UNIVERSIDAD OBERTA DE CATALUNYA

Titulación

Ingeniería Informática de Sistemas

PROYECTO FIN DE CARRERA

Diseño y Desarrollo de un cliente Web Processing Services (WPS) para gvSIG

Alumno: Mariano Velamazán Martínez
Dirigido por: Jesús de Diego

CURSO 2008-09 (Febrero)

Índice

Índice 3
1. Introducción 4
2. Planificación: tareas 5
3. Agenda de trabajo. 7
5. Presentación de los SIG. 14
6. Presentación de gvSIG 18
7. GvSIG desde el punto de vista del desarrollador 20
8. Estándares del OGC. 22
9. Web Processing Services (WPS). 23
10. Requisitos de la aplicación 26
11. Casos de uso. 28
12. Análisis de la aplicación 31
13. Diseño. 33
14. Implementación. 35
15. Conclusiones 39
Anexo 1. Balance de tareas realizadas. 41
Anexo 2. Bibliografía 43
Anexo 3. Tareas pendientes y futuras mejoras 46

1. Introducción

El presente documento es una exposición de los trabajos realizados dentro de la asignatura Proyecto Fin de Carrera de la Titulación Ingeniería Técnica en Informática de Sistemas de la Universidad Oberta de Catalunya, para conseguir los objetivos marcados en su plan docente. Dichos objetivos son:

1. Conocer la herramienta gvSIG y configurar su entorno de desarrollo en el IDE Eclipse.
2. Conocer los estándares más extendidos del Open Geospatial Consortium (OGC) y en especial del Web Processing Services.
3. Analizar, diseñar y desarrollar un cliente capaz de consumir servicios WPS.

Para conseguir dichos objetivos en general, comenzamos con un plan de trabajo del que se presenta un resumen de los puntos más importantes. La función de este plan de trabajo es identificar las tareas, organizarlas y distribuirlas en el tiempo.

Para el primer objetivo, continuamos con una presentación de los Sistemas de Información Geográfica en general y de la aplicación gvSIG en particular ya que como se ve en dicho objetivo es la herramienta sobre la que vamos a trabajar.

Para el objetivo 2 hemos dedicado los apartados “Estándares del OGC” y “Web Processing Services”. En ellos se ofrece una visión del Open Geospatial Consortium y sus estándares más importantes y a ésta le sigue una descripción detallada del estándar Web Processing Services (WPS).

El cliente WPS debe servir para permitir la comunicación entre gvSIG y un servidor WPS. Los siguientes puntos de este documento están encaminados a analizar cómo debe ser esa comunicación y a implementar una solución técnica.

Por último, ya que el Proyecto Fin de Carrera también ha de servir como síntesis de todo lo aprendido durante los estudios de Ingeniería Técnica de Informática de Sistemas, se ofrece un apartado de conclusiones donde se valoran los conocimientos previos, los adquiridos y la consecución de los objetivos expuestos aquí en particular y en el plan docente en general.

2. Planificación: tareas

En este punto se ofrece una descripción general de las principales tareas definidas en este plan para dividir y organizar el proyecto final de carrera. Comenzaremos con un listado de todas ellas:

1. Definición
2. Planificación
3. Configurar entorno de desarrollo en el IDE Eclipse.
4. Introducción a los SIG. Usos, utilidades y ejemplos. Conocer la herramienta gvSIG
5. Conocer los estándares más extendidos del Open Geospatial Consortium (OGC) y en especial del Web Processing Services.
6. Analizar, diseñar y desarrollar un cliente capaz de consumir servicios WPS.
7. Elaboración de la memoria
8. Presentación
9. Debate virtual

De un total de nueve tareas, las dos primeras son totalmente preparativas. Los objetivos principales de la "Definición" son las primeras lecturas para familiarizarnos con el mundo de los SIG y con el proyecto encargado en particular. Los puntos más importantes a cubrir por la planificación serán los expuestos (objetivos, tareas, calendario, disponibilidad de horas, etc.)

A esas dos tareas le siguen otras tres teórico prácticas de preparación técnica del proyecto. Como se expone en el enunciado la función de la tarea 3 es preparar el entorno básico de desarrollo en Java, Eclipse.

Respecto a la tarea 4, su función principal es la de instalar gvSIG y familiarizarnos con los SIG. Esta tarea es más importante que la anterior porque es más nueva para nosotros y más específica del proyecto por eso en el plan tendrá más horas dedicadas. El fin de esta tarea es saber utilizar básicamente gvSIG.

El objetivo de la tarea 5 es obtener un conocimiento teórico y práctico de los SIG en Internet, en concreto de los Web Processing Services tal y como los especifica el Open Geospatial Consortium. Esta tarea es posterior a la 4 porque para ella es necesario que conozcamos ya el programa gvSIG y estemos familiarizados con él. Esta tarea es fundamental porque termina de centrarnos en la parte práctica del proyecto ya que con ella concluyen las preparaciones técnicas y teóricas.

A partir de aquí, comienza el núcleo duro del proyecto que es la fase de resolución del proyecto solicitado: un clliente WPS para gvSIG. Es la tarea más difícil y la que nos llevará más tiempo. También ha sido encadenada a las anteriores porque sin ellas no podríamos empezar a trabajar. De la tarea 6 y de sus subtareas debe salir un producto que funcione, que cumpla con los

requisitos y dentro del plazo previsto. Para llevarla a cabo pasaremos por todas las etapas convencionales del desarrollo de software: casos de uso, diseño, implementación, errores, compilación final, etc.

Las dos tareas siguientes son las más importantes en la medida en que su función es servir de presentación y conclusión de todo el proyecto. Son la Memoria y la Presentación Virtual.

Respecto a la Memoria, además de recoger un "registro" del desarrollo del proyecto y una exposición de sus características más importantes, también se expondrá una evaluación personal y unas conclusiones sobre cómo ha evolucionado el proyecto. Para que el proyecto no quede como un producto estancado se expondrán posibles desarrollos posteriores y una lista de errores conocidos del software.

La Presentación será una tarea muy cuidada porque la imagen con que presentemos el proyecto influirá en la percepción que el público tendrá de él. Haremos un guión previo para dotarlo de estructura y coherencia narrativa.

La última tarea está más enfocada a una valoración de la asignatura y de los proyectos con los consultores. Suponemos también que tendremos que aclarar algunos puntos que no han quedado claros en la Memoria o en la Presentación. Es una tarea de conclusión, de cierre y de despedida que tendrá lugar después de haber entregado la Memoria y la Presentación y durará hasta la tercera semana de enero.

3. Agenda de trabajo.

Una vez establecidas las tareas, expondremos la disponibilidad horaria y el total de horas posible que se podrán dedicar al proyecto. Hemos intentado ofrecer una visión realista conociendo nuestras limitaciones dejando una amplia horquilla de horas "extra" con respecto a la estimación de horas a dedicar a la asignatura y una posible ampliación extra los fines de semana para casos imprevistos y emergencias.

Concretamente se dedicarán 7 horas semanales que se repartirán de la siguiente manera:

Martes y viernes: 2 horas diarias
Miércoles: 3 horas diarias

Desde el 15 de noviembre hasta el final de la asignatura (5 de enero) en caso de que no se cumpla con el plan previsto se añadirán hasta 8 horas semanales durante el fin de semana (4 horas el sábado y 4 el domingo). Excepto el 24, 25, 31 de diciembre y 1 de enero.

En total serán entre 101 horas y 193 horas dependiendo de las necesidades del proyecto. En la Tabla 1, se desglosan pormenorizadamente.

Semana	Fecha inicio	Dedicación
1	22/09/2008	7 horas
2	29/09/2008	7 horas
3	06/10/2008	7 horas
4	13/10/2008	7 horas
5	20/10/2008	7 horas
6	27/10/2008	7 horas
7	03/11/2008	7 horas
8	10/11/2008	7 horas
9	17/11/2008	7 horas
10	24/11/2008	7 horas
11	01/12/2008	7 horas
12	08/12/2008	7 horas

13	15/12/2008	7 horas
14	22/12/2008	5 horas
15	29/12/2008	5 horas
	TOTAL	101 horas

Tabla 1. Distribución de horas.

Una vez expuesta la disponibilidad horaria pasamos a detallar el diagrama de Gantt. En la siguiente Tabla 2, ofrecemos una sola vista general del proyecto como ilustración esquemática de lo visto en estos apartados.

Id	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
1	Definición	2,5 días	17/09/2008	24/09/2008	
7	Planificación	2,67 días	24/09/2008	01/10/2008	1
17	Configurar entorno de desarrollo en	1,33 días	01/10/2008	07/10/2008	7
22	Introducción a los SIG. Usos, utilidade	1,83 días	07/10/2008	10/10/2008	17
27	Conocer los estándares más extendi	3 días	10/10/2008	21/10/2008	22
36	Analizar, diseñar y desarrollar un clie	11,33 días	21/10/2008	21/11/2008	27
44	Memoria	7,67 días	25/11/2008	16/12/2008	36
57	Presentación	5 días	17/12/2008	06/01/2009	44
65	Debate virtual	7,67 días	06/01/2009	17/01/2009	57

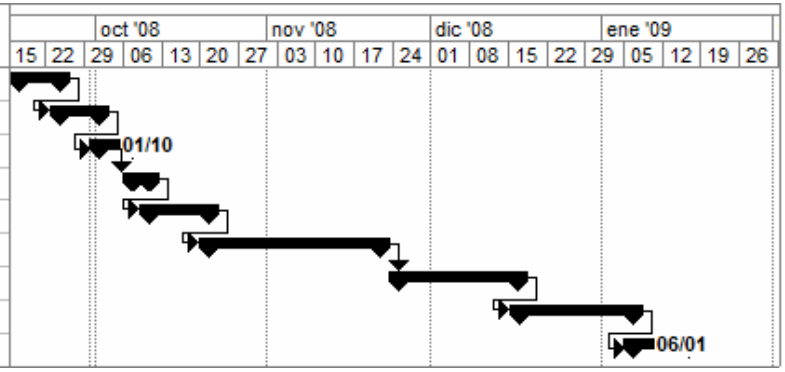

Tabla 2. Diagrama de Gantt

4. Calendario de trabajo

Ya que el diagrama de Gantt lo hemos expuesto de manera general, exponemos más abajo la Tabla 3, con el desarrollo de todas las tareas, subtareas y sus fechas para tener una visión global (aunque sin gráfico visual) del proceso de puesta en práctica del proyecto.

Id.	Tarea	Actividad	Duración	Inicio	Fin
1	Definición		3d	17/09/2008	24/09/2008
		Lecturas y primera toma de contacto	3h	17/09/2008	17/09/2008
		Otra documentación aportada por el consultor	3h	19/09/2008	23/09/2008
		Objetivos	1h	23/09/2008	23/09/2008
		Participantes	1h	23/09/2008	23/09/2008
		Elementos y estimaciones	1h	24/09/2008	24/09/2008
2	Planificación		3d	24/09/2008	01/10/2008
		Requerimientos y necesidades	0,5h	24/09/2008	24/09/2008
		Objetivos	0,5h	24/09/2008	24/09/2008
		Tareas	1h	24/09/2008	24/09/2008
		Organización	1h	24/09/2008	26/09/2008
		Control y revisión	0,5h	26/09/2008	26/09/2008
		Errores	1h	26/09/2008	26/09/2008
		Entrega previa al consultor	0,5h	30/09/2008	30/09/2008
		Posibles correcciones	3h	30/09/2008	01/10/2008
		Entrega definitiva	1h	01/10/2008	01/10/2008
3	Configurar entorno de desarrollo en el IDE Eclipse.		2d	01/10/2008	07/10/2008
		Lectura de la documentación	1h	01/10/2008	01/10/2008
		instalación última versión de Eclipse	1h	01/10/2008	03/10/2008
		instalación cliente subversión para Eclipse	1h	03/10/2008	03/10/2008
		Pruebas y toma de contacto con todo el entorno	3h	03/10/2008	07/10/2008
4	Introducción a los SIG. Usos, utilidades y ejemplos. Conocer la		4,5d	07/10/2008	10/10/2008

	herramienta gvSIG				
		Descargar la herramienta	0,5h	07/10/2008	07/10/2008
		Leer documentación de instalación	2h	07/10/2008	07/10/2008
		instalar herramienta	2h	08/10/2008	08/10/2008
		Pruebas y primera toma de contacto con su funcionamiento e interfaz	9h	08/10/2008	10/10/2008
5	Conocer los estándares más extendidos del Open Geospatial Consortium (OGC) y en especial del Web Processing Services.		7,67d	10/10/2008	21/10/2008
		Descargar Tomcat	1h	10/10/2008	10/10/2008
		Leer documentación de instalación	1h	10/10/2008	14/10/2008
		instalación Tomcat	2h	14/10/2008	14/10/2008
		Descargar WPS 52North	1h	14/10/2008	14/10/2008
		Leer documentación de instalación	1h	15/10/2008	15/10/2008
		instalación WPS 52North	2h	15/10/2008	15/10/2008
		Pruebas con 52North	9h	15/10/2008	17/10/2008
PEC		Ver y probar otros clientes WPS de otros entornos	6h	17/10/2008	21/10/2008
6	Analizar, diseñar y desarrollar un cliente capaz de consumir servicios WPS.		25d	21/10/2008	21/11/2008
		Elaboración del catálogo de requisitos	6h	21/10/2008	22/10/2008
		Casos de uso	12h	22/10/2008	24/10/2008
		Diseño de la aplicación	15h	28/10/2008	04/11/2008
		Feedback del consultor	2 días	21/10/2008	24/10/2008
		implementación	30h	05/11/2008	18/11/2008
		Pruebas	9h	19/11/2008	19/11/2008

PEC		Generación y configuración de un producto instalable y un manual de usuario	3h	21/11/2008	21/11/2008
7	Memoria		14,33d	25/11/2008	16/12/2008
		Redactar toda la documentación generada durante el proyecto	9h	25/11/2008	28/11/2008
		Posibles desarrollos posteriores al proyecto	3h	28/11/2008	02/12/2008
		Esquemas y gráficos explicativos / ilustrativos	9h	02/12/2008	05/12/2008
		Conclusiones y evaluación	3h	05/12/2008	05/12/2008
		Documentación de errores conocidos	3h	09/12/2008	09/12/2008
		Lista de tareas por hacer	3h	09/12/2008	09/12/2008
		Feedback del consultor		25/11/2008	28/11/2008
		Estilo, diseño y capítulos de organización (portada, índice, abstract, etc.)	6h	10/12/2008	10/12/2008
		Repaso general	3h	12/12/2008	12/12/2008
		Entrega previa al consultor	0,5h	12/12/2008	12/12/2008
		Posibles correcciones	3h	12/12/2008	16/12/2008
		Entrega definitiva	0,5h	16/12/2008	16/12/2008
8	Presentación		5,5d	17/12/2008	06/01/2009
		Guión de la presentación. Destacar los puntos fuertes del proyecto. Exponer de manera clara y sencilla el funcionamiento y posibilidades del cliente diseñado.	3h	17/12/2008	17/12/2008
		Captura de pantallas y de acciones “demo”	3h	19/12/2008	23/12/2008
		implementación en Flash	6h	23/12/2008	02/01/2009
		Repaso general	1h	02/01/2009	02/01/2009
		Entrega previa al consultor	0,5h	05/01/2009	05/01/2009
		Posibles	2,5h	05/01/2009	06/01/2009

		correcciones			
		Entrega definitiva	0,5h	06/01/2009	06/01/2009
9	Debate virtual		1d	06/01/2009	17/01/2009
		Visitar el foro diariamente un par de veces (mañana y tarde) para comprobar si hay preguntas o temas de discusión	3h	09/01/2009	10/01/2009
		Contestar de manera clara y concisa	3h	06/01/2009	17/01/2009

Tabla 3. Desglose de tareas.

5. Presentación de los SIG.

La redacción de este apartado está basada en el contenido ofrecido por el módulo 4 “Introducción a los sistemas de información geográfica” (2), por el módulo 6 “Nuevas tendencias en SIG” (6) y por Wikipedia (3).

Aunque la definición de SIG no está cerrada y depende del momento y la utilización que del sistema se haga, podemos intentar llegar a un mínimo común diciendo que son aplicaciones informáticas que gestionan datos espaciales.

La obtención de los datos se hace a través de:

- Sistemas globales de navegación por satélite
- Teledetección y sensores remotos
- Los propios Sistemas de Información Geográfica

La aplicación SIG, combinada con el hardware necesario, los datos, el usuario y los procedimientos es responsable de procesar, almacenar, analizar y distribuir la información sobre los territorios de la Tierra para ayudar en la toma de decisiones estratégicas relacionadas con los problemas y la planificación territorial.

Los SIG tienen numerosas funciones en diferentes tareas: medio ambiente, planificación urbana, cartografía, logística, marketing, etc.

Dada la naturaleza de este Proyecto Fin de Carrera, se detallan a continuación algunos puntos referentes a los datos y su estructura dentro de un SIG general.

Los modelos de datos son objeto relacionales. Los datos geográficos son la representación digital de entidades, objetos o fenómenos que ocurren sobre la superficie de la Tierra o cerca de ésta. Las bases de datos geográficos deben almacenar y permitir la exportación, modificación y consulta de entidades que recopilen información tanto geográfica como alfanumérica. Un elemento geográfico es, pues, el conjunto de la geometría que lo representa sobre el territorio y los atributos alfanuméricos que describen las propiedades. (4)

El sistema permite separar la información en diferentes capas temáticas y las almacena independientemente, permitiendo trabajar con ellas de manera rápida y sencilla, y facilitando la posibilidad de relacionar la información existente a través de la topología de los objetos, con el fin de generar otra nueva que no podríamos obtener de otra forma.

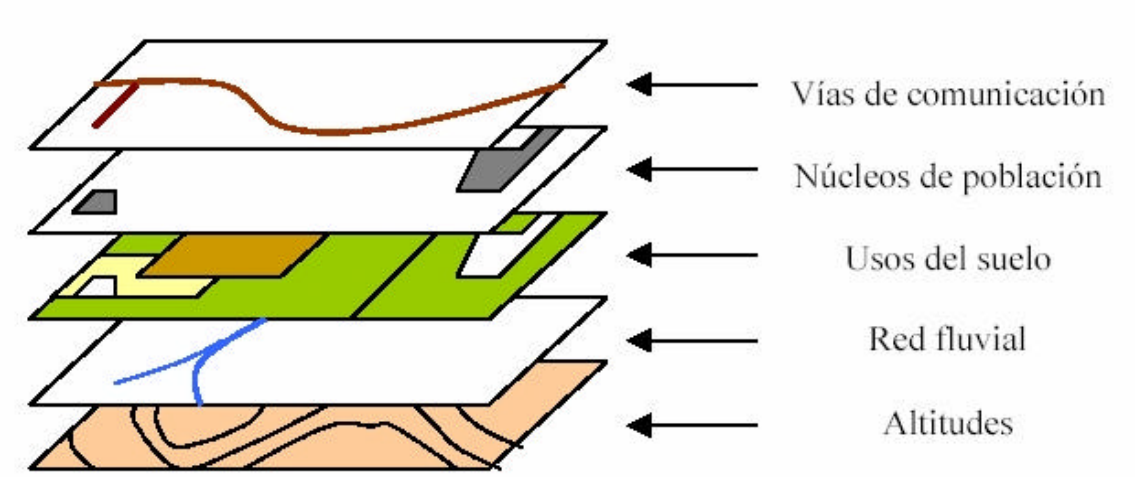

Figura 1. Ejemplo de información geográfica combinada en capas. Fuente: Wikipedia (3).

Los datos se introducen a partir de sobre todo, la digitalización de datos existentes y de imágenes orto-rectificadas. Dichos datos se almacenan de dos formas: según el modelo raster o según el modelo vectorial.

Los datos raster se componen de filas y columnas de celdas. Los datos raster pueden ser imágenes (imágenes raster), con un valor de color en cada celda (o píxel). Otros valores registrados para cada celda pueden ser un valor discreto, como el uso del suelo, valores continuos, como temperaturas, o un valor nulo si no se dispone de datos.

Figura 2. Ejemplo de imagen ráster.

En los datos vectoriales, el interés de las representaciones se centra en la precisión de localización de los elementos geográficos sobre el espacio. Se usa este modelo vectorial cuando los fenómenos a representar son discretos, es decir, de límites definidos. Cada una de estas geometrías está vinculada a una fila en una base de datos que describe sus atributos. Por ejemplo, una base de datos que describe los lagos puede contener datos sobre la batimetría del lago, la calidad del agua o el nivel de contaminación.

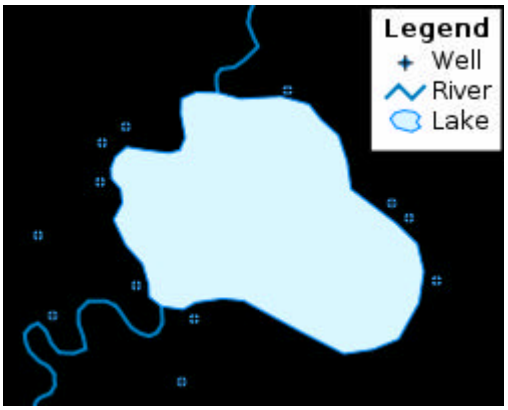

Figura 3. Ejemplo de imagen vectorial. Fuente Wikipedia (3).

Todas las capas de información vectorial tienen su “Tabla de atributos”. Cada elemento gráfico de una determinada capa tiene su correspondiente registro en dicha“Tabla de atributos”.

Para modelar digitalmente las entidades del mundo real se utilizan tres elementos geométricos: el punto, la línea y el polígono.

- Puntos

Los puntos se utilizan para las entidades geográficas que mejor pueden ser expresadas por un único punto de referencia. En otras palabras: la simple ubicación. Por ejemplo, las ubicaciones de los pozos, picos de elevaciones o puntos de interés. También se pueden utilizar para representar zonas a una pequeña escala. Por ejemplo, las ciudades en un mapa del mundo estarán representadas por puntos en lugar de polígonos.

- Líneas o polilíneas

Las líneas unidimensionales o polilíneas son usadas para rasgos lineales como ríos, caminos, ferrocarriles, rastros, o líneas topográficas. De igual forma que en las entidades puntuales, en pequeñas escalas pueden ser utilizados para representar polígonos. En los elementos lineales puede medirse la distancia.

- Polígonos

Los polígonos bidimensionales se utilizan para representar elementos geográficos que cubren un área particular de la superficie de la tierra. Estas entidades pueden representar lagos, limites de parques naturales, edificios, provincias, o los usos del suelo, por ejemplo. Los polígonos transmiten la mayor cantidad de información en archivos con datos vectoriales y en ellos se puede medir el perímetro y área.

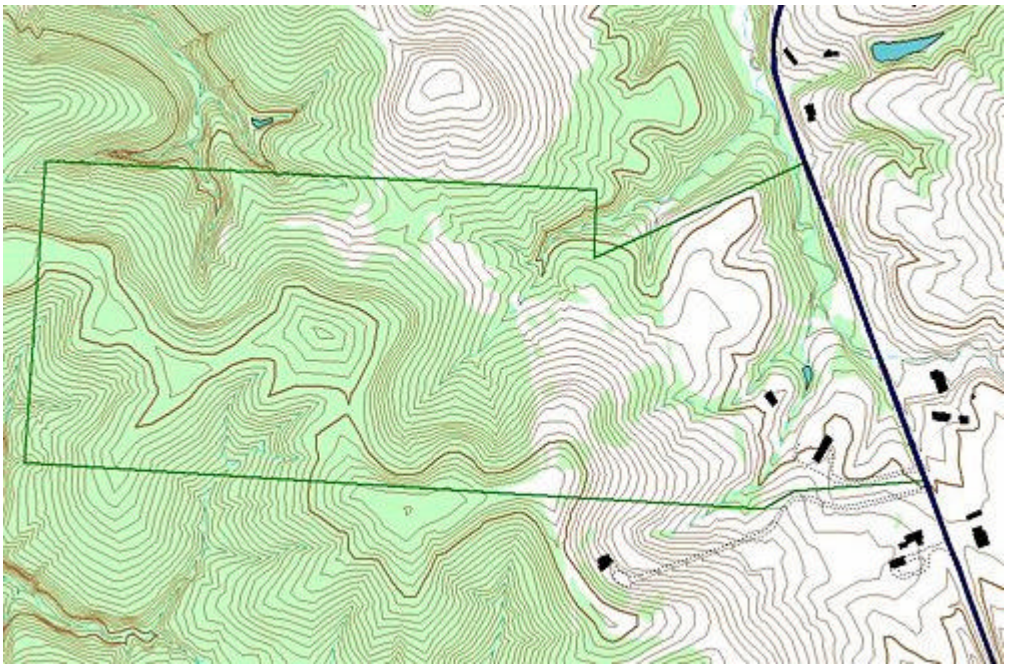

Figura 4. En esta imagen se pueden apreciar líneas y polígonos. Fuente Wikipedia (3)

Las principales cuestiones que puede resolver un Sistema de Información Geográfica, ordenadas de menor a mayor complejidad, son:

- Localización: preguntar por las características de un lugar concreto.
- Condición: el cumplimiento o no de unas condiciones impuestas al sistema.
- Tendencia: comparación entre situaciones temporales o espaciales distintas de alguna característica.
- Rutas: cálculo de rutas óptimas entre dos o más puntos.
- Pautas: detección de pautas espaciales.
- Modelos: generación de modelos a partir de fenómenos o actuaciones simuladas.

Hoy en día, gracias sobre todo a la popularidad de la web, de Google Maps y los sistemas GPS, los usuarios se han familiarizado con muchas operaciones comunes y en esto se basa el prometedor futuro de este campo.

Entre los software SIG que hemos probado están ArcGIS, GeoMedia, GRASS, JUMP, uDig o el propio gvSIG que ahora pasamos a presentar.

6. Presentación de gvSIG

Según el manual de usuario de gvSIG (10) y el curso de gvSIG (9),

"El programa gvSIG está orientado al manejo de información geográfica. Se caracteriza por una interfaz amigable y sencilla, con capacidad para acceder ágilmente a los formatos más usuales (ráster y vectoriales). gvSIG además es capaz de integrar datos en una vista, tanto locales como remotos, a través de un origen WMS (Web Map Service), WFS (Web Feature Service) WCS (Web Coverage Service) o JDBC (Java Database Connectivity).

Está orientado a usuarios finales de información geográfica, profesionales o personal de Administraciones Públicas (Ayuntamientos, Diputaciones, Consejerías o Ministerios). También resulta de especial interés para los ambientes universitarios, debido a su componente I+D+I (Investigación + Desarrollo + Innovación).

La aplicación es de código abierto, con licencia GPL (General Public License o licencia pública general) y gratuita. Se ha hecho especial hincapié desde sus inicios, en que gvSIG sea un proyecto extensible, de forma que los desarrolladores puedan ampliar las funcionalidades de la aplicación fácilmente, así como desarrollar aplicaciones totalmente nuevas a partir de las librerías utilizadas en gvSIG (siempre y cuando cumplan la licencia GPL).

En gvSIG toda la actividad se localiza en un proyecto, el cual está formado por diferentes documentos. Los documentos en gvSIG son de tres tipos: Vistas, tablas y mapas.

• **Vistas**: Son documentos donde se trabaja con datos gráficos (11).
• **Tablas**: Son documentos donde se trabaja con datos alfanuméricos (12).
• **Mapas**: Constructor de mapas que permite insertar los distintos elementos cartográficos que componen un plano (vista, leyenda, escala...) (13).

Las vistas son los documentos de gvSIG que constituyen el área de trabajo de la información cartográfica.

Dentro de una vista pueden existir distintas capas de información geográfica (hidrografía, comunicaciones, divisiones administrativas, curvas de nivel, etc.).

Las tablas son documentos que contienen información alfanumérica. Las tablas se componen de **filas o registros** (que representan cada uno de los elementos de la base de datos) y **columnas o campos** (que definen los distintos atributos de cada elemento).

• **Fila o registro**: Es la representación de los distintos elementos de la tabla.
• **Columna o campo**: Son los tipos de atributos que definen a cada elemento.
• **Celda**: La intersección de un registro y un campo es una celda. La celda es el elemento mínimo de trabajo y puede contener información.

• **Información de registros**: Informa del total de elementos (registros) que contiene la tabla.

Los documentos de tipo “Mapa” le permiten GLVHxar y combinar en una Si gina todos los elementos que desea que aparezcan en un mapa impreso.

7. GvSIG desde el punto de vista del desarrollador

El proyecto gvSIG se presenta como un framework sobre el que se pueden ir añadiendo plugins que le doten de nuevas funcionalidades.

La plataforma que proporciona gvSIG se sustenta en una arquitectura abierta en la que cada equipo que está desarrollando un plugin se puede centrar en su área de experiencia. Puede haber equipos especializados en librerías base, mientras que otros equipos se centran en desarrollar interfaces de usuario para que las herramientas de base estén accesibles para los usuarios finales.

gvSIG usa un modelo que consiste en presentar un conjunto de herramientas de forma homogénea desde el punto de vista del usuario. Las herramientas que se desarrollan se integran dentro del marco de gvSIG usando unos mecanismos ya definidos llamados plugins.

La plataforma gvSIG en sí misma está construida a modo de capas, cada una de las cuales define sus propios puntos de extensión. A su vez, cada plugin puede definir sus propios puntos de extensión. Este modelo de plugins permite a los desarrolladores añadir gran variedad de funcionalidades a la plataforma base de gvSIG, de forma que los artefactos de cada herramienta, como pueden ser los distintos tipos de capas, o botones, se presentan al usuario desde la plataforma común.

Los desarrolladores de plugins también se benefician de esta arquitectura. El framework base de gvSIG les proporciona una serie de servicios de los cuales ellos no tienen que preocuparse, pudiéndose centrar en las tareas específicas de su extensión.

gvSIG está estructurada en una serie de subsistemas los cuales están implementados como librerías y como plugins en si mismos.

La plataforma de gvSIG está conformada en su núcleo por tres subsistemas:

- gvSIG. Representa los datos geográficos manejados por Fmap. En este subsistema encontraremos las clases que implementan la mayor parte de cuadros de diálogo que utiliza la aplicación final, así como las clases de soporte a esos cuadros de diálogo. Por ejemplo, aquí se encuentran los formularios para asignar leyendas, crear mapas y vistas, definir escalas, etc.
- FMap. Es el corazón SIG de la plataforma. Incluye todas las clases necesarias para manejar objetos SIG, así como drivers y adaptadores para manejar los formatos más usados para el almacenamiento de los datos cartográficos. Dentro de esta librería encontramos clases para leer y escribir los formatos soportados, dibujar los mapas a las escalas adecuadas, asignar leyendas, definir simbologías, realizar búsquedas, consultas, análisis, etc.
- Subdriver. En este subsistema encontramos todas las clases que permiten el acceso y la gestión de los datos.

Andami es el framework sobre el que se construye gvSIG. Está diseñado para ser extensible mediante plugins (módulos que añaden nueva funcionalidad a la aplicación). Además de proveer el mecanismo de plugins, Andami da soporte a gvSIG en la composición básica de la interfaz de usuario y en la gestión de ventanas y eventos. Proporciona una simplificación importante en la gestión de estos aspectos, de forma que resulta muy sencillo crear un plugin funcional de gvSIG.

La mayor parte de la funcionalidad de gvSIG la proporcionan los plugins, que a su vez se apoyan en librerías. El propio gvSIG es un plugin de Andami, aportándole el concepto de documento (vista, tabla, mapa), la capacidad de cargar y salvar proyectos, el gestor de proyectos, y muchas otras funcionalidades.

8. Estándares del OGC.

Según Wikipedia (5), el Open Geospatial Consortium (OGC) fue creado en 1994 y agrupa a más de 250 organizaciones públicas y privadas. Su fin es la definición de estándares abiertos e interoperables dentro de los Sistemas de Información Geográfica. Persigue acuerdos entre las diferentes empresas del sector que posibiliten la interoperación de sus sistemas de geoprocesamiento y facilitar el intercambio de la información geográfica en beneficio de los usuarios. Anteriormente fue conocido como Open GIS Consortium.

Las especificaciones más importantes surgidas del OGC son:

- GML - Lenguaje de Marcado Geográfico (no confundir con Lenguaje de Marcado Generalizado, también GML)
- WFS - Web Feature Service o Servicio de entidades vectoriales que proporciona la información relativa a la entidad almacenada en una capa vectorial (cobertura) que reúnen las características formuladas en la consulta.
- WMS - Web Map Service o Servicio de mapas en la web que produce mapas en formato imagen a la demanda para ser visualizados por un navegador web o en un cliente simple.
- WCS - Web Coverage Service
- CSW - Web Catalogue Service

9. Web Processing Services (WPS).

El contenido de este subapartado está basado en los textos citados en (7) y en (8).

La especificación que a nosotros nos ocupa, Web Processing Services (WPS) define una interfaz estandarizada que facilita la publicación de procesos geoespaciales, el uso y el acceso a ellos por parte de clientes. Por "proceso" se entiende cualquier algoritmo, cálculo que opere en datos espacialmente referenciados. Por "publicación" se entiende poner a disposición de los clientes (otras máquinas o personas) la información necesaria para la lectura de información y uso de servicios geoespaciales.

Un WPS puede ser configurado para ofrecer cualquier tipo de funcionalidad SIG a un cliente a través de una red, incluyendo el acceso a cálculos preprogramados o modelos de computación que operen sobre datos espacialmente referenciados. Los datos requeridos por el WPS pueden ser proporcionados a través de la red o pueden estar en el servidor.

Esta especificación de interfaz proporciona mecanismos para identificar los datos espaciales requeridos para el cálculo, iniciar dicho cálculo y gestionar la salida para que el cliente pueda acceder a ella. Se pueden procesar tanto datos vectoriales como raster.

La especificación de los WPS está diseñada para permitir a un proveedor de servicios ofrecer un proceso vía Web (como por ejemplo intersección de polígonos), de manera que los clientes puedan introducir datos y ejecutar procesos sin tener conocimiento del API que lo permite. La interfaz WPS ofrece un estándar para normalizar el modo en que los procesos, sus entradas y sus salidas son descritas, cómo un cliente puede requerir la ejecución de un proceso y cómo la salida de dicho proceso es entregada al cliente.

La interfaz WPS especifica tres operaciones que pueden ser requeridas por un cliente y llevadas a cabo por un servidor WPS. Todas son de obligado cumplimiento por parte de todos los servidores. Dichas operaciones son:

a) GetCapabilities – Esta operación permite a un cliente solicitar y recibir metadadtos de servicios (Capabilities) en documentos que describen las posibilidades ofrecidas por la implementación de un servidor dado. La operación GetCapabilities proporciona los nombres y descripciones generales de cada uno de los procesos ofrecidos por el servidor WPS.
b) DescribeProcess – Esta operación permite a un cliente solicitar y recibir información detallada sobre los procesos que pueden ser ejecutados en el servidor, incluyendo los requerimientos de entrada, los formatos permitidos y las salidas que puede producir.
c) Execute – Esta operación permite a un cliente ejecutar un proceso determinado implementado por el servidor WPS usando los valores de entrada

proporcionados por el cliente y devolviendo la salida producida. Estas operaciones tienen muchas similitudes con otros servicios Web del OGC, incluyendo WMS, WFS y WCS.

9.1. Detalles técnicos para el diseño.

A partir de ahora, se va a desarrollar con más detalle las características de los WPS ya que algunas de ellas han sido determinantes en la elaboración de los requisitos y análisis de nuestra aplicación.

Los tipos de datos de entrada y de salida son:

- LiteralData: cadenas, números enteros y reales (Double)
- ComplexValue: los datos de los vectores y los raster. Pueden ser de gran tamaño.
- ComplexValueReference: igual que el anterior pero la respuesta al cliente es una referencia al archivo, no los datos en sí.
- BoundingBox: pares de coordenadas

Por tanto, hay dos modos de gestión de los datos (16):

- Los datos son enviados como parte del documento de petición: Son parte del archivo XML que se envía.
- Sólo se envían referencias: el servidor se descarga los datos de entrada de otro sitio en Internet. El cliente sólo va apuntando dónde están los datos.

Muy relacionado con el punto anterior están los tipos de ejecución del proceso (17):

- Síncrono: El cliente pide la ejecución de un proceso, el servidor lo hace y el cliente espera hasta que lo recibe.

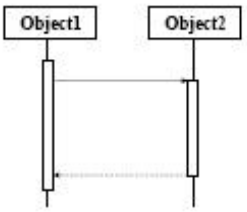

Figura 5. Comunicación síncrona. Fuente (17)

- Asíncrono: Hay dos modelos, push y pull. En el primero de ellos, el cliente recibe mensaje del servidor cuando ha concluido la ejecución.

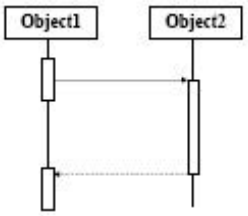

Figura 6. Comunicación asíncrona. Fuente (17)

En el segundo modelo, el cliente incluso puede consultar el estado de la ejecución.

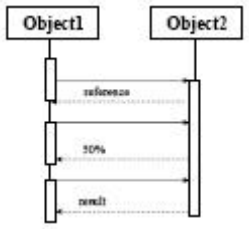

Figura 7. Comunicación asíncrona con consulta de estado. Fuente (17)

Como conclusión de lo expuesto, basándonos en las características técnicas de la definición de WPS (18), creemos que el uso óptimo de los WPS se consigue cuando el proceso a ejecutar es largo y complejo. Los WPS también son muy apropiados para el desarrollo de nuevos procesos que están sometidos a cambios continuos ya que están centralizados y siempre se ofrece la última versión. Esta parece ser la tendencia actual, tener las aplicaciones y los datos en internet porque es más accesible y fácil de mantener.

10. Requisitos de la aplicación

Además de las consideraciones anteriores, para establecer los requisitos de la aplicación llevamos a cabo un análisis de varios clientes anteriores para extraer las características más importantes. Algunas de ellas de las hechas en Java:

uDig (20): desde el principio fue nuestra referencia sobre todo porque permite utilizar ComplexValueReference (muy útil para poder encadenar peticiones) y la posibilidad de exportar a KML que es un desarrollo futuro de integración con Google Earth muy interesante.

JUMP (21): la limpieza y claridad de su código

OpenLayers WPS Control (22): Aunque hecho en javascript, lo más interesante, su uso completamente adaptado a Internet.

Otros clientes escritos en java que hemos analizado han sido Deegree (23) y WPSint (24) pero no han aportado nada reseñable.

Con este trabajo previo se establecieron dos listas de requisitos: uno de requisitos generales y otra de específicos.

Requisitos generales:

1. Mantener el carácter de código libre de todo el proyecto.
2. Mantener la utilización de estándares (GML, KML)
3. Hacerlo fácilmente adaptable a otros sistemas y mantenerlo lo más simple posible. Una arquitectura modular, código bien comentado, y lenguaje Java.
4. Que fuera fácil adaptarlo a Internet ya que es probable que los mapas, igual que otros contenidos y aplicaciones tiendan a estar disponibles en Internet para estar siempre actualizados y disponibles para ser mejorados.
5. Reutilizar el máximo código posible.

Dado el limitado tiempo de todo proyecto, este último requisito era especialmente importante, así que se buscaron clientes WPS existentes para gvSIG. Aparecieron dos; uno desarrollado por la empresa valenciana Lógica Extrema (25). Nos pusimos en contacto con ellos para ver el estado y la naturaleza del proyecto pero fue infructuoso. De todas formas algunas de sus ideas nos parecieron interesantes. Basándonos en una presentación suya disponible en la página de gvSIG (26) sacamos algunos de nuestros requisitos específicos. El otro proyecto existente está siendo desarrollado desde Sextante (27). También nos pusimos en contacto con ellos pero el proyecto no está lo suficientemente maduro[1].

[1] Olaya, Víctor. (25 de noviembre de 2008). ***WPS en gvSIG.*** [Correo electrónico de Mariano Velamazán], [On-line]. Dirección de correo electrónico: mvelamazanm@uoc.edu

```
El cliente se esta desarrollando en este momento, aunque queda bastante
trabajo. La verdad es que con los tiempos que manejas, y en esa situación,
quizas no haya mucho margen para hacer cosas, pero si estas interesado, no dudes en
consultarme, ok? Un saludo
Victor volaya@unex.es
```

Requisitos específicos:

1. Permitir seleccionar el servidor al que nos conectamos
2. Mostrar las capacidades (*Capabilities*) del servidor
3. Mostrar las descripciones de los procesos (*describeProcess*)
4. Permitir seleccionar un proceso (*process*) de los disponibles
5. Pedir los parámetros de entrada para ejecutar un proceso. Debe ser capaz de pedir los parámetros de cualquier tipo de proceso. La generación de los campos de la interfaz es totalmente dinámica.
6. Integrarlo bien con WMS y WFS.
7. Basar toda la información en GML. Convertir la entrada a GML para que sea fácil poner el cliente en Internet y en otros programas. La respuesta también será generada en GML.
8. Pestaña “Advanced” para seleccionar que la ejecución del proceso sea síncrona o asíncrona. Poder seguir trabajando en la aplicación mientras se ejecuta el proceso de manera asíncrona.
9. Opción que permita seleccionar si queremos una respuesta ComplexValueReference.
10. Ya que los WPS son procesos, el lugar más adecuado para integrar el cliente es, sin duda, el gestor de Geoprocesos.

11. Casos de uso.

A partir de los requisitos anteriores es fácil definir los actores y casos de uso.
Actores: Usuario gvSIG. Es un usuario final que conoce gvSIG y conoce las posibilidades y ventajas de los WPS.
Guión: El usuario está trabajando con una vista del programa gvSIG, necesita la ejecución de un geoproceso y decide encargar su ejecución a un servidor bien porque el geoproceso es muy nuevo o bien porque su cálculo es lento y costoso. Muchos de los procesos que se han implementado en el marco de proyectos WPS no están disponibles en gvSIG: generalización cartográfica, modelización de procesos, etc… Por tanto no es sólo una cuestión de comodidad acceder a servicios WPS, sino de necesidad. Abre el panel de geoprocesos, selecciona el tipo WPS y apunta al servidor que dispone de ese geoproceso. Se conecta a él, selecciona el geoproceso y lo ejecuta de manera síncrona o asíncrona según le interese. Según las especificaciones del Open Geospatial Consortium (8)(páginas 39-41) recibe como respuesta un archivo XML.

Identificación de casos de uso:

1. Conexión al servidor
2. Ver capabilities del servidor y seleccionar un proceso
3. Obtener información detallada de lo que hace un proceso (DescribeProcess)
4. Seleccionar la ejecución de un proceso a partir de una capa de entrada. Dar los otros valores de entrada necesarios.
5. Poder seleccionar que la ejecución sea asíncrona y que la respuesta sea un ComplexValueReference.
6. Visualizar respuesta del servidor

En la siguiente figura se muestra el diagrama de casos de uso

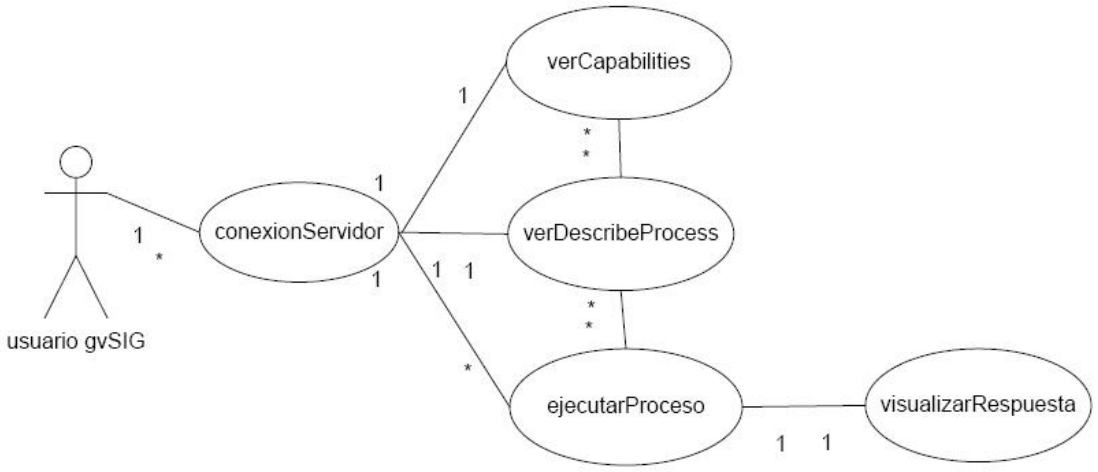

Figura 8. Principales casos de uso del cliente WPS.

Es el momento de detallar la documentación textual de los casos de uso. Se muestra en las Tablas 4 a 8.

Documentación Textual

Caso de uso número 1:	conexiónServidor
Resumen	el usuario se conecta a un servidor WPS para acceder a algún geoproceso
Papel	es el paso previo a cualquiera de los otros casos de uso. Sin conexión no hay WPS.
Actores	usuario gvSIG
Casos de uso relacionados	verCapabilities, verDescribeProcess, ejecutarProceso
Precondición	el usuario no está conectado al servidor WPS
Postcondición	el usuario está conectado al servidor WPS
Otros	

Tabla 4. Caso de uso 1.

Caso de uso número 2:	verCapabilities
Resumen	el usuario se ve los procesos disponibles en el servidor WPS.
Papel	Le sirve para comprobar que el geoproceso que necesita está disponible.
Actores	usuario gvSIG
Casos de uso relacionados	verDescribeProcess, ejecutarProceso
Precondición	el usuario está conectado al servidor WPS. El servidor ofrece geoprocesos e implementa esta interfaz.
Postcondición	el usuario ve los servicios de geoprocesamiento disponibles en el servidor
Otros	

Tabla 5. Caso de uso 2.

Caso de uso número 3:	verDescribeProcess
Resumen	el usuario ve los detalles de un geoproceso.
Papel	para saber, sobre todo, el tipo de valores de entrada que tendrá que introducir para un geoproceso y qué hace el algoritmo.
Actores	usuario gvSIG
Casos de uso relacionados	verCapabilities, ejecutarProceso
Precondición	el usuario está conectado al servidor

	WPS. El servidor ofrece geoprocesos e implementa esta interfaz.
Postcondición	el usuario ve los detalles del geoproceso disponible en el servidor.
Otros	

Tabla 6. Caso de uso 3.

Caso de uso número 4:	ejecutarProceso
Resumen	el usuario introduce los parámetros de entrada, decide la capa sobre la que quiere efectuar el cálculo y nombra una capa de salida.
Papel	es el caso de uso principal. Ejecuta el geoproceso a partir de unos valores de entrada.
Actores	usuario gvSIG
Casos de uso relacionados	verCapabilities, verDescribeProcess, visualizarSalida.
Precondición	el usuario está conectado al servidor WPS. El servidor ofrece geoprocesos e implementa esta interfaz. Introduce los valores de entrada.
Postcondición	el usuario obtiene una capa de salida escrita en XML.
Otros	

Tabla 7. Caso de uso 4.

Caso de uso número 5:	visualizarSalida
Resumen	el usuario ve el resultado del geoproceso como una capa o como un fichero XML.
Papel	es el caso de uso final.
Actores	usuario gvSIG
Casos de uso relacionados	ejecutarProceso.
Precondición	el usuario está conectado al servidor WPS. El servidor ofrece geoprocesos e implementa esta interfaz. El proceso se ha ejecutado satisfactoriamente y el archivo GML se ha generado correctamente.
Postcondición	el usuario ve la capa de salida escrita en GML.
Otros	

Tabla 8. Caso de uso 5.

12. Análisis de la aplicación

Una vez definidos los casos de uso, vamos a detallar todos los pasos de la fase de análisis. Como sabemos, la función de este apartado es traducir las necesidades descritas en los requisitos en un esquema que sirva de punto de partida para el desarrollo. En la siguiente figura se muestra el diagrama de paquetes.

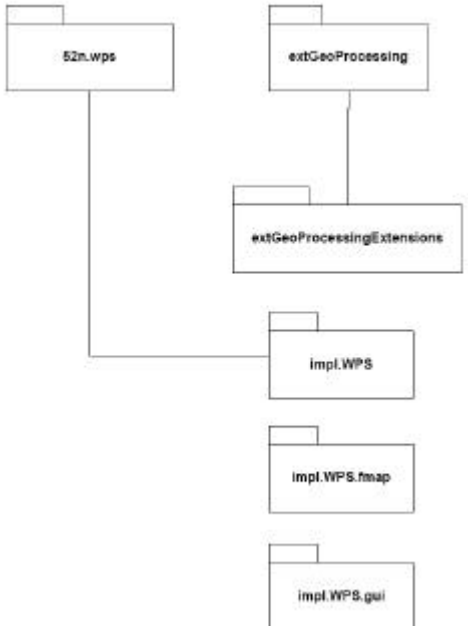

Figura 9 . Diagrama de paquetes.

En esta figura se aprecian las clases con anotaciones de su función:

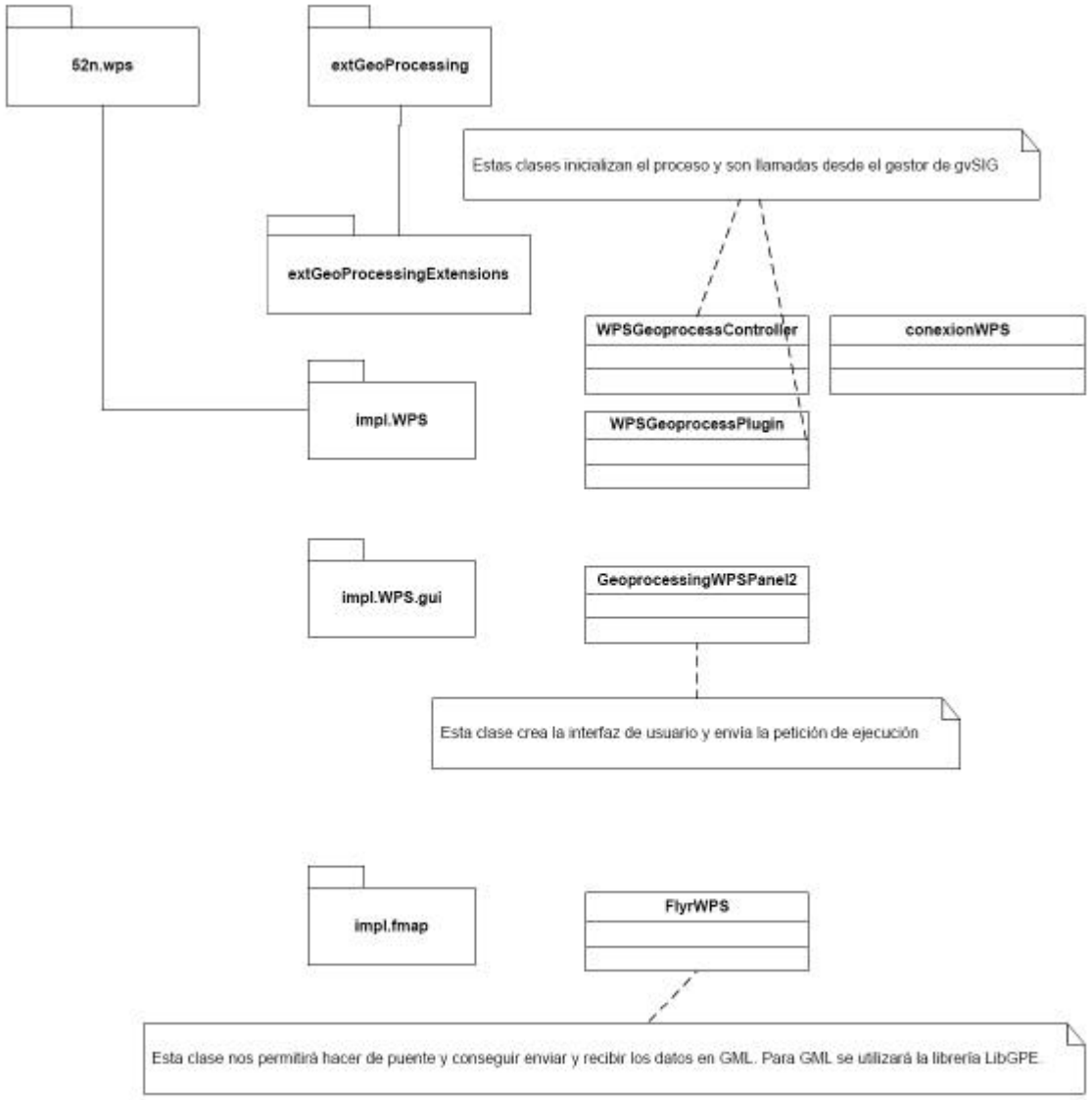

Figura 10. Diagrama de clases. Notación UML

La parte crítica del esquema es la clase FlyrWPS que ha de encargarse de conectarse con la librería libGPE y ser capaz entre las dos de traducir a GML la capa de entrada y luego volver a crear una capa GML a la salida de la ejecución del proceso en el WPS.

13. Diseño.

En esta fase se decide más concretamente cómo pretendemos dar respuesta al problema de partida haciendo hincapié en lo que reutilizamos.

Comenzaremos diciendo que la arquitectura de gvSIG basada en extensiones impone una estructura para su ampliación. Este hecho define las clases de las que heredamos y las interfaces que implementamos. El modelo de extensión de geoproceso de gvSIG define tres paquetes para un geoproceso; uno encargado de la interfaz (.gui), otro encargado de integrarse con la aplicación (.WPS) y otro encargado del geoproceso en sí. Como en nuestro caso, el geoproceso se calcula en el servidor sólo tenemos que encargarnos de la interfaz, de integrar la extensión en el programa y de traducir las entradas necesarias desde gvSIG al WPS.Si hacemos un símil con el patrón de diseño Model-View-Controller, nosotros nos encargaríamos de la parte View y Controller mientras que el WPS y gvSIG se encargarían de la parte Model.

De la persistencia también se encarga gvSIG creando el fichero de la capa cuyo contenido le enviamos filtrado nosotros en lenguaje GML.

Por último, vamos a ofrecer algunas capturas de pantalla del prototipo de interfaz gráfica de usuario.

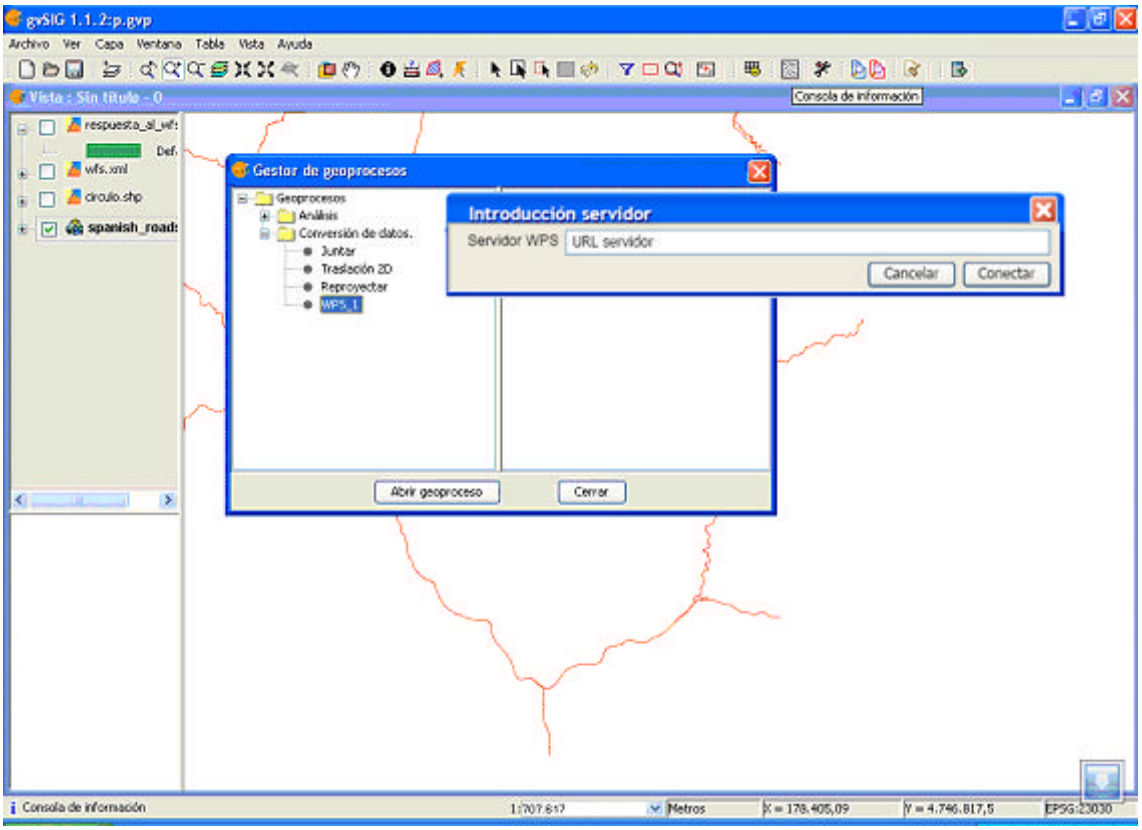

Figura 11. Detalle de la interfaz: petición de la URL del servidor.

En esta captura se ve el campo de texto de introducción de la dirección del servidor.

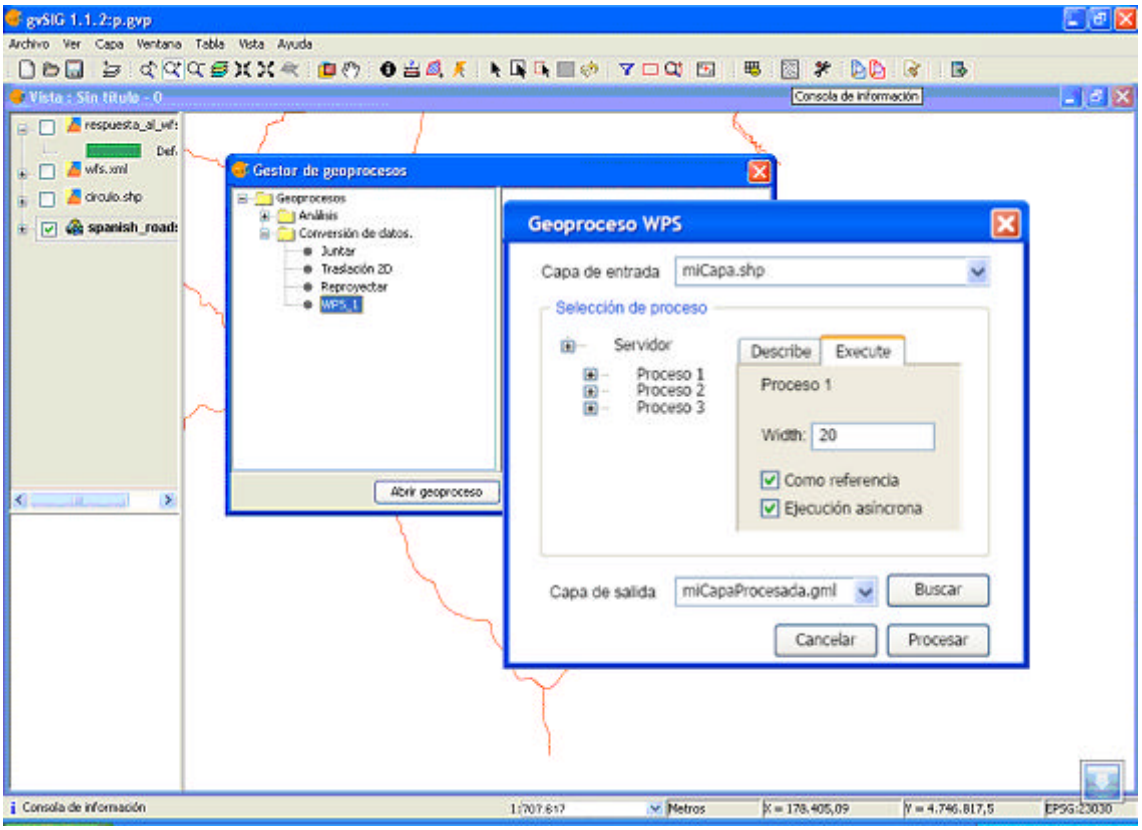

Figura 12. Detalle de la interfaz: petición de parámetros propios del proceso seleccionado.

En esta pantalla se ven los parámetros de ejecución de un proceso. En la misma ventana se define la entrada y la salida.

14. Implementación.

Al llegar a la fase de implementación, encontramos un problema que ha complicado demasiado los planes iniciales. Para ver si era viable pasar toda la información a través de GML (cuestión que hemos justificado más arriba), hicimos pruebas exportando archivos GML pero esta función tiene un bug en gvSIG. Los correos de respuesta son de Jorge Piera, el autor de la librería LibGPE[2].

> Hola a todos,
>
> ¿dónde se puede ver el schema cit.xsd?
>
> Gracias
>
> Mariano Velamazán
> Informática Sistemas (UOC)
>

Hola Mariano.

El esquema cit.xsd no existe. Cuando tu exportas a GML tienes que seleccionar el nombre del esquema que quieres generar. En las primeras versiones de gvSIG se generaba por defecto el esquema cit.xsd pero en las últimas se genera un esquema con el mismo nombre del archivo GML pero con extensión xsd.

Un saludo.

Jorge.

> Gracias por responder tan rápido Jorge,
>
> el tema es que cuando exporto una capa a gml, hace siempre referencia a ese schema. Necesito que un servidor WPS (web processing service) lea los gml que genera gvSIG. Estoy haciendo pruebas a exportar el gml con el schema de referencia (GML Feature Schema, que está en schemas.opengis.net/gml/2.1.2/feature.xsd) pero no consigo que el servidor interprete el GML que genero con gvSIG. No sé si puedes orientarme un poco... Estamos intentando hacer un cliente wps para gvsig.
>
> Gracias.
>
> Mariano Velamazán
> Informática Sistemas

Hola Mariano.

Acabo de comprobar lo que dices y es cierto, gvSIG añade el esquema por defecto en lugar de añadir el esquema que el usuario ha seleccionado en el proceso de exportación. Eso es un bug y no será corregido hasta la siguiente versión de gvSIG. Puedes hacer dos cosas: o bien mirar el servidor de WPS para que no sea tan restrictivo a la hora de comprobar la validez del esquema del GML o bien modificar gvSIG para que no añada el esquema por defecto a los GML's que genera. Si eres un desarrollador y tienes los fuentes de gvSIG te puedo indicar donde esta el error para que lo puedas modificar.

Un saludo.

Jorge.

El problema era difícil de solucionar y suponía añadir más complicación a unos requisitos ya de por sí ambiciosos. La opción de tomar un camino más seguro basado en utilizar el cliente JUMP se convirtió en la más razonable.

[2] Peira, Jorge. (21 de noviembre de 2008). *cit.xsd.* [Correo electrónico de Mariano Velamazán], [On-line]. Dirección de correo electrónico: mvelamazanm@uoc.edu

Una vez analizado el cliente de JUMP, se utilizó como base para implementar la interfaz de usuario junto con la interfaz de los otros geoprocesos existentes para que el resultado visual del cliente fuera coherente con el resto de la aplicación. Se adaptaron los métodos y se resolvieron las dependencias de librerías. Nuestro cliente consigue conectarse con el servidor WPS, recibe las capacidades (capabilities) de dicho servidor y las descripciones de los procesos (describeProcess). También solicita correctamente los parámetros de entrada aunque queda algún detalle de la interfaz por perfilar (eliminar una de las dos peticiones de capa de entrada).

El problema está en que cuando se ejecuta el método executeProcess, necesitamos enviar una featureCollection de GeoTools al servidor WPS y de aquí no hemos podido pasar. Se intentó haciendo un cast de capa vectorial a capa GeoTools (GT2):

```
FLayer layer = getInputLayer();
FLyrGT2 lyr = (FLyrGT2)layer;
DataStore store = ((FeatureSource) lyr).getDataStore();
FeatureSource features = store.getFeatureSource(lyr.getName());
org.geotools.feature.FeatureCollection fc = features.getFeatures();
```

Desafortunadamente, esta opción no es correcta porque el casting a FLyrGT2 no es soportado desde una Flayer ni desde una FLyrVect.

Intentamos seguir los pasos apuntados por el compañero Víctor Velarde pero tampoco se consiguió compilar el servidor WPS con la versión más antigua de GeoTools; la que utiliza gvSIG (2.1 y no 2.4).

Por tanto, seguimos buscando soluciones sencillas y rápidas porque el tiempo restante empezaba a ser insuficiente. Así, encontramos una segunda línea de trabajo basada en una extensión de gvSIG llamada extJumpAdapter que básicamente garantiza la comunicación entre capas gvSIG y capas JUMP (28).

El objetivo de esta extensión es crear un puente entre el modelo de feature de JUMP y el de gvSIG y así poder traspasar algunas funcionalidades de OpenJUMP a gvSIG con facilidad.

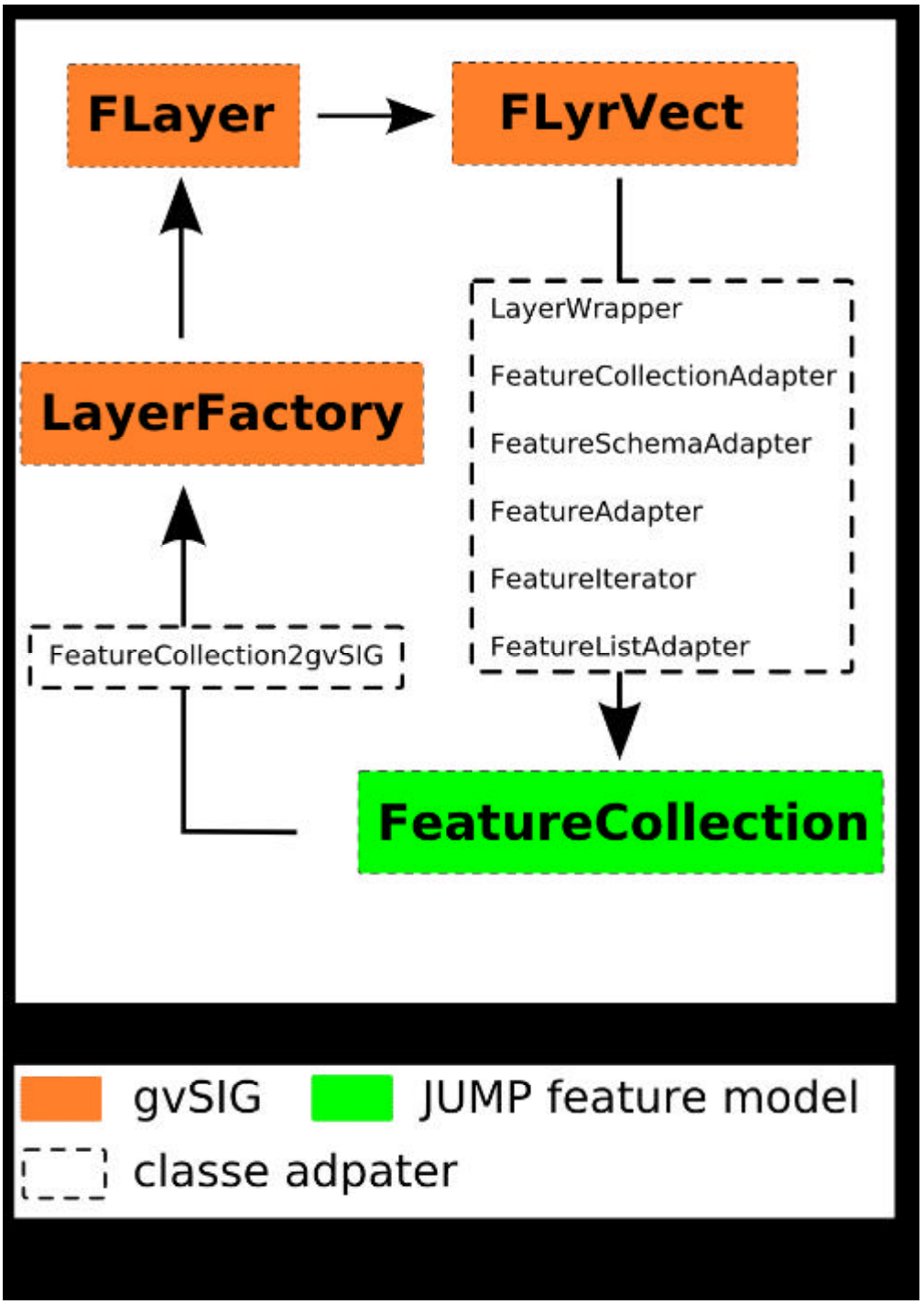

Figura 13. Modelo de clases de JumpAdapter

Añadiendo la clase perteneciente al cliente JUMP JUMPConversionUtil a esta extensión y modificando las dependencias y el tipo de parámetros de entrada del método convert2GTFeatures conseguimos enviar la featureCollection al servidor WPS con éxito. Devuelve un error generado porque no encuentra el archivo de configuración wps-config.xml.

El código clave que consigue enviar los datos al servidor es el siguiente:

```
FLyrVect lyr = super.getInputLayer();
```

```
            feature.FeatureCollection jFC =
gvSIGToFeatureCollection.getFeatureCollection(lyr);
            org.geotools.feature.FeatureCollection gt = null;
      try {
            gt = JUMPConversionUtil.convert2GTFeatures(jFC);
      }
      catch(Exception e) {
            LOGGER.warn(e);
      }
            erb.addComplexData(parameterName, gt);
```

El siguiente paso, para el que ya no hemos tenido tiempo, es recibir correctamente la respuesta del servidor WPS.

En este punto es recomendable ver el anexo 3 con el listado de las tareas pendientes más importantes.

Como conclusión, diremos que esta solución también dista de ser elegante pero al menos levanta el estado de bloqueo en el que se encontraba el proyecto, y abre la puerta a nuevas posibilidades al permitir incorporar funcionalidades de JUMP.

15. Conclusiones

El contenido de este apartado está determinado por el hecho de que no se ha conseguido implementar el cliente WPS. Los párrafos siguientes apuntan algunas de las causas más importantes.

Desde el principio no se cumplió el plan de trabajo porque aparecían problemas de los que no habíamos sido conscientes. Compilar gvSIG, compilar el servidor de 52 north, compilar la API del cliente (con Maven, sin Maven,...) fueron horas y horas dedicadas en las que en más de una ocasión nos encontramos perdidos entre versiones y detalles que no permitían seguir avanzando.

Una vez configurado todo el entorno necesario para el desarrollo del trabajo se volcaron todos los esfuerzos en conseguir un buen catálogo de requisitos, un buen análisis y un buen diseño. Creíamos que la implementación, en el peor de los casos, sería cuestión de encontrar qué clases de las existentes a uno y otro lado (52north y gvSIG) hacían las tareas definidas por nosotros añadiendo algunos retoques. La realidad fue bien distinta. La búsqueda de clientes existentes o en desarrollo tampoco fue satisfactoria como hemos detallado en el apartado “Requisitos de la aplicación”.

Otro “detalle” que indirectamente ha costado muchas horas fue la decisión de implementar el cliente como un Geoproceso. A nivel lógico era una decisión correcta y coherente pero a nivel técnico implicaba compartir dependencias con los otros geoprocesos. Debido a nuestra falta de experiencia y conocimientos respecto a las dependencias de proyectos y librerías en un programa de estas características (Workspace de Eclipse) resolver este tipo de imprevisto fue más duro que si hubiéramos tenido una extensión hecha desde cero por nosotros.

En la fase de documentación de requisitos y análisis tomamos como referencia el cliente uDIG porque seguramente por un error de apreciación y/o concepto creímos que estaba mejor preparado para incluir la posibilidad de ejecutar los procesos de manera asíncrona y mediante ComplexValueReference. La realidad es que al estar hecho sobre la Rich Application Platform de Eclipse, gran parte de su código estaba empleado en tareas ajenas al propio cliente y también perdimos mucho tiempo intentando descifrar qué nos podía ser útil.

Las pruebas con el cliente JUMP mostraron que había que pasar por Geotools y tal y como apuntó Víctor Velarde en el foro de la asignatura era necesario además recompilar el servidor de 52north, tarea que no se consiguió ya que en ese momento se necesitaban urgentemente resultados inmediatos.

En general, creo que por falta de conocimientos previos se ha perdido mucho tiempo en detalles técnicos de mayor o menor importancia y en intentos infructuosos que no han llevado a ningún sitio. Al final, **ha quedado muy poco tiempo para la implementación en sí y el análisis de las clases existentes**

en gvSIG para deducir qué hacer de manera sistemática (no mediante prueba-error y búsqueda de soluciones pre-existentes).

Todo lo dicho no quiere decir que el balance sobre mi proyecto no sea positivo ya que he aprendido mucho. En la parte positiva está el enfrentarme por primera vez con un proyecto del que no sé nada a priori y que requiere que muchas piezas de software existente se comuniquen entre sí. Ha sido también positivo, quizá lo más positivo, lo que he aprendido sobre la redacción de proyectos ya que en este punto sí he logrado avances más importantes. Pero, desafortunadamente, no es suficiente para evitar un cierto sabor amargo por no haber sido capaz de cumplir todos los objetivos.

Anexo 1. Balance de tareas realizadas.

Para que quede registro del tiempo empleado y los caminos seguidos se ofrece un listado de las tareas realizadas. Los primeros puntos pueden servir como ayuda para configurar el entorno de desarrollo y para instalar y utilizar nuestro futuro cliente en otros equipos y para los estudiantes de próximos años. Los puntos finales son más una muestra de los intentos fallidos y los caminos comenzados que no llevaron a ningún resultado satisfactorio. Pueden servir para no tomarlos.

- Leer sobre los SIG en general. Leer la memoria de muestra de otros cursos.
- Descargar la aplicación de escritotio gvSIG y probarla siguiendo el documento “Curso de gvSIG 1.1”. Descargar el manual de usuario.
- Descarga de documentación del OGC. Descarga de los WPS Schema. No se utilizan posteriormente.
- Lectura del documento base de WPS. Open GIS Web Processing Services. Leer el documento del Ministerio de Fomento titulado Web Processing Services 0.2.0 del Consejo Superior Geográfico (15).
- Ver uDIG y JUMP. Probarlos. .jar de uDig y .jar de JUMP. 52NWPS1.0.0_JUMP_CLIENT-20071204-CVS y el documento “An integrated client for Web Processing Services. Upgrading uDIG with processing power.”
- Instalar tomcat (la versión probada con WPS 52North) apache-tomcat-5.5.27. Instalar el admin de tomcat y el manager
- Instalar eclipse versión 3.4.1.
- Instalar subclipse versión 1.4.2. Leer un poco sobre los SVN y los repositorios.
- Instalar maven siguiendo las especificaciones del documento entregado por el consultor “Setting up the WPS as an Multiple Module Eclipse Project (14).
- Instalar el servidor WPS de 52North (52n-wps-webapp-1.0-rc2.war) en Tomcat. Probar GetCapabilities y DescribeProcess desde el archivo test.html que viene de prueba. Probarlo en xml y abandonarlo. Sólo se utilizará el método de peticiones GET. Compilar los proyectos del repositorio de 52North. No se consigue. Visitar http://www.idee.es/WPS/ para hacer pruebas. Descargar el javadoc del API de 52North WPS.
- Intentar compilar el API cliente en un archivo .jar. No se consigue. Se decide utilizar el .jar de uDIG.
- Crear una clase de prueba para asegurar que todo funciona correctamente ya que crea una instancia de WPSClientSession, se conecta al servidor Tomcat y devuelve la salida de las operaciones GetCapabilities y DescribeProcess.
- Descargar fuentes de gvSIG. Compilar y ejecutar gvSIG desde los fuentes instalados en Eclipse. Se siguen exactamente las especificaciones del archivo Léeme.txt: descargar librerías de imagen, descargar archivo de

conexión a bases de datos Oracle, etc. pero aún así se encuentran problemas que se solucionan compilando manualmente cada uno de los proyectos en el orden del citado archivo, se resuelven los problemas de dependencias uno tras otro. Otro problema encontrado fue al tratar de compliar gvSIG desde un SVN repository siguiendo las especificaciones de un documento publicado en la lista de desarrolladores de gvSIG.

- Demo del cliente JUMP.
- Se comprueba la disponibilidad en gvSIG de conexión a fuentes OGC como WMS y WFS. En el manual.
- Se averigua dónde se gestionan los elementos de la interfaz de gvSIG
- Se comprueba cómo está implementado en gvSIG el WMS y sobre todo el WFS. Driver de conexión.
- Se analiza el cliente uDig. Me lo he descargado del SVN de 52north
- Pruebas con execute en mi servidor. Hay que hacerlas en:
- http://localhost:8080/wps/test.html y meter en el área de texto el contenido de alguno de los archivos que están en: C:\Documents and Settings\mariano\apache-tomcat-5.5.27\webapps\wps\testDocs. La salida es un documento XML-GML
- Se pruebo a ejecutar geoprocesos, a añadir capasc(shp, wfs, wms) pero no se consigue ver ningún archivo gml. Parece que hay un error en cómo gvSIG exporta las capas a GML
- Decidi si es mejor incorporarla como geoproceso: como geoproceso se crea una capa nueva. Como nueva capa, también. A nivel conceptual es más correcto ponerlo como un geoproceso.
- Se analiza el código fuente de los geoprocesos, de WFS y de udig para ver de cuál se puede reutilizar más
- Se hacen pruebas enviando el código GML generado por un WFS, por gvSIG y abriendo desde gvsig el GML generado por el WPS. Nada funciona.
- Averiguar las especificaciones que entiende gvSIG (Simple Feature Profile (SFP-X)) y mirar cómo hay que enviar el xml a wps
- Averiguar qué partes de un xml se deben enviar para obtener respuesta válida del wps.
- Importo las librerías de JUMP en vez de las de udig
- Averiguar qué es un featureCollection y ver cómo se envía al servidor wps el complexData y cómo lo recibe éste.

Anexo 2. Bibliografía

1. **Sáenz Higueras, Nita y Vidal Oltra, Rut** (2008): "Las referencias bibliográficas". En: *Redacción de textos científico técnicos.* UOC. ("Módulo 2", Asignatura Trabajo Final de Carrera).

2. **Rodríguez Lloret, Jesús y Olivella, Rosa** (2008): "Qué son los sistemas de información geográfica". En: *Introducción a los sistemas de información geográfica.* UOC. ("Módulo 4", Asignatura Trabajo Final de Carrera. Documentación específica sobre SIG).

3. Wikipedia (2008) : Sistemas de Información Geográfica. En: http://es.wikipedia.org/wiki/Sistema_de_Información_Geográfica

4. **Botella Plana, Albert** (2008): "Bases de datos geográficos". En: *Bases de datos geográficos.* UOC. ("Módulo 2", Asignatura Trabajo Final de Carrera. Documentación específica sobre SIG).

5. Wikipedia (2008) : OGC. En: http://es.wikipedia.org/wiki/OGC

6. **Botella Plana, Albert** (2008): "La apertura de los SIG". En: *Nuevas tendencias en SIG.* UOC. ("Módulo 6", Asignatura Trabajo Final de Carrera. Documentación específica sobre SIG).

7. **Schut, Peter** (2007): "Introducción". En: *OpenGIS Web Processing Service.* Open Geospatial Consortium Inc. En: http://www.opengeospatial.org/standards/wps

8. **Schut, Peter** (2007): "Operaciones WPS". En: *OpenGIS Web Processing Service.* Open Geospatial Consortium Inc. n http://www.opengeospatial.org/standards/wps

9. **Conselleria de Infraestructuras y Transportes** (2007): *Curso de gvSIG 1.1.* Generalitat Valenciana. En: http://www.gvsig.org/web/home/docusr

10. **IVER, Tecnologías de la Información** (2007): "Introducción a gvSIG". En: *Manual de usuario versión 3.* Generalitat Valenciana. En: http://www.gvsig.org/web/home/docusr

11. **IVER, Tecnologías de la Información** (2007): "Vistas". En: *Manual de usuario versión 3.* Generalitat Valenciana. En: http://www.gvsig.org/web/home/docusr

12. **IVER, Tecnologías de la Información** (2007): "Tablas". En: *Manual de usuario versión 3.* Generalitat Valenciana. En: http://www.gvsig.org/web/home/docusr

13. **IVER, Tecnologías de la Información** (2007): "Mapas". En: *Manual de usuario versión 3.* Generalitat Valenciana. En: http://www.gvsig.org/web/home/docusr

14. **52North**: *Setting up the WPS as a múltiple module Eclipse project.* 52North. En: https://52north.org/twiki/pub/Processing/52nWebProcessingService/Setting_up_the_WPS_as_a_Multiple_Module_Eclipse_Project_svn.pdf

15. **Ministerio de Fomento** (2008): *Web Processing Service.* Consejo Superior Geográfico.

16. **Cepický, Jáchym** (2008): ***OGC Web Processing Services and it´s usage***. GIS Ostrava. En: http://**gis.vsb.cz**/GIS_Ostrava/GIS_Ova_2008/sbornik/Lists/Papers/025.pdf

17. **Foester, Theodor** (2008): ***Web processing with OGC WPS Specification***. En: https://52north.org/twiki/bin/viewfile/Processing/GIdaysWPSworkshop?rev=1;filename=GI-daysWorkshop2008.pdf

18. **Open GIS Consortium, Inc**.(2007) ***OpenGIS Geographic Markup Language (GML)*** Encoding Standard. OpenGIS Standard, Open GIS Consortium, Inc. En: http://www.opengeospatial.org/standards/gml

19. **Open Geospatial Consortium, Inc**. (2007) ***Web Processing Service (WPS) Specification***. En: http://portal.opengeospatial.org/files/?artifact_id=24151.

20. **uDig**: http://udig.refractions.net

21. **JUMP**: http://www.jump-project.com. Este y el anterior basados en el cliente de 52north en: http://52north.org/maven/project-sites/wps/52n-wps-site/

22. **OpenLayers**: http://www.openlayers.org/

23. **Deegree**: http://deegree.org

24. **WPSint**: http://wpsint.tigris.org

25. **Lógica Extrema**: http://www.logex.es

26. **Celda Ibáñez, José Luis y Araque Vilches, Enrique** (2007): ***Callejeros en gvSIG, WPS y GML***. Documento en internet: http://www.jornadasgvsig.gva.es/cas/talleres_3as/Callejero_WPS.zip

27. **Sextante**: http://www.sextantegis.com.

28. **Brocher, Erwan (2006) : *JumpAdapter*** En: http://r1.bocher.free.fr/index.php?n=Main.JumpBridge4gvSIG.

Anexo 3. Tareas pendientes y futuras mejoras

- Solucionar la detección del archivo wps-config.xml
- Solucionar la respuesta del servidor: convertir la información en una capa gvSIG y mostrarla
- Corregir los errores de la interfaz de usuario: hay dos peticiones de capa de entrada. Hay dos botones para ejecutar el proceso
- Probar en todos los tipos de procesos
- Probar en más tipos de capas
- Crear el cuadro de diálogo de petición de conexión a otro servidor WPS.
- Generar error si no se conecta porque no haya servicio en el servidor
- Refrescar la ventana después de cambiar de capa de entrada
- Traducir los términos de la extensión a otros idiomas
- Crear una distribución ejecutable
- Ofrecer la posibilidad de seleccionar modo síncrono o asíncrono
- Ofrecer la posibilidad de encadenar procesos y ComplexValueReference