
mvn clean install
```

4.3.7.2. Desplegar mi proyecto

Si nuestro proyecto es una extensión de gvSIG, existe una tarea adicional, además de la compilación y generación de archivos .jar: la instalación de la extensión sobre gvSIG.

-

Para ello se ha creado el siguiente perfil en maven:

install-extension: copia los archivos .jar generados, junto con el resto de recursos de la extensión (configuración, properties con textos multiidioma, imágenes, etc.) a gvSIG.

Dicho perfil está activado por defecto en los proyectos de tipo extensión, cuando se invoca al objetivo:

```
mvn install
```

4.3.7.3. Publicar mi proyecto

Una vez que se haya terminado un determinado desarrollo en nuestro proyecto, y se desse que esté disponible para otros proyectos, sin que éstos tengan que descargar el código fuente de nuestro proyecto y compilarlo, se puede publicar en el repositorio de maven de gvSIG.

Este paso adicional se realiza a través del objetivo:

```
mvn deploy
```

Este objetivo realiza un *mvn install* y, a continuación, sube los artefactos generados, como el archivo .jar con las clases compiladas, al repositorio de maven.

para que el deploy funcione se debe tener un usuario con permisos en el proyecto gvSIG, y haber realizado correctamente la configuración definida en el documento *Configuración inicial de maven* (ver contenido relacionado al final del documento).

Los proyectos de gvSIG están configurados de forma que el repositorio al que se suben los artefactos es el de gvSIG.

En gvSIG se genera, por defecto, tanto el archivo .jar con las clases compiladas como el archivo .jar con el código fuente, que también se subirá al repositorio.

Además, si estamos publicando una versión con cambios significativos, será conveniente publicar también el javadoc en un archivo .jar. Para ello activaremos el perfil *release* con el siguiente objetivo desde eclipse:

```
mvn deploy release
```

Desde consola, el uso de maven directo es el siguiente:

```
mvn -P release deploy
```

4.3.7.4. Generación de documentación e informes

Maven permite la generación de documentación a partir de los fuentes de un proyecto, como por ejemplo el *javadoc*.

Para ello existen una serie de *plugins* especiales de maven para la generación de informes.

Por ejemplo, para generar el javadoc usaremos el plugin correspondiente de la siguiente forma:

```
mvn javadoc:javadoc
```

Además de la generación de informes individuales, maven permite la generación de lo que llaman el *site*. Este consiste en un conjunto de páginas de informes de maven enlazadas, con un menú que nos

permite navegar por todos los documentos generados.

De hecho, este mini website generado con maven es lo que emplean muchos de los proyectos Java de apache como website del proyecto.

Para los proyectos de gvSIG, se ha preconfigurado una serie de plugins con las opciones adecuadas del proyecto. Además, existen una serie de páginas que muestran información extraída del archivo *pom.xml* del proyecto, por lo que será importante completar y mantener actualizada dicha información.

En los proyectos que se generen usando los arquetipos de maven, el archivo *pom.xml* ya lleva los atributos necesarios predefinidos, a falta de rellenar y completar con valores reales.

Para generar el site invocaremos el objetivo:

```
mvn site
```

Esto genera la documentación dentro de nuestro proyecto, en la carpeta:

```
target/site
```

4.3.7.5. Cómo añadir o actualizar una dependencia

Si se desea añadir o actualizar una dependencia externa en un *pom.xml* en un proyecto de gvSIG, se debe hacer lo siguiente:

Primero, nos asegurarnos que la dependencia y la versión que nos interesa está ya disponible en algunos de los repositorios oficiales de maven. Para ello tenemos dos opciones:

- Usar el buscador del repositorio de maven.org
- Realizar una búsqueda manual, mediante google o inspección directa de alguno de los repositorios alternativos de maven.

Si hemos encontrado la dependencia deseada, ya podemos modificar el *pom.xml*, pues ya sabemos que está disponible para todos.

Si no hemos encontrado la dependencia, habría que pedir al proyecto original que suban la dependencia al repositorio oficial de maven, proporcionándoles la información necesaria para ello para que les sea más sencillo:

Así, contribuiremos con la comunidad a facilitar el uso de la librería correspondiente a través de maven. En algún proyecto es posible incluso que nos propongan que nos encarguemos de actualizar nosotros el repositorio de maven con nuevas versiones de una librería.

Si desde el proyecto original no se quiere hacer o se tarda mucho, podemos subirla temporalmente al repositorio maven de gvSIG, quitándola cuando esté disponible en el repositorio oficial.

4.3.7.6. Cómo subir una libraría al repositorio maven de gvSIG

Lo ideal sería que todas las librerías de las que depende un proyecto de gvSIG tuviesen un repositorio maven para poder descargarlas. Lamentablemente esto siempre no es así y a veces hay que subir una librería al repositorio maven de gvSIG.

En ese caso, para hacer pruebas lo primero que debemos hacer es incluir la dependencia en nuestro

-

repositorio local de maven. Para ello usaremos el siguiente comando:

```
mvn deploy:deploy-file -Durl=REPO_URL
                    -DrepositoryId=some.id
                    -Dfile=your-artifact-1.0.jar
                    [-DpomFile=your-pom.xml]
                    [-DgroupId=org.some.group]
                    [-DartifactId=your-artifact]
                    [-Dversion=1.0]
                    [-Dpackaging=jar]
                    [-Dclassifier=test]
                    [-DgeneratePom=true]
                    [-DgeneratePom.description="My Project Description"]
                    [-DrepositoryLayout=legacy]
                    [-DuniqueVersion=false]
```

El comando admite muchas opciones, sobre las cuáles podemos encontrar más documentación en la web del proyecto maven, en el apartado del plugin *deploy*. Sin embargo, lo más habitual será emplear lo siguiente:

```
mvn deploy:deploy-file -DgroupId=$LIBRARY_GROUPID
                    -DartifactId=$LIBRARY_ID
                    -Dversion=$LIBRARY_VERSION
                    [-Dclassifier=$LIBRARY_CLASSIFIER]
                    -Dpackaging=jar
                    -Dfile=$LIBRARY_JAR
                    -Durl=$REPOSITORY_URL
                    -DrepositoryId=gvsig-repository
```

Los valores de las variables (empiezan por $) deberemos sustituirlos por:

LIBRARY_GROUPID:
: un valor típico suele ser el dominio principal del proyecto. Por ejemplo, para los proyectos gvSIG es *org.gvsig*, para los proyectos apache *org.apache*, etc.

LIBRARY_ID:
: identificador o nombre de la librería. Suele ser un nombre único o bien un nombre con estilo de paquete java. En gvSIG los proyectos emplean este último formato (ej: *org.gvsig.tools.lib*).

LIBRARY_VERSION:
: versión del jar que vamos a subir.

LIBRARY_CLASSIFIER:
: para proyectos que generan varios jars, se emplea este valor para distinguir entre ellos. Un valor típico es *tests*, para el jar con los tests unitarios del proyecto. Es opcional y no suele ser necesario.

LIBRARY_JAR:
: path absoluto al archivo .jar que vamos a subir, o bien relativo desde el directorio donde ejecutamos maven

REPOSITORY_URL:

: url del repositorio maven al que vamos a subir. En general usaremos dos opciones:

  - El repositorio local de maven: *$USER_HOME/.m2/repository*.

    En este caso hay que tener en cuenta que la dependencia sólo estará disponible para nuestro usuario en local. Suele emplearse para pruebas antes de subir la dependencia al

repositorio de gvSIG.

- El repositorio maven de gvSIG: dav:https://devel.gvsig.org/m2repo/j2se

  Una vez subido a este repositorio ya estará disponible para el resto de desarrolladores.

Para que el deploy funcione se ha tenido que configurar previamente el fichero settings.xml tal y como se indica en el documento "Configuración inicial de maven" de la guía del desarrollador, apartado "Acceso de escritura al repositorio maven de gvSIG".

A modo de ejemplo, supongamos que tenemos un jar de la librería JEP en su versión 2.4.0 que queremos incluir como dependencia. La instrucción completa a emplear sería:

```
mvn deploy:deploy-file -DgroupId=org.nfunk \
               -DartifactId=jep \
               -Dversion=2.4.0 \
               -Dpackaging=jar \
               -Dfile=jep.jar \
               -Durl=dav:https://devel.gvsig.org/m2repo/j2se \
               -DrepositoryId=gvsig-repository
```

-

CAPÍTULO V

RESULTADOS

A continuación se recogen y analizan críticamente los resultados obtenido de los ensayos realizados con el algoritmo de Vector de Cambios con la ayuda de tablas e imágenes. Estos resultados se muestran para cada escenario establecido en el capítulo III. Posteriormente se muestra el resultado final del desarrollo de la herramienta, explicando su uso con una pequeña guía de usuario con ayuda de ilustraciones.

5.1. Resultados obtenidos en la experimentación del algoritmo de Vector de Cambios

En el apartado actual se muestran los resultados obtenidos por el método de Vector de Cambios para detección de cambios.

Los resultados que se muestran a continuación son los obtenidos por la detección de cambios entre las épocas de 1999 y 2002, con lo que en teoría, se ha producido el mayor número de cambios posibles sobre todo en los escenarios urbanos y/o agrícolas. También se han calculado los valores de las zonas invariantes para la comparación de estas con los resultados de los cambios.

Con el vector de cambios los resultados obtenidos se representan en gráficas de dispersión donde se compara las distancias entre los píxeles (distancias espectrales) con su orientación (ángulo entre ellos).

5.1.1. Análisis de vector de cambios en zona de escenario incendiado

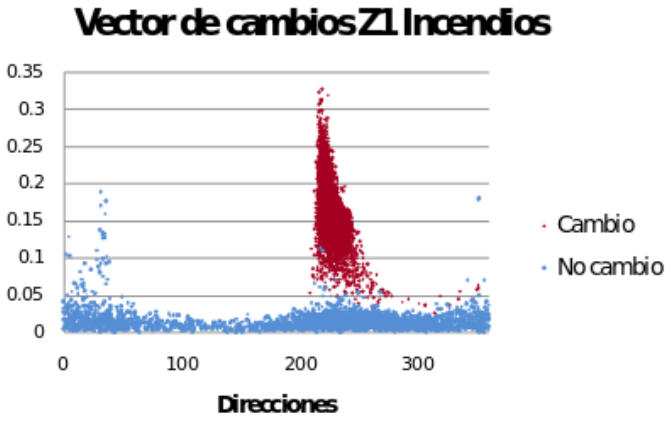


Figura 15: Diagrama de dispersión de vector de cambios en zonas incendiadas.

En la figura 15, se representa en función del módulo o distancia y la dirección, los valores de las zonas de cambio y no cambio. Como se puede observar, las zonas de cambio está muy bien definida con una agrupación de píxeles con un valor de distancias superior 0.05 y direcciones que se sitúan entre los 200º y 250º sexagesimales, mientras que los valores de las direcciones de las zonas sin cambios se sitúan sobre todo el rango – desde 0º a 360º - y las distancias rara vez son superiores a 0.05.

Las estadísticas de la zona de cambio se muestran en la Tabla 15. En ella se puede ver que la media de las direcciones está en torno a los 228º con una desviación de 9.71º. Si se deseara automatizar el proceso, se podría seleccionar directamente la zona de cambio centrándose en esa zona del diagrama, obteniendo el resultado del cambio sin considerar las zonas invariantes de la imagen.

	Direcciones	***Distancias***
Max	351.21	0.33
Min	209.11	0.02
Media	227.50	0.16
Desviación	9.71	0.04

Tabla 3: Estadísticas básicas del vector de cambios de zonas incendiadas de zonas de cambio.

5.1.2. Análisis de vector de cambios en zonas de escenario agrícola

Para las zonas agrícolas de la zona 1, el diagrama presentado (figura 16) muestra mayor dispersión en las zonas de cambio. Aún así, se podría concretar cierta área del gráfico con el que considerar una zona de cambio ya que la mayoría de estos píxeles están situados dentro de estos umbrales. En la se muestran las estadísticas básicas del gráfico.

-

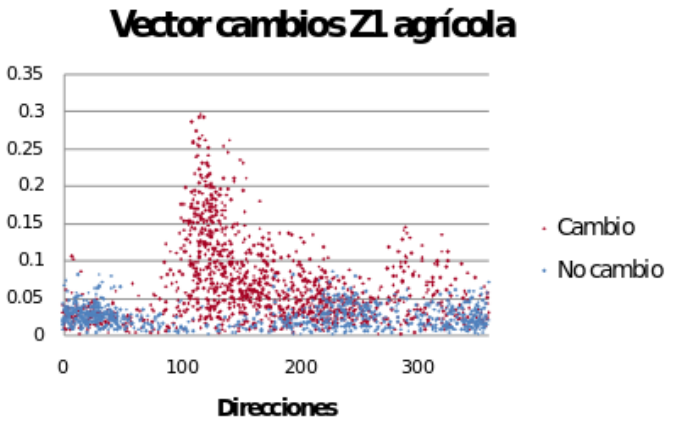


Figura 16: Diagrama de dispersión de vector de cambios en zona1 del escenario agrícola.

	Direcciones	Distancias
Max	359.54	0.30
Min	0.24	0.00
Media	171.21	0.07
Desviación	80.08	0.06

Tabla 4: Estadísticas básicas del vector de cambios de la zona 1 agrícola de zonas de cambio.

Los valores de las direcciones, a pesar de que se tener valores por todo el rango, están centrados alrededor de los 170º, lo que da una idea por donde se sitúan los cambios. Cercanos a este valor, se encuentran los picos de mayor distancia, indicando dónde se han producido los mayores cambios. Viendo el gráfico y las estadísticas, los cambios en esta zona 1 agrícola se sitúan entre los 100 y los 200 grados, donde las distancias y densidades de puntos son mayores.

La dispersión que se observa en este diagrama (figura 16) es el reflejo del tamaño de parcelas en contraste con el tamaño del píxel Landsat. En la zona 2 de cambios agrícolas, al ocurrir justo lo contrario, se puede observar en la figura 17 las agrupaciones bien localizadas en dos núcleos diferentes correspondientes a los dos tipos de cambio (de menos a más húmedo y viceversa).

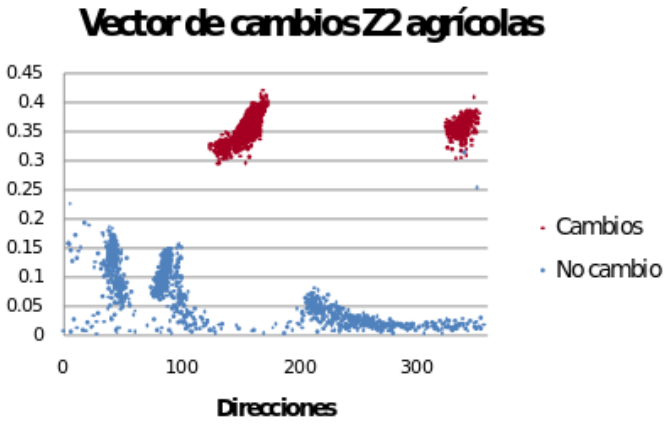


Figura 17: Diagrama de dispersión de vector de cambios en zona 2 del escenario agrícola.

Con este diagrama se puede definir correctamente los dos tipos de cambio para la zona de cultivos, con direcciones que están entre los 120º y 180º para un tipo de cambio y 320º y 360º para el otro

-

tipo, con distancias iguales o superiores a 0.3. El inconveniente de esta distribución es que las estadísticas básicas de los cambios no reflejan bien las zonas ya que, al haber dos núcleos, los valores de meda y desviación no se centran en ninguno. En cambio, los valores medios y de las distancias son totalmente aceptables, con una valores mínimos de 0.3, con un mínimo solape con los valores máximos de las zonas de no cambio del mismo gráfico.

	Direcciones	Distancias
Max	352.38	0.42
Min	123.87	0.29
Media	214.59	0.35
Desviación	85.73	0.02

Tabla 5: Estadísticas básicas del vector de cambios de la zona 2 agrícola de zonas de cambio.

	Direcciones	Distancias
Max	356.61	0.31
Min	0.10	0.00
Media	126.17	0.08
Desviación	90.41	0.05

Tabla 6: Estadísticas básicas del vector de cambios de la zona 2 agrícola de zonas de no cambio.

5.1.3. Análisis de vector de cambios en zona de escenario urbano

En la zona urbana se pueden distinguir dos núcleos diferentes de cambio (Figura 18), el primero situado entre los 0º y los 60º y el segundo entre los 170º y los 210º. En el diagrama se puede observar que los valores de las distancias de cambio se encuentran en el rango de las zonas de no cambio, dificultando así la selección de un umbral mínimo para las zonas de cambio. A pesar de ello, este método es con el que mejor se distinguen los cambios urbanos dada la problemática del tamaño de los píxeles en contraste con el tamaño del cambio.

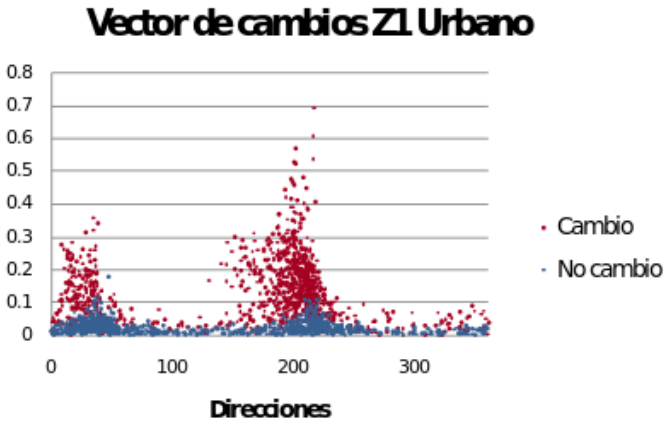


Figura 18: Diagrama de dispersión de vector de cambios en la zona urbana.

Con las tablas de resultados estadísticos y la Figura 18, se puede obtener un valor aproximado, no de las direcciones que se adquieren directamente del diagrama, de las distancias mínimas a partir de las cuales se obtendrían las zonas de cambio sin contar con las zonas invariantes.

	Direcciones	***Distancias***
Max	359.88	0.69
Min	0.29	0.00
Media	163.57	0.14
Desviación	82.25	0.09

Tabla 7: Estadísticas básicas del vector de cambios urbanos de zonas de cambio.

	Direcciones	***Distancias***
Max	359.79	0.18
Min	0.03	0.00

-

Media	132.46	0.02
Desviación	99.48	0.02

Tabla 8: Estadísticas básicas del vector de cambios urbanos de zonas de no cambio.

Mediante las gráficas se puede tomar el valor mínimo para las distancias siendo este valor el de la media más el de la desviación multiplicada por un factor que, dependiendo del porcentaje que se desee tomar o excluir, será mayor o menos. Se recomienda el factor con valor de 3, siendo este un buen factor de confianza si se considera una distribución normal de los valores invariantes. También habría que considerar que a mayor factor, menor número de píxeles se toman como cambio.

5.1.4. Análisis de vector de cambios en zona de escenario de humedal

El vector de cambios para la zona de humedales actúa, al igual que el resto de métodos, correctamente en su detección, generando un núcleo claramente identificable y separado de las zonas que no varían. Esto se puede observar en la Figura 19 , correspondiente al diagrama de dispersión de la zona de estudio de humedal.

En la Tabla 9 de estadísticas, se puede observar que el rango de valores de cambio está bien definido, con una desviación de tan solo 8.5º para las direcciones y un núcleo definido en 115º.

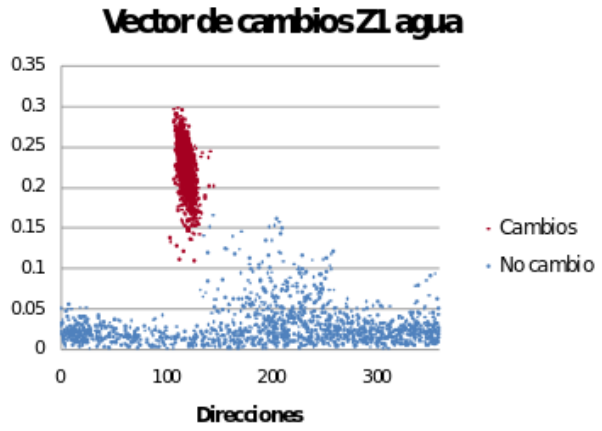


Figura 19: Diagrama de dispersión de vector de cambios en la zona de humedal.

	Direcciones	Distancias
Max	171.82	0.37
Min	96.78	0.01
Media	114.48	0.24

Desviación	8.52	0.04

Tabla 9: Estadísticas básicas del vector de cambios en humedales de las zonas de cambio.

5.2. Resultado del desarrollo de la extensión

Como resultado de la implementación se ha obtenido un plugin (o extensión) de gvSIG Desktop 2.0 llamada **org.gevsig.changedetection.app.** s una herramienta que tiene como objetivo la detección de cambios a partir de una secuencia multi-temporal de imágenes de satélite. La técnica permite obtener zonas de cambio a partir del análisis de las banda del Rojo y del Infrarrojo. Esta herramienta requiere de unos parámetros mínimos y máximos de las variables de Orientación y Distancia, característicos de cada ámbito o escenario de estudio (vegetación, incendios, agua, urbanización).

Para lanzar la herramienta de Vector de Cambios se utiliza el menú desplegable "Detección de Cambios" de la ventana principal, presionando la opción "Vector de Cambios".

5.2.1. Proceso de aplicación del algoritmo

La interfaz de usuario principal de la extensión se muestra en la figura siguiente:

-

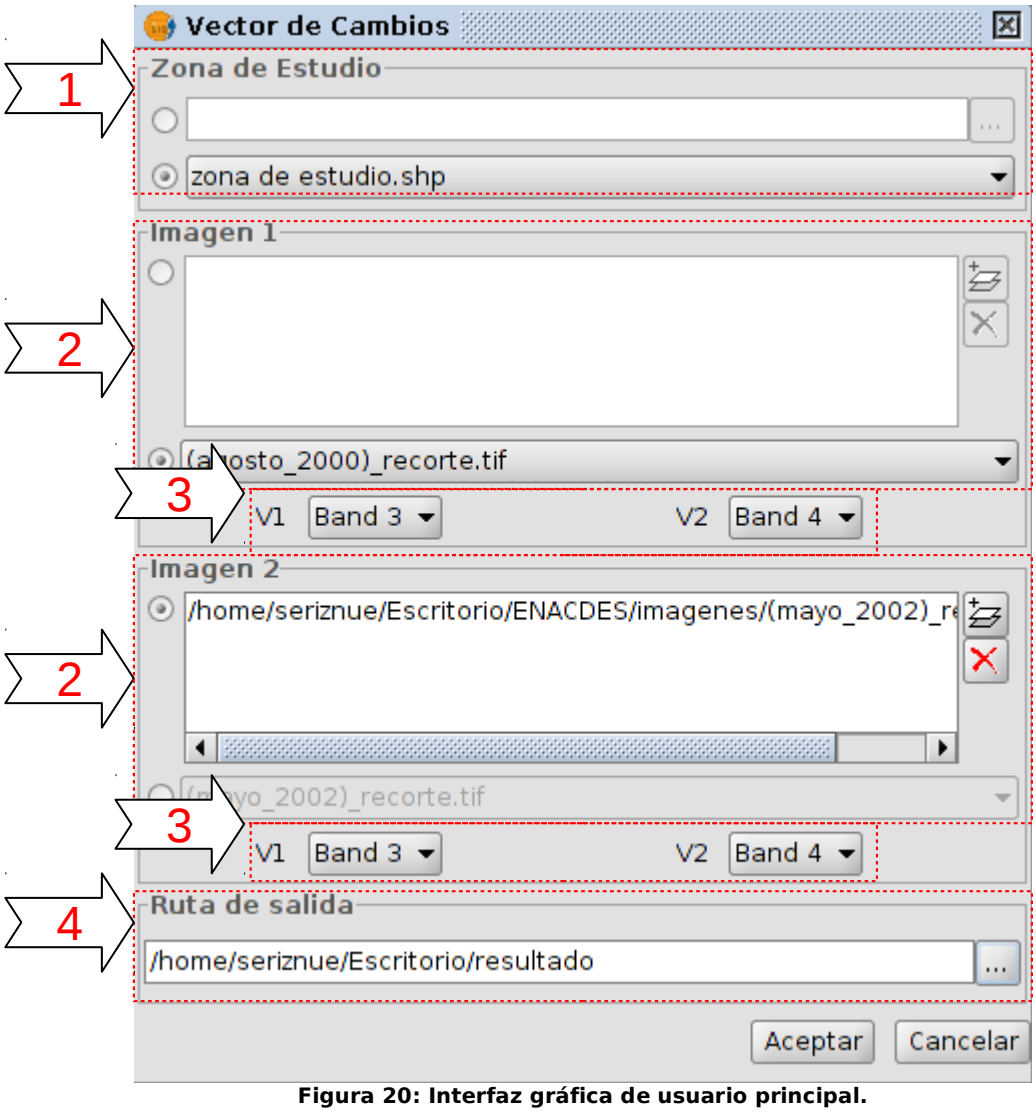


Figura 20: Interfaz gráfica de usuario principal.

1. Se elige un shape de polígonos que delimitará el área de estudio (área o áreas donde se aplicará el análisis). Esta elección es posible realizarla tanto desde la vista activa como desde disco en el selector de zona de estudio(1). Si no se elige ningún shape el análisis se realizará para toda la imagen.
2. Se eligen la imágenes satélite cuyas bandas se aplicará el algoritmo. Esta elección se realiza en los selectores de imagen 1 y 2 (2). Al igual que en el paso anterior es posible obtener las imágenes tanto desde la vista activa como desde disco. En el selector de imagen 1 se carga la imagen mas antigua y en el selector de imagen 2 la mas reciente de la secuencia multi-temporal.
3. Una vez elegidas las imágenes, automáticamente, se cargan en los combos de selección (3) las bandas numeradas por orden de la imagen (o las imágenes en caso de que se hayan cargado mas de una imagen desde disco). El usuario debe tener conocimiento del orden de las bandas de las imágenes. En los combos de selección V1 y V2 (3) se eligen la banda del Rojo y la banda del Infrarrojo respectivamente.
4. Se elige una ruta de salida donde se almacenarán las imágenes resultantes del análisis

(Distancia.tif y Orientación.tif). En caso de no escoger ninguna ruta las imágenes se guardan en la carpeta temporal.

Se lanza el proceso con el botón Aceptar o se cancela con el botón Cancelar.

A continuación, y tras el proceso de análisis se mostrará un cuadro de diálogo similar al que aparece en la siguiente figura.

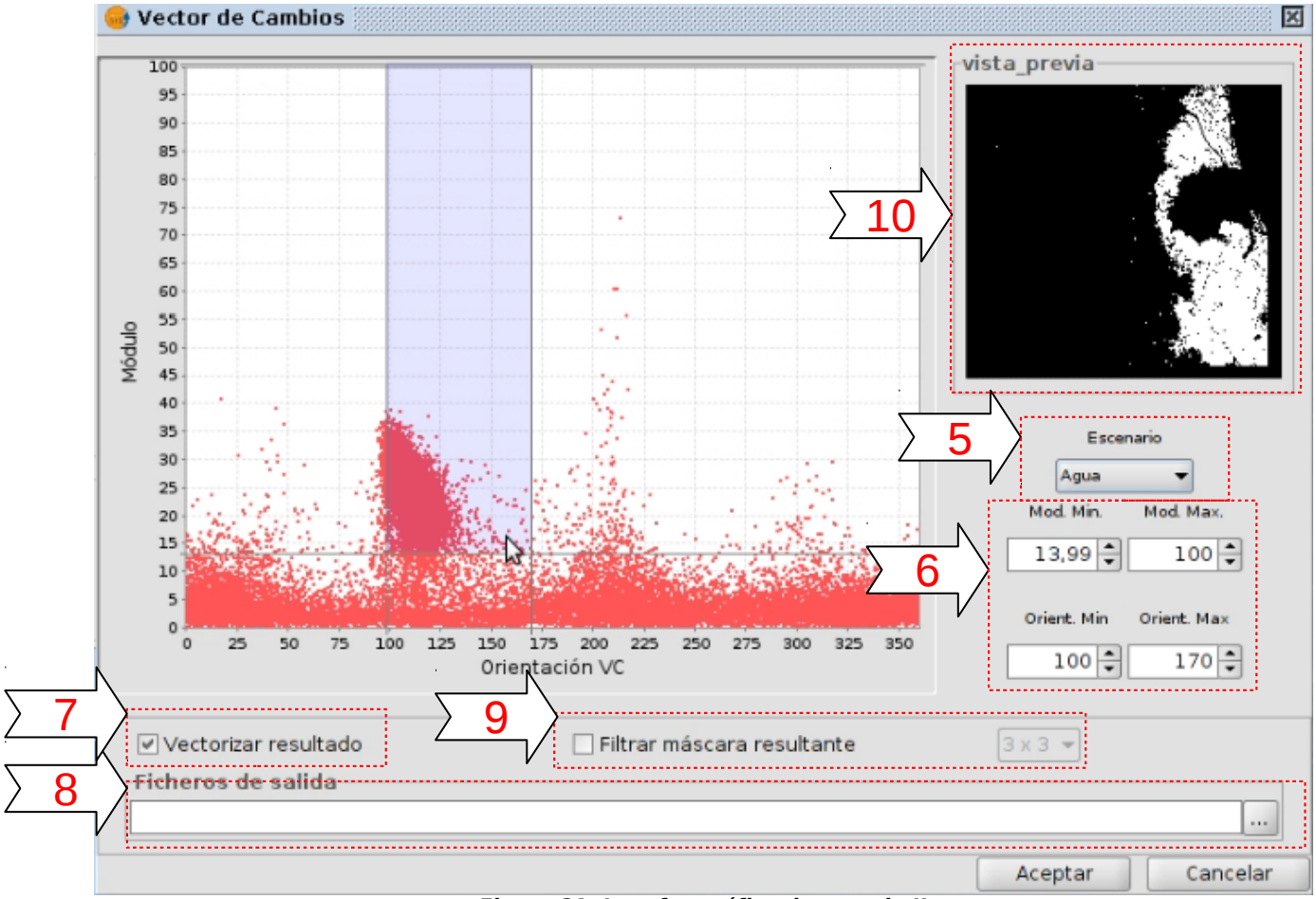


Figura 21: Interfaz gráfica de usuario II

Este cuadro de dialogo muestra una gráfica con los valores obtenidos de Orientación y Distancia de los píxeles de las zonas de estudio. En el deciden los parámetros máximos y mínimos de Orientación y Distancia que determinen los píxeles de cambio y no cambio. Para ello se siguen los siguientes pasos:

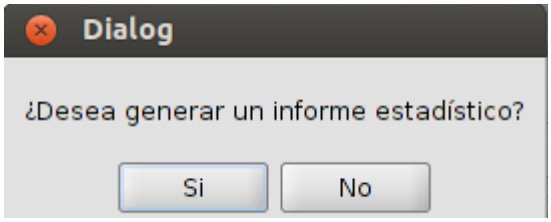
5. Se elige un escenario en el combo de selección de escenario (5). Al escoger un escenario se establecen unos parámetros orientativos para cada ámbito (vegetación, incendios, agua, urbanización o ninguno).
6. El usuario puede escoger manualmente los parámetros que mejor se adecuen a la gráfica obtenida en el análisis mediante los controles del panel (6) o directamente dibujando un rectángulo sobre la gráfica.

-

7. Se decide si se desea obtener resultado vectorizado o simplemente un resultado raster en el checkbox de Generar shp (7). En el caso seleccionar esta opción, se lanzará un proceso de vectorización y ofrecerá al usuario de obtener un informe estadístico de los resultados.
8. Se elige la ruta de salida donde se almacenará la imagen resultante (cuyos píxeles considerados como no cambio y cambio tendrán asignados los valores de 0 y 1 respectivamente) y el shape resultante de la vectorización (en caso de haber escogido la opción (7)). En caso de no escoger ninguna ruta las imágenes se guardan en la carpeta temporal.
9. Esta interfaz de usuario ofrece la posibilidad de aplicar un filtro morfológico (3x3, 5x5 o 7x7) al resultado final, para eliminar posibles píxeles aislados.
10. En (10) se puede previsualizar el resultado antes de lanzar el proceso.

Se lanza el proceso con el botón Aceptar o se cancela con el botón Cancelar.

A continuación, y tras el proceso se cargarán en la vista los resultados y se mostrará un cuadro de diálogo, solo en caso de haber seleccionado la opción de Generar shp (7), similar al que aparece en la siguiente figura.



11. Se elige si se desea generar un informe estadístico del resultado del análisis. En caso de presionar el botón Si aparece una ventana con un informe estadístico en formato html, dando la posibilidad de guardarlo en disco.

5.2.2. Ejemplo de aplicación

En la figura 23 se muestra una Imagen LandSat del área de la Albufera de Valencia, donde se han escogido dos pequeñas zonas de estudio, una de las cuales se pueden encontrar anualmente cambios notables debido a sus características naturales y a su explotación agrícola.

Figura 23: Imagen landsat y poligonos de zona de estudio

Tras aplicar la herramienta y fijar los parámetros de módulo y orientación específicos para un "escenario de agua" se obtiene el resultado tal y como aparece en las figura 24 y 25.

-

Figura 24: Mascara resultante del proceso de detección de cambios

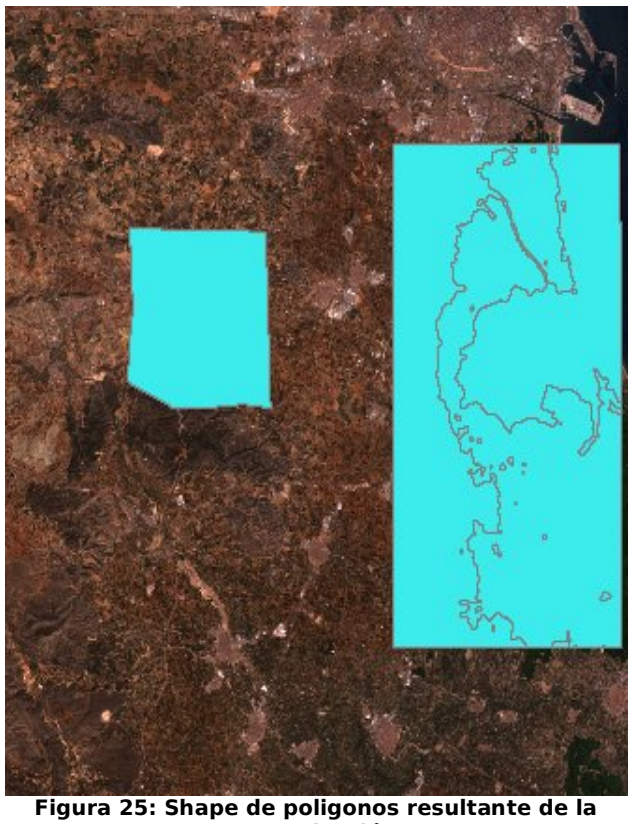


Figura 25: Shape de poligonos resultante de la vectorización

En la figura 24 se observa de color blanco todos aquellos píxeles que corresponden a cambios hídricos en la secuencia multitemporal. Como es de esperar en la zona de estudio de la izquierda no aparece ningún pixel de cambio. En la zona de la derecha los píxeles de cambio corresponde a aquellos píxeles de la imagen original que representan la zona de la Marjal de la Albufera.

En la figura 25 se observa el mismo resultado en formato vectorial. Esta capa de polígonos es la resultante de aplicar el proceso de vectorización a la capa raster de cambios.

Es importante tener en cuenta que en muchos casos las zonas de cambio no están tan definidas como aparecen en este caso. Para ello, la herramienta nos da la opción de aplicar un filtro morfológico (3x3, 5x5 o 7x7) a la mascara resultante. La función de este tipo de filtro es eliminar ruido, o mejor dicho, eliminar aquellos píxeles aislados que darían lugar a pequeños polígonos insignificantes tras aplicar el proceso de vectorización. En la figura 26 se muestra el efecto que tiene la aplicación del filtro morfológico.

Figura 26: Mascara resultante del proceso sin filtrar, resultado con filtro 3x3, resultado con filtro 5x5 y resultado con filtro 7x7, respectívamente

Por último se muestra en la siguiente figura el aspecto del informe estadístico que se obtiene al final del proceso (en caso de que el usuario lo solicite).

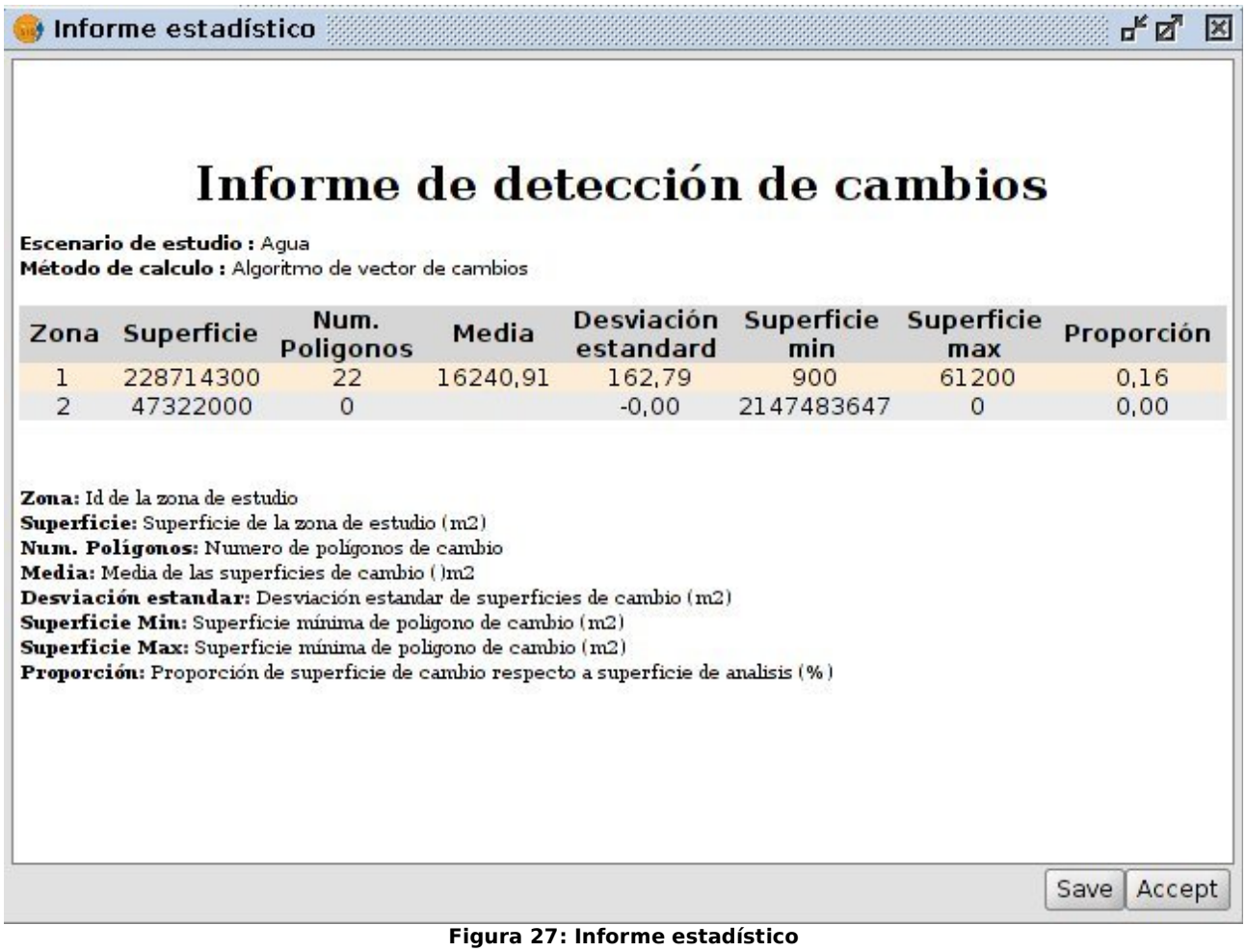


Figura 27: Informe estadístico

-

CAPÍTULO VI

CONCLUSIONES

Las técnicas de seguimiento de la evolución del territorio son de gran importancia para la correcta gestión de los diferentes entornos, tales como agrícola, urbano, forestal, etc. Por otra parte, en los últimos años, gracias a la creciente disponibilidad de imágenes satélites comerciales con una resolución espacial muy alta y la cobertura periódica (1C/1D IRS, Ikonos, QuickBird, etc.) existe una amplia gama de aplicación y técnicas para la gestión y seguimiento en el campo de la detección de cambios en el medio ambiente.

La detección de cambios en la secuencia multi-temporal de imágenes de satélite es una de las aplicaciones más importantes de la teledetección. La comparación multitemporal de imágenes se ha utilizado principalmente para la detección de cambios en la cobertura terrestre, para seguir la evolución de las áreas forestales, áreas quemadas, los desastres naturales, los recursos naturales, el crecimiento urbano, etc

Para este trabajo se considera que el algoritmo de vector de cambios es el algoritmo mas efectivo, ya que solo depende de dos variables – bandas R e IR – los resultados de cambios se obtienen directamente, y no como en otro tipo de algoritmos(como por ejemplo Componentes Principales).

A pesar de que el vector de cambios presente buenos resultados para conseguir la automatización en la detección de cambios y depende de un menor número de variables que influyen sobre ellos, se considera necesaria la aplicación de ajustes radiométricos para la normalización de los datos de las imágenes para conseguir unos resultados óptimos, dando por hecho que el registro entre imágenes ya se ha realizado correctamente en las imágenes Landsat.

En cuanto a la plataforma de desarrollo, quizá gvSIG Desktop actualmente y en un futuro cercano, no sea una herramienta de Teledetección que esté a la altura de softwares propietario como puedan ser ENVI, Erdas o cualquier otro, pero si puedo decir que gracias a la comunidad de gvSIG, se convertirá en una gran herramienta, destacando sobre el resto su coste para el usuario, que continuando con la filosofía actual, será 0. Por ello, pongo de manifiesto, que gvSIG será una gran herramienta y sobre todo el hecho de ser un software libre multiplataforma, abrirá fronteras y no las cerrará, ya que no será necesario disponer de un sistema operativo determinado, para instalar un software concreto, como ocurre con otros SIG.

Por último destacar que la utilización de software libre ya no es una promesa, es una realidad y se utiliza en sistemas de producción por algunas de las empresas tecnológicas mas importantes como IBM, SUN Microsystems, Google, Hewlett-Packard, etc. Podemos augurar sin lugar a dudas un futuro crecimiento de su empleo y una consolidación bien merecida.

-

CAPÍTULO VII

FUTUROS TRABAJOS

Actualmente tengo el placer de estar participando en el programa de becas para estudiantes Google Summer of Code 2012, donde fui aceptado mediante una propuesta de proyecto (http://seriznue.wordpress.com/category/osgeo/) que consiste en la continuidad de este trabajo. El objetivo es el desarrollo de una extensión de gvSIG Desktop 2.0, con el mismo propósito que el presente proyecto, pero aplicando el algoritmo de Descomposición de Componentes Principales.

Semanalmente publico un pequeño informe sobre los avances del proyecto en mi blog personal: http://seriznue.wordpress.com/category/gsoc-2012-reports/

Además, este proyecto no terminará aquí, ya que existen otros algoritmos para la detección de cambios, como por ejemplo NDVI (*The Normalized Difference Vegetation Index*).

CAPÍTULO VIII

REFERENCIAS

Lunetta. R.S. y Elvidge. C.D. (Eds.). 1999. *Remote Sensing Change Detection: Environmental Monitoring Methods and Applications*. Taylor & Francis.

Duveiller, G., Defourny, P., Desclée, B. y Mayaux, P., 2008. *Deforestation in Central Africa: Estimates at regional, national and landscape levels by advanced processing of systematically-distributed Landsat extracts.* Remote Sensing of Environment 112(5), 1969-1981.

Diermayer, E. y Hostert, P., 2007. Assessing post-socialist urban change with Landsat data: Case Study Berlin, Germany. Urban Remote Sensing Joint Event, 11-13 April 2007. pp.1-4.

Yuan, F., Sawaya, K. E., Loeffelholz, B.C. y Bauer, M.E., 2005. *Land cover classification and change analysis of the Twin Cities (Minnesota) Metropolitan Area by multitemporal* Landsat remote sensing. Remote Sensing of Environment 98 (2-3), 317-328.

Loveland, T.R., Cochrane, M.A. y Henebry, G.M., 2008. Landsat still contributing to environmental research. Trends in Ecology and Evolution 23 (4), 182-183.

Phua, M., Tsuyuki, S., Furuya, N. y Lee, J.S., 2008. *Detecting deforestation with a spectral change detection approach using multitemporal Landsat data: A case study of Kinabalu Park, Sabah, Malaysia.* Journal of Environmental Management 88(4), 784-795

MacAlister, C. y Mahaxay, M., 2009. Mapping wetlands in the Lower Mekong Basin for wetland resource and conservation management using Landsat ETM images and field survey data. Journal of Environmental Management 90 (7), 2130-2137.