UNIVERSIDAD DE CASTILLA-LA MANCHA

ESCUELA POLITÉCNICA SUPERIOR

INGENIERÍA

EN INFÓRMATICA

PROYECTO FIN DE CARRERA

Ejecución de consultas de localización en ambientes indoor

María José Montoya Martínez

Junio, 2007

Resumen

Los sistemas basados en localización en exteriores se han convertido en una herramienta de uso cotidiano, para una gran mayoría de usuarios. Entre estos sistemas cabe destacar el sistema de posicionamiento global (GPS) que permite determinar la posición de un objeto en un determinado momento.

Dentro de los sistemas de localización, un campo que ha evolucionado mucho en los últimos años han sido los servicios basados en localización Indoor (RFID, Bluetooth, etc.). Estos sistemas en interiores permiten ofrecer unas prestaciones similares al GPS dentro de recintos como pueden ser: museos, hospitales, parque de atracciones, etc.

Los sistemas de localización indoor, unidos con las tecnologías que existen actualmente para el cálculo de rutas y los Servidores de Mapas por Internet (IMS), permitirán ofrecer al usuario una serie de funcionalidades básicas como son: *cálculo de rutas entre un origen y un destino, conocer los objetos más cercanos al usuario, consultar la información de un objeto, conocer la posición actual, etc.*

Por tanto, lo que se pretende conseguir es desarrollar una arquitectura web que permita ejecutar tanto cálculo de rutas como consultas espaciales destinadas a interrogar elementos de interés situados en interiores. La interfaz gráfica de esta arquitectura web debe seguir unas pautas de usabilidad.

En la presente memoria se hace un recorrido por lo principales Servidores de Mapas por Internet (IMS) y tecnologías para el cálculo de rutas disponibles en el mercado. También, se ha aportado una solución al problema expuesto anteriormente, diseñando un sistema de localización Indoor para un museo.

Agradecimientos y dedicatorias

Todavía recuerdo el primer día de universidad, en el que no conocía a casi nadie por los pasillos y todo lo que se decía en clase me parecía extraño. Nunca pensé que iba a estar escribiendo estas palabras, ya que mi intención era hacer solo Ingeniería Técnica Informática de Gestión, y con un gran empujón estoy terminando la Superior.

En primer lugar, quiero dedicar este proyecto a mis padres por haberme dado la oportunidad de hacer esta carrera, ya que sin su ayuda la realización de este proyecto no hubiera sido posible. Muchas gracias por todo el apoyo recibido, gracias por todo el esfuerzo que habéis hecho por mí.

También, quiero dedicárselo a mi hermano, ya que siempre me andaba preguntando: ¿Cuándo acabas? ¿No vienes este fin de semana? ¿Hoy también tienes que estudiar? Pues parece que sí, que ya se ve el fin, ya podré pasar con vosotros todo el tiempo necesario.

Y no puedo olvidar a mi novio, por haberme convencido en su día de hacer la Superior, por haberme dado siempre mucho ánimo para continuar con los estudios. Ya que él también ha vivido todas estas situaciones y me ha ayudado siempre en los momentos más duros que he tenido hasta llegar aquí. Además, le agradezco todas las opiniones intercambiadas sobre este proyecto, y por aguantar todas las charlas que le he dado sobre este trabajo. Muchas gracias por todo tu apoyo recibido.

También, agradecer a todos los amigos que he ido haciendo durante mis años de universitaria. A toda aquella gente que conocí cuando hice Ingeniería Técnica Informática de Gestión, ya que son muy buena gente y nunca me olvidaré de ellos. Al ser la primera carrera que inicias, es cuando más gente conoces y mejores amigos consigues hacer. Muchas gracias a todos, en especial a Rosario.

Por supuesto, quiero hacer una mención especial a mi director de proyecto, José Eduardo Córcoles, que nunca ha tenido ningún inconveniente en recibirme siempre que lo he necesitado. Además, me ha prestado una gran ayuda a lo largo de todo el proceso de realización del proyecto, tanto en el desarrollo como en la redacción de la memoria.

Finalmente, aunque no por ello menos importante, me gustaría dar las gracias a Juan Peralta, por haber realizado a la perfección las tareas de Jefe de Proyecto. También, me gustaría agradecer a todos mis compañeros de SITESA, empresa donde he desarrollado este trabajo, por todos los momentos que hemos compartido juntos y por toda la ayuda recibida.

Índice de Contenidos

Listado de Figuras

Listado de Tablas

Capítulo 1. INTRODUCCIÓN

"La formulación de un problema es más importante que su solución"

Albert Einstein (1879-1955)
Científico estadounidense de origen alemán.

En el presente capítulo se realiza una introducción al proyecto desarrollado, haciendo hincapié en el objetivo perseguido con el mismo. Además, se pretende ubicar el proyecto dentro de ámbito empresarial en el cual ha sido llevado a cabo.

1.1 MOTIVACIÓN

Uno de los ámbitos más atractivos de aplicación de los Sistemas de Información Geográfica (SIG), sobretodo gracias a los espectaculares avances vividos en el ámbito de las telecomunicaciones y de los sistemas de posicionamiento globales (GPS), es el de los sistemas basados en la localización (LBS).

La logística de estos sistemas se basa en la existencia de un núcleo central en el cual se reciben los datos, especialmente de ubicación de un número ilimitado de usuarios conectados con distintos tipos de dispositivos móviles (teléfono móvil, PDA, tabletPC). Estos, a parte de encontrarse perfectamente localizados dentro del ámbito geográfico, pueden realizar peticiones de información a la central y recibir estos datos o las órdenes pertinentes. Estos servicios se pueden definir de la siguiente manera:

> *Los servicios de localización son servicios que integran una localización o ubicación de un dispositivo móvil con otra información para proveer un valor agregado a un usuario (Definición Jochen Schiller)*

Dentro de los LBS, un campo que, gracias a la tecnología, ha evolucionado mucho en los últimos años han sido los servicios basados en localización *Indoor*. Mediante sistemas de posicionamiento en interiores (indoor location) como (RFID, Wi-Fi, UWB, Bluetotth, Zigbee, etc) usuarios y/o dispositivos pueden ser localizados en interiores con el fin de ofrecerles servicios basados en la ubicación que ocupan en un determinado momento. Muchos son los campos de aplicación de este tipo de servicios indoor: museos, hospitales, parques de atracciones, etc.

Así, por ejemplo, en el ámbito de un museo, un sistema basado en localización puede detectar en cada momento la posición de un dispositivo móvil que porte un usuario (por ejemplo, PDA). En base a esta localización, el sistema le puede ofrecer al usuario en su dispositivo móvil servicios como: información de la obra de arte que tiene en frente, información sobre las piezas que hay expuestas en la sala en donde se encuentra situado, la ruta que debe seguir desde su posición actual hasta una determinada obra, la ruta a la salida más cercana en caso de emergencia, etc.

1.2 OBJETIVOS A ALCANZAR

Tomando como base los sistemas de información geográfica (SIG) y los servicios basados en localización (LBS), este proyecto pretende definir una arquitectura para satisfacer el siguiente escenario: un usuario con un sistema móvil (PDA, teléfono móvil, etc.) puede obtener información contextual asociada a la localización que tiene en un determinado momento dentro de un ambiente indoor.

El objetivo se puede enunciar de la siguiente manera:

Estudiar, diseñar e implementar una arquitectura software que, basada en sistemas de localización indoor, permita la ejecución de lo que se han considerado funcionalidades básicas en este tipo de sistemas, como son: cálculo de rutas en interiores, conocer la posición actual de un usuario, obtener información de un objeto, conocer los objetos más cercanos a uno dado, etc.

Aunque muchos pueden ser los puntos de vista para llevar a cabo este objetivo, el trabajo se ha centrado en definir una arquitectura basada en sistemas de información geográfica (SIG). Teniendo presente como funcionan los sistemas de información geográfica basados en GPS, a este proyecto se le ha dado la misma perspectiva pero aplicada a sistemas indoor. En este trabajo, la información espacial que representa las diferentes partes de un edificio (salas, pasillos, escaleras, plantas,...) tienen exactamente la misma importancia que la información alfanumérica (cuadros en un museo, puestos de trabajo en una oficina, etc.). Esta equiparación es significativa a la hora de abordar los requisitos funcionales, como son por ejemplo: *conocer los objetos más cercanos a*

uno dado. Estos requisitos son imprescindibles en este tipo de sistemas y no podrían desarrollarse eficientemente con otra perspectiva que no se basara en sistemas espaciales.

Al trabajar con sistemas móviles es obligado tener en cuenta aspectos como: (i) la ejecución eficiente, ya que la capacidad de ejecución de un dispositivo móvil no es muy elevada, (ii) la definición de las funcionalidades más adecuadas para ser ofertadas a este tipo de dispositivos, y (iii) la manera de presentar y acceder a la información desde estos dispositivos.

La figura 1.1 muestra un esbozo de la arquitectura diseñada. En ella interviene: (i) un Mediador, encargado de atender las consultas del usuario, procesarlas y enviar la respuesta; (ii) un sistema de localización, que se encargará de enviar la posición del usuario; (ii) y por último, un IMS (servidor de mapas por internet), que será el encargado de publicar los mapas con los datos espaciales y/o alfanuméricos.

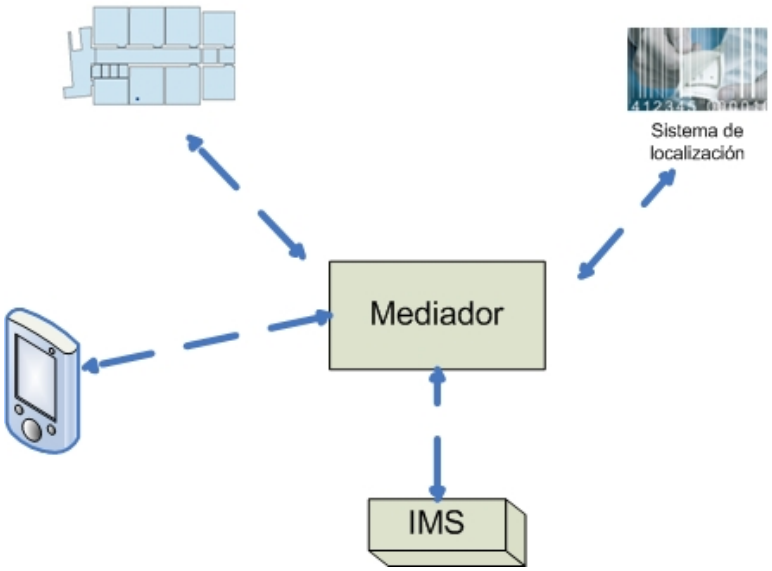


Figura 1.1. Esbozo de la arquitectura diseñada.

Así, en un escenario de ejecución, el usuario realiza una consulta desde el entorno de la aplicación (para ello, se tendrá una interfaz adaptada con las diferentes consultas que podrá realizar el usuario). La consulta es tratada por el mediador quien se la enviará al servidor de mapas para obtener la respuesta a la consulta solicitada.

Del esquema de la figura 1.1 se obtienen los puntos clave considerados en este proyecto y que han sido tratados como tareas dentro de la metodología aplicada.

- La familiarización con el concepto de sistemas de información espacial y su implementación con **Servidores de Mapas por Internet (IMS).** Esta tarea conlleva el estudio de varias alternativas comerciales y de software libre.

- La caracterización de tecnologías para **cálculo de rutas** y el estudio de varias alternativas presentes actualmente en el mercado.

- Estudio con detalle de las **características** del servidor de mapas y de la tecnología para el **cálculo de rutas** seleccionado para el desarrollo del proyecto.

- Diseño e implementación de un **sistema mediador** que será el encargado de atender las peticiones de los usuarios y dar una respuesta.

- Diseño e implementación de un **sistema para el cálculo de rutas** utilizando la tecnología seleccionada **y un sistema de localización** que permitirá al mediador obtener la posición del usuario.

- El caso de estudio abordado será para un **museo**, como si se tratará de una guía museística, donde se podrá realizar diferentes consultas como son: *consultar las características de un determinado cuadro, consultar los objetos más cercanos a la posición del usuario, etc*. Para la implementación de esta guía museística se deberá tener en cuenta unos criterios de **usabilidad**.

- Y, finalmente, la **valoración y análisis de conclusiones** de la solución obtenida.

Además, el desarrollo de estas tareas lleva consigo la realización de otra tarea implícita pero de especial relevancia dentro del ámbito de un proyecto fin de carrera, como es: la adecuada aplicación de aquellos conceptos teóricos estudiados en la carrera de Ingeniería Informática. Metodologías de desarrollo software, etc.

1.3 UBICACIÓN DEL PROYECTO

Este proyecto fin de carrera ha sido desarrollado íntegramente en la empresa SITESA, cuya oficina de Albacete está ubicada en el Parque Científico y Tecnológico de Albacete. SITESA es la compañía del Grupo EP especializada en las Tecnologías de la Información que acerca a todos sus clientes el potencial que ofrecen los Sistemas de Información Geográfica como ventaja competitiva dentro de las pautas que marca la nueva economía [WSITESA].

El origen de este proyecto fin de carrera ha sido satisfacer las necesidades de la empresa privada dentro del ámbito de la Innovación Tecnológica, integrando los conocimientos académicos con la experiencia y practicidad empresarial.

Como se comentó en la Introducción de este Capítulo, la incorporación de los sistemas de localización es una necesidad empresarial actual. Por ello, SITESA, interesada en buscar soluciones a esta necesidad, ha dotado a este proyecto de toda la infraestructura necesaria (software, hardware y lugar de trabajo) para la realización del mismo.

En el marco del proyecto, el tutor del proyecto y el jefe proyectos de SITESA jugaban los papeles de clientes que establecen los requisitos de la arquitectura a desarrollar. Por su lado, el proyectando juega el papel real de analista/desarrollador encargado de plasmar los modelos necesarios para llegar a una solución que valida los requisitos. Debido al carácter innovador del proyecto, la inestabilidad de los requisitos software, lo novedoso de la tecnología empleada y la premura de los plazos de finalización, se ha necesitado del uso de *metodologías ágiles* para su desarrollo.

1.4 INTRODUCCIÓN A LOS SISTEMAS ESPACIALES

En esta sección se introducirán algunos términos y conceptos que serán de utilidad para comprender el resto de capítulos de esta memoria. Se pretende hacer una introducción somera a los sistemas espaciales para personas noveles en el tema, siendo posible comprender el trabajo en su totalidad sin acudir a bibliografía suplementaria. Por ello, esta sección puede ser eludida por iniciados en el mundo de los sistemas espaciales.

Los términos Sistema de Información Geográfica (SIG) y sistema espacial pueden ser utilizados indistintamente en este trabajo, debido a que los sistemas espaciales utilizados para desarrollar este proyecto han sido sistemas geográficos. SIG es un término multidisciplinar, donde se pueden incluir especialidades como cartografía, topografía, informática, etc., mientras que el término sistema espacial parece estar más restringido al ámbito informático. Por ello, a lo largo de este documento se utilizará el término sistema espacial.

1.4.1 Definiciones de sistema espacial

Los sistemas espaciales han sido una verdadera revolución conceptual y práctica en el manejo y análisis de la información espacial. De hecho, los sistemas espaciales son el paso adelante más importante desde la invención del mapa en cuanto a la utilización de los datos espaciales. Un mejor entendimiento de la realidad de este fenómeno y de utilidad de estas herramientas de análisis se puede obtener revisando algunas de las definiciones sobre lo que son y lo que hacen los sistemas espaciales desarrolladas por diversos autores.

En algún caso, la explicación del contenido de un sistema espacial se basa en el tipo de información que maneja, y así se define como: "*Base de datos computerizada que contiene información espacial*" [CM1986].

En otro planteamiento se insiste en las capacidades y funciones de que están dotados los sistemas espaciales: "*Un sistema hardware, software y procedimientos elaborados para facilitar la obtención, gestión, manipulación, análisis, modelado, representación y salida de datos espacialmente referenciados para resolver problemas complejos de planificación y gestión*" [NCGIA].

Una de las aplicaciones más importantes de los sistemas espaciales son los Sistemas de Información Geográfica que según lo concibe algún autor se puede definir como: "[*Un*] *Modelo informatizado del mundo real, descrito en un sistema de referencia ligado a la Tierra, establecido para satisfacer unas de las necesidades de información específicas respondiendo a un conjunto de preguntas concreto*" [RP1993].

De acuerdo con esta última definición, un SIG es un intento, más o menos logrado, según los casos, de construir una visión esquemática de una realidad compleja.

Varios hechos son importantes en las citadas definiciones: la capacidad de este dispositivo informático para gestionar/analizar datos espaciales y la combinación de distintas funciones operativas definidas sobre este tipo de información: (i) introducir datos espaciales en el ordenador; (ii) creación de una base de datos que conserve sus características de modo económico y coherente; (iii) gestión y manipulación para interrogar a la base de datos; (iv) análisis y generación de nueva información a partir de la ya incluida en la base de datos; (v) representación cartográfica y por otros medios de la base de datos.

Desde otro punto de vista, con la misma denominación (sistema espacial) se suele hablar de dos cosas ligeramente deferentes: en primer lugar, de una aplicación informática especialmente diseñada para cumplir las anteriores funciones y, por otro lado, de una base de datos especializada que contiene los datos espaciales.

1.4.2 Representación digital de los datos espaciales

La primera cuestión en el desarrollo de un sistema espacial, entendido como base de datos, es como representar de forma digital la información espacial. Dos cuestiones son fundamentales en el proceso de esquematizar la realidad para convertirla en el modelo representado en la base de datos. En primer lugar, la manera en que se concibe el mundo real, y después, cómo sintetizar los diversos componentes de un dato espacial.

En la actualidad, predomina en los sistemas espaciales una visión del mundo real que se puede denominar **estratos**, **layers** o **capas**. Según esto, el mundo está compuesto de infinitos lugares cuya localización se puede medir en cualquier grado de precisión espacial a través de un sistema de coordenadas. La geografía de ese mundo se organiza en distintas variables temáticas, cuyos valores se pueden estimar en cualquier lugar. Cada variable es una capa (*layer*) de la base de datos. En cada capa los datos tienen los mismos componentes conceptuales.

Los componentes de un dato espacial son:

- **Espacial**:
  - **Geometría:** posición absoluta de cada objeto respecto a unos ejes de coordenadas (x/y).
  - **Topología:** relaciones entre los objetos (polígono A es vecino de B, la línea E corta el polígono C, el punto D está fuera del polígono A, etc.).
- **Temático:** variables ligadas a cada objeto (número de habitantes en cada municipio, etc.).

En general, en un dato espacial se pueden diferenciar dos aspectos conceptuales: el espacial (geometría más topología) y el temático. Un sistema espacial debe ser capaz de representar digitalmente ambos. Existen varias posibilidades para organizar esta doble base de datos (espacial y temática). En primer lugar, el modelo de sistema espacial denominado híbrido, que utiliza dos bases de datos diferentes, una para cada una de los dos elementos fundamentales, y por lo tanto está constituido por la base de datos espacial y la base de datos temática. En segundo lugar, la otra posibilidad de organización es incluir ambos tipos de datos en una única base de datos mixta, que reúna tanto las características espaciales como las temáticas. En la actualidad el modelo híbrido tiene más éxito y difusión entre los programas espaciales comerciales. El modelo integrado es más lento en su funcionamiento por lo que requiere ordenadores más potentes.

A continuación se describen diferentes alternativas de representar los componentes espaciales (datos vectoriales y datos raster).

1.4.3 El modelo de datos vectorial

Este modelo define un objeto espacial de la realidad a través de sus límites o fronteras con el exterior. Para ello se establece, mediante unos ejes de coordenadas, la posición de una serie de vértices que unidos dos a dos forman líneas rectas y facilitan la delimitación de esas fronteras de los objetos geográficos. Estas líneas rectas se denominan **segmentos**, y una sucesión de segmentos forman una **polilínea**. Aunque estas definiciones son las más extendidas, algunos autores llaman segmento a lo que aquí se ha definido como polilínea. En este proyecto se emplearán ambas definiciones indistintamente. El contexto de cada cita aclarará el sentido concreto empleado.

Dentro de este enfoque vectorial, existen dos formas distintas de organizar y/o estructurar los datos: (i) en lista de coordenadas, y (ii) en estructuras con topología.

1.4.3.1 Lista de coordenadas

En este caso el modelo vectorial es simple, pero también menos potente y capaz; en realidad esta forma del modelo de datos vectorial es especialmente adecuada para la representación cartográfica, pero no lo es tanto para el análisis espacial.

La solución consiste en representar los objetos espaciales puntuales por un par de coordenadas y un identificador unívoco para cada uno de ellos. Los objetos lineales aparecerán representados por una serie de pares de coordenadas, las suficientes para aproximar, mediante segmentos lineales, el recorrido de la línea reproducida. Igualmente se emplea un identificador para nombrar de manera única a cada una de las líneas existentes. En tercer lugar, los objetos poligonales se representan de manera similar a las líneas, con la diferencia de que los segmentos se cierran y delimitan una superficie.

De ese modo, con una serie de ficheros, uno al menos para cada tipo de objeto espacial (puntos, líneas y polígonos), se representan digitalmente los aspectos geométricos de los objetos espaciales. La cuestión temática asociada a cada uno de ellos se incluye en otro fichero o base de datos.

1.4.3.2 Estructura con topología

En este caso la descripción digital de los aspectos geométricos de los objetos espaciales es más compleja. Ahora se añade a la mera posición geométrica de las fronteras de los objetos espaciales alguna información sobre las relaciones de proximidad, contigüidad, etc. existentes entre los diversos objetos representados (la llamada *topología*, que completa a la simple geometría recogida por las coordenadas).

Para conseguir esto, se establecen dos tipos de vértices en la descripción de las fronteras: los vértices propiamente dichos y los nodos. Estos últimos son aquellos donde se unen tres o más líneas, las cuales forman los llamados arcos. Un arco es un conjunto de segmentos rectos orientados (con origen en un nodo concreto y destino otro nodo) que poseen la misma topología, en este caso medida simplemente por tener, todos ellos, a la izquierda y a la derecha siempre los mismos polígonos. De este modo, se puede organizar los ficheros de la siguiente forma: fichero 1, que recoge esencialmente la geometría o posición de los vértices de las fronteras de los objetos poligonales; fichero 2, que recoge la relación entre segmentos rectas/arcos y vértices, y el fichero 3, que muestra la topología de los polígonos.

1.4.4 El modelo de datos raster

El modelo de datos raster representa digitalmente la información espacial de un modo diferente y, en cierto modo, complementario al anterior. Ahora lo que se codifica en el ordenador es el contenido de los objetos espaciales, en lugar de sus límites exteriores. Para ello, el procedimiento consiste en superponer al mapa a representar una rejilla formada de unidades regulares, normalmente cuadrados o rectángulos, con lo cual el espacio geográfico queda particionado en forma sencilla y regular, y por ello fácil de representar. A continuación se trata de representar qué objeto espacial/valor temático existe en cada una de las unidades de la rejilla, y estos valores son almacenados en el ordenador de manera secuencial, conservando así su posición relativa, que representa la posición geográfica. Básicamente, dos son las formas de estructurar esta información:

- **Estructura descriptiva:** en esta forma, la representación raster utiliza un número o valor para cada elemento de la rejilla (también denominado *píxel*), lo que la convierte en muy premiosa y detallada, y por lo tanto más complicada de manejar y de guardar en el ordenador.

- **Estructura *run length:*** esta otra posibilidad de almacenar la información raster, aprovecha la circunstancia usual de muchas situaciones geográficas, se trata de la repetición de los mismos valores en píxeles contiguos. Por ello, basta con indicar, para cada fila de la rejilla el valor temático que aparece y la columna final hasta la cual este valor se repite.

1.5 ESTRUCTURA DEL PRESENTE DOCUMENTO

La presente memoria está organizada en forma de capítulos con la siguiente distribución:

El presente Capítulo, denominado ***Introducción*** trata de aportar una visión sobre los sistemas espaciales. En él se aporta notas de la motivación del proyecto fin de carrera y se discute los objetivos principales del mismo.

En el Capítulo 2, ***Estado del arte de los Servidores de Mapas y Cálculo de Rutas,*** se realiza una descripción de los conceptos *servidores de mapas* y *cálculo de rutas* ofreciendo una serie de alternativas comerciales para implementar dichos conceptos. En una de sus secciones se realiza un estudio de las diferentes alternativas indicando cual ha resultado ser la más conveniente para el desarrollo del proyecto.

El Capítulo 3, ***ArcIMS y Network Analyst***, se entra en profundidad en el funcionamiento de ArcIMS como IMS y de Network Analyst como tecnología utilizada para realizar cálculo de rutas.

Es ya en el Capítulo 4, ***Metodología. Análisis del sistema***, donde se detalla la metodología seguida en el desarrollo del proyecto y el análisis de la funcionalidad requerida por los módulos que componen la solución.

El Capítulo 5, ***Arquitectura del sistema***, es una continuación del anterior, donde se muestra la arquitectura de la solución de forma detallada, incluyendo los diferentes módulos desarrollados. Se detallan dos ejemplos para ver como intervienen cada uno de los componentes que forman la arquitectura. Además, se aporta una descripción de la tecnología empleada para el desarrollo de los módulos.

En el Capítulo 6, ***Caso de Estudio***, se aportan una serie de casos prácticos para una mejor comprensión de la forma de funcionar de los diferentes módulos desarrollados.

El Capítulo 7, ***Conclusiones y trabajos futuros***, se enumeran las conclusiones obtenidas del trabajo desarrollado con este proyecto fin de carrera y los trabajos futuros que se pueden llevar a cabo relacionados con lo realizado en este proyecto fin de carrera.

Capítulo 2. ESTADO DEL ARTE

"Nunca consideres el estudio como un deber,
sino como una oportunidad para penetrar en
el maravilloso mundo del saber"

Albert Einstein (1879-1955)
Científico estadounidense de origen alemán.

En este capítulo se describirán dos componentes fundamentales para la desarrollo de este proyecto fin de carrera como son los Servidores de Mapas por Internet (Internet Map Server, IMS) y tecnologías para el cálculo de rutas. Se comenzará dando una descripción genérica de lo que son IMS y para que se emplean. Se estudiarán diferentes alternativas de IMS, seleccionando la que mejor se ajuste a las necesidades del proyecto. Al final del capítulo, se hace un estudio de diferentes alternativas para los gestores de cálculo de rutas, seleccionando la que mejor se adecua al proyecto.

2.1 SERVIDORES DE MAPAS POR INTERNET

2.1.1 Introducción

A medida que Internet se convierte día a día en un canal de comunicación más importante y ofrece mayores posibilidades para transmitir y recibir todo tipo de información, los sistemas de información geográfica (SIG) se están complementando con este desarrollo, y en consecuencia, reemplazando por medios de comunicación interactivos a través de la red.

En muy pocos años, Internet ha evolucionado desde un sistema hipermedia hasta una completa plataforma informática. Para los usuarios de información

geográfica eso significa que gran parte del trabajo que se realiza en una computadora local se puede obtener a través de Internet.

Este paso en el desarrollo de la tecnología Cliente-Servidor ha posibilitado la implementación de aplicaciones que ha permitido movilizar desde los documentos preparados y estáticos hasta una plataforma interactiva y dinámica. De forma virtual, cualquier computadora conectada a Internet puede ofrecer un servicio y usando un navegador, se podrá acceder al mismo.

El factor clave que ha permitido lograrlo, ha sido el uso de la tecnología Internet Map Server (IMS). A partir de esta tecnología, se han desarrollado varios sistemas como MapObject IMS, MapServer, ArcView IMS, ArcIMS o MapGuide entre otros, que permiten crear aplicaciones SIG en Internet / Intranet para visualizar, consultar y analizar información geográfica por la red.

Pero para interactuar de forma adecuada con algunos servicios que presentan una tecnología muy avanzada, se necesita mejorar las aptitudes (HTML) del navegador, aumentando sus funcionalidades con la instalación de otro software. Dependiendo de la técnica que se use, se habla de un programa de "Plugin" Java o un "Control" ActiveX. Otra opción es usar el entorno Java básico que los navegadores más comunes suministran.

Con la tecnología IMS, la información espacial publicada en la red es dinámica. La distribución de información geográfica vía Internet permite la integración en tiempo real de datos procedentes de cualquier parte del mundo. El usuario tiene acceso a los recursos de Internet, se desplaza libremente por toda la información con herramientas funcionales, cambia la representación gráfica en línea, enlaza elementos gráficos con informaciones procedentes de bases de datos, y trabaja en tiempo real con funciones de análisis [WSMI].

Las opciones de intercambiar, integrar o analizar datos de una nueva forma a través de la red, facilitan, agilizan y favorecen el proceso para la toma de decisiones. Los usuarios pueden combinar datos e información accesible vía Internet con datos locales, visualizarlos, hacer consultas y el análisis pertinente.

Este sistema distribuido de información, en comparación con herramientas "stand-alone" o instaladas en un ordenador personal ofrece, entre otras, las siguientes ventajas:

- Compartir e intercambiar datos.

- Dar acceso a aplicaciones y herramientas para el análisis y toma de decisiones a un público mucho más amplio.

- Facilitar la actualización continuada de la información, ayudando a reducir redundancias (duplicaciones) y mejorando el acceso a bases de datos.

- Facilitar la actualización de aplicaciones e información divulgada.

La arquitectura IMS consta de tres niveles:

Figura 2.1. Arquitectura IMS

- **Aplicaciones Cliente:** Entorno de trabajo del usuario. Cualquier navegador que soporte el estándar HTML puede actuar como cliente. Será necesario que también soporte Applet (Plugin) de Java o tecnología ActiveX si los servicios a los que se accede contienen estos componentes. A través de Internet y con el navegador como interfaz, el Cliente envía peticiones a la aplicación servidor para obtener la información que le interesa visualizar, consultar o analizar.

- **Aplicaciones Servidor:** Son las encargadas de canalizar y atender las operaciones que el usuario solicita sobre los datos: ArcView IMS, MapObjects IMS, ArcIMS, MapGuide, Geomedia, MMS, MapServer, etc.

- **Bases de Datos:** Las aplicaciones servidoras acceden a los datos que pueden estar almacenados en archivos o en bases de datos espaciales (ArcSDE, Oracle Spatial, etc) [ArcSDE] [OSW].

2.1.2 Definición de IMS

Los IMS permiten a los usuarios la máxima interacción con la información geográfica. Por un lado el usuario o cliente accede a la información en su formato original, de manera que es posible realizar consultas tan complejas como las que haría un SIG. Un IMS funciona enviando, a petición del cliente, desde su navegador de Internet, una serie de páginas HTML (normalmente de contenido dinámico DHTML), con una cartografía asociada en un formato de imagen (por ejemplo: una imagen GIF o JPEG). Un IMS es, de hecho, un SIG a través de Internet [WMI].

Las primeras versiones de servidores de mapas sólo permitían realizar funciones básicas de visualización y consultas alfanuméricas simples. En las versiones más recientes es posible realizar funciones mucho más avanzadas. El tiempo dirá si los servidores de mapas tendrán toda la funcionalidad de los SIGs.

El IMS se puede personalizar, es decir, se pueden preparar o programar las herramientas (los iconos de la aplicación) de manera que sean intuitivas para el usuario no experto en SIG.

2.1.3 Tipos de IMS

- **Servidores de Imágenes en Formato Mapa de BIT:** "Este es el nivel básico del IMS, basado en un servidor HTTP ordinario como el Internet Information Server (IIS) de Microsoft, el Enterprise Server de Netscape, o Apache de código abierto. Los mapas son gráficos sencillos, normalmente GIF o JPEG. Todos estos servidores ofrecen imágenes estáticas, es decir, sólo sirven para visualizar dichas imágenes.

- **Servidores de Mapas Interactivos:** El nivel siguiente, en cuanto a complejidad de los servidores de mapas, lo constituye este grupo, que realiza varias tareas de manipulación cartográfica y conexión con bases de datos, enviando imágenes vectoriales de mapas a través del servidor web, Por lo tanto, dicho servidor se utiliza únicamente para aceptar peticiones del cliente, pasarlas al IMS y devolver la información geográfica solicitada al cliente.

2.1.4 Funcionalidad de los IMS

Las principales funciones que permiten realizar los servidores de mapas son:

- Visualización, zoom para alejar o acercar los elementos cartográficos. En servidores de mapas más avanzados el usuario puede definir la extensión de los "zooms"; también puede activar y desactivar la visualización de las capas de elementos cartográficos; información dinámica al pasar el ratón sobre cada elemento cartográfico.

- Identificación de atributos alfanuméricos en cada elemento cartográfico.

- Consultas de atributos alfanuméricos sencillas, como la búsqueda de topónimos o más complejas, con operadores booleanos.

- Conexión de bases de datos locales a la base de datos remota del IMS, de cara a la creación de mapas temáticos con datos alfanuméricos propios, o para el

volcado masivo o una a una de direcciones postales como puntos en una capa de ejes de calles (geocodificación de direcciones postales o “addressmatching”).

- Selección de elementos por combinación de capas o análisis con operadores espaciales de superposición, contención, intersección, etc. de dos capas (con la creación de nuevas capas) y creación de zonas de influencia.

- Cálculo de rutas óptimas para la navegación de vehículos (“routing”). Edición básica de líneas (“redlining”) por parte del cliente, de manera que el administrador del IMS pueda recuperar esas líneas e incorporarlas a la cartografía.

- Capacidad de imprimir el mapa manteniendo la escala.

Por lo general los IMS que disponen entre sus familias de productos de una herramienta SIG cuentan con funciones más avanzadas que aquellos servidores de mapas que proceden de herramientas CAD tradicionales (AutoCad, MicroStation), a las que se ha añadido un módulo de SIG (AutoCad Map, MicroStation Geographics).

2.1.5 Arquitectura de los IMS

La arquitectura de los servidores de mapas es de tipo Cliente/Servidor. El cliente (un navegador de Internet) solicita los recursos del servidor. El servidor gestiona todas las peticiones y responde de manera ordenada a estas. La red es la estructura física a través de la cual el cliente y servidor se comunican.

El cliente, al recibir los datos del servidor los interpreta y los presenta al usuario (en el navegador como texto con un determinado estilo, tamaño de fuente, color, etc.).

En el caso de los IMS, el formato de los datos que son leídos por el cliente puede determinar el tipo de cliente, es decir, cuando el formato de cartografía que llega al cliente es de imagen (formato genérico como JPG, PNG o GIF, por ejemplo), un explorador simple HTML, que es un lenguaje totalmente transparente al navegador, por lo general es suficiente. En cambio, cuando el cliente debe leer un formato vectorial encriptado (no se trata del formato vectorial nativo de la cartografía), de manera que se puedan ejecutar funciones más sofisticadas, puede ser necesario instalar algún componente en el computador local, como “plug-ins” para Netscape, “applet” de Java o ActiveX COM de Microsoft.

Normalmente esos componentes pueden descargarse gratuitamente de Internet y no tardan más que unos instantes o breves minutos en instalarse. Aún así, no cabe duda

de que suponen un cierto inconveniente para el usuario, sobre todo si no cuenta con privilegios de administrador o ese contenido esta restringido por el “proxy” o “firewall”.

La figura 2.2 muestra un posible ejemplo de la arquitectura de un IMS.

El cliente puede ser de dos tipos: el primero, universal, preparado para leer documentos HTML estándar; y el segundo, en el que ha sido necesario añadir un “plug-ins”, es decir, un programa que aumenta las prestaciones del cliente HTML.

El cliente HTML es útil cuando se trata de publicar información geográfica en Internet, mientras que, en ocasiones se recomienda que el cliente “plug-in” se instale sólo en una Intranet, por dos razones: por una parte, el usuario no deberá instalar el componente, sino que se encargará de eso el servicio de informática corporativo; en segundo lugar, las funciones que realice el cliente “plug-in” tendrán probablemente un menor tiempo de respuesta en una intranet que en Internet

En el flujo descendente de las flechas, el cliente, realiza una petición que llegará al IMS, a través de Internet/Intranet y la recibe en primera instancia el servidor web. En el flujo ascendente el IMS atiende la petición y extrae la información del servidor de datos, presentándola al servidor web, que la envía a través de Internet/Intranet hasta el cliente.

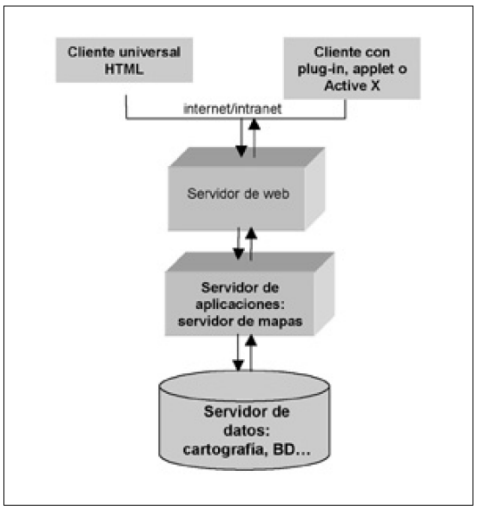


Figura 2.2. Arquitectura de los servidores de mapas

En las siguientes secciones de este Capítulo, se muestran diferentes alternativas del mercado para los IMS. Dada las características de este trabajo, en el estudio de estas alternativas se da una definición, características y funcionamiento de cada una de ellas, desarrollando en más detalle la opción elegida en el Capítulo siguiente.

2.2 ANÁLISIS DE LOS IMS MÁS UTILIZADOS

A la hora de elegir el IMS apropiado, se tiene varias alternativas en el mercado, ya sean comerciales o gratuitos, los más usados son ArcIMS, MapServer, MapGuide, GeoTools, Gis Viewer, MapObjects IMS, y muchos más.

Las alternativas estudiadas son MapServer, como estandarte de servidor de mapas de código libre y ArcIMS, como alternativa comercial desarrollada por ESRI como una aplicación muy potente, escalable y basada en estándares.

También se describe otras alternativas de software libre pero en menos detalle debido a la poca información que se ha encontrado. En el campo de sistemas de software libre se hace un estudio de los IMS conocidos como Degree, GeoServer y en el campo de los sistemas espaciales comerciales MapGuide.

2.2.1 MapServer 4.6.1

2.2.1.1 Definición

Es un sistema para la creación de aplicaciones GIS en Internet/Intranet para visualizar, consultar y analizar información geográfica por la red mediante la tecnología Internet Map Server (IMS) [WMS].

MapServer es una aplicación Common Gateway Interface (CGI) de carácter libre, la cual corre bajo plataformas Linux/Apache, Windows NT/98/95, distribuida bajo licencia GPL, desarrollada por la Universidad de Minnesota para construir aplicaciones que sirvan mapas a través de Internet.

Su modo de funcionamiento está basado en la generación de lado del servidor web de imágenes estáticas (JPEG, GIF, PNG, etc.) como resultado del proceso de las peticiones realizadas por los clientes. Estas imágenes son referenciadas posteriormente dentro de la interfaz de usuario que se le envía al cliente (código HTML).

Como consecuencia de los trabajos de estandarización en materia de SIG de OGC (OpenGeoSpatial Consortium) [WOGC], se ha avanzado mucho en el terreno de las aplicaciones de WebMapping. En este sentido, MapServer cuenta con extensiones que le permiten escuchar peticiones de mapas formuladas según el protocolo WMS (Web Map Server) de OGC y de enviar respuestas según dicho protocolo.

2.2.1.2 Características

MapServer es un proyecto de software libre puesto a disposición de la comunidad, que también se ha aprovechado de contribuciones realizadas por parte de otros proyectos de software libres. Algunos de los proyectos de software libres sobre los

que se ha apoyado MapServer son: ShapeLib, FreeType, Proj.4, GDAL/OGR, etc. [MT2005].

MapServer es totalmente autosuficiente, no necesita de otro programa servidor para procesar datos o crear informes. Entre sus características destacan:

- Dibujo y etiquetado dependiente de la escala.
- Valores de escala.
- Símbolo y color adaptable.
- Acceso en función de las características a datos sobre atributos.
- Generación automática de leyendas.
- Utilización de datos en forma de mosaico.
- Es adaptable.
- Corre bajo plataformas Linux/Apache y Windows.
- Entre los formatos raster soportados se encuentran: TIFF/GeoTIFF, GIF, PNG, ERDAS, JPEG y EPPL7.
- Entre los formatos vectoriales soportados se encuentra: Shapefiles.
- Soporta fuentes TrueType.
- Permite dibujar sobrecargas en datos tanto raster como vectoriales.

Entorno a MapServer también han surgido otros proyectos de aplicaciones de mayor riqueza como: MS4W, MapLab, VisualBasicGIS, etc.

Además, cualquier desarrollador puede crear sus propias aplicaciones utilizando el modelo de objetos de MapServer gracias a la biblioteca de componentes MapScript.

El API de MapScript puede ser utilizada por lenguajes script como PHP, Perl, Python, o incluso Java si se desarrollan los conectores JNI necesarios para acceder al API en C de MapScript.

2.2.1.3 Funcionamiento

En un primer momento, MapServer fue desarrollado por la Universidad de Minnesota como herramienta para un proyecto de distribución de datos de gestión

medioambiental por Internet. Actualmente, es mantenido por el proyecto TerraSip, patrocinado por la NASA, en el que también trabaja la universidad, y trabajan más de veinte desarrolladores en su evolución.

MapServer no es un sistema SIG completo, ni pretende llegar a serlo. Es simplemente una herramienta que permite construir aplicaciones web interactivas que permitan la visualización y consulta de información geográfica en forma de mapas, es lo que se ha venido a denominar dentro de la industria SIG una aplicación "WebMapping".

Se puede obtener diferentes versiones en la página oficial de MapServer [WMS].

MapServer generalmente se ejecuta como una aplicación CGI en un Servidor HTTP. Esto será así a menos que se construya una aplicación más avanzada con MapScript, el cual accede directamente a la API de MapServer.

Las aplicaciones CGI utilizan los siguientes recursos:

- Un servidor HTTP como Apache o Internet Information Server.
- El programa MapServer.
- Un archivo de inicialización que lance la primera vista de una aplicación con MapServer (opcional).
- Un Mapfile que controle lo que MapServer hará con los datos.
- Un archivo plantilla que controle la interfaz de usuario de la aplicación con MapServer en la ventana del explorador de Internet.

Por tanto, para la creación de los servicios de mapas se tiene:

- **El Archivo de Inicialización:** Este archivo puede ser parte de un archivo plantilla HTML, pero por simplicidad, este también puede ser otro archivo. El Archivo de Inicialización utiliza un formulario para enviar una consulta inicial al servidor HTTP, que retorna un resultado desde MapServer. MapServer es dinámico, comienza y se ejecuta cada vez que recibe una consulta, por lo tanto, el archivo de inicialización sólo se requiere para pasar una serie de parámetros iniciales (ocultos) hacia la aplicación. El archivo de Inicialización es un archivo HTML regular, por ello, su extensión es *.htm o *.html.

  Alternativamente, un hipervínculo hacia la aplicación con mapserver puede ser construido. Este pasaría los parámetros básicos requeridos por la aplicación con el CGI de Mapserver.

- **El MapFile**: define parámetros de los datos, el despliegue y las consultas que serán usados en una aplicación con MapServer. Se puede hablar del Mapfile como un archivo de configuración de la aplicación. El Mapfile también incluye información sobre como dibujar el mapa, la leyenda, y los mapas resultantes desde una consulta. Los Mapfiles normalmente tienen una extensión *.map.

- **El Archivo Plantilla**: Controla como saldrán los mapas y las leyendas desde MapServer hacia la página HTML, opera como cualquier otro archivo HTML excepto porque contiene celdas que pueden ser modificadas por el CGI de MapServer. El archivo Plantilla permite al autocolocar el mapa y la leyenda en una página, y determina la manera en que el usuario interactúa con MapServer (navegar, consultar, hacer *zoom*, etc.).

  MapServer usa el archivo plantilla y reemplaza las palabras clave de las celdas, con información de su estado actual o del estado del conjunto de datos SIG, para generar el archivo HTML final que será inicializado por el navegador. Debido a que el archivo plantilla será usado para crear un archivo HTML, debe ser guardado con la extensión *.html.

2.2.2 ArcIMS

2.2.2.1 Definición

ArcIMS constituye una aplicación muy potente, escalable y basada en estándares que permite, de manera rápida y sencilla, diseñar y gestionar servicios de cartografía en Internet [WESRI].

ArcIMS (Internet Map Server), permite la distribución de información geográfica vía Internet, así como la integración en tiempo real de datos procedentes de diferentes fuentes. ArcIMS es la solución de ESRI que proporciona una plataforma común para este intercambio.

2.2.2.2 Características

Características esenciales de ArcIMS:

- Capacidad para servir imágenes y vectores.

- Integra datos locales con datos de Internet.

- Sencilla administración y mantenimiento de los servicios publicados.

- Disponible para Windows NT y UNIX.

- Arquitectura del servidor altamente escalable.

- Representación de cartografía de alta calidad.

Las características principales de ArcIMS incluyen:

- Capacidad de combinar datos procedentes de múltiples fuentes en un mismo servicio GIS de ArcIMS.

- Amplia gama de funcionalidad GIS, incluida la posibilidad de calcular rutas y geocodificar direcciones.

- Asistentes muy intuitivos que permiten la creación, el diseño y la gestión de sitios Web de forma muy sencilla y sin necesidad de desarrollo.

- Soporte para una gran variedad de clientes.

- Arquitectura del servidor altamente escalable que permite ampliar la capacidad del servicio según van aumentando las necesidades, sin necesidad de rediseñar el sistema.

- Integración con los productos de **ArcGIS Desktop**, tanto en cuanto a posibilidad de acceso desde estos productos a servicios ofrecidos por ArcIMS, como a la posibilidad de publicar mapas generados con **ArcMap**.

- Sencilla instalación, implementación y administración mediante asistentes y plantillas.

2.2.2.3 Funcionamiento

La puesta en marcha de un servicio de ArcIMS es muy sencilla e intuitiva gracias a los asistentes que incorpora el producto [ESRI4].

- **ArcIMS Author**: es el asistente para la generación del fichero de configuración del servicio de mapas. [ESRI4]. Se basa en establecer que capas de datos van a definir el servicio, es decir, que capas son las necesarias para crear el servicio de mapas, etc.

  En esta fase es donde se fijan las propiedades de las capas, las escalas de visibilidad, el modelo de geocodificación, las consultas predefinidas, el etiquetado a utilizar, así como el origen de los datos, en la figura 2.3 se muestra la interfaz del asistente ArcIMS Author.

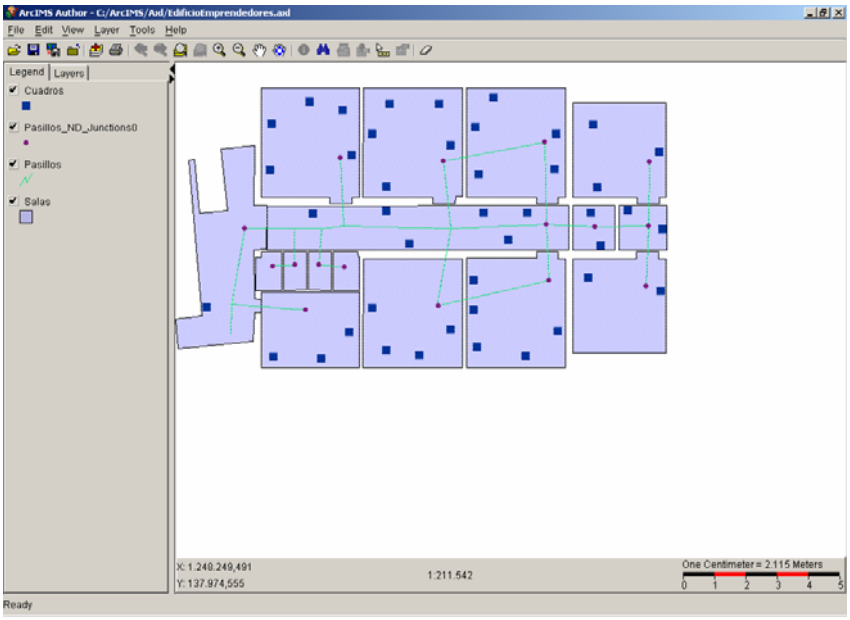


Figura 2.3. ArcIMS Author

- **ArcIMS Administrator**: es el asistente para la administración de los servicios publicados y de los servidores espaciales. [ESRI4].

  ArcIMS Administrator es el encargado de la explotación de los distintos servicios de mapas. Entre sus funcionalidades se encuentran:

  - Añadir y configurar los servicios de mapas al *Website.*
  - Realizar el balanceo de carga.
  - Administrar los servidores espaciales.
  - Asignar tareas a los servidores.
  - Monitorizar la comunicación entre cliente y servidor.
  - Actualizar automáticamente la configuración del *Website.*
  - Proporcionar información estadística.

  ArcIMS está diseñado para poder añadir y borrar servicios sin necesidad de detener aquellos que no se ven afectados por los cambios, en la figura 2.4 se muestra la pantalla principal de ArcIMS Administrator.

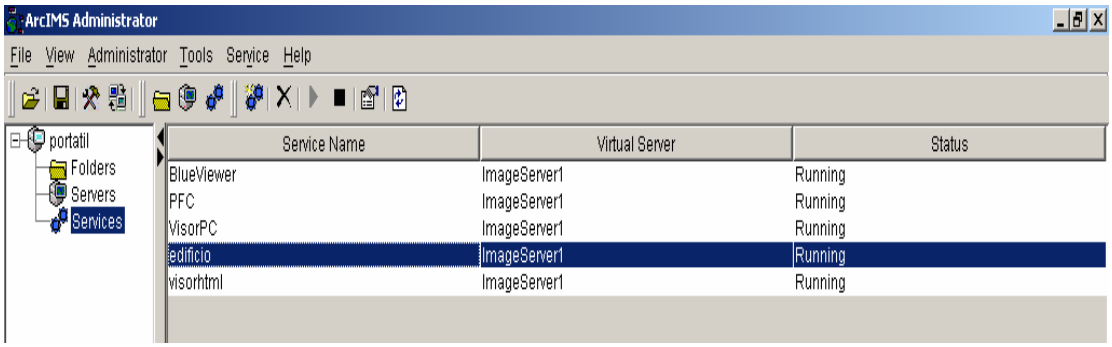


Figura 2.4. ArcIMS Administrator

El punto fuerte del ArcIMS Administrator es manejar todos estos componentes con el fin de crear un sistema que permita la distribución de datos y la funcionalidad SIG en Internet.

- **ArcIMS Designer**: es el asistente para el diseño del sitio web que permite definir la funcionalidad a la que tendrá acceso el cliente. [ESRI4].

  La funcionalidad del ArcIMS Designer es construir el *website* que finalmente será accesible por los clientes. En esta fase se establecen las funcionalidades que estarán presentes en el navegador y es cuando se decide que tecnología utilizar en el cliente: HTML o Java.

  El diseño del *website* es mucho más que un servicio de mapas. Este incluye la leyenda, la barra de herramientas, la escala y la vista global. La salida de ArcIMS Designer es una serie de páginas HTML, que pueden ser utilizadas directamente o personalizadas para satisfacer necesidades específicas.

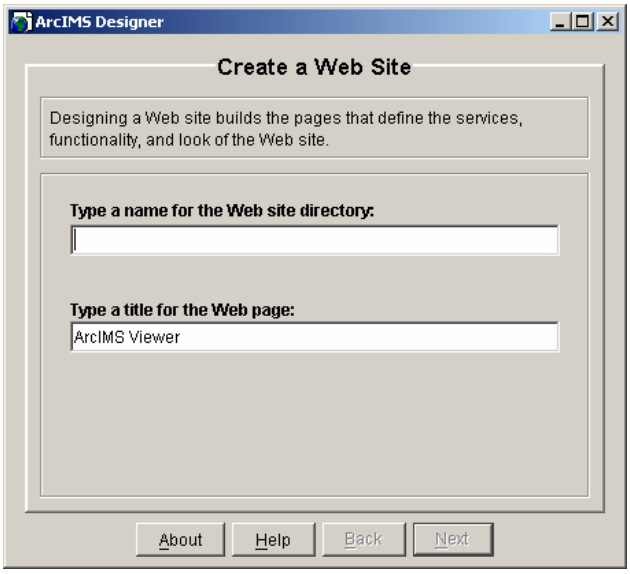


Figura 2.5. ArcIMS Designer

- **ArcIMS Manager**: es el asistente que permite crear aplicaciones web desde cualquier navegador, que permitirá al usuario realizar todas las tareas necesarias para la puesta en marcha de un servicio GIS. ArcIMS Manager combina tres aplicaciones que pueden utilizarse de manera independiente: ArcIMS Author figura 2.3, ArcIMS Administrator figura 2.4 y ArcIMS Designer figura 2.5.

Por tanto, la creación y explotación de nuestros servicios de mapas se divide en tres fases:

- **Fichero de Configuración:** La puesta en marcha de un servicio de mapas comienza con la creación del fichero de configuración del servicio, el cual recoge la información que contendrá el mapa, y la forma de representar dicha información (disposición de capas, simbología, control de escalas, …). El fichero de configuración se puede crear con la aplicación **ArcIMS Author** y con ArcMap (**ArcView**, **ArcEditor** y **ArcInfo**).

- **Tipo de Servicio:** La aplicación **ArcIMS Administrator** permite determinar qué tipo de servicio se va a implementar (servicio de imágenes o vectores, servicio de ArcMap Server, metadatos, rutas o servicio de extracción).

- **Funcionalidad GIS del Cliente:** La aplicación ArcIMS Designer permite sin desarrollo de ningún tipo crear el cliente web y definir la funcionalidad GIS definida en ese cliente.

2.2.3 Otros IMS

2.2.3.1 El servidor Deegree

Deegree es una aplicación servidor para la construcción de un SIG. Degree es un software libre, bajo la licencia GPL diseñado por el Departamento de Geografía de la Universidad de Bonn (Alemania) para cubrir la demanda de servidores web en el campo de los sistemas de información geográfica [WGT].

En 1997 se creó una iniciativa en el ámbito de la investigación “El servidor experimental EXSEGIS” en el Departamento de Geografía de la universidad de Bonn. Para trabajar en el campo de los sistemas de información geográficos a través de Internet, este proyecto ha derivado en el actual Degree.

Uno de los primeros resultados experimentales fue el “atlas interactivo de las elecciones federales” para Alemania, un Java applet, fácil de utilizar que permite visualizar mapas, definidos por los usuarios, de las elecciones.

Degree se caracteriza por su interoperabilidad al implementar, fielmente, los estándares del OGC [WOGC]. Es independiente de las diversas plataformas ya que está implementado en Java.

Entre sus características se puede encontrar:

- Soporta datos vectoriales en formatos: ESRI shapefiles, ORACLE Spatial, PostGres/PostGIS, MySQL y JDBC-enabled database.
- Soporta datos raster en formato: JPEG, GIF, PNG, (Geo) TIFF, PNM y BMP.
- Construcción de mapas temáticos usando expresiones lógicas o clases regulares.
- Detección de colisión de las anotaciones.
- Soporta distintos tipos de proyecciones.
- Distintos tipos de formatos de salida: GIF, PNG, JPEG, WBMP

2.2.3.2 EL servidor GeoServer.

Servidor cartográfico que permite consultar y modificar datos geográficos a través de Internet. Con GeoServer se puede publicar datos de mapas/imágenes utilizando WMS y WFS, se pueden modificar los elementos de las capas ya que implementa WFS-T. Es un software libre, bajo la licencia GNU [WGT].

Entre sus características se encuentra:

- Soporta datos vectoriales en formatos: ESRI shapefiles, ORACLE Spatial, PostGIS, MySQL, ArcSDE.
- No soporta datos raster.
- Construcción de mapas temáticos usando expresiones lógicas o clases regulares.
- Detección de colisión de las anotaciones.
- Soporta distintos tipos de proyecciones.

2.2.3.3 MapGuide

Mapguide dispone de capacidades similares a las de ArcIMS y ha sido diseñado para dar acceso en línea a datos espaciales en formato raster y vector. El sistema es compatible con sistemas operativos estándares (MS Windows, Sun Solaris, Apple MacIntosh) y desarrollado a partir de la arquitectura cliente-servidor. Al igual que

ArcIMS, dispone de clientes de mapas que proveen toda la funcionalidad necesaria para desplegar datos espaciales y navegar en la información publicada [WGT].

2.2.4 Comparativa entre IMS

Una vez se ha realizado una visión general de las diferentes alternativas seleccionadas para IMS, se va a evaluar cuales son las diferencias más significativas que se han encontrado en el estudio de las diferentes propuestas.

En la tabla 2.1 se muestra una comparativa entre los diferentes servidores IMS que se han estudiado en este proyecto y otros que existen en el mercado, teniendo en cuenta sus principales características.

Entre las características que se van a comparar se encuentran: si se trata de un software libre o comercial, la plataforma (sistema operativo) donde se puede aplicar, el formato de los datos de entrada (si soporta datos de entrada en formato shapefile, base de datos espacial PostGis y Oracle), el formato de datos de salida (en el caso de que la salida sea una imagen que formato puede tener), y si sigue los estándares de OGC (Open Geospatial Consortium) [WOGC], en concreto WMS (Servicio de mapas en la web que produce mapas en formato imagen bajo demanda para ser visualizados por un navegador web o cualquier otro cliente) y WFS (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).

	Plataforma	**OGC**		**Libre**	**Formato de entrada de los datos**					**Formato de la salida de los datos**			
		WMS	**WFS**		**shp**	**PostGis**	**Oracle**	**TIFF**	**JPEG**	**SWF**	**TIFF**	**JPEG**	**GIF**
MapServer	Windows Linux	SI	SI	Si	Si	Si	No	Si	Si	Si	No	SI	SI
Degree	Independiente	SI	¿	Si	Si	Si	Si	Si	Si	No	No	SI	SI
GeoServer	Independiente	SI	SI	Si	Si	Si	Si	No	No	No	No	SI	SI
ArcIMS	Windows Linux	SI	SI	No	Si	Si	Si	Si	Si	No	Si	SI	SI
MapGuide	Windows Solaris Macintosh	¿	¿	No	No	¿?	¿?	Si	Si	No	¿?	SI	SI
Alov Map	Independiente	NO	NO	¿	Si	¿?	¿?	Si	Si	Si	Si	SI	SI

Tabla 2.1. Comparativa entre diferentes IMS

En las tablas 2.2 y 2.3 se muestra una comparativa teniendo en cuenta los requerimientos funcionales y no funcionales de los IMS.

En la tabla 2.2 se puede ver una comparación de las principales funcionalidades de los IMS de carácter libre. De dicha tabla se puede concluir que las limitaciones en este software son muy variadas y están condicionadas al tipo de aplicación que se desea implementar.

Por otro lado, la tabla 2.3 muestra las principales funcionalidades de los servidores de mapas comerciales. De esta tabla se puede concluir que las funcionalidades del software comercial son muy similares a las funcionalidades del software libre, aunque se destacan algunas mejoras en ciertas funcionalidades como son: *visualización de documentos, generación de informes de consultas, operaciones geométricas básicas, etc.*

La diferencia más importante entre el software comercial y el software libre la marca el precio de la licencia, los requerimientos mínimos de hardware y la velocidad de respuesta a las peticiones Web.

Por tanto, realizando un análisis de las tablas comparativas de software libre y comercial, se puede concluir decidiendo que el software comercial cuenta con mayores funcionalidades que las alternativas de software libre. Por ello, se elige un software comercial para el desarrollo del proyecto.

Requerimientos funcionales y no funcionales	Capacidades del software		
	GeoTools	GIS Viewer	MapServer
Navegación y visualización dinámica e interactivo	B	B	B
Selección de elementos	B	B	B
Control de visualización según detalle	B	B	B
Consultas gráficas y lógicas	B	B	B[X]
Operaciones geométricas básicas (corredor, distancia)	N	N	N
Variación de simbología y colores	B	N	B
Creación de elementos gráficos temporales	B	N	N
Operaciones geométricas avanzadas (unión, intersección)[6]	N	N	N
Generación de reportes sobre consultas	N	N	B
Visualización de documentos	N	N	N
Generación e impresión automatizada de mapas	P	P	P
Desarrollo en idioma español	N	P	P
Acceso directo a Base de Datos	B/P	P	B
Implemento de sistema de Metadatos	B	P	B

B = Funcionalidad básica, P = Programable, (X) = Limitada, N = No implementada.

Tabla 2.2. Comparación de funcionalidades de softwares libres

Requerimientos funcionales y no funcionales	Capacidades software		
	ArcIMS	MOIMS	MapGuide
Navegación y visualización dinámica e interactivo	B	B	B
Selección de elementos	B	P	B
Control de visualización según detalle	B	P	B
Consultas gráficas y lógicas	B	P	B^{λ}
Operaciones geométricas básicas (corredor, distancia)	B	P	B
Variación de simbología y colores	B	P	N
Creación de elementos gráficos temporales	B	P	P
Agregación de capas adicionales (local/web)[b]	B/B	P/P	P/N
Operaciones geométricas avanzadas (unión, intersección)[b]	N	P	N
Actualización de datos geográficos y de atributos	B^{λ}	P	P
Generación de reportes sobre consultas	P	P	B
Visualización de documentos	P	P	P
Generación e impresión automatizada de mapas	B^{λ}/P	P	B
Desarrollo en idioma español	B	P	B
Seguridad de datos y restricción de acceso	B	P	B^{λ}
Acceso directo a SQLServer	B/P	P	B
Implemento de sistema de Metadatos	P	P	P

B = Funcionalidad básica, P = Programable, (X) = Limitada, N = No implementada.

Tabla 2.3. Comparación de funcionalidades de software comerciales

A continuación se muestra un gráfico comparando diferentes alternativas de IMS, como estudio realizado por la revista electrónica de tecnologías geoespaciales, Directions Magazine, durante el "Primer Concurso Anual de WebMapping" [WSGT].

Según los resultados publicados al término del concurso, la tecnología ArcIMS de ESRI está presente dentro del 29% (ver figura 2.6) de aplicaciones, mientras que la segunda tecnología que se utilizó en las aplicaciones representa un 11%, reafirmando el liderazgo de ArcIMS dentro del mercado de Servidores de Mapas por Internet.

Por tanto, teniendo en cuenta el gráfico sobre aplicaciones GIS se puede ver que ArcIMS es el líder indiscutible dentro de las preferencias de los usuarios a nivel mundial que han desarrollado Aplicaciones de Mapas para Internet y observando la tabla 2.3 se muestra que el software comercial que tiene más ventajas al resto es ArcIMS, siendo éste el elegido para el desarrollo del proyecto.

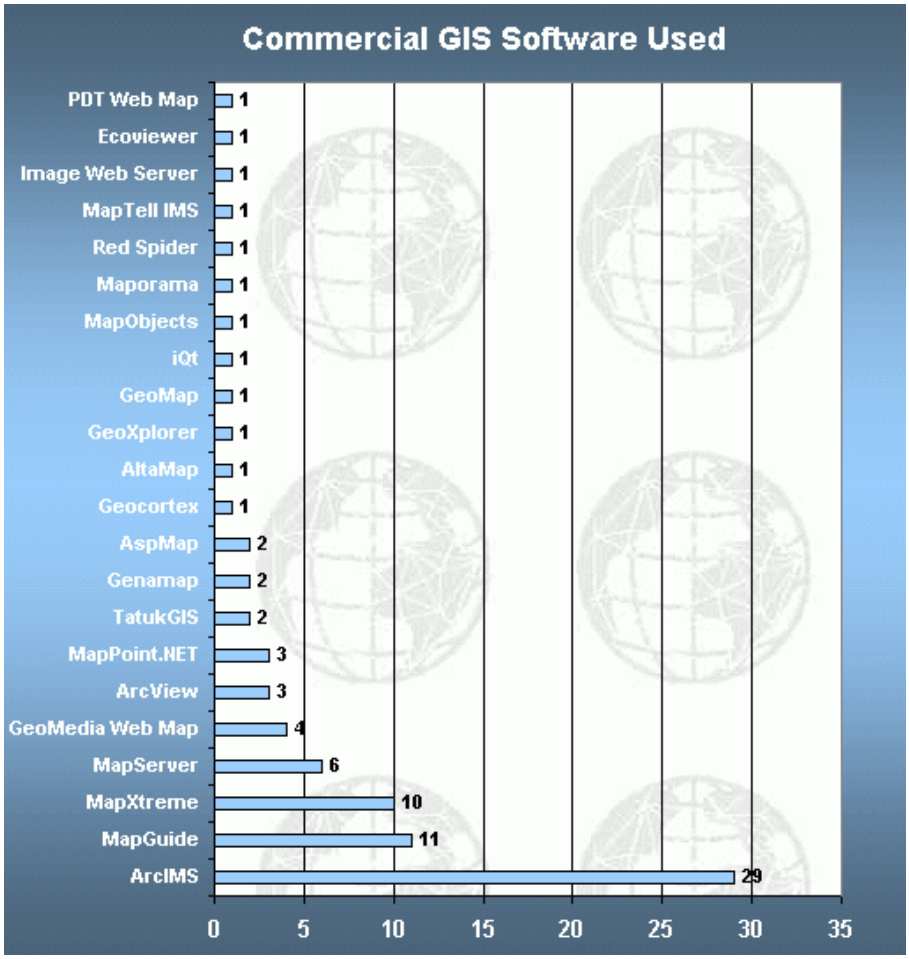


Figura 2.6. Gráfico comparativo de aplicaciones GIS

2.3 GESTORES DE CÁLCULO DE RUTAS

Es un software que a partir de un conjunto de posiciones iniciales y un conjunto de posiciones finales es capaz de obtener varios tipos de caminos, como pueden ser: el camino más corto, caminos alternativos, etc. Este software detallará las pautas que permitirán recorrer el camino obtenido [ESRI2005].

2.3.1 Network Analyst

Network Analyst es la extensión de ArcGIS Desktop 9.1 que resuelve problemas de rutas con redes multimodales como la generación de la ruta más eficiente, localización de las ubicaciones más próximas, generación de áreas de servicio basadas en tiempos de viaje, cálculo de matriz de origen-destino y listado de informe de direcciones.

Esta extensión incorpora un potente dataset de Redes que permite la modelización de redes multimodales.

2.3.1.1 Características y Funcionalidad de Network Analyst

- Permite trabajar con **redes complejas** y **de gran tamaño**.

- Se ha desarrollado un **modelo avanzado de atributos** para establecer impedancias en las redes y en los giros (incluyendo giros multiparte), restricciones de cambios de sentido o el acercamiento al bordillo en las paradas. Todo ello junto con jerarquías en las redes que permiten mejorar los análisis y generar rutas más realistas, por ejemplo compartiendo paradas.

- Completamente **integrada** con las **herramientas** de geoprocessing, modelos y scripts.

- Los datos de la red pueden ser shapefile, geodatabase o datos SDC (Smart Data Compressed).

- Se puede trabajar con ventanas de **tiempos** y definir la **duración** de las paradas en las rutas, además de establecer una señalización adecuada de las mismas, definir sus atributos (como la altura máxima en túneles, por ejemplo) y establecer barreras de forma dinámica.

- Permite realizar **personalizaciones de los informes de direcciones** y **generar matrices de origen/destino**.

En la figura 2.7 se muestra un ejemplo de uso de NetWork Analyst sobre la herramienta ArcMap.

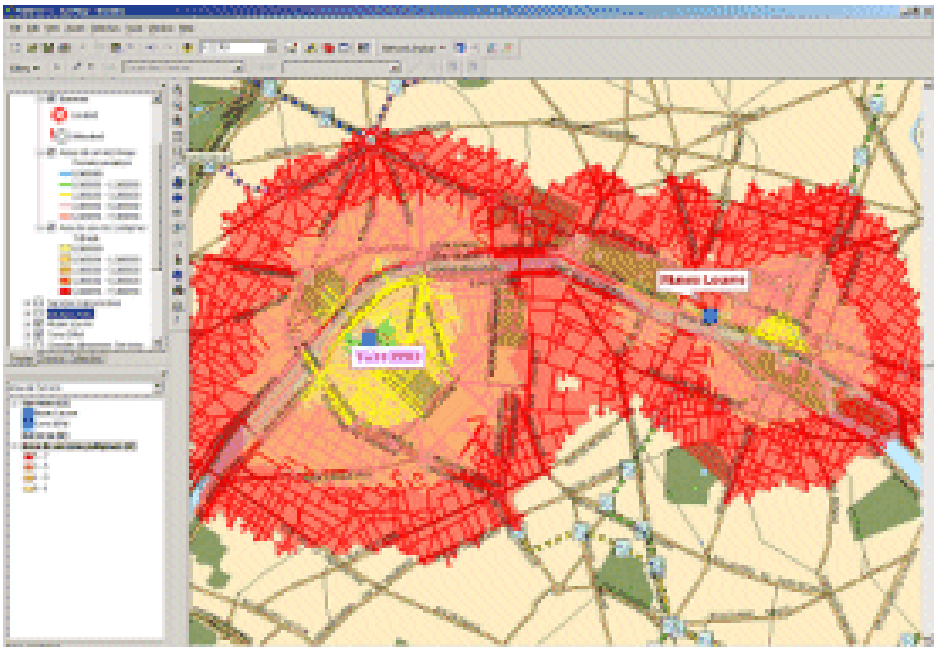


Figura 2.7 Ejemplo de NetWork Analyst

2.3.2 ArcIMS Route Server

ArcIMS Route Server permite añadir de forma rápida y fiable, funcionalidad de cálculo de rutas y geocodificación (directa e inversa) dentro del Servidor ArcIMS. ArcIMS Route Server, es por tanto de utilidad para todo aquel que use ArcIMS y necesite además la capacidad de realizar cálculos de rutas y funciones de geocodificación inversa [WESRI].

2.3.2.1 Características

- **Cálculo de Rutas:** Es posible realizar un cálculo de rutas punto a punto, con dos o con múltiples paradas, mostrando rutas a través de un barrio, una ciudad, un país o entre varios países, y recibiendo los resultados en forma de mapas y listados de instrucciones de viaje. ArcIMS Route Server permite especificar distintos parámetros que influirán en el resultado obtenido (preferencias en el uso de determinadas vías, ruta más corta, ruta más rápida, etc.)

- **Geocodificación:** ArcIMS Route Server proporciona la capacidad de localizar direcciones en un callejero, así como la capacidad de realizar geocodificación inversa, el proceso que encuentra una dirección al interactuar con un punto del mapa.

- **Formato optimizado:** ArcIMS Route Server utiliza el formato de datos SDC (Smart Data Compression), especialmente optimizado para conseguir un alto rendimiento en procesos de geocodificación y cálculo de rutas.

- **Clientes:** Todos los clientes de ArcIMS pueden acceder a servicios de ArcIMS Route Server y aprovechar las ventajas de los servicios de rutas y geocodificación y rutas que éste ofrece.

En la figura 2.8 se muestra un ejemplo de una página web utilizando Route Server para realizar el cálculo de rutas.

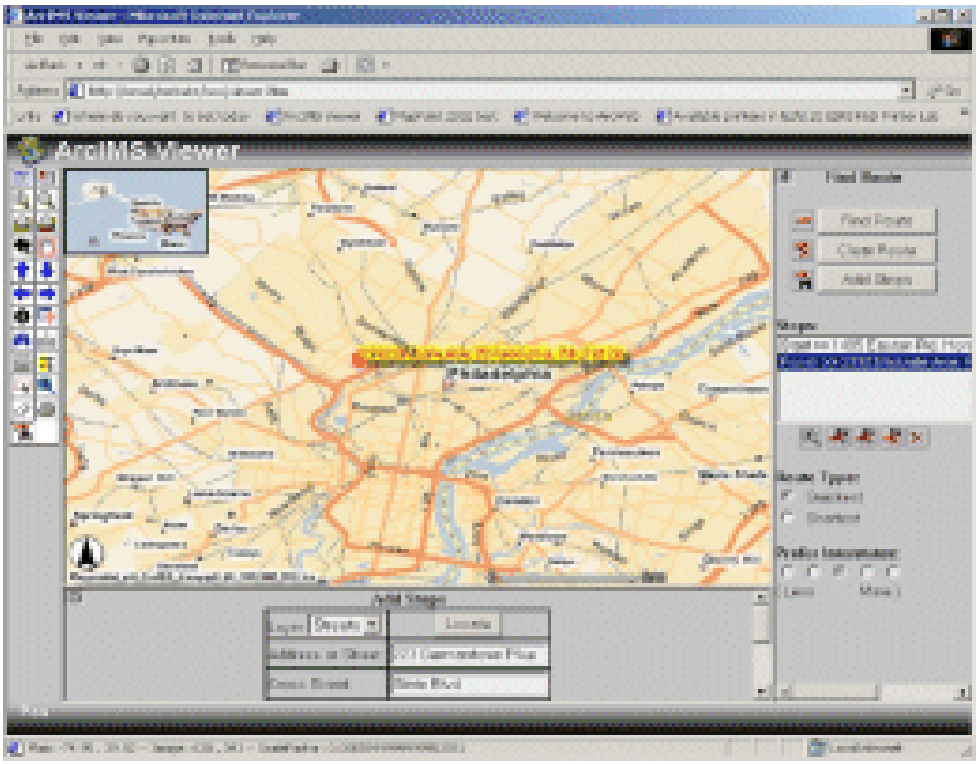


Figura 2.8. Website con Route Server

2.3.3 Comparativa entre gestores de rutas

Según las definiciones encontradas, junto con sus características y funcionalidad, se plantea un problema a la hora de la elección del gestor de rutas, ya que las dos posibles alternativas se adaptarían a las características del proyecto.

Por ello, se realiza un estudio más detallado en términos de facilidad de uso, coste de licencia y mejor integración con diferentes productos.

Si se atiende a la característica de facilidad de uso es más conveniente elegir Network Analyst en vez de Route Server, ya que este último necesita preparar las capas con toda la información necesaria para abordar el problema, enviarlas a ESRI que será el encargado de procesarlas y encontrar una solución para las rutas y una vez echo esto ESRI las devuelve y ya pueden ser integradas en Route Server.

Sin embargo, con NetWork Analyst la preparación de los datos es mucho más simple, se define una serie de capas, una red sobre estas capas que permite conocer como están conectados los diferentes puntos de origen y destino. Todo este proceso se puede realizar a través de una herramienta visual de fácil manejo llamada **ArcMap**.

En cuanto a los costes de licencia hay que mencionar que la licencia de Route Server supone un coste mayor que la licencia de Network Analyst.

Por último, hablando de integración con otros productos, es una mejor opción Network Analyst ya que no necesita de ningún otro software para su funcionamiento, y Route Server necesitaría de un servidor de mapas como es ArcIMS. Por tanto, sólo se podría utilizar el cálculo de rutas si se dispone de este servidor de mapas.

Por todo lo comentado anteriormente, se ha decidido seleccionar NetWork Analyst como la tecnología para implementar el cálculo de rutas para el presente proyecto fin de carrera.

Capítulo 3. ARCIMS y NETWORK ANALYST

"Mil rutas se apartan del fin elegido, pero hay una que llega a él"

Michel deMontaigne (1533-1592).
Pensador y escritor francés

En el actual capítulo se amplían las características expuestas en el Capítulo 2 sobre ArcIMS y Network Analyst. En primer lugar, se van a estudiar con detalle los componentes, arquitectura, funcionalidad y lenguaje de comunicación que utiliza el servidor de mapas ArcIMS y a continuación, se hace un estudio detallado de los algoritmos que utiliza Network Analyst para su funcionamiento, así como la importancia de los parámetros necesarios para el cálculo de rutas.

3.1 ARCIMS

Entre algunas de las razones que se tomaron en cuenta para escoger este sistema para desarrollar el proyecto se encuentra:

- Instalación y mantenimiento fácil desde *wizards*.
- Arquitectura fácilmente escalable.
- Capacidades de despliegue de mapas de alta calidad.
- Usuarios web con capacidades de geoprocesamiento.

La arquitectura de estos sistemas ha sido específicamente diseñada para brindar capacidades de análisis geográficos a Internet, y de esta forma soportar los requerimientos de múltiples usuarios accediendo a grandes bases de datos.

Por lo que es necesario establecer una plataforma común, para el intercambio de información GIS y de servicios, haciendo uso de un sistema de Geoprocesamiento distribuido en Internet. Esta tecnología de publicación de Mapas, soporta una gran variedad de clientes.

Desde el punto de vista del Servidor, se debe proveer una plataforma para integrar las bases de datos geográficas con las bases descriptivas y desarrollando específicamente para distribuir los Servicios GIS en Internet. También, se debe diseñar la creación de los servicios de geoprocesamiento, diseñar páginas Web para los clientes y administrar los recursos realizando balanceo de cargas, de manera que se pueda operar en una ambiente distribuido compuesto por recursos informáticos repartidos entre los clientes y los servidores que permiten un máximo aprovechamiento de los recursos.

Típicamente un cliente envía un requerimiento al servidor, este accede a las bases de datos, genera el análisis/informe y devuelve la información al cliente en forma de mapas, datos tabulares y gráficos de fácil comprensión. Esta tecnología es parte de una arquitectura múltiple que consiste en la unión sinérgica de clientes, servicios y datos. La gran variedad de clientes que se deben soportar (HTML, Java, JavaScript, etc.) hacen de ArcIMS una solución adecuada para los diferentes entornos de hardware, software y comunicaciones.

A lo largo de esta sección se describe ArcIMS, así como los componentes necesarios para su funcionamiento, la arquitectura que maneja, tipo de acceso al servidor, lenguaje de comunicación, etc.

3.1.1 Componentes

ArcIMS es el servidor de aplicaciones integrado dentro de la **arquitectura ArcGIS** que ha sido diseñado para la distribución y difusión de información geográfica, mapas y servicios GIS en entornos Internet / Intranet, en la figura 3.1 se muestra la posición de ArcIMS en la arquitectura de ArcGIS [WAGIS].

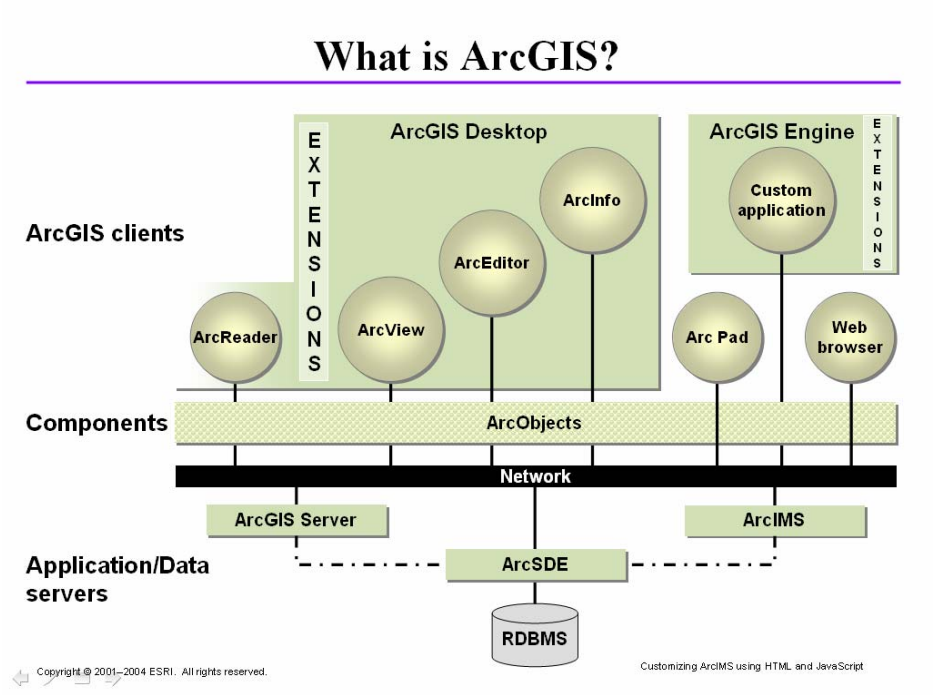


Figura 3.1. ArcGIS

Tanto si se opera en un entorno limitado, como en la intranet de una organización, como si se hace a través del entorno universal de Internet, es posible el empleo de ArcIMS para distribución de datos y funcionalidad GIS a múltiples usuarios.

ArcIMS es un producto del Internet que trabaja en un ambiente de Java™. Para que funcione correctamente, se debe de tener una serie de componentes necesarios, aunque no son parte de ArcIMS.

Estos componentes son un servidor web, JavaVM, y un motor del servlet (ver figura 3.2). Estos componentes, junto con ArcIMS, proporcionan la fundación para un sitio de trabajo de ArcIMS [ESRI2004].

Figura 3.2. Componentes necesarios para ArcIMS

- **Servidor web:** maneja las peticiones del cliente usando el protocolo HTTP. El servidor web transmite la petición de forma adecuada y envía una respuesta de nuevo al cliente. El servidor web no se incluye con ArcIMS.

- **JavaVM:** muchos de los componentes de ArcIMS son componentes Java y requieren de la existencia de la Maquina Virtual de Java, que proporciona la interfaz de programación básica (API) para la ejecución de esas aplicaciones. La Maquina Virtual de Java está incluida en Java Runtime Environment (JRE) o en Java Developer Kit (JDK). ArcIMS puede utilizar cualquier JavaVM existente en la máquina, siempre que sea compatible.

- **Servlet:** ArcIMS requiere un motor de servlet. Un motor de servlet es una extensión al JavaVM y proporciona ayuda para los servlets con un API de servlet.

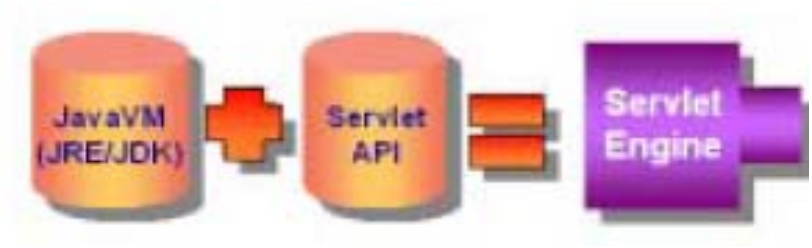


Figura 3.3. JavaVM+Servlet API

3.1.2 Arquitectura

ArcIMS se enmarca, dentro de una arquitectura multicapa, en la que se integran: los datos, el servidor de aplicaciones, el servidor WEB y los clientes [ESRI2004].

- **Clientes**: En el nivel superior de la arquitectura se encuentra la gran variedad de clientes soportados por ArcIMS que incluye herramientas profesionales como **ArcView**, **ArcEditor** y **ArcInfo**, visualizadores gratuitos como **ArcExplorer**, **ArcReader** y ArcGIS **Explorer**, o clientes que se ejecutan en navegadores estándar, así como desarrollos hechos a medida y dispositivos inalámbricos (por ejemplo, agendas electrónicas). Esta gran variedad permite elegir en cada momento la herramienta adecuada para satisfacer unas necesidades concretas.

- **Servicios**: En la siguiente capa de la arquitectura, conocida como capa de lógica de negocio, se encuentran los componentes encargados de recibir las peticiones del cliente (Servidor Web), traducirlas al lenguaje en el que se comunica ArcIMS (Conectores) y encaminarlas mediante el servidor de aplicaciones hacia los componentes encargados de resolverlas (Servidores Espaciales).

- **Gestión de Datos**: En el último nivel de la arquitectura se encuentra la información a explotar, almacenada en sistemas de archivos y/o bases de datos. ArcIMS soporta una gran variedad de formatos espaciales, incluidos shapefile, coberturas, Geodatabase, CAD, múltiples formatos raster y, en general, cualquier formato soportado por **ArcGIS Desktop**.

La arquitectura de ArcIMS se ha diseñado específicamente para servir datos SIG y dar nuevos servicios en Internet. Permite crear servicios de mapas, diseñar sitios de red, así como realizar tareas de administración de proyectos, dentro del mismo entorno de explotación. Además implementa una arquitectura multinivel propia de los entornos distribuidos, en la figura 3.4, se muestra la arquitectura de ArcIMS [ESRI-ES – 2002].

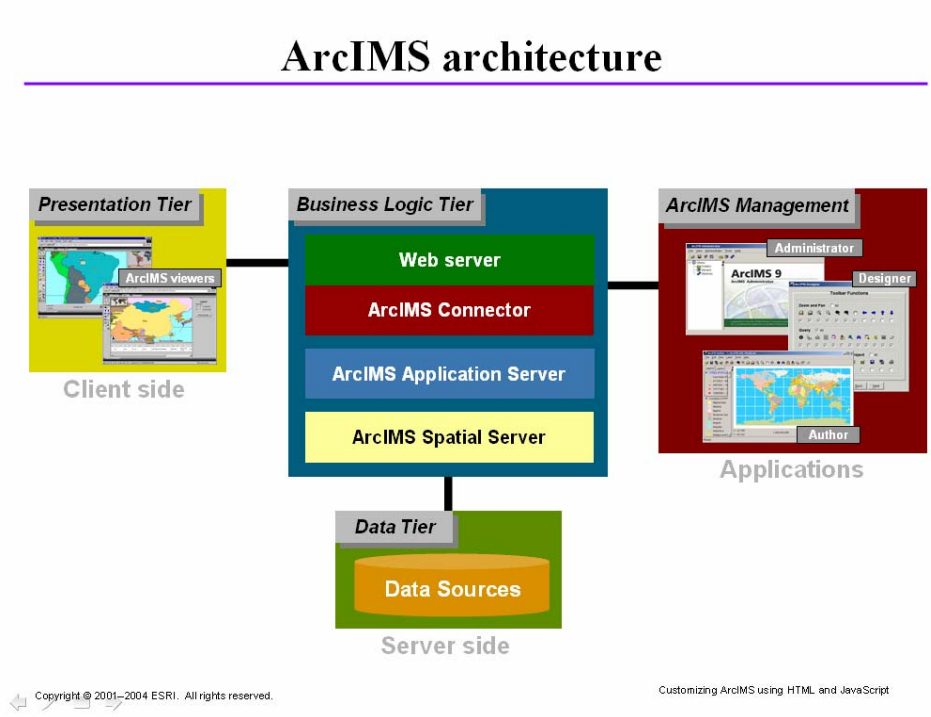


Figura 3.4. Arquitectura ArcIMS.

3.1.2.1 Componentes en la capa de lógica de negocio

La capa de lógica de negocio incluye los componentes necesarios para ejecutar los servicios y procesar peticiones y respuestas. Los componentes incluyen: el Conector del Servidor de Aplicaciones, el Servidor de Aplicaciones y los Servidores Espaciales. Estos componentes se muestran en la figura 3.5.

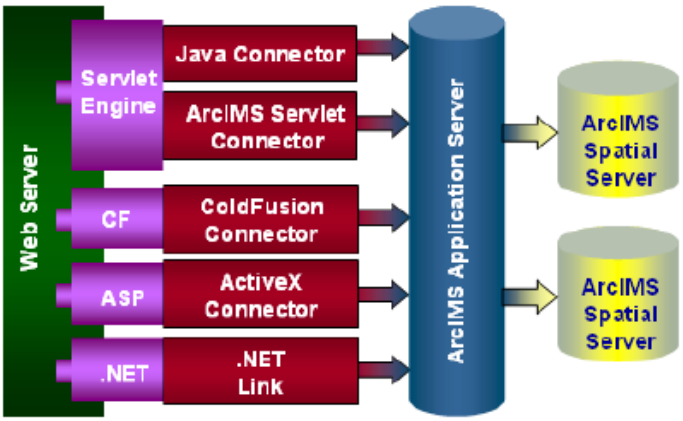


Figura 3.5.Componentes de la capa de negocio

3.1.2.2 Servidores Espaciales

Crea imágenes digitales de datos vector y raster. Da acceso a elementos geográficos y procesa consultas en la base de datos.

Son los componentes encargados de resolver las peticiones de los clientes accediendo a los datos necesarios. Dependiendo del tipo de petición que se realice, entrarán en funcionamiento uno o varios de los servidores disponibles, ver figura 3.6.

Figura 3.6. Servidores espaciales

A continuación, se detalla cada uno de los servidores espaciales:

- **Image Server:** Accede a la información geográfica y genera imágenes como resultado de las peticiones realizadas por el cliente.

- **Feature Server:** Genera respuestas a las peticiones del cliente en formato vectorial. El cliente deberá ser capaz de manejar datos vectoriales.

- **Query Server:** En combinación con el servidor Image Server permite resolver las consultas espaciales y alfanuméricas que realice el cliente contra un servicio de imágenes.

- **Geocode Server:** Permite resolver consultas en las que se necesita la geocodificación de localizaciones en función de direcciones postales.

- **Extract Server:** Permite al cliente descargarse en local la información vectorial visualizada en un momento dado.

- **Metadata Server:** Ofrece las herramientas necesarias para la creación y la gestión de catálogos de metadatos centralizados.

- **ArcMap Server:** Es un servidor de imágenes que permite publicar cualquier mapa generado con **ArcMap** (MXDs), o publicado con la extensión **ArcGIS Publisher** (PMFs). De este modo es posible aprovechar todo el potencial de los clientes de **ArcGIS Desktop** para la generación de mapas de alta calidad y su publicación posterior en Internet.

- **Route Server:** Se trata de una extensión opcional de ArcIMS que permite crear servicios de cálculo de rutas con múltiples paradas, así como la geocodificación de localizaciones. Este Servidor Espacial fue visto en el Capítulo 2 como alternativa para el cálculo de rutas.

La comunicación entre los diferentes componentes de ArcIMS se realiza a través del protocolo ArcXML. Es un derivado de XML (HTML extendido) y se diferencia de éste en tanto ArcXML no describe la estructura o apariencia de una página Web, sino la estructura del siguiente contenido y funcionalidad:

- **Configuración de servicios de mapas**: Definen el contenido y el diseño de mapas a diseminar, incluyendo las capas de información espacial y su simbología.

- **Consultas**: Emplean un filtro a servicios de mapas existentes y especifican la parte del mapa y datos relacionados que va a ser procesado.

- **Respuestas**: Mandan la información requerida al cliente. ArcXML está basado en tecnología XML, el nuevo estándar en intercambio y manejo de información, clave para bases de datos espaciales distribuidas por Internet.

3.1.2.3 Servidor de Aplicaciones

Maneja el balance de los procesos y demandas entrantes, y mantiene un registro de los servicios de mapas ejecutados en los servidores espaciales.

El servidor de aplicaciones cuenta con unos conectores que generan peticiones ArcXML antes de enviar peticiones al servidor de aplicaciones ArcIMS, como se puede ver en la figura 3.7:

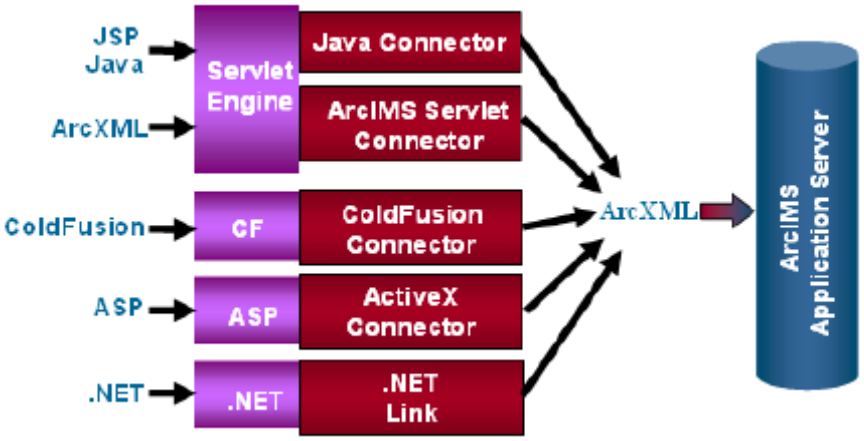


Figura 3.7. Conectores al servidor de aplicaciones

3.1.2.4 Conectores del Servidor de Aplicaciones

Conecta el Servidor Web al Servidor de Aplicaciones. ArcIMS tiene varios tipos de conectores:

- El *conector ArcIMS Servelt*, es el conector por defecto de ArcIMS. Este conector utiliza el motor de servlet para proporcionar un enlace de comunicación entre el Servidor Web y el Servidor de Aplicaciones ArcIMS. Utiliza ArcXML para comunicar el servidor Web con el Servidor de Aplicaciones.

  El formato ArcXML se ha diseñado como un protocolo para el intercambio de información entre los diferentes componentes de los productos ArcIMS. Una descripción más detallada de este lenguaje se puede ver en la sección 3.1.7.

- El *conector ArcIMS ColdFusion*, trabaja con clientes propios y traduce su lenguaje interno a ArcXML es decir, procesa las peticiones desde el Servidor ColdFusion antes de enviar la petición al Servidor de Aplicaciones ArcIMS.

- El *conector ArcIMS ActiveX*, es un componente DLL del modelo de Objetos (COM), que se puede utilizar en aplicaciones ASP. Este conector solo está disponible en Windows.

- El *enlace .NET ArcIMS*, es un conector para facilitar el desarrollo de aplicaciones ArcIMS para la plataforma .NET. Consiste de clases y funciones utilizadas para construir una Aplicación Servidor ArcIMS a través de conexiones HTTP o TCP. Solo está disponible en Windows.

- El *conector Java ArcIMS*, es un sistema JavaBeans que permite que los usuarios creen aplicaciones cliente, servidor, servlets y aplicaciones JSP. Se incluye una librería para dar soporte a aplicaciones JSP. Este conector está disponible en todas las plataformas.

3.1.2.5 Componentes de la capa Administrador

Conjunto de asistentes de fácil uso para el manejo de todas las funciones y tareas relacionadas con el servidor. Existen asistentes para crear y manejar servicios de mapas, diseñar los mapas a publicar, crear los sitios Web que proveen el acceso al usuario y administrar los servidores espaciales, la finalidad de estos componentes se detalló en la sección 2.1.7.3 [ESRI1].

En la figura 3.8 se puede ver el proceso que tiene lugar con la herramienta ArcIMS Author. A partir de unos datos y a través de la herramienta se genera el fichero de configuración del mapa.

Figura 3.8. Proceso con la herramienta ArcIMS Author

Una vez que ya se tiene el fichero de configuración del mapa creado, se utilizará la herramienta ArcIMS Administrator para crear el servicio a través del cual se accederá al mapa con uno de los Servidores Espaciales seleccionado. Este proceso se puede ver en la figura 3.9.

Figura 3.9. Proceso con la herramienta ArcIMS Administrator

Cuando se hacen peticiones ArcXML al servicio creado del mapa, la herramienta Administrator se comunicará en el conector utilizado y este conector se comunicará con el servidor de aplicaciones ArcIMS, como se puede ver en la figura 3.10:

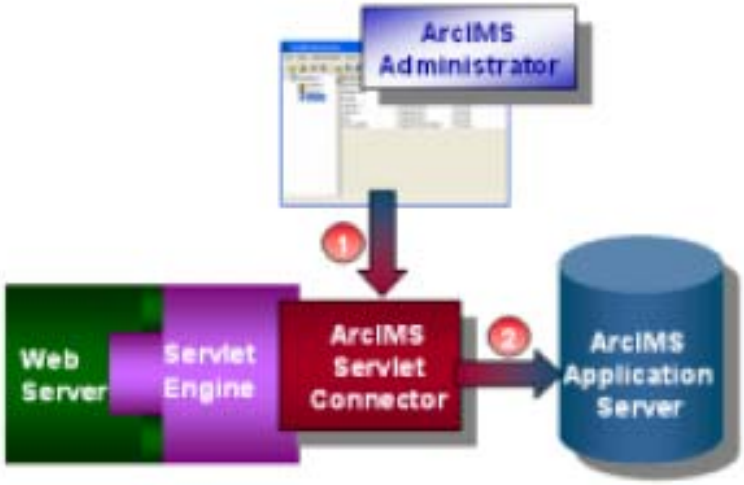


Figura 3.10. ArcIMS Administrator, Conector y Servidor de Aplicaciones

3.1.3 Funcionalidades

Las funcionalidades más importantes ofrecidas por los clientes de ArcIMS son:

- Navegación por el mapa cartográfico: Acercar, alejar, desplazar, etc.
- Consulta de datos espaciales y sus atributos.
- Creación de buffers alrededor de elementos.
- Posibilidad de añadir notas, gráficos o imágenes sobre el mapa.

- Posibilidad de enviar propuestas de edición sobre los datos espaciales y sus atributos.
- Guardar y recuperar proyectos.
- Activación y desactivación de capas, así como la interacción con la leyenda.
- Vista global del *website*.
- Medida de distancias sobre el mapa.
- Localización de direcciones.
- Impresión de salidas gráficas.

3.1.4 Tipos de Acceso al Servidor ArcIMS

ArcIMS maneja dos tipos de acceso:

- A través de componentes en el servidor, este tipo de acceso es el estudiado en los puntos anteriores (Servidores Espaciales, Servidor de Aplicaciones,…)
- A través de componentes del cliente, se detalla en la sección siguiente.

3.1.4.1 Componentes en el cliente

Ofrecen la posibilidad de elegir entre clientes ligeros, que sólo utilizan HTML, o clientes Java que permiten explotar al máximo todas las novedades tecnológicas de ArcIMS.

Los programadores también pueden construir aplicaciones cliente a medida, programando con lenguajes estándar como Visual Basic, Visual C++ y Java, y haciendo uso de los API que ofrecen los Conectores de ArcIMS (ActiveX, ColdFusion,…) [ESRI2004].

Las peticiones a un Servidor de Aplicaciones ArcIMS pueden ser enviadas desde tres tipos diferentes de clientes:

- HTML/DHTML: Clientes que envían peticiones directamente usando ArcXML.
- HTML/DHTML: Clientes que utilizan los conectores ActiveX o ColdFusion de ArcIMS. Son los clientes más ligeros puesto que todo el proceso se realiza en el lado del servidor.

- Java Viewers incluyendo ArcExplorer 3 (en su versión Java).

- El tipo de cliente utilizado, determinará la funcionalidad y el aspecto estético del sitio de red, siendo posible realizar todo tipo de modificaciones como insertar logos y gráficos, cambiar los colores o añadir nuevas funcionalidades.

3.1.5 La capa de presentación. Visores ArcIMS

La capa de presentación consiste en diferentes clientes, para acceder, manipular y gestionar datos geográficos. Un cliente básico incluye un mapa y algunos métodos para interactuar con el mapa.

Los clientes pueden ser generados utilizando el ArcIMS Designer, usando un conector del Servidor de Aplicaciones ArcIMS, así como aplicaciones independientes como ArcExplorer.

En general, los clientes hacen una petición a un servicio que se encuentra en un servidor ArcIMS. El servidor ArcIMS procesa la petición y envía los resultados. El proceso para generar las peticiones depende del cliente y del conector utilizado. A continuación, se muestra los diferentes tipos de clientes que puede haber:

- **Visor ArcIMS:** Los conocidos visores ArcIMS son los visores que vienen por defecto con ArcIMS, y que son generados utilizando la herramienta ArcIMS Designer. La comunicación con el servidor ArcIMS se realiza utilizando el conector servlet, estos visores se pueden modificar utilizando HTML y Javascript.

  Los visores ArcIMS generan peticiones y respuestas al lado del cliente. Cuando un usuario hace *click* en el mapa, se genera una petición ArcXML al lado del cliente y se envía al servidor ArcIMS, como se puede ver en la figura 3.11.

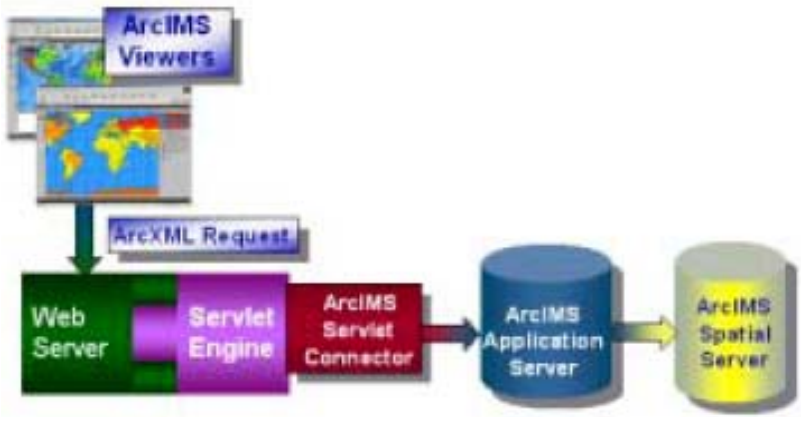


Figura 3.11. Visor ArcIMS

Cuando una petición es enviada, pasará por el servidor Web, el Servlet, el conector de Servlet ArcIMS y el Servidor de Aplicaciones ArcIMS. Entonces, el Servidor de Aplicaciones encuentra el servidor virtual donde el servicio del mapa reside y envía una petición a una instancia que se encuentra ejecutándose en este servidor virtual. La respuesta seguirá el mismo camino hacia el cliente.

- **Visor HTML**: este visor está escrito en HTML, DHTML y JavaScript. Un ejemplo de un visor HTML se muestra en la figura 3.12:

Figura 3.12. Visor HTML

En este visor solo un servicio de Imagen o un servicio ArcMap puede ser visualizado. Todas las peticiones son generadas utilizando JavaScript, y todas las respuestas son gestionadas utilizando JavaScript. Para manejar las peticiones y respuestas el cliente web debe ser Internet Explorer o Netscape [ESRI3].

- **Visor Java:** Los visores Java soportan servicios Imagen, ArcMap y servicios Feature. Se pueden combinar múltiples servicios, con datos locales y visores con el mismo visor Java. Los visores utilizan un Applet de Java para visualizar la información y procesar la petición.

  Las peticiones son enviadas al servidor para obtener los datos y el procesamiento de los datos reside en el servidor. La memoria temporal se elimina cuando el visor se cierra.

  ArcIMS tiene dos visores Java: Java personalizado y Java Standard. Ambos tienen la misma funcionalidad. Los visores personalizados de Java utilizan JavaScript para comunicarse con los applets. Pueden ser personalizados utilizando métodos del API del Modelo de Objetos del Visor. El visor

Personalizado de Java sólo soporta la versión de Internet Explorer 4.0 o posterior. Un ejemplo del visor personalizado de Java se muestra en la figura 3.13.

Figura 3.13. Visor Java

El visor Estándar de Java no utiliza JavaScript. Las herramientas y funciones son predefinidas y no pueden ser personalizadas utilizando el Modelo de Objetos. El visor Estándar de Java es soportado por Netscape y versiones de Internet Explorer 4.0 o posterior.

ArcIMS lleva consigo ArcExplorer 9, que es una aplicación para visores Java que no requiere de navegador. En el visor Estándar de Java, las herramientas y funciones son predefinidas y no pueden ser personalizadas. ArcExplorer no es software libre, y no se puede obtener de la página de ESRI [WESRI]

- **Visores utilizando el conector ActiveX:** el conector ActiveX es un COM DLL que puede ser usado en una aplicación COM así como en páginas Servidor de Microsoft Active. Los visores que utilizan este conector están basados en HTML. Estos visores no generan las peticiones ArcXML al lado del cliente como lo hace los visores HTML, si no que permite el procesamiento al lado del servidor. El proceso para realizar una petición se muestra en la figura 3.14.

Figura 3.14. Conector ActiveX

El cliente hace llamadas al método al API Conector del Modelo de Objetos. El conector ActiveX recibe esta información y transforma la petición a ArcXML. Una vez que la petición es transformada a ArcXML, el Servidor de Aplicaciones ArcIMS y el Servidor Espacial procesa la petición de la misma forma como si fuera un Conector Servlet de ArcIMS.

Las respuestas utilizan el mismo camino que las peticiones. La respuesta ArcXML es manejada por el Conector ActiveX, y una pagina HTML se genera utilizando ASP. La ventaja en este tipo de visor es que el visor al lado del cliente no necesita generar una petición o procesar la respuesta, esto hace que se trate de un cliente ligero.

El usuario final, utiliza un visor Conector ActiveX para identificar y tener funcionalidad similar al Visor ArcIMS HTML, pero la forma de manejar las peticiones y respuestas son mucho más diferentes. Este conector puede ser utilizado para construir aplicaciones cliente para Intranet utilizando lenguajes basados en COM así como Visual Basic y C++.

- **Visores utilizando conector ColdFusion:** un visor que utiliza Conector ColdFusion esta basado en HTML, pero todo el procesamiento se realiza al lado del servidor.

  El proceso para hacer las peticiones se muestra en la figura 3.15. En esta implementación, el cliente lanza eventos para ejecutar etiquetas ColdFusion para el servidor de aplicaciones ColdFusion. Las etiquetas personalizadas son manejadas por el Conector ColdFusion ArcIMS, que transforma las peticiones a ArcXML. Cuando una petición es transformada a ArcXML, el servidor de Aplicaciones ArcIMS y el Servidor Espacial manejan la petición de la misma forma que el Conector de Servlet ArcIMS.

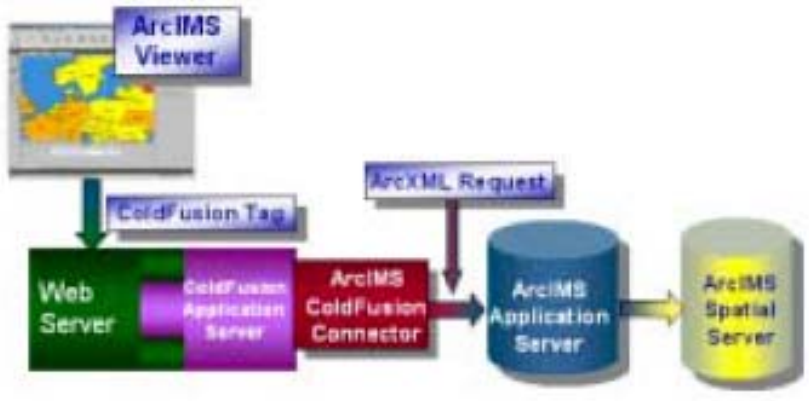


Figura 3.15. Conector ColdFusion

- **Visores utilizando el conector de Java:** El Conector de Java ArcIMS permite al usuario implementar una solución personalizada utilizando páginas JavaServer o una personalización de servlet. El conector consiste de un grupo de JavaBeans y librería de etiquetas JSP. La librería de etiquetas es un grupo de etiquetas y atributos que en muchos casos es similar a ArcXML. La librería de etiquetas proporciona una interfaz a los JavaBeans sin necesidad de que el programador necesite utilizar JavaScript.

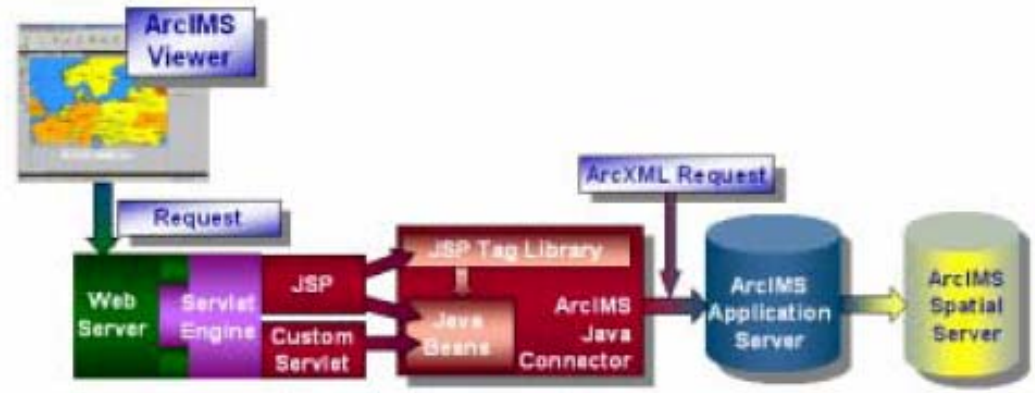


Figura 3.16. Conector Java

- **Visores utilizando el enlace .NET:** el enlace .NET facilita el desarrollo de aplicaciones ArcIMS en plataformas .NET. Consiste de clases y funciones que pueden ser utilizadas para construir conexiones a un Servidor de Aplicaciones ArcIMS con conexiones HTTP o TCP. Una vez que se establece la conexión con el servidor de aplicaciones ArcIMS, se puede enviar peticiones ArcXML y recibir respuestas ArcXML. Mientras que otros conectores tienen un API para crear peticiones ArcXML y procesar las respuestas, el enlace .NET requiere que se construya mecanismos para crear y procesar ArcXML en la aplicación [BT2005].

3.1.6 Lenguaje de comunicación ArcXML

3.1.6.1 Introducción

En estos últimos años, y sobre todo en estos últimos meses se ha estado hablando del (Lenguaje Extensible de "Etiquetado"), e**X**tensible **M**arkup **L**anguage, o **XML** [W3C2006].

Antes de nada conviene repasar su historia y precedentes. La versión 1.0 del lenguaje XML es una recomendación del W3C desde Febrero de 1998, pero se ha trabajado en ella desde un par de años antes. Está basado en el anterior estándar **SGML** (**S**tandard **G**eneralized **M**arkup **L**anguage, ISO 8879), que data de 1986, pero que empezó a gestarse desde principios de los años 70, y a su vez basado en el **GML** creado por IBM en 1969. Esto significa que aunque XML pueda parecer moderno, sus conceptos están más que asentados y aceptados de forma amplia.

Está además asociado a la recomendación del W3C **DOM** (**D**ocument **O**bject **M**odel), aprobado también en 1998. Éste no es más que un modelo de objetos (en forma de API) que permite acceder a las diferentes partes que pueden componer un documento XML o HTML.

SGML proporciona un modo consistente y preciso de aplicar etiquetas para describir las partes que componen un documento, permitiendo además el intercambio de documentos entre diferentes plataformas.

Así que, manteniendo su misma filosofía, de él se derivó XML como subconjunto simplificado, eliminando las partes más engorrosas y menos útiles. Como su padre, y este es un aspecto importante sobre el que se incidirá después, XML es un metalenguaje: es un lenguaje para definir lenguajes. Los elementos que lo componen pueden dar información sobre lo que contienen, no necesariamente sobre su estructura física o presentación, como ocurre en HTML.

En el proyecto se utiliza una extensión de **XML**, creado para el contexto de las herramientas ESRI. Esta extensión se denomina **ArcXML**.

El formato **ArcXML** se ha diseñado como un protocolo para el intercambio de información entre los diferentes componentes del producto ArcIMS [ESRI2].

Un ejemplo de ArcXML se muestra en la tabla 3.1, donde se puede ver un ejemplo de una petición (**REQUEST**) y una respuesta (**RESPONSE**). Este tipo de peticiones será las que intercambian los diferentes componentes de ArcIMS para comunicarse. Cuando el cliente hace una petición (un acercamiento o un movimiento en el mapa), esta petición se envía en formato ArcXML usando el elemento REQUEST.

Simultáneamente, la respuesta es de un servidor espacial de ArcIMS, el cual responde, usando el elemento RESPONSE [WAIMS].

REQUEST	RESPONSE
<?xml version="1.0"?> <ARCXML version="1.1"> **<REQUEST>** <GET_IMAGE> <PROPERTIES> <ENVELOPE minx="1246016,40629494" miny="115698,292874632" maxx="1281602,1778626" maxy="140717,110439662" /> <IMAGESIZE width="458" height="322" /> <LAYERLIST> <LAYERDEF id="0" visible="true" /> <LAYERDEF id="1" visible="true" /> <LAYERDEF id="2" visible="true" /> </LAYERLIST> </PROPERTIES> </GET_IMAGE> **</REQUEST>** </ARCXML>	<?xml version="1.0" encoding="ISO-8859-1"?> <ARCXML version="1.1"> **<RESPONSE>** <IMAGE> <ENVELOPE minx="1246016,40629494" miny="115698,292874629" maxx="1281602,1778626" maxy="140717,110439665" /> <OUTPUT url="http://portatil/output/edificio_PORTATIL120022683.jpg" /> </IMAGE> **</RESPONSE>** </ARCXML>

Tabla 3.1. Request y Response

El archivo de configuración del mapa del sitio web, también está formado con el lenguaje ArcXML, este archivo de configuración de mapa consta de dos partes: la primera sección es de elementos de ArcXML y la segunda corresponde a los elementos de las capas (LAYER), como se muestra en el listado 3.1.

```
<?xml version="1.0" encoding="UTF-8"?>
<ARCXML version="1.1">
<CONFIG>
<ENVIRONMENT>
<LOCALE country="MX" language="es" variant="" />
<UIFONT color="0,0,0" name="Arial" size="12" style="regular" />
</ENVIRONMENT>
<MAP>
<PROPERTIES>
<ENVELOPE minx="56.69" miny="20.5" maxx="57.13" maxy="20.5" name="Initial_Extent" />
<MAPUNITS units="decimal_degrees" />
</PROPERTIES>
<WORKSPACES>
<SHAPEWORKSPACE name="shp_ws-2" directory="C:\capas" />
</WORKSPACES>
<LAYER . . .
</LAYER>
<LAYER . . .
</LAYER>
</MAP>
</CONFIG>
```

```
</ARCXML>
<?xml version="1.0" encoding="UTF-8"?>

<ARCXML version="1.1">
 <CONFIG>
  <ENVIRONMENT>
   <LOCALE country="ES" language="es" variant="" />
   <UIFONT color="0,0,0" name="SansSerif" size="12" style="regular" />
   <SCREEN dpi="96" />
  </ENVIRONMENT>
  <MAP>
   <PROPERTIES>
        <ENVELOPE minx="1247130,080368759" miny="118701,24905529567"
maxx="1280488,5037887928" maxy="137714,15425899986" name="Initial_Extent" />
        <MAPUNITS units="meters" />
    </PROPERTIES>
    <WORKSPACES>
        <SHAPEWORKSPACE name="shp_ws-12" directory="C:\capas" />
   </WORKSPACES>
    <LAYER type="featureclass" name="Salas" visible="true" id="0">
        <DATASET name="Salas_OK" type="polygon" workspace="shp_ws-12" />
        <SIMPLERENDERER>
          <SIMPLEPOLYGONSYMBOL boundarytransparency="1,0" filltransparency="1,0"
filltype="gray" fillcolor="153,153,255" boundarycaptype="round" />
        </SIMPLERENDERER>
    </LAYER>
    ……
    <LAYER type="featureclass" name="Cuadros" visible="true" id="2">
        <DATASET name="Cuadros" type="point" workspace="shp_ws-12" />
        <SIMPLERENDERER>
          <SIMPLEMARKERSYMBOL color="0,51,153" type="square" width="10" />
        </SIMPLERENDERER>
    </LAYER>
  </MAP>
 </CONFIG>
</ARCXML>
```

Listado 3.1. Archivo ArcXML (edificio.axl)

Este tipo de archivo de mapa de configuración muestra una jerarquía de ArcXML, todo código de ArcXML comienza con el elemento **ARCXML**, los sub.-elementos pueden ser:

- **CONFIG:** Es el elemento principal para definir el archivo de configuración, y su ubicación.

- **REQUEST:** Define la petición que será enviada al Servidor Espacial ArcIMS para procesarla.

- **RESPONSE:** Contiene resultados de una petición de Servidor Espacial ArcIMS como se muestra en la tabla 3.1.

El elemento **CONFIG**, indica que es un archivo de mapa de configuración, en este elemento se define: las propiedades del mapa y las capas de datos. Además se incluye los siguientes elementos:

- **ENVIRONMENT**: Este elemento define el lenguaje a usar por este visualizador, como se muestra en el listado 3.1.

- **MAP**: indica la información del mapa utilizado, además contiene los siguientes elementos:

    - **PROPERTIES:** Este elemento describe las propiedades generales del mapa. Es posible definir la extensión de un mapa usando el elemento ENVELOPE, este elemento se muestra en el listado 3.1:

        - **ENVELOPE:** Que define la unidad del mapa, proyección, leyendas, imagen de salida, archivo de salida y el color de fondo [WAIMS]

    - **WORKSPACES:** Este elemento lista los tipos y localizaciones de las capas de datos del mapa, dentro de este elemento hay un gran número de sub-elementos, algunos de los elementos que usa son los siguientes, y se pueden ver en listado 3.2:

        - **SHAPEWORKSPACE:** Define la ubicación del archivo shapefile.

        - **SDEWORKSPACE:** Define la ubicación de los archivos en la base datos geográfica ArcSDE.

        - **IMAGEWORKSPACE:** Especifica un espacio de trabajo para los archivos de la imagen.

        - **AVIMSWORKSPACE:** Especifica un espacio de trabajo para un mapa servido en el IMS de ArcView.

        - **FEATURESERVERWORKSPACE:** Especifica un espacio de trabajo para un servicio de característica de ArcIMS.

        - **IMAGESERVERWORKSPACE:** Especifica un espacio de trabajo para un servicio de imagen de ArcIMS.

    - **LAYER:** Proporciona un marco de trabajo, para dar las propiedades de como se dibuja cada capa en el mapa, como se puede ver en el listado 3.2.

```
<ARCXML . . .
<LAYER type=”featureclass” name=”limite” visible=”true” id=”0”>
<DATASET name=”limite” type=”polygon” workspace=”shp_ws-0” />
</LAYER>
<LAYER type=”featureclass” name=”caminos” visible=”true” id=”2”>
<DATASET name=”caminos” type=”line” workspace=”shp_ws-0” />
<GROUPRENDERER>
<SIMPLELABELRENDERER field=”CARRETERA” linelabelposition=”placeabove”>
<TEXTSYMBOL antialiasing=”true” font=”Arial” fontstyle=”regular” fontsize=”10” />
</SIMPLELABELRENDERER>
<SIMPLERENDERER>
<SIMPLEMARKERSYMBOL color=”128,128,128”, width=”8” type=”circle” />
</SIMPLERENDERER>
</GROUPRENDERER>
</LAYER>
. . .</ARCXML>
```

Listado 3.2. Elementos de capa (LAYER)

El elemento **DATASET** define el nombre actual del archivo shapefile, imagen de las capas de ArcSDE y el espacio de trabajo de los datos almacenados, como se muestra en el listado 3.2.

El elemento **LABELRENDERER**, define como dibujar las etiquetas de cada elemento. Hay dos tipos:

- **SIMPLELABELRENDERER:** Proporciona la información de cada elemento, como se muestra en el listado 3.2.
- **VALUEMAPLABELRENDERER:** Dibuja las etiquetas basados en rangos o valores de atributos [WAIMS].

El elemento **RENDER** proporciona un marco de trabajo para la simbología de las capas, hay tres tipos:

- **SIMPLELABELRENDER:** Dibuja todos los elementos con la misma interpretación.
- **VALUEMARENDER:** Dibuja elementos basados en rangos o valores de atributo.
- **SCALEDEPENDENTRENDER:** Define los factores de escala cuando se dibuja la capa.

Un elemento **LABELRENDERER** o **RENDER** dentro del elemento **LAYER** puede ser agrupado usando un elemento llamado **GROUPRENDERER,** como se muestra en el listado 3.2.

El elemento **SYMBOL,** define el elemento que va a ser definido, hay muchos elementos **SYMBOL:**

- **SIMPLEMARKERSYMBOL:** Es la simbología que usa ArcIMS para representar un punto, usando uno de los tipos predefinidos del símbolo tenemos: círculo, triángulo, cuadrado, cruz, o estrella.

- **SIMPLELINESYMBOL:** Símbolo para las características de la línea.

- **SIMPLEPOLYGONSYMBOL:** Símbolo para las características del polígono.

Además el elemento **SYMBOL** usa siempre sub-elementos de un elemento **RENDERER**, también los elementos **SYMBOL** usan atributos que define la simbología de los elementos e incluye atributos de color, fuente de texto, tamaño, ancho, modo de etiqueta, y modo de impresión. En el atributo color puedes usar el **R**ed, **G**reen, **B**lue (**RGB**) y para el atributo fuente de texto siempre tienen un caso sensitivo.

3.1.6.2 Sintaxis de ArcXML

Esta sintaxis se basa en el consorcio de XML [W3C2006]. Las siguientes características son una convención, que la empresa ESRI definió para el uso de los elementos de ArcXML como extensión de XML.

- Cuando abres un elemento se debe poner “<”, seguido por el nombre del elemento (todos los nombres de elementos deben estar es mayúsculas), después los atributos de ese elemento y se cierra con “>”.

- Para cerrar los elementos se sigue la misma sintaxis pero se pone un “/”, antes del nombre del elemento.

- Todos los sub-elementos son insertados antes de cerrar el elemento.

- Los nombre de atributos (sí es necesario), deben ser en mayúscula.

- Algunos atributos para algunos elementos deben de seguir un cierto orden para que funcione correctamente.

- Un atributo puede usar solo uno valor y siempre esta dentro de dos comillas (“).

- El valor de atributos no es un caso sensitivo.

- El nombre de los sub-elementos debe ser en mayúsculas.

- Algunos sub-elementos pueden usarse múltiples veces.

- Para hacer comentarios, se usa de la misma manera que en HTML. <!-- →.

Para más información, se puede consultar el manual de ArcXML Programmer´s Referente Guide [WAIMS].

Las etiquetas y atributos de Arc**XML** describen la estructura de:

- Archivos de configuración de servicios de mapas (MapService) (archivos AXL). Estos archivos describen, entre otras cosas, qué capas utilizar, qué simbología y que escala.

- Peticiones. (REQUEST) Es un filtro sobre el servicio de mapas que especifica qué parte de este servicio y qué datos asociados se quiere consultar.

- Respuestas. (RESPONSE) Constituyen la información solicitada por el cliente.

3.1.7 Formato de Fichero Shapefile

El formato **ESRI Shapefile** (SHP) es un formato propietario de datos espaciales desarrollado por la compañía ESRI, quien desarrolla y comercializa software para Sistemas de Información Geográfica como ArcInfo o ArcGIS. Originalmente se creó para la utilización con su producto ArcView GIS, pero actualmente se ha convertido en formato estándar *de facto* por la importancia que los productos ESRI tienen en el mercado SIG [ESRI5].

Es un formato vectorial. Este formato no es topológico, almacena localización geométrica e información de atributos de los elementos geográficos. La localización geométrica para cada elemento geográfico es almacenada como un conjunto de vectores de coordenadas.

Este formato Shapefile, es el estándar admitido por el OGC como estructura de datos para información geográfica vectorial, por lo que se utilizará este formato.

Los elementos geográficos son representados por un conjunto reducido de elementos geométricos (puntos, líneas y polígonos), también llamados primitivas gráficas o formas (Shape). Un Shapefile almacena la geometría y la información de elementos espaciales, en un paquete de datos.

Un archivo SHP (SHAPE), es un formato de vector creado por ESRI. Este archivo soporta geometrías punto, multi-punto, polígono, polilínea, multi-patches,…

Este formato SHP, contiene tres archivos:

- Archivo principal: *.shp
- Archivo de índice: *.shx
- Archivo de base de datos: *.dbf

El archivo principal es un acceso directo, en el que se describe la forma de cada elemento como una lista de sus vértices. En el archivo de índice, cada registro contiene la dirección de comienzo y fin de los registros del archivo principal. De esta forma, la tabla dBASE contiene atributos de los elementos espaciales (registros del archivo principal), con un registro por elemento. La relación exacta entre la geometría y atributos está basada en el número de registro índice. Así, los atributos de los registros geográficos en el archivo dBASE deben estar en el mismo orden en el que aparecen en el archivo principal.

Además de estos tres archivos requeridos, opcionalmente se pueden utilizar otros para mejorar el funcionamiento en las operaciones de consulta a la base de datos, información sobre la proyección cartográfica, o almacenamiento de metadatos. Estos archivos son:

- **.sbn** y **.sbx** – Almacena el índice espacial de las entidades.
- **.fbn** y **.fbx** – Almacena el índice espacial de las entidades para los shapefiles que son inalterables (solo lectura).
- **.ain** y **.aih** – Almacena el índice de atributo de los campos activos en una tabla o el tema de la tabla de atributos.
- **.prj** – Es el archivo que guarda la información referida a sistema de coordenadas.
- **.shp.xml** – Almacena los metadatos del shapefile.

A continuación, se va a describir la estructura general y la organización del Shapefile. El **fichero principal** (.shp) se organiza como se puede ver en la figura 3.17:

Cabecera del fichero	
Cabecera del registro	Contenido del Registro
Cabecera del registro	Contenido del Registro
….	….
Cabecera del registro	Contenido del Registro

Figura 3.17 Fichero principal Shapefile

La cabecera del fichero tiene una longitud fija de 100 bytes y posee los siguientes campos: código del fichero, longitud del fichero, versión, tipo de shape (forma) y el Bounding Box.

Value	Shape Type
0	Null Shape
1	Point
3	PolyLine
5	Polygon
8	MultiPoint
11	PointZ
13	PolyLineZ
15	PolygonZ
18	MultiPointZ
21	PointM
23	PolyLineM
25	PolygonM
28	MultiPointM
31	MultiPatch

Tabla 3.2. Valores Tipo Shape

El tipo de Shape, indica el tipo de formas o geometrías que va a contener el fichero, en un mismo fichero Shape todas las geometrías contenidas deben de ser del mismo tipo. En la tabla 3.2, se muestran los tipos de Shapes que pueden aparecer en un Shapefile. Los tipos no especificados (2, 4, 6,…) son reservados para uso futuro. Los tipos soportados para ArcIMS son puntos, líneas, polígonos, poli línea, etc.

A continuación de la cabecera del fichero se encuentran todos los registros. Dichos registros tienen dos partes. Primero aparece la cabecera del registro, que es de longitud fija 8 bytes, y es donde se indica el número de registro y la longitud del contenido, y después se indica el contenido del registro. Este consta del tipo de Shape, seguido de su información geométrica, la longitud depende el número de partes y vértices del Shape.

El **fichero de índices** tiene una estructura similar al fichero principal, dicha estructura se puede ver en la figura 3.18:

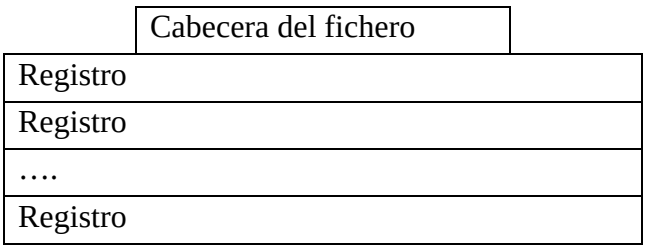


Figura 3.18. Fichero índices Shapefile

La cabecera del fichero de índices esta organizada igual que la del fichero principal, el campo longitud ahora contendrá la longitud total del fichero de índices en palabras de 16 bits. Cada uno de los registros de dicho fichero guarda el desplazamiento y la longitud del contenido para cada uno de los registros del fichero principal. El desplazamiento vendrá dado en palabras de 16 bits desde el principio del fichero principal hasta el primer byte de la cabecera del registro correspondiente. La longitud del contenido almacenada en el registro de índices es la misma que la almacenada en la cabecera del registro correspondiente, en el fichero principal.

El **fichero Dbase** (.dbf) es el encargado de guardar los atributos para cada geometría. Este fichero debe cumplir tres requisitos:

- Este fichero debe tener el mismo nombre que los dos anteriores con la extensión .dbf.
- La tabla debe contener un registro por cada Shape del fichero principal.
- El orden de los registros debe ser el mismo que el orden de las geometrías (Shapes) en el fichero principal.
- El valor correspondiente al año que aparece en la cabecera del dBASE debe ser desde 1900.

Para una información mas detallada del formato Shapefile y dBASE, véase la documentación técnica [ESRI5].

Este formato de ficheros se utilizará en el desarrollo del proyecto teniendo distintas capas de información del sistema indoor en este formato.

3.1.8 Conclusión

ArcIMS, es una herramienta de gran ayuda para los Sistemas de Información Geográficos, ya que permite compartir datos de diferentes fuentes y visualizar la información vía Internet.

ArcIMS soporta dos tipos de cliente: el cliente ligero HMTL que es el cliente que no tiene experiencia en los SIG y el cliente JAVA que es el cliente que analiza y explota las aplicaciones SIG.

ArcIMS, es una herramienta fácil de utilizar, ya que no se necesita ser un programador para explotar esta herramienta. Además cuenta con tres aplicaciones que se pueden examinar de manera independiente para la creación de un servicio vía Internet.

3.2 NETWORK ANALYST

El objetivo de esta sección es informar sobre como funciona el algoritmo de Network Analyst en ArcGIS y dar una visión general de sus características.

Se realiza una comparación de los dos algoritmos que utiliza NetWork Analyst. Se trata dar una visión del algoritmo cuando se plantea un problema entre un origen y un destino, o entre múltiples orígenes y múltiples destinos. Cómo se maneja los obstáculos dentro de la red, y la importancia de tener diferentes niveles jerárquicos.

Se detalla los parámetros necesarios para hace funcionar el algoritmo, dando una definición del mismo, y su valor que pueden tener. Y se finaliza dando soluciones a un problema planteado.

Network Analyst se basa en una serie de algoritmos jerárquicos muy útiles para encontrar una ruta entre un origen y un destino en grandes redes.

El principal objetivo de estos algoritmos es conseguir **exactitud** junto con **velocidad**. En algunos casos, como por ejemplo, en callejeros donde hay una jerarquía natural, la velocidad y la exactitud es importante.

Los algoritmos jerárquicos han contribuido al éxito de varios productos de cálculo de rutas en ESRI como por ejemplo: NetEngine, ArcLogistics Route, la extensión de StreetMap ArcGIS y ArcIMS Route Server [ESRI2005].

La extensión de ArcGIS, conocida como Network Analyst, incluye un algoritmo jerárquico que ha sido diseñado para soportar capacidades de modelado importantes de la red disponible en ArcGIS 9.1, y soportar datos StreetMap, que pueden ser usados para las rutas. Además, ha sido diseñado para ser el motor base de las 3 funciones conocidas como: *calculo de rutas, localización de las ubicaciones más próximas y la matriz de coste Origen-Destino.*

En ArcGIS se cuenta con dos posibilidades a la hora de analizar una red utilizando Network Analyst. Por un lado, se puede utilizar ArcMap, utilizando una herramienta de Geoprocesamiento, independiente de ArcGIS Engine, o por otro lado, a través de una aplicación Web desarrollada usando ArcGIS Server [WESRI].

ArcMap es una herramienta de ESRI que permite visualizar, explorar, consultar datos, así como capacidad para crear y editar datos geográficos y alfanuméricos [WESRI].

ArcGIS Server constituye la plataforma adecuada para compartir recursos GIS entre usuarios locales o a través de Internet.

El objetivo de esta sección es proporcionar la utilidad e información práctica, para entender como usar Network Analyst de modo eficiente. Se da un conocimiento básico de lo que se conoce como red dataset y del funcionamiento de Network Analyst ArcGIS.

Una red dataset es un fichero que contiene toda la lógica necesaria para realizar el cálculo de rutas. Por ejemplo, intersecciones que se pueden encontrar, caminos que se pueden tomar, etc.

A la hora de utilizar Network Analyst se puede distinguir dos algoritmos, por un lado, el algoritmo de la ruta más exacta y por otro lado, el algoritmo de la mejor ruta jerárquica.

En esta sección se realiza una comparación de estos dos algoritmos y se introduce conceptos de una solución jerárquica. Además se proporciona información sobre el modelado de las redes jerárquicas, con datos básicos y datos comerciales.

Por tanto, se podrá ver como el uso eficiente de opciones jerárquicas durante el análisis de red, ayuda a la realización y a la calidad de la solución.

3.2.1 Comparativa de lo jerárquico contra la exactitud

Encontrar la mejor ruta en Network Analyst, consiste en el cálculo de la ruta más rápida o la ruta más corta, dependiendo de la elección realizada. En esta sección, se realiza una comparación de los dos algoritmos implementados: el **algoritmo de la ruta más exacta**, y el **algoritmo de mejor ruta jerárquica**.

Entre las características de ambos algoritmos se encuentra que:

- Minimizan una impedancia (ejemplo: tiempo de viaje o longitud).
- Encuentran rutas de nodo a nodo, de nodo a arista, de arista a arista, y de arista a nodo.
- Manejan restricciones en aristas y nodos (ejemplo: una restricción de camino).
- Manejan obstáculos sobre elementos de la red (por ejemplo: en ciertas aristas o nodos).
- Prohibición de maniobras (Restricciones de giros).

- Añadir maniobras al retardo (Penalizar el retardo).

- Mencionar el acercamiento específico entre un origen y un destino (por ejemplo, En frente de la calle).

A continuación, se detallan las principales diferencias entre el algoritmo de la ruta más exacta, y el algoritmo de mejor ruta jerárquica:

- El algoritmo de la ruta más exacta, está basado en el algoritmo de Dijskstra[1], calcula la ruta óptima entre un origen y un destino, recorriendo la red completa.

  Por el contrario, el algoritmo de la mejor ruta jerárquica debe calcular una ruta subóptima, más larga mirando términos de impedancia, pero más realista.

  La opción jerárquica cree que los caminos primarios (por ejemplo: autovías) son más rápidos que los caminos secundarios (por ejemplo: carreteras secundarias o locales).

  Así por ejemplo, si un usuario crea una ruta que no favorece a la jerarquía, el coste total en la ruta exacta debe ser inferior que en la ruta jerárquica. Aunque no sea una representación exacta de la realidad. La ruta exacta no tiene en cuenta los retardos de espera ni las señales de parada (como por ejemplo: semáforos ni stops). Las rutas jerárquicas están a favor de las carreteras primarias, basadas en la suposición que los retardos en estas carreteras son más pequeños.

- El clásico algoritmo de la ruta más exacta, también conocido como el algoritmo del camino más corto, no soporta consultas a tiempo real en redes grandes. Su funcionamiento sofisticado, hereda técnicas heurísticas y algoritmos jerárquicos, y reduce gramáticamente el tiempo necesario para solucionar el problema.

- Las direcciones de viaje, instrucciones paso a paso para la navegación de la ruta, son más fáciles y más concisos en rutas generadas por el algoritmo de la mejor ruta jerárquica.

  Según lo indicado anterior, un algoritmo jerárquico está a favor de una carretera principal, que reduce el número de transiciones entre diferentes clases de caminos o nombres de carreteras. El algoritmo de la ruta más exacta genera muchas más instrucciones para realizar el viaje.

[1] Se obtiene el algoritmo de la ruta más corta entre en el origen y el destino.

- Una ruta jerárquica, puede ser creada cuando un cierto vehículo tiene preferencia sobre ciertas clases de carreteras. Por ejemplo, para calcular la ruta que tiene que seguir un camión de reparto, se puede crear una jerarquía para rutas primarias.

- El cálculo de todas las rutas entre múltiples orígenes y múltiples destinos es requerida en muchos problemas, como por ejemplo, para un viajante, y el reparto de un camión. Estos problemas requieren un cálculo computacional de **n** por **m** rutas de una manera eficiente. Un algoritmo exacto calculará todas las rutas óptimas pero debe consumir eficientemente recursos computacionalmente caros.

  Un algoritmo jerárquico sacrificará algo de exactitud por algo de velocidad.

3.2.2 Definición de algoritmo jerárquico

Para proporcionar un algoritmo jerárquico eficiente para los usuarios, se tienen que diseñar e implementar dos procedimientos:

- Un preprocesamiento, que clasifica a la red en niveles jerárquicos.

- Y un análisis, en este caso el algoritmo de mejor ruta jerárquica explora el preprocesamiento de la red, para computar la mejor ruta entre paradas.

En esta sección se discute el primer paso, el del preprocesamiento, seguido por una descripción del algoritmo jerárquico usado durante el análisis.

Por tanto, se presentan tres ventajas: una solución al viaje, una solución alternativa, y por último un cálculo de n por m rutas.

3.2.2.1 Etapa de preprocesamiento

La etapa de preprocesamiento clasifica las aristas de una red en niveles jerárquicos, reduciendo el espacio de búsqueda durante el procedimiento del cálculo de ruta.

La clasificación jerárquica es un agrupamiento espacial multinivel que se realiza cuando se construye la red dataset. Durante el proceso de construcción, el usuario selecciona un atributo jerárquico, que se utiliza para crear hasta tres niveles de jerarquía:

- Caminos principales, que están en el nivel más alto. Por ejemplo: autovías y carreteras de uso autorizado.

- Caminos secundarios, que están en el nivel intermedio. Por ejemplo: carreteras nacionales.

- Caminos locales, que están en el nivel más inferior. Por ejemplo: calles locales.

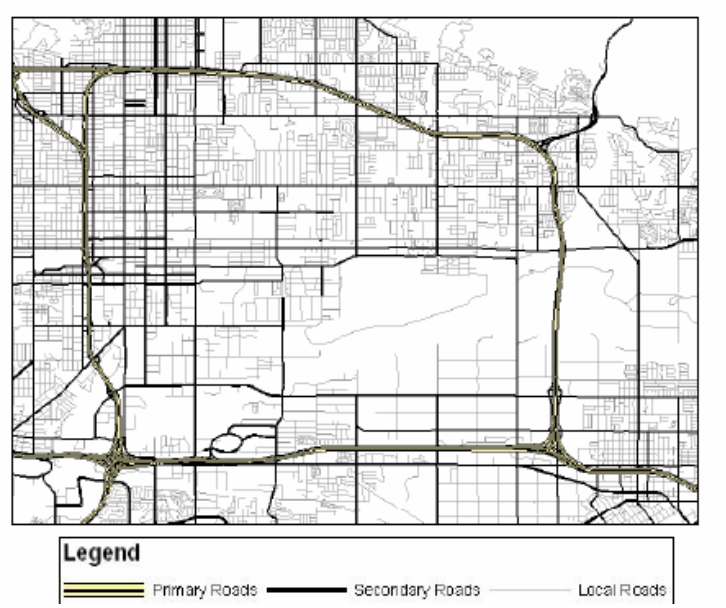


Figura 3.19 Ejemplo de red Jerárquica.

3.2.2.2 Etapa de análisis

Una vez que la red jerárquica está creada, se utiliza un algoritmo jerárquico bidireccional durante la etapa de análisis, para calcular la ruta entre un origen y un destino.

El objetivo final de esta técnica es minimizar la impedancia mientras que se utiliza el nivel más alto de la jerarquía. La técnica consiste en:

- Simultanear la búsqueda entre un origen y un destino hasta un número específico de punto de conexión, hasta el siguiente nivel más alto de la jerarquía encontrado.

- Explorar el nivel más alto de la jerarquía desde el origen, sin hacer caso del resto de los niveles.

- Parar la búsqueda, cuando ambos segmentos de la ruta (uno desde el origen y otro desde el destino) se encuentran.

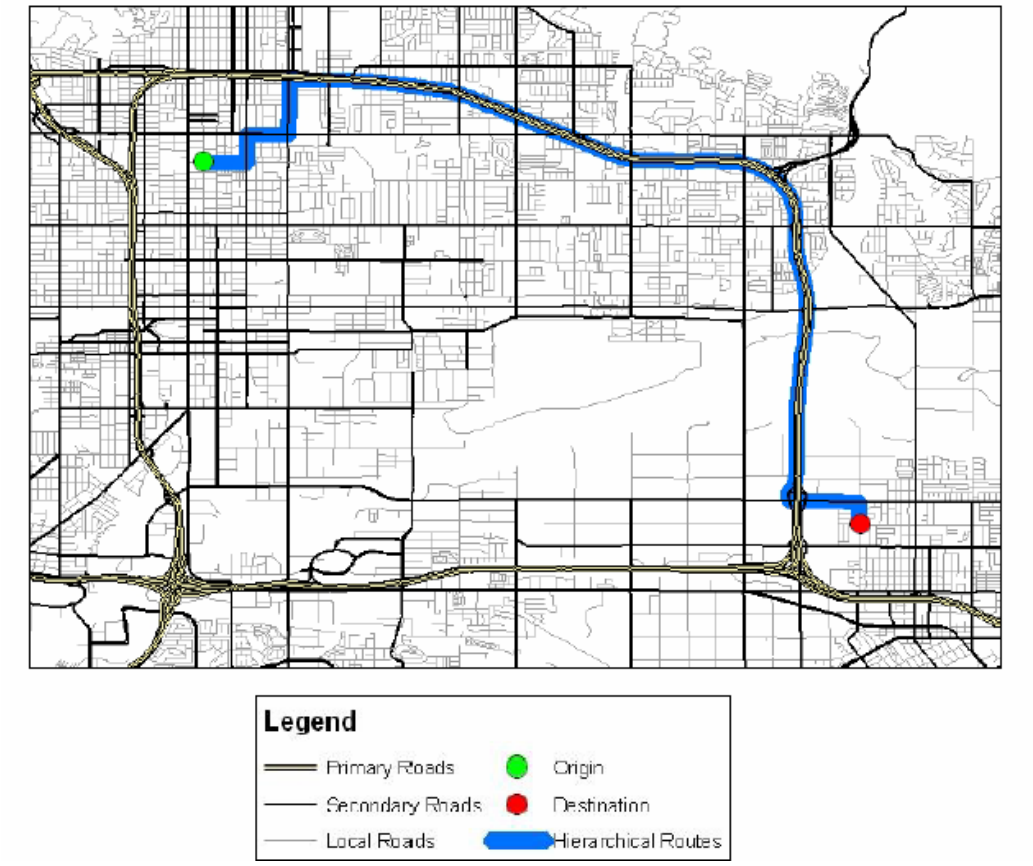


Figura 3.20. Ejemplo de ruta jerárquica

3.2.2.3 Consideraciones de giro

Cuando se construye el árbol descendente desde el origen y el árbol ascendente desde el destino, se ha de tener en cuenta ciertas restricciones. Los giros deben involucrar aristas pertenecientes al mismo o a diferentes niveles de la jerarquía.

Los giros son manejados para guardar información durante el cálculo de la solución, que puede determinar si los árboles ascendente y descendente se han encontrado de una forma válida.

Por ejemplo, cuando el árbol ascendente y descendente se encuentra en un cruce común, y la ruta que atraviese ese cruce no puede ser travesada debido a una restricción de giro, como se puede ver en la figura 3.21.

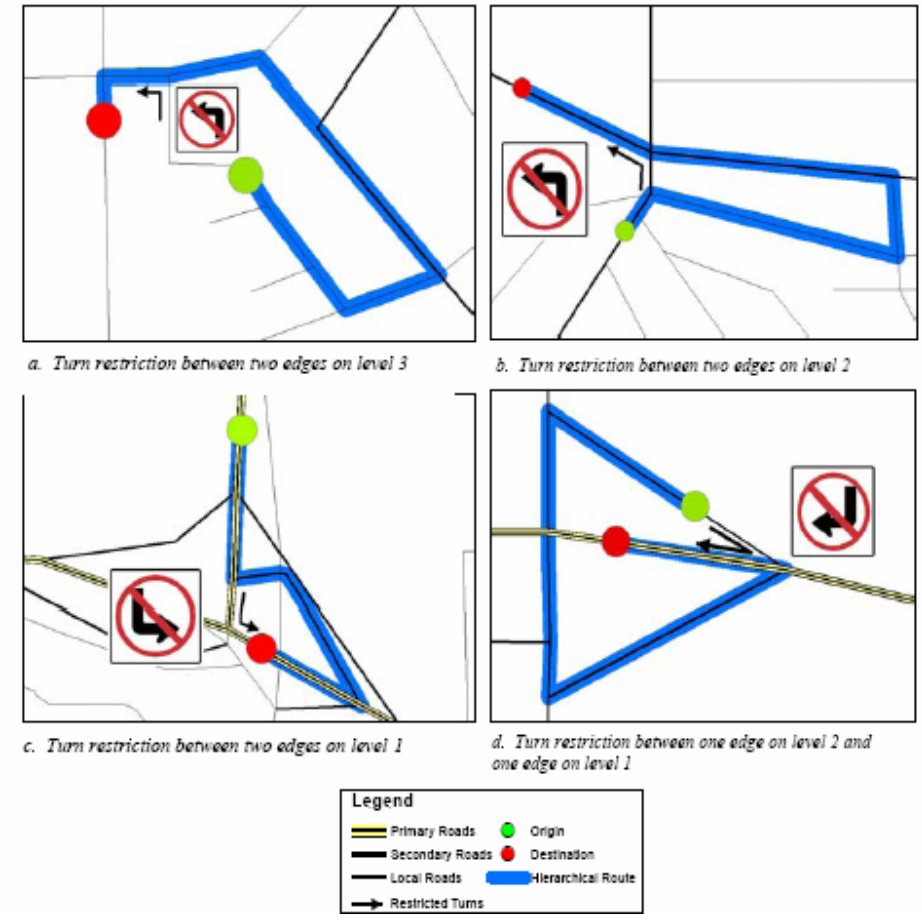


Figura 3.21. Ejemplo de rutas jerárquicas con consideraciones de giro

3.2.2.4 Consideraciones de los obstáculos

El algoritmo jerárquico puede manejar obstáculos localizados en ciertos elementos de red (en nodos y aristas).

Entre las propiedades de obstáculos se encuentra que una arista esté restringida por ambos lados, o sólo por la derecha o sólo por la izquierda.

Cuando un obstáculo se encuentra en una arista en el nivel más inferior de la jerarquía (nivel 3), y se encuentra durante la búsqueda, este elemento de red no se considerará en la búsqueda ascendente ni en la descendente.

Se ha implementado un procedimiento especial para manejar los obstáculos en los elementos de red del nivel 1 y 2. Para gestionar sus propiedades, si un obstáculo se encuentra se pasa al nivel de la jerarquía inferior, rodeando el obstáculo, y a continuación se volverá al nivel de jerarquía superior origen, como se puede ver en la figura 3.22:

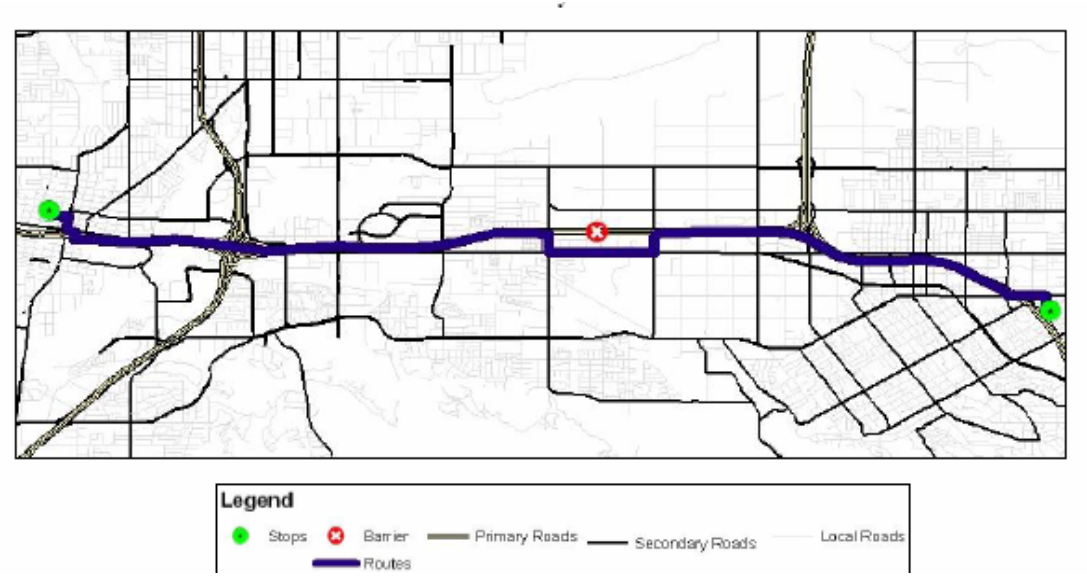


Figura 3.22. Ejemplo de ruta jerárquica con un obstáculo localizado

3.2.2.5 Las mejores rutas de n por m

La solución de una matriz de coste OD (Origen-Destino) y el problema de las localizaciones más próximas en Network Analyst, consiste en una solución de múltiples rutas computacionales desde n múltiples orígenes hasta m múltiples destinos.

Esto requiere encontrar los k destinos más próximos, o encontrar las rutas a todos los destinos, atendiendo a categorías de términos de coste. El problema de n por m es descompuesto en n_1 por m subproblemas que son solucionados de forma iterativa.

Si la solución de 1 por m utiliza el algoritmo de la ruta más exacta, el árbol descendente simplemente va creciendo desde el origen hasta los m posibles destinos, entonces todas las rutas serán clasificadas basadas en el coste computacional.

La solución al problema de 1 por m, será diferente si se utiliza el algoritmo de la mejor ruta jerárquica. Por tanto, se utilizará el algoritmo jerárquico con algunas modificaciones simples.

En la figura 3.23 se muestra una ruta desde un origen hacia dos destinos diferentes. Las operaciones que se tienen que realizar son las siguientes:

- Para encontrar la ruta a los m destinos, se utilizará un árbol descendente desde el origen y m árboles ascendentes desde los destinos.

- Cuando se hayan alcanzado todos los destinos, deben ser calculados todos los árboles ascendentes.

- Cuando sólo se necesita los k destinos más próximos, sólo se calculará los árboles ascendentes necesarios.

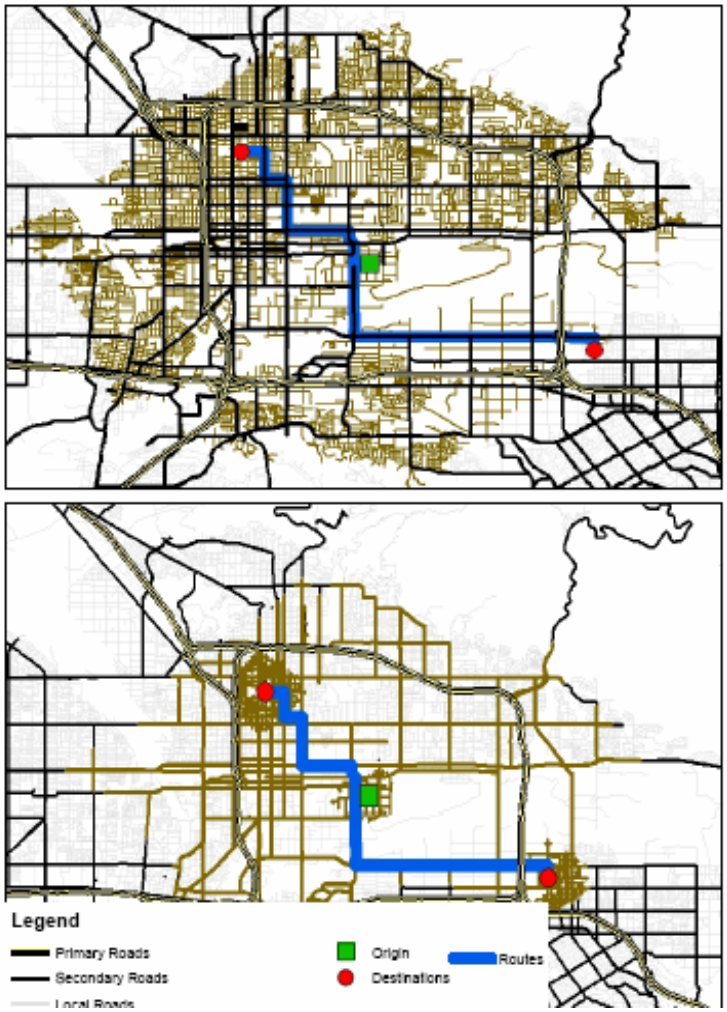


Figura 3.23. Ejemplo de una ruta desde un origen hacia dos destinos, con el algoritmo de la mejor ruta jerárquica y la ruta más exacta.

3.2.3 Conceptos de Modelado de red

La calidad de los datos es esencial para conseguir buenas rutas y llegar a obtener una solución factible. Además, el procesamiento de los datos es quizá la cuestión más importante cuando se realiza el cálculo de rutas en grandes redes.

La estructura de los datos almacenados, la conectividad y la información de los atributos (por ejemplo: costes y restricciones) se unen a menudo al algoritmo de encaminamiento. En esta sección se mostrará información sobre el modelado de las redes jerárquicas en Network Analyst y se conseguirá rutas correctas según las expectativas del usuario.

3.2.4 Crear niveles jerárquicos

Como se mencionó en la sección 3.2.2, los niveles jerárquicos de una red son creados durante el proceso de construcción de la red dataset. Se utiliza un algoritmo eficiente para construir la conectividad de la red, permitiendo navegación en el mismo nivel jerárquico o entre niveles jerárquicos (por ejemplo: ascendiendo desde carreteras locales a carreteras secundarias). Este algoritmo también agrupa espacialmente elementos de red basados en grupos jerárquicos.

El algoritmo jerárquico se respalda en una red dataset compuesta de 3 niveles de jerarquía. Si se crea una nueva red dataset, el usuario debe especificar como le gustaría crear los niveles jerárquicos de las aristas fuentes. Para crear una red dataset con clasificación jerárquica la realización de los dos siguientes pasos es fundamental:

- Primero, para cada elemento de arista de la red, se tiene un nuevo tipo de atributo, llamado atributo jerárquico. Este atributo se utiliza para asignar una categoría, dar una importancia de orden, así como la clasificación de carreteras.

  Esta categoría es representada por un valor entero. El elemento de red de mayor preferencia queda representado con el valor entero de menor valor.

- Segundo, se realizarán grupos en 3 niveles de jerarquía, clasificando los valores jerárquicos indicados.

  El algoritmo jerárquico crea agrupaciones (cluster) basados en esas 3 categorías. Las categorías por defecto, asignan valores jerárquicos de 1, 2 y 3, al nivel jerárquico 1 (carreteras primarias), al nivel jerárquico 2 (carreteras secundarias) y al nivel jerárquico 3 (locales), respectivamente. El usuario puede especificar categorías personalizadas, acordes con sus datos.

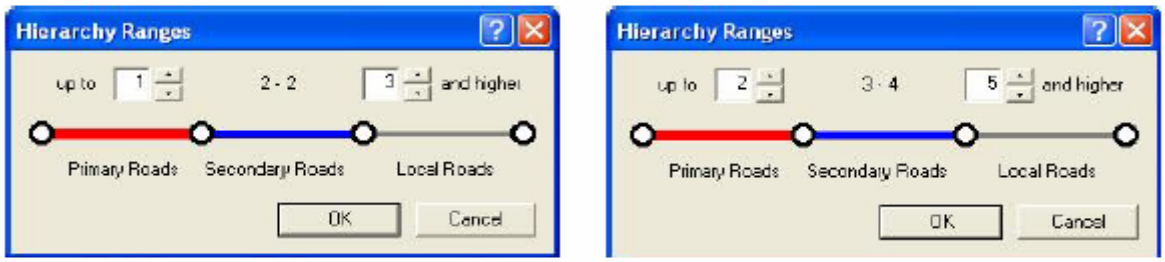


Figura 3.24. Categorías jerárquicas por defecto en la figura de la izquierda y categorías jerárquicas personalizadas en la figura de la derecha.

3.2.4.1 Consideraciones de Modelado

El algoritmo de la mejor ruta jerárquica implementado en Network Analyst, requiere adecuadamente construir niveles jerárquicos para obtener resultados razonables.

A parte de eso, debe producir substancialmente rutas más largas, tomando tiempo de cálculo importante, o en el peor de los casos, debe fallar para encontrar la solución. Es importante analizar la prioridad de los datos origen para construir la red dataset. A continuación, se muestran algunas pautas prácticas en construcción de red dataset por orden de importancia:

- Comprobar si las aristas de la red están conectadas correctamente, por ejemplo, no debería haber isletas en el nivel jerárquico 1. De forma similar, los callejones sin salida en el nivel 1 y 2 afectan a la calidad de la ruta. También, es importante verificar que hay compatibilidad de jerarquía ascendente.

  Por ejemplo, si una rampa de acceso entre una carretera primaria y una carretera secundaria es codificada como carretera local, no sería posible coger esta carretera primaria desde esta carretera secundaria.

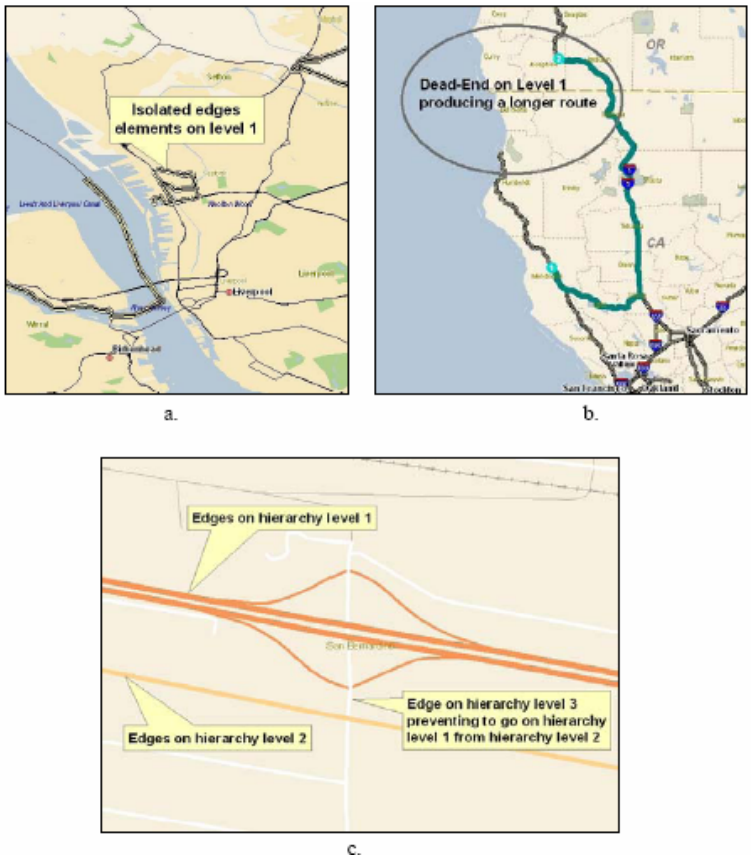


Figura 3.25. a) Aristas aisladas, b) Puntos finales y c) Sin conexión entre la jerarquía 2 y 1

- Crear un atributo jerárquico y asignarle los valores apropiados. El evaluador debe ser creado cuidadosamente, asignarle el valor del atributo jerárquico para cada arista de la red. Por ejemplo, un atributo jerárquico puede ser asignado usando un campo, poner un valor constante o usar un evaluador de VBScript.

- Tener una buena proporción de aristas pertenecientes a diferentes niveles jerárquicos. Se recomienda una propuesta piramidal, donde el nivel más alto contiene un número más pequeño de aristas que el nivel inferior (aproximadamente 10 al 20 % del nivel anterior). Sin embargo, el nivel más alto debería tener suficientes aristas para proporcionar una buena calidad de la ruta.

- Definir que atributos de coste y restricción pueden ser utilizados durante una operación de cálculo con el algoritmo jerárquico. Las redes jerárquicas y los atributos de coste están intrínsecamente unidos, esto significa que ciertos atributos de coste no deben ser usados durante el análisis.

- Entender que elementos de red desconectados, puede pasar como atributos de coste con valores negativos, como la impedancia y/o atributos de restricción. Además, la causa de tener elementos de red desconectados se debe a la definición de categorías jerárquicas por los usuarios.

3.2.4.2 Consideraciones de las redes de Callejeros

Las redes de callejeros pueden ser consideradas por Network Analyst de forma similar a otros formatos de redes dataset (base de datos geográfica personal, geodatabase enterprise y shapefile). Esto significa que el dataset de los callejeros, proporcionado por ArcGIS StreetMap puede ser utilizado para rutas con Network Analyst. Los usuarios no pueden construir redes de callejeros sin la herramienta Network Analyst.

3.2.4.3 Usando jerarquía en Network Analyst

ArcGIS ofrece una completa y flexible herramienta, donde el análisis de la red se puede realizar en ArcMap, usando una herramienta de Geoprocesamiento, en una aplicación desarrollada desde ArcGIS Engine, o vía aplicación Web desarrollada usando ArcGIS Server.

Independientemente, de cómo el análisis de red es realizado en ArcGIS, es importante entender como el uso eficiente de las opciones jerárquicas conseguirán rutas adecuadas con realización óptima.

En esta sección se detalla los parámetros del algoritmo de mejor ruta, expuestos al usuario y se discute la importancia de esos parámetros cuando se realiza el análisis.

3.2.4.4 Parámetros del algoritmo de la mejor ruta

En la tabla 3.3 se muestra los parámetros comunes para la realización del análisis en Network Analyst, con los algoritmos de la ruta más exacta y el algoritmo de la mejor ruta jerárquica. En esta tabla se muestra la definición de cada parámetro, su valor por defecto, y si es necesario para el análisis.

El requisito mínimo para realizar un análisis es especificar la impedancia para reducir al mínimo. Los atributos de restricción y los atributos acumulativos son opcionales. La política de giro toma el valor por defecto.

Nombre del parámetro	Definición	Valor por defecto	Requerido
Impedancia	La impedancia mínima mientras se determina la ruta. El atributo de coste puede ser elegido como impedancia.	N/A	Si
Atributos de restricción	Listado de atributos de restricción utilizados para limitarse transversalmente con un dataset de la red.	N/A	No
Atributos acumulativos	Listado de los atributos de coste acumulados una vez que se ha realizado el cálculo de la ruta. Por ejemplo: si el atributo tiempo de viaje es utilizado como impedancia y otro atributo de coste como longitud es utilizado como acumulativo, la ruta devolverá el tiempo total de viaje y la longitud total.	N/A	No
Política de giro	Mientras se calcula una ruta, la política de giro, puede ser puesta en todas partes, en ninguna o únicamente en los callejones sin salida.	En todas partes	Si

Tabla 3.3. Parámetros comunes para la realización del algoritmo

La tabla 3.4 muestra los parámetros específicos para ejecutar al algoritmo jerárquico.

El requisito mínimo para calcular una ruta utilizando algoritmo jerárquico es indicar el parámetro “Uso jerárquico”. Los otros parámetros: *nombre jerárquico, categorías jerárquicas, número de transiciones a la jerarquía,* son opcionales o automáticamente se les pone su valor por defecto.

Nombre del parámetro	Definición	Valor por defecto
Uso jerárquico	Se utiliza para llamar al algoritmo de mejor ruta jerárquico en el análisis. Esta opción solo está disponible si la red dataset tiene un atributo jerárquico.	Falso en ArcObjects Verdadero en ArcMap y herramientas de geoprocesamiento.
Nombre jerárquico	Especifica el nombre del atributo jerárquico.	N/A
Categorías jerárquicas	Categorías utilizadas para reclasificar las aristas antes de encontrar una ruta jerárquica. Las categorías jerárquicas son definidas por dos parámetros, especificando el valor del atributo jerárquico perteneciente respectivamente a los niveles jerárquicos 1 y 2. Las categorías jerárquicas son especificadas cuando la red dataset es creada, y esas categorías se convierten en valores por defecto durante el análisis. Si el usuario no pone categorías jerárquicas cuando construye la red dataset, los siguientes valores serán utilizados para proporcionar una agrupación jerárquica inicial: • Todas las aristas con el valor del atributo jerárquico igual o inferior a 1, están en el nivel 1. • Todas las aristas con el valor del atributo jerárquico a 2 están en el nivel 2. • El resto de aristas están en el nivel 3.	Se define cuando se crea la red Dataset.
Número de transiciones a la jerarquía	Número máximo de puntos de entrada (o salida) para alcanzar un nivel dado de la jerarquía desde un nivel inferior. Hay dos parámetros disponibles en ArcObjects para definirlo: • Número máximo de puntos para alcanzar el nivel 1 desde el nivel 2 o el 3. • Número máximo de puntos para alcanzar el nivel 2 desde el nivel 3.	Número de transiciones para la jerarquía en el nivel 1 es igual a 9. Número de transiciones para la jerarquía en el nivel 2 es igual a 6.

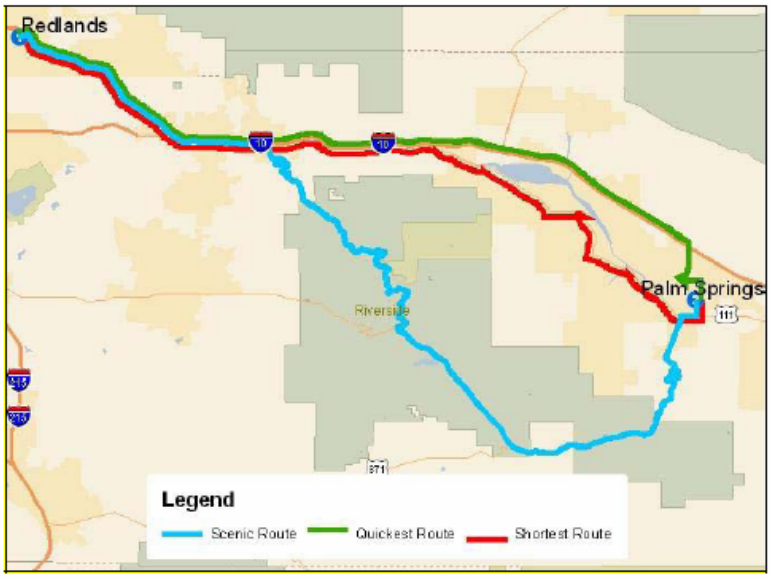
Tabla 3.4. Parámetros específicos para ejecutar el algoritmo

Los parámetros importantes, se pueden clasificar en tres grupos que son:

- La impedancia para reducir al mínimo.
- Aplicar los atributos de restricción, los atributos acumulativos y la política de cambio de giro.
- Las opciones jerárquicas (Uso jerárquico, categorías jerárquicas, y número de transiciones en la jerarquía).

La impedancia es el parámetro más importante para reducir al mínimo. El cálculo de la forma de la ruta, es el coste total, y las direcciones de viaje relacionadas dependen del coste del atributo seleccionado como impedancia.

En la figura 3.26, se muestra un ejemplo de tres rutas diferentes, con las mismas paradas, pero diferentes atributos de coste, hablando de tiempo de viaje, distancia y escenario. Esas tres rutas representan, la ruta más corta, la ruta más rápida, y la ruta que ofrece mejores vistas (mejor escenario). Cada una de ellas es óptima para el atributo de impedancia elegido.



Quickest (Drive Time = 1 hour, 7 minutes; Distance = 94.7 kilometers), Shortest (Drive Time = 1 hour, 12 minutes; Distance = 92.5 kilometers), and the Most Scenic Routes (Drive Time = 2 hours, 6 minutes; Distance = 143 kilometers) between Redlands and Palm Springs, California

Figura 3.26. Ejemplo con 3 rutas (la más rápida, la más corta y la más interesante)

Además, los usuarios deberán saber que los niveles jerárquicos y los atributos de coste están intrínsecamente unidos. Por ejemplo, considerar una red dataset que ha sido construida basada en el atributo de coste tiempo de viaje, y el atributo jerárquico, conocido como “mayor preferencia, las carreteras principales”. El cálculo del tiempo de viaje es calculado según la longitud de las aristas, la clasificación de las carreteras, y los

límites de velocidad. Sin embargo, el atributo “mayor preferencia, las carreteras principales”, es calculado únicamente teniendo en cuenta los límites de velocidad.

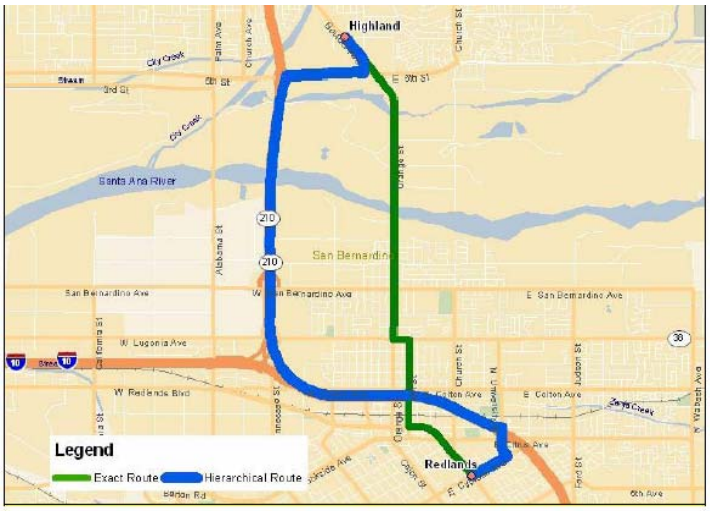
Hay una relación directa, si la ruta a obtener es la ruta más corta en tiempo de viaje, ya que se usará las aristas más rápidas de la red si se ha seleccionado está opción, como opción jerárquica.

Inversamente, puede ser irrelevante usar algunos atributos de coste cuando se usan opciones jerárquicas. Por ejemplo, no tiene sentido usar opciones jerárquicas a favor de las carreteras primarias, mientras se está haciendo una búsqueda de ruta para peatones en zonas urbanas.

Para conseguir una ruta exacta mientras se encaminan vehículos, es importante aplicar adecuadamente las restricciones, así como restricciones unidireccionales y restricciones de giro. También, la política de giro debe de afectar a la solución de la ruta. Los atributos de coste acumulativo, no afectan al cálculo de la ruta.

Usando opciones jerárquicas durante el análisis, se afecta a la calidad y al tiempo de cálculo de la solución. La ruta puede ser subóptima (por ejemplo, en términos de impedancia, pero puede ser más realista).

En la figura 3.27, se muestra la ruta más rápida según el algoritmo de ruta más exacta y el algoritmo de la mejor ruta jerárquica. Además, se puede destacar, que el tiempo computacional es menor en el algoritmo jerárquico que en el algoritmo más exacto, ya que como se indicó en una de las secciones anterior, el tiempo computacional en el algoritmo es menor cuando se trata de grandes redes y de grandes distancias.



Exact Quickest Route (Drive Time = 8 minutes, 30 seconds) versus Hierarchical Quickest Route (Drive Time = 10 minutes, 15 seconds) from Redlands to Highland, California

Figura 3.27. Ruta jerárquica y ruta más exacta

Las categorías jerárquicas y el número de transiciones entre los niveles jerárquicos, juegan un papel importante en el comportamiento del algoritmo jerárquico. Esos parámetros directamente afectan a la calidad de la ruta y al tiempo computacional.

Las categorías jerárquicas indican los diferentes niveles jerárquicos, junto con los valores de sus atributos.

Como se explicó, en la sección Consideraciones de Modelado, se debe de respetar una forma piramidal como descomposición jerárquica. Aunque, un mínimo número de aristas es requerido en cada nivel para conseguir una buena calidad de la solución.

Por otro lado, muchas aristas en el primer nivel de la jerarquía aumenta el tiempo computacional. La compensación entre la calidad y la velocidad puede depender de varios factores, por ejemplo, el objetivo del usuario, la calidad de los datos, y la extensión de los datos.

Varias tentativas deben ser necesarias, para determinar unas buenas categorías jerárquicas para una solución particular. Los tres pasos detallados a continuación, pueden repetirse las veces que sea necesario para conseguir unas buenas prestaciones y una ruta correcta:

1. Construir la red dataset con unos valores determinados de una categoría de la jerarquía.

2. Calcular un sistema de rutas bien definido.

3. Analizar la calidad de la solución y la ejecución de las rutas calculadas.

Una vez que las categorías han sido elegidas y la red dataset ha sido creada, las categorías se convierten en las opciones por defecto. Es posible personalizar los valores de las categorías jerárquicas antes de calcular una ruta, sin reconstruir la red dataset.

Figura 3.28. Ejemplo de ruta jerárquica utilizando categorías jerárquicas por defecto y personalizadas

3.2.4.5 Determinar la calidad y el funcionamiento de la solución

El proceso de validación es una tarea importante al desarrollar una aplicación de encaminamiento, es decir, al desarrollar una aplicación para cálculo de rutas. El proceso de validación debe incluir en su evaluación la calidad y el funcionamiento de la solución.

La evaluación de la calidad de la solución debe depender del tipo de aplicación desarrollada, por ejemplo, cómo de óptima, realista y factible es la solución encontrada.

La evaluación del funcionamiento puede implicar medidas del tiempo computacional y evaluar los recursos (tamaño de memoria, velocidad de CPU, tamaño de disco duro,…) necesarios para configurar la máquina de desarrollo.

Las siguientes pautas proporcionan información práctica para evaluar la calidad de la solución y el tiempo computacional del algoritmo de la mejor ruta jerárquica.

Estas recomendaciones pueden ser realizadas usando el sistema bien conocido de paradas, y comparando con los resultados esperados. Estas recomendaciones se deben de realizar siguiendo con lo que se dijo en la sección de Consideraciones de modelado.

- **Inspección visual de las rutas en ArcMap**. Se podrá visualizar la ruta para ver si se cumple con las restricciones de giro, con la propiedad jerárquica de utilizar el nivel más superior,…

- **Comparar las rutas jerárquicas con las rutas exactas**. Aquí se puede comparar el tiempo computacional y el coste total de cada algoritmo, partiendo desde el mismo origen y el mismo destino. Esto determinará si el ratio es bueno en las carreteras primarias y las carreteras secundarias, respecto al ámbito piramidal.

- **Calcular rutas en áreas de gran densidad y en áreas de baja densidad**. Esto puede ayudar a determinar si hay una buena cobertura espacial en las aristas, en la red de carreteras primarias.

- **Cambiar las categorías jerárquicas por defecto**. Esto ayuda a determinar si la red dataset necesita ser reconstruida con las nuevas categorías jerárquicas.

- **Poner obstáculos en calles, en áreas, etc**., para comprobar si es posible llegar con una ruta alternativa.

3.2.4.6 Soluciones a ciertos problemas en el cálculo de rutas

Las siguientes pautas diagnostican las causas del comportamiento inesperado que algunos usuarios pueden encontrar, al usar el nuevo algoritmo de la mejor ruta jerárquica en Network Analyst. El comportamiento inesperado incluye que:

- Una ruta incorrecta sea calculada.

- La aplicación tome mucho tiempo de cálculo para la solución.

- En el peor caso, la ruta no se encuentre.

Las causas de esos problemas pueden variar: ediciones de modelado de datos, limitaciones del software, o que la ruta sea simplemente inaccesible. Por ejemplo, uno puede pensar que el resultado de un análisis de ruta, usando jerarquía, sea demasiado largo en términos del tiempo total mínimo. Esto se debe a un incorrecto modelado de la jerarquía en la red dataset o debido a que el algoritmo de la mejor ruta jerárquica, es heurístico y no garantiza ser óptimo.

En el siguiente listado se propone unos diagnósticos cuando los usuarios se encuentran con una ruta inesperada entre dos paradas, calculadas por el algoritmo de la mejor ruta jerárquica, y encontrar una solución para fijar un entendimiento al problema. Estos diagnósticos pueden ser:

- Verificar que el atributo de coste apropiado es utilizado como impedancia.

- Verificar que las restricciones de los atributos están correctamente modeladas, se utilizan cuando se realiza el análisis, y están en la ruta.

- Verificar que hay un atributo jerárquico en la red dataset y que el atributo de "Utilización jerárquico" se está utilizando durante el análisis.

- Verificar que el nivel de jerarquía 1 no esta desconectado. Visualizar el nivel de jerarquía 1 utilizando los ficheros fuente. Verificar que el atributo de jerarquía está correctamente asignado.

- Verificar que la solución del algoritmo de la ruta más exacta encuentra una solución óptima con las mismas paradas.

- Si la solución contiene obstáculos, verificar que hay una ruta accesible sin obstáculos.

- Solución con las mismas paradas, con categorías jerárquicas personalizadas.

- Reconstruir la red dataset utilizando formatos de ficheros fuentes diferentes, y entonces resolver el mismo problema.

- Solucionar el mismo problema utilizando los datos fuentes en la misma máquina que la aplicación.

3.2.5 Conclusiones

Una vez estudiado las características y funcionamiento de los algoritmos disponibles en Network Analyst, se pasará a detallar algunas de las conclusiones más importantes en este estudio:

- La propuesta descrita sobre un método de búsqueda heurística comparado con el algoritmo de la ruta más exacta, encuentra soluciones válidas, pero no garantiza una ruta óptima.

- Esta estrategia supone que el nivel más alto de la jerarquía está conectado. Si el nivel 1 de la jerarquía está desconectado, el algoritmo jerárquico no desciende a niveles inferiores. Como por ejemplo, si un callejón sin salida estuviera en el nivel 1. En el peor caso, no se encontrará una ruta.

- La exploración de la red desde el origen al destino, se realiza usando un árbol de búsqueda ascendente y descendente, respectivamente.

- El número de transiciones en los parámetros de la jerarquía, especifican el número de puntos de conexión en el siguiente nivel superior de la jerárquica para el origen y el destino. Por ejemplo, en una red de carreteras, un punto de conexión puede ser una señal que indique acceso a un camino principal a la izquierda.

- Como el procedimiento explora más soluciones alrededor de un obstáculo, el tiempo computacional se incrementa. Este tiempo computacional será significativo si a lo largo de la red se encuentra con muchos obstáculos.

- La estrategia descendente para los obstáculos de la jerarquía en el nivel 1 y 2, evita que una ruta vuelva al nivel 1 si el obstáculo desconecta a la jerarquía del nivel 1.

- Cuando múltiples orígenes están en la solución, todos los árboles destino calculados para el primer origen son reutilizados para el resto de orígenes.

- La reutilización de los árboles ascendentes aumenta la velocidad. Ya que todos los árboles ascendentes calculados se almacenan en memoria, y sólo el árbol descendente desde cada origen debe ser lanzado.

- La solución del algoritmo jerárquico de la ruta n por m, completa 1 por m problemas en grandes redes (por ejemplo, 1 origen y múltiples destinos), por el contrario, el algoritmo la ruta más exacta falla por memoria. Inversamente, para redes pequeñas, la solución del algoritmo de la ruta más exacta es más rápido que el algoritmo de la mejor ruta jerárquica.

  Esto es una de las causas de por qué se ha utilizado el algoritmo de la ruta más exacta, más rápida y más corta (Dijktra) para el desarrollo del proyecto.

- La solución para el problema de n por m rutas, requiere la utilización de mucha memoria RAM, para poder ejecutarse.

- Cuando se crea la red dataset, las categorías creadas por defecto pueden ser modificadas en tiempo de análisis para ajustarlas apropiadamente. Aunque, cambiar las categorías jerárquicas antes de una operación de cálculo, no cambiará la agrupación espacial inicial creada por la red al construir el algoritmo.

- Cuando se usan datos de caminos comerciales, un campo de la clasificación del camino, puede ayudar a definir los diferentes grupos de red para calcular las

rutas lógicas y eficientes. Este campo puede ser utilizado para asignar valores a un atributo de la jerarquía.

- Si una red dataset tiene múltiples clases de características fuente, el atributo evaluador jerárquico debe ser asignado cuidadosamente. Por ejemplo, para una red dataset compuesta por tres clases de caminos (carreteras principales, carreteras secundarias, y locales), el evaluador jerárquico será constante para cada origen y puesto respectivamente a 1, 2 y 3.

- Modelar niveles jerárquicos cuando los modos múltiples de la red se incluyen en un dataset (por ejemplo, calles para coches y peatones, líneas de metro, y líneas de tren para uso público), puede ser una desafiante tarea que la solución planea.

  Para esta tarea, los usuarios deberán analizar para cada nivel jerárquico, que modos de viaje serán incluidos, o excluidos. Además, los usuarios tendrán que analizar si está permitido transferir entre diferentes modos de red y lanzar los diferentes modos jerárquicos. En otras palabras, una arista que representa una transferencia desde un modo (por ejemplo, línea de metro) a otro modo (por ejemplo, línea de tren), debería ser codificada con el apropiado valor jerárquico para anticiparse a los resultados de la ruta.

Capítulo 4. METODOLOGÍA. ANÁLISIS DEL SISTEMA.

"La formulación de un problema es más importante que su solución"

Albert Einstein (1879-1955)
Científico estadounidense de origen alemán.

En este capítulo se explica la metodología de desarrollo seguida para este proyecto fin de carrera, junto con el análisis del sistema. La metodología abarca desde las primeras etapas del proyecto (captura de requisitos) hasta la solución final. Además se describirá el marco en el que se encuentra la metodología seleccionada. Finalmente, se expondrá el análisis del sistema.

4.1 INTRODUCCIÓN

Como se ha visto en los Capítulos 2 y 3, los Servidores de Mapas por Internet y los gestores de rutas son piezas importantes para el tratamiento de datos espaciales. Por tanto, existe la necesidad de integrar ambas herramientas para proporcionar grandes beneficios al usuario con el fin de desarrollar un sistema de localización indoor.

Cómo se ha comentado en el Capítulo 1, el objetivo de este proyecto fin de carrera es realizar consultas espaciales de localización dentro de un sistema *indoor*, con el fin de encontrar información alfanumérica o imágenes asociadas dentro de este sistema. Para ello, es necesario contar con:

1) Un servidor de mapas que proporcionará un soporte de gestión de la información geográfica disponible.

2) Unos ficheros con formato especial, conocidos con el nombre de Shapefile, para almacenar la información alfanumérica asociadas al sistema indoor.

3) Un servidor de rutas, orientados a calcular las rutas entre un origen y un destino desde un punto de vista dinámico y calcular rutas de forma estática ya predefinidas.

4) Un simulador de localización que proporcione la posición actual del usuario en el sistema indoor.

Una vez que tenemos todos los servidores necesarios preparados, será necesario desarrollar una interfaz de usuario que permite al usuario realizar consultas al sistema. La usabilidad de la interfaz es sin duda uno de los requisitos imprescindible para obtener una solución viable.

Un factor importante a la hora de llevar a cabo un proyecto es la elección de la metodología a seguir para el desarrollo del mismo. En este caso la metodología seleccionada ha sido una adaptación de la **eXtreme Programming** (**XP**) dentro del marco de las conocidas como metodologías ágiles. Este tipo de metodologías están especialmente orientadas para proyectos pequeños, y constituyen una solución a medida para ese entorno, aportando una elevada simplificación que a pesar de ello no renuncia a las prácticas esenciales para asegurar la calidad del producto.

Las metodologías ágiles, aunque recientes, gozan de un gran interés por su visión industrial del desarrollo de software. Además de ser muy aptas para proyecto de pequeño y mediano tamaño, las metodologías ágiles son especialmente interesantes en aquellos proyectos en los cuales los equipos de desarrollo son pequeños, con plazos reducidos, requisitos volátiles, y/o basados en nuevas tecnologías. Estas son la causas principales por las que se ha optado por una metodologías de este tipo para el desarrollo de este proyecto, ya que cumple cada una de ellas.

En el Anexo I se muestra una introducción a las metodologías ágiles y en particular a una de las más difundidas, *eXtreme Programming* (XP). Esta introducción ha sido extraída de un trabajo previo de José Hilario Canós y Patricio Letelier ambos pertenecientes al Departamento de Sistemas Informáticos de la Universidad Politécnica de Valencia.

Las metodologías ágiles se basan en un conjunto de manifiestos desarrollados por *The Agile Alliance*, una organización, sin ánimo de lucro, dedicada a promover los conceptos relacionados con el desarrollo ágil de software y ayudar a las organizaciones para que adopten dichos conceptos.

Los valores anteriores inspiran los doce principios del manifiesto. Son características que diferencian un proceso ágil de uno tradicional. Los dos primeros principios son generales y resumen gran parte del espíritu ágil. El resto tienen que ver

con el proceso a seguir y con el equipo de desarrollo, en cuanto a metas a seguir y organización del mismo. Los principios son:

- La prioridad es **satisfacer al cliente** mediante tempranas y continuas entregas de software que le aporte un valor.
- **Dar la bienvenida a los cambios**. Se capturan los cambios para que el cliente tenga una ventaja competitiva.
- **Entregar frecuentemente software que funcione** desde un par de semanas a un par de meses, con el menor intervalo de tiempo posible entre entregas.
- **La gente del negocio y los desarrolladores deben trabajar juntos** a lo largo del proyecto.
- **Construir el proyecto en torno a individuos motivados**. Darles el entorno y el apoyo que necesitan y confiar en ellos para conseguir finalizar el trabajo.
- El **diálogo cara a cara** es el método más eficiente y efectivo para comunicar información dentro de un equipo de desarrollo.
- El **software que funciona es la medida principal** de progreso.
- **Los procesos ágiles promueven un desarrollo sostenible**. Los promotores, desarrolladores y usuarios deberían ser capaces de mantener una paz constante.
- **La atención continua, la calidad técnica y el buen diseño mejora la agilidad**.
- La **simplicidad** es esencial.
- Las mejores arquitecturas, requisitos y diseños surgen de los **equipos organizados por sí mismos**.
- En intervalos regulares, **el equipo reflexiona respecto a cómo llegar a ser más efectivo**, y según esto ajusta su comportamiento.

Con idea de comparar las metodologías ágiles con las tradicionales, en la tabla 4.1 se muestra un resumen:

Metodologías Ágiles	Metodologías Tradicionales
Basadas en heurísticas provenientes de prácticas de producción de código	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo
Especialmente preparados para cambios durante el proyecto	Cierta resistencia a los cambios
Impuestas internamente (por el equipo)	Impuestas externamente
Proceso menos controlado, con pocos principios	Proceso mucho más controlado, con numerosas politicas/normas
No existe contrato tradicional o al menos es bastante flexible	Existe un contrato prefijado
El cliente es parte del equipo de desarrollo	El cliente interactúa con el equipo de desarrollo mediante reuniones
Grupos pequeños (<10 integrantes) y trabajando en el mismo sitio	Grupos grandes y posiblemente distribuidos
Pocos artefactos	Más artefactos
Pocos roles	Más roles
Menos énfasis en la arquitectura del software	La arquitectura del software es esencial y se expresa mediante modelos

Tabla 4.1. Comparativa entre las metodologías ágiles y metodologías tradicionales.

En las siguientes secciones se hace un estudio mas detallado de la metodología empleada en el proyecto, así como su adaptación al mismo. Seguidamente se realizará el análisis del sistema mediante casos de uso y diagramas de actividad.

La estructura del capítulo queda como sigue. En la sección 4.2 se explica la adaptación de la metodología de desarrollo de software *eXtreme Programming* (XP) al presente proyecto fin de carrera.

En la sección 4.3 se realiza el análisis de la funcionalidad del sistema global, identificando y describiendo, en formato tabular, los diferentas casos de uso. Es importante destacar que para la redacción de esta sección no se ha considerado el resultado final tanto de requisitos como de soluciones de análisis, obviando que el proceso seguido ha sido iterativo e incremental.

Finalmente, en la sección 4.4 se explica el momento, dentro del desarrollo del proyecto, en el que se realizó el estudio del arte de las diferentes tecnologías empleadas para la realización del proyecto fin de carrera.

4.2 METODOLOGÍA UTILIZADA EN EL PROYECTO

Como se comentó en la sección anterior la metodología empleada en el desarrollo del proyecto ha sido **eXtreme Programming (XP),** el ciclo de desarrollo de esta metodología consiste (a grandes rasgos) en los siguientes pasos:

1. El cliente define el valor de negocio a implementar.
2. El programador estima el esfuerzo necesario para su implementación.
3. El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.
4. El programador construye ese valor de negocio.
5. Vuelve al paso 1.

En todas las iteraciones de este ciclo tanto el cliente como el programador aprenden. No se debe presionar al programador a realizar más trabajo que el estimado, ya que se perderá calidad en el software o no se cumplirán los plazos. De la misma forma el cliente tiene la obligación de manejar el ámbito de entrega del producto, para asegurarse que el sistema tenga el mayor valor de negocio posible con cada iteración. El ciclo de vida ideal de XP consiste en seis fases: (i) Exploración, (ii) Planificación de la Entrega (*Release*), (iii) Iteraciones, (iv) Producción, (v) Mantenimiento y (vi) Muerte del Proyecto.

En este proyecto se han abordado las cuatro primeras, siendo imposible por las características de un proyecto fin de carrera abordar el mantenimiento y la muerte del mismo.

En la fase de **Exploración** se han capturado los requisitos mediante diagramas de casos de uso. Posterior al desarrollo de cada requisito, una vez validado por el cliente, se ha documentado con detalle.

La fase de **Planificación de la Entrega** también ha sido abordada para cada iteración, definiendo plazos acorde a la complejidad de la solución. El grado de complejidad ha sido calculado con el ratio novedad y dificultad de la tecnología a utilizar y la experiencia objetiva del programador. El valor de novedad y dificultada de la tecnología era estimado por el jefe de proyecto de SITESA, y la experiencia objetiva del programador era calculada según el expediente académico de éste y su experiencia previa. Fueron definidas tres iteraciones del proyecto, abordando en cada una de ellas, captura de requisitos, diseño y pruebas.

La fase de **Producción** no ha sido abordada estrictamente como tal debido a la falta de existencia de un cliente final que explorase el resultado. Sin embargo, esta fase sirvió como fase de pruebas del desarrollo para cada iteración.

La principal suposición que se realiza en *XP* es la posibilidad de disminuir la mítica curva exponencial del costo del cambio a lo largo del proyecto, lo suficiente para que el diseño evolutivo funcione. Esto se consigue gracias a las tecnologías disponibles para ayudar en el desarrollo de software y a la aplicación disciplinada de un conjunto de prácticas que *XP* define. A continuación se definen las prácticas y como has sido abordadas en nuestro proyecto.

- **El juego de la planificación.** *Hay una comunicación frecuente entre el cliente y los programadores. El equipo técnico realiza una estimación del esfuerzo requerido para la implementación de las historias de usuario y los clientes deciden sobre el ámbito y tiempo de las entregas de cada iteración.* En nuestro caso, el cliente era la propia empresa en donde se ha implementado este proyecto, SITESA, por lo que la planificación ha sido llevada formal y estrictamente, pudiendo calcular en cada momento el tiempo de ejecución dimensionando los objetivos.

- **Entregas pequeñas**. *Producir rápidamente versiones del sistema que sean operativas, aunque no cuenten con toda la funcionalidad del sistema. Esta versión ya constituye un resultado de valor para el negocio. Una entrega no debería tardar más de 3 meses.* Así ha sido abordado el proyecto, definiendo e implementando con ciclos incrementales funcionalidades básicas que posteriormente se han ido incrementando. El escaso tiempo exigido para la ejecución del proyecto así lo requería.

- **Metáfora**. *El sistema es definido mediante una metáfora o un conjunto de metáforas compartidas por el cliente y el equipo de desarrollo. Una metáfora es una historia compartida que describe cómo debería funcionar el sistema (conjunto de nombres que actúen como vocabulario para hablar sobre el dominio del problema, ayudando a la nomenclatura de clases y métodos del sistema).* La metáfora del sistema para este proyecto estaba basada en los servidores de mapas y gestores de rutas.

- **Diseño simple**. *Se debe diseñar la solución más simple que pueda funcionar y ser implementada en un momento determinado del proyecto.* Esto ha dado lugar a diferentes soluciones de implementación en un corto espacio de tiempo.

- **Pruebas**. *La producción de código está dirigida por las pruebas unitarias. Éstas son establecidas por el cliente antes de escribirse el código y son ejecutadas*

constantemente ante cada modificación del sistema. Las pruebas unitarias han sido llevadas a cabo en la implementación. Además se establecieron casos de prueba de aceptación en la propia definición de los requisitos.

- **Refactorización (Refactoring)**. *Es una actividad constante de reestructuración del código con el objetivo de remover duplicación de código, mejorar su legibilidad, simplificarlo y hacerlo más flexible para facilitar los posteriores cambios.* Se ha mejorado la estructura interna del código sin alterar su comportamiento externo en las diferentes iteraciones.

- **Programación en parejas**. *Toda la producción de código debe realizarse con trabajo en parejas de programadores. Esto conlleva ventajas implícitas (menor tasa de errores, mejor diseño, mayor satisfacción de los programadores, etc.).* Esta práctica, uno de los buques insignia de XP, no ha sido abordada en el proyecto debido a que era unipersonal.

- **Propiedad colectiva del código.** *Cualquier programador puede cambiar cualquier parte del código en cualquier momento.* Al ser unipersonal, se ha llevado esta práctica estrictamente.

- **Integración continúa.** *Cada pieza de código es integrada en el sistema una vez que esté lista. Así, el sistema puede llegar a ser integrado y construido varias veces en un mismo día.*

- **40 horas por semana.** *Se debe trabajar un máximo de 40 horas por semana. No se trabajan horas extras en dos semanas seguidas. Si esto ocurre, probablemente está ocurriendo un problema que debe corregirse. El trabajo extra desmotiva al equipo.* Sin duda esta práctica ha sido especialmente útil en nuestro proyecto.

- **Cliente in-situ.** *El cliente tiene que estar presente y disponible todo el tiempo para el equipo. Éste es uno de los principales factores de éxito del proyecto XP. El cliente conduce constantemente el trabajo hacia lo que aportará mayor valor de negocio y los programadores pueden resolver de manera inmediata cualquier duda asociada. La comunicación oral es más efectiva que la escrita.* Práctica abordada también muy estrictamente al estar el cliente a escaso metros del programador.

- **Estándares de programación**. *XP enfatiza que la comunicación de los programadores es a través del código, con lo cual es indispensable que se sigan ciertos estándares de programación para mantener el código legible.*

4.3 USABILIDAD

Esta sección está dedicada al concepto de usabilidad de un producto software y a la recopilación e identificación de los mecanismos y recursos disponibles para su logro, teniendo en cuenta que, siendo nuestro principal interés los aspectos relacionados con la Interfaz de Usuario (IU), el punto de vista ofrecido será la visión del usuario, pero no podemos caer en el error de suponer que la usabilidad afecta solo a la IU, es cierto que para el usuario, la interfaz es su aplicación, pero como se irá viendo es más que la simple cualidad de la interfaz del usuario, que acompaña a la aplicación, y así queda reflejado en la propia evolución de la características distintivas asociadas a la IU.

A continuación, se mostrará como surge la necesidad y conciencia de la búsqueda de la usabilidad en el desarrollo de productos software, seguidamente se analiza cual es el concepto confuso aun existiendo estándares internacionales, para terminar se muestra como considerar la usabilidad en el desarrollo de un producto software y como se aplica en el caso práctico.

4.3.1 Introducción

En los últimos tiempos con el auge de las nuevas tecnologías, a fomentado la extensión de la informática y más concretamente de Internet a todas las capas de la sociedad y ha creado la necesidad de estudiar a estos usuarios, para ajustar los productos al máximo. Las aplicaciones cada vez tienen más usuarios y más heterogéneos, primeros fueron los académicos, después las empresas y ahora el ciudadano en general. Su comportamiento, hábitos, expectativas, necesidades o satisfacción es lo que quieren saber los estudios que realizan empresas o instituciones para definir sus productos, servicios o imagen.

Por otro lado, los usuarios exigen cualidades al software destinadas a facilitar su trabajo, ahorrar tiempo, evitar y corregir los errores. Las empresas dedican cada vez mas tiempo y recursos a resolver estas cuestiones.

Todo esto ha llevado a la necesidad y conciencia de añadir la usabilidad desde las primeras fases del proceso de desarrollo de productos software para conseguir así productos más usables.

4.3.2 Concepto de usabilidad

La usabilidad como concepto no tiene un significado académicamente claro, por lo que es importante definir este concepto para después ver sus relaciones con otras disciplinas de las que hereda elementos de estudio como el diseño gráfico de la IU

(GUI, Graphical User Interface), IS, IPO, ergonomía, psicología, sociología y lingüística.

En la literatura científica el termino usabilidad esta extensamente utilizado y hay muchas definiciones propuestas. Por ejemplo, algunas de ellas:

En el modelo de Jacob Nielsen [NJ1993], la usabilidad es “Parte de la utilidad del sistema, la cual es parte de la aceptabilidad práctica y, finalmente parte de la aceptabilidad del sistema”.

Jacob Nielsen, es uno de los pioneros en la difusión de la usabilidad, sugiere que la usabilidad es un término multidimensional. Indica que un sistema usable debe poseer los siguientes atributos: Capacidad de aprendizaje, eficiencia en el uso, facilidad de memorizar, tolerante a errores y subjetivamente satisfactorio.

Jenny Preece [PJ1994], autora de multitud de estudios de usabilidad y de varios reconocidos libros, propone la definición más corta pero quizás la más intuitiva. Se refiere a la usabilidad como el “desarrollo de sistemas fáciles de usar y de aprender”.

Niegel Bevan [BKM1991], la define como la “facilidad de uso y la aceptabilidad de un sistema o producto APRA una clase particular de usuarios que llevan a cabo tareas específicas en un entorno específico”.

Cuando veamos, un poco más abajo el estándar ISO 9241-11, adivinaremos sin mucha dificultad, que Niegel Bevan ha contribuido de forma muy directa en la definición propuesta de esta propuesta.

Janice (Ginny) Redish (Redish & Associates, Inc. USA), reconocida profesional de la usabilidad defiende la idea de que el objetivo de las personas que trabajan en la usabilidad no es otro que el de producir “trabajos para sus usuarios” proporcionando a los usuarios las herramientas para poder: (i) encontrar lo que necesitan, (ii) entender lo que encuentran, (iii) actuar apropiadamente sobre ese entendimiento, y (iv) hacer todo esto con el tiempo y esfuerzo que ellos creen necesarios porque el término usabilidad no se refiere solamente a hacer que los sistemas sean simples, si no que comprende además la comprensión de los objetivos de los usuarios, el contexto de su trabajo y cual es el conocimiento y la experiencia de que disponen [RJ1995].

Whitney Quesenbery [QW2001] propone extender la definición de la ISO 9241 para hacerla, según ellas, más comprensible. Propone definir la usabilidad en base a las 5 características que los usuarios deben encontrar en el sistema interactivo: Effective (Efectividad), Efficiency (Eficiencia), Engaging (ser atractivo), Error-Tolerant (tolerante a errores) y Easy-to-Learn (fácil de aprender).

La Organización Internacional de Estándares (ISO) proporciona dos definiciones de usabilidad en sendos estándares:

- ISO/IEC 9241-11: “Nivel con el que un producto se adapta a las necesidades del usuario y puede ser utilizado por el mismo para lograr unas metas con efectividad, eficiencia y satisfacción en un contexto especifico de uso”. Esta es una definición centrada en el concepto de la calidad en el uso, es decir, se refiere a cómo el usuario realiza tareas específicas en escenarios específicos con efectividad, eficiencia y satisfacción. Por efectividad se entiende la precisión y la plenitud con las que los usuarios llegan a los objetivos especificados. A esta idea va asociada la facilidad de aprendizaje, la tasa de errores del sistema y la facilidad del sistema para ser recordado. La eficiencia se medirá por los recursos empleados en relación con esta precisión y plenitud con la que los usuarios han llegado a los objetivos especificados. La satisfacción se relaciona con la ausencia de incomodidad y una actitud positiva en el uso del producto.

- ISO/IEC 9126: [BM1994] “La usabilidad se refiere a la capacidad de un software de ser comprendido, aprendido, usado y ser atractivo para el usuario, en condiciones especificas de uso”. Esta definición pone énfasis en los atributos internos y externos del producto, que contribuyen a su usabilidad, funcionalidad y eficiencia. La usabilidad no solo depende del producto sino también del usuario. Por eso, un producto no es ningún caso intrínsecamente usable, solo tendrá la capacidad de ser usado en un contexto particular y por usuarios particulares. La usabilidad no puede ser valorada estudiando un producto de manera aislada.

La ISO-9241-11 es una definición centrada en el concepto de la calidad en el proceso, es decir, se refiere a como el usuario realiza tareas especificas en escenarios específicos, mientras que la segunda definición, ISO 9126 se centra en la calidad como producto, ve la usabilidad de la aplicación como una parte del producto en si, y valora esa parte sin tener que desarrollar ninguna acción [ISO9126].

Pasamos a mostrar que es un modelo de calidad y como se logra integrar en el desarrollo de productos software.

4.3.3 Descomponiendo la usabilidad

La elaboración de un modelo de calidad pasa por la identificación de aquellas características, tanto internas como externas, que influyen en la calidad de un producto software. Un modelo de calidad es una descomposición jerárquica, y su elaboración puede abordarse de diferentes formas, por ejemplo identificar objetivos y métricas que permitirán la caracterización de la calidad.

La IS ha elaborado y están disponibles diferentes modelos de calidad y estándares relacionados con la calidad que ofrece un producto software, Principalmente, destacamos dos tendencias, los modelos de calidad que persiguen identificar características de producto (McCall, boehm, ISO 9216,…) y por otros modelos de calidad que buscan características de calidad en el proceso (SPICE, COCOMO,…), por otro lado tenemos la IPO, que se centra en proponer modelados que capten la sensación que procesa un usuario al hacer uso del producto, esta interacción se realiza a través de la IU.

4.3.4 Qué y cómo considerar la usabilidad

Hasta ahora se han mostrado un conjunto de factores, criterios y modelos para aplicar al desarrollo pero no tenemos unas pautas fijas de cómo realizarlo, falta integrar dichos elementos en el propio proceso de desarrollo. Dependiendo del punto de vista del desarrollador y la experiencia del mismo, tiende más a aplicar unos factores de usabilidad que otros.

Para aplicar usabilidad al desarrollo del producto, tendríamos que ver que estándares internacionales queremos obtener, y en función de ellos ver que modelos tenemos que aplicar para obtenerlos, pero aun así los ingenieros con poca experiencia en desarrollos les falta dirección a la hora de seleccionar qué y cómo desarrollar su proyecto, y además muchos criterios son difíciles de conseguir aplicar.

Como se pude apreciar, no hay un modelo de calidad cerrado, pero si que existe un primer nivel de descomposición de calidad extendido y aceptado, el estándar 9126, a partir de ahí se apuesta por la utilización de los modelos de calidad abiertos donde cada uno considere aquello que más le interese.

A continuación, se muestra la propuesta de Montero [Montero2005], que apuesta por una mezcla entre los estándares internacionales y los criterios ergonómicos que recogemos en la Tabla 4.2.

Sus características son:

- Está basado en estándares internacionales como la ISO 9126.
- Es abierto, puede incorporarse nuevos criterios de calidad, ligándose a los ya identificados en el mismo.
- Utiliza criterios ergonómicos.
- Permite la puesta en práctica de técnicas de usabilidad.

En la propuesta, a los criterios ergonómicos, se han añadido nuevos criterios, asociándolos a criterios de calidad de la ISO 9126 y ligados directamente con la usabilidad, con los factores de understandability, learnability y operability.

Calidad	Factor	Criterios	Nivel Importancia
Usability	Understandability	Compatibility	Alto
		Legibility	Medio
		Prompting	Medio
		Inmediate feed-back	Alto
		Significance of codes and behaviours	Alto
		Helpfulness	Medio
	Learnability	Grouping	Alto
		Minimal Action	Alto
		Conciseness	Bajo
		Consistency	Alto
		Information density	Medio
	Operability	Explicity user action	Alto
		User control	Alto
		User experience's	Alto
		Flexibility	Medio
		Error protection	Alto
		Quality of error messages	Bajo
		Error policies	Medio
		Privacy policies	Bajo
		Accesibility	Alto

Tabla 4.2. Propuesta de modelo de calidad centrado en la usabilidad

Este es el modelo de calidad que se aplicará como requisito no funcional.

4.3.5 Conclusiones

La usabilidad se considera más y más como parte del proceso del desarrollo de producto en vez de ser una actividad separada realizada por un departamento responsable de examinar la usabilidad de los productos diseñados. Así, los métodos de

calidad se aplican en fases tempranas de la definición del concepto, y disminuyen costes y esfuerzo económicos.

En el último punto, se estudió la propuesta de (Montero, 2005), que se basa en una mezcla entre los estándares internacionales y los criterios ergonómicos que se recoge en la tabla 3.3. Este modelo de calidad será el utilizado en el capítulo de caso de estudio.

Lo que se intenta es crear un proceso centrado en el usuario y basado en el conocimiento profundo de sus tareas. En este proyecto se ha abundado en los requisitos de usuario considerando de forma empírica la necesidad de tener presentes requisitos no explícitamente funcionales relacionados con la usabilidad, estas consideraciones darán lugar a descubrir nuevos casos de uso y consideraciones, fuera de la funcionalidad del sistema, que se añadirán al mismo y servirán para llegar a productos más óptimos.

A continuación, se muestran los requisitos del sistema, funcionales y no funcionales que son necesarios para llevar a cabo el caso práctico.

4.4 REQUISITOS DEL SISTEMA

En esta fase se trata de familiarizarnos con el entorno del caso práctico, capturando los requisitos funcionales y los requisitos no explícitamente funcionales, para ello se usarán los diagramas de casos de uso para los requisitos funcionales, y criterios ergonómicos y estándares internacionales para los requisitos no explícitamente funcionales.

4.4.1 Requisitos funcionales

A continuación se muestra la funcionalidad aportada por el sistema (en forma de casos de uso), junto con una descripción de dicha funcionalidad (en forma de tablas que describen los casos de uso identificados). Además se adjuntan los diagramas de actividad para los casos de uso más completos.

Se ha definido dos perfiles de usuario, el usuario conectado a través de un puesto móvil y el usuario conectado a través de un puesto fijo, por ello, se muestra un diagrama de actores, como se muestra en la figura 4.1, junto con una descripción (en forma de tablas que describe a cada actor).

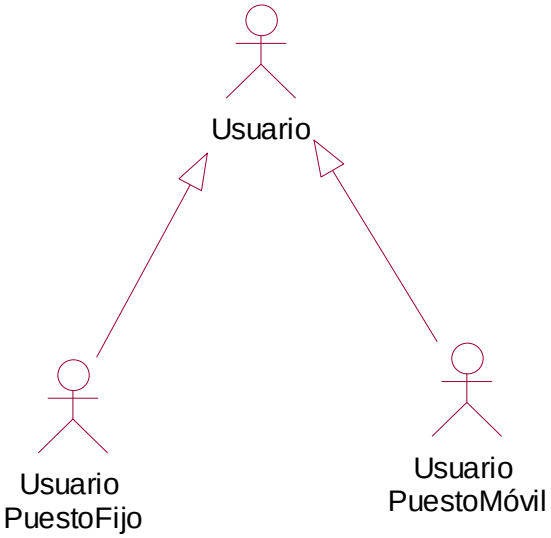


Figura 4.1. Diagrama de Actores

ACT-01	Usuario
Descripción	Este actor representa a cualquier usuario conectado a la aplicación, sin distinción de su hardware.

Tabla 4.3. Descripción del actor Usuario

ACT-02	Usuario PuestoFijo
Descripción	Este actor representa a un usuario conectado a la aplicación web a través de un puesto fijo, como puede ser un PC.

Tabla 4.4. Descripción del actor Usuario PuestoFijo

ACT-01	Usuario PuestoMóvil
Descripción	Este actor representa a un usuario conectado a la aplicación web a través de un puesto móvil, como puede ser una PDA o un teléfono móvil.

Tabla 4.5. Descripción del actor Usuario Puesto Móvil

4.5 Requisitos funcionales para dispositivo móvil

En la figura 4.2 se puede ver el diagrama de casos de uso para el caso particular de dispositivo móvil y desde la tabla 4.6 hasta la 4.15 se muestra las correspondientes descripciones para dichos casos de uso.

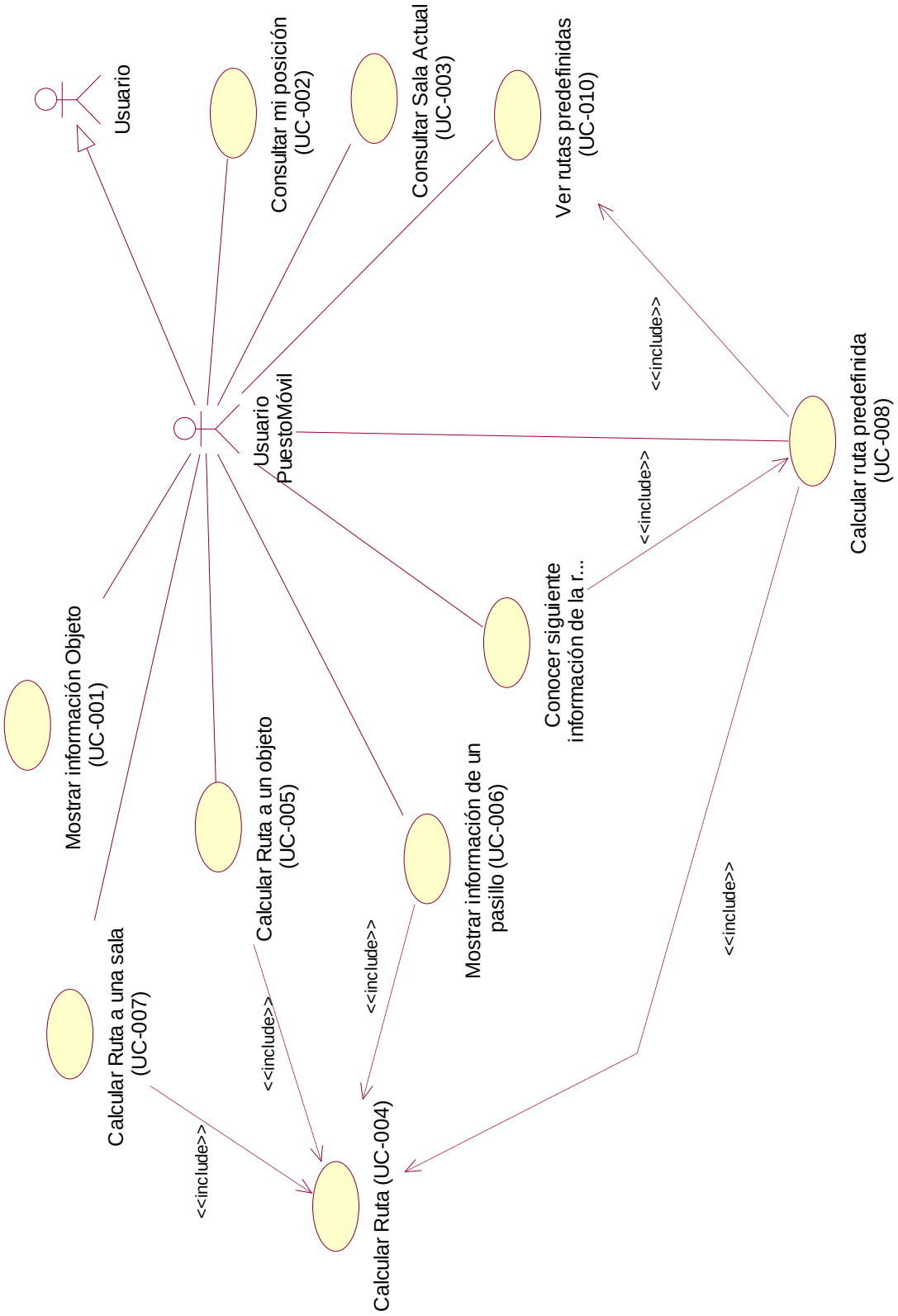


Figura 4.2. Diagrama de casos de uso para el usuario PuestoMóvil

La descripción de los casos de uso se muestra desde la tabla 4.6 hasta la tabla 4.15:

UC-001	Mostrar información objeto	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario desee conocer información de los objetos, como título, autor, año, etc.	
Precondición	El usuario PuestoMóvil debe de de estar conectado a la página de la aplicación.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario PuestoMóvil solicita a la aplicación mostrar información sobre los objetos.
	2	El sistema muestra la página donde se visualizará la información.
	3	El usuario selecciona el objeto del que quiere conocer la información.
	4	El sistema recoge el nombre del objeto seleccionado.
	5	El sistema envía una petición al servidor de mapas para obtener la información asociada al objeto.
	6	El sistema muestra al usuario la información de dicho objeto en pantalla.
Poscondición	Se muestra la información del objeto.	
Flujo alternativo	**Paso**	**Acción**
	4	Si el usuario no ha seleccionado un objeto se muestra un mensaje de error.

Tabla 4.6. Descripción del caso de uso Mostrar información objeto (UC-001)

UC-002	Consultar mi posición	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario PuestoMóvil desee conocer su posición en el plano e información de su entorno.	
Precondición	El usuario Puesto Móvil debe de tener una posición (x,y).	
Secuencia normal	**Paso**	**Acción**
	1	El usuario PuestoMóvil solicita al sistema conocer su posición.
	2	El sistema manda una petición al servidor de coordenadas para obtener la posición (x,y) del usuario PuestoMóvil.
	3	El sistema recoge la posición (x,y) del usuario PuestoMóvil.
	4	El sistema envía una petición al servidor de mapas para obtener la imagen del plano con la posición.
	5	El sistema muestra al usuario PuestoMóvil el plano junto con su posición en pantalla, así como otra información como los objetos más cercanos.
Poscondición	Se muestra una imagen del plano con la posición actual.	
Flujo alternativo	**Paso**	**Acción**
	5	Si el usuario no tiene objetos cercanos sólo se muestra la imagen del plano junto con su posición.

Tabla 4.7. Descripción del caso de uso Consultar mi posición (UC-002)

UC-003	Consultar Sala Actual	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario Puesto Móvil desee conocer cual es su sala actual y en que sala estaba anteriormente.	
Precondición	El usuario PuestoMóvil debe de haber cambiado de sala.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario PuestoMóvil pide al sistema conocer su sala actual.
	2	El sistema envía una petición al servidor de coordenadas para obtener la posición (x,y) del usuario PuestoMóvil.
	3	El sistema recoge la posición (x,y) del usuario PuestoMóvil actual.
	4	El sistema comprueba si el usuario PuestoMóvil ha cambiado de Sala.
	5	El sistema envía una petición al servidor de mapas para obtener la imagen del plano con la sala actual y la sala anterior.
	6	El sistema muestra al usuario PuestoMóvil el plano indicando la sala actual y la sala anterior.
Poscondición	Se muestra una imagen del plano con la sala actual y la sala anterior.	
Flujo alternativo	**Paso**	**Acción**
	4	Si el usuario PuestoMóvil no ha cambiado de Sala, se muestra un mensaje de aviso.

Tabla 4.8. Descripción del caso de uso Consultar Sala Actual (UC-003)

UC-004	Calcular ruta	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso durante la realización de los casos de uso: • UC-005 Calcular ruta a un objeto. • UC-006 Calcular ruta a una sala. • UC-007 Mostrar información de un pasillo. • UC-008 Calcular ruta predefinida. • UC-022 Calcular ruta entre salas.	
Precondición	El sistema tiene almacenado una serie de parámetros.	
Secuencia normal	**Paso**	**Acción**
	1	El sistema recoge los parámetros necesarios para iniciar el cálculo de ruta.
	2	El sistema envía una petición al servidor de rutas para que le calcule la ruta según los parámetros seleccionados.
	3	El servidor de rutas calcula la ruta.
	4	El sistema reordena la información obtenida del servidor de rutas.
	5	El sistema almacena la información de la ruta.
	6	El sistema muestra una imagen de la ruta a seguir e información de la ruta.
Poscondición	Se almacena la información de la ruta seleccionada.	

Flujo alternativo	**Paso**	**Acción**
	1	Si el sistema no tiene los parámetros necesarios, no inicia el cálculo de ruta.
	2	Si la ruta es entre dos salas, enviará una petición al servidor de rutas, indicando la sala origen y la sala destino
	2	Si la ruta es una ruta predefinida, enviará una petición al servidor de rutas, indicando el nombre de la ruta que se quiere obtener.

Tabla 4.9. Descripción del caso de uso Calcular ruta (UC-004)

UC-005	**Calcular ruta a un objeto**	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario PuestoMóvil desee consultar cual es la ruta a seguir desde su posición (x,y) a un objeto.	
Precondición	El usuario debe de estar conectado a la página de la aplicación.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario PuestoMóvil quiere conocer la ruta a un objeto.
	2	El usuario selecciona un objeto.
	3	El sistema manda una petición al servidor de coordenadas para obtener la posición (x,y) del usuario PuestoMóvil.
	4	El sistema recoge la posición (x,y) del usuario PuestoMóvil actual.
	5	El sistema recoge el nombre del objeto seleccionado.
	6	El sistema manda una petición al servidor de mapas para obtener la sala donde está el objeto.
	7	El sistema manda una petición al servidor de mapas para obtener la sala donde está el usuario Puesto Móvil.
	8	El sistema almacena la sala donde está el objeto y la sala del usuario para el cálculo de rutas.
	9	Se realiza el caso de uso Calcular Ruta (UC-004)
Poscondición	Se muestra una imagen con la ruta y su información.	
Flujo alternativo	**Paso**	**Acción**
	5	Si el usuario PuestoMóvil no ha seleccionado un objeto se muestra un mensaje de error.
	9	Si la sala del usuario PuestoMóvil y la sala del objeto es la misma, se manda un mensaje de aviso al usuario informándole de la situación.

Tabla 4.10. Descripción del caso de uso Calcular ruta a un objeto (UC-005)

UC-006	**Mostrar información de un pasillo**	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario desee consultar la información almacenada sobre un pasillo determinado.	
Precondición	El usuario se encuentra conectado a la página de la aplicación.	
Secuencia normal	**Paso**	**Acción**
	1	Se realiza el caso de uso Calcular Ruta (UC-004)

	2	El usuario selecciona el pasillo del que quiere conocer su información.
	3	El sistema recoge el nombre del pasillo seleccionado.
	4	El sistema manda una petición al servidor de mapas para que le muestre información detallada de este pasillo, como imagen zoom a él, objetos cercanos,….
	5	El sistema recibe la información, y la reordena.
	6	El sistema muestra al usuario la información solicitada.
Poscondición	Se muestra una imagen *zoom* al pasillo, listado de objetos de este pasillo, y una imagen indicando el pasillo en el plano, con una posición (x,y)si el usuario se encontrara en él.	
Flujo alternativo	**Paso**	**Acción**
	6	Si el pasillo seleccionado, no tiene objetos cercanos, no se muestra el listado de objetos.

Tabla 4.11. Descripción del caso de uso Mostrar información de un pasillo (UC-006)

UC-007	**Calcular ruta a una sala**	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario PuestoMóvil desee consultar cual es la ruta a seguir desde su posición (x,y) a una sala determinada.	
Precondición	El usuario PuestoMóvil debe de haber seleccionado una sala.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario PuestoMóvil quiere conocer la ruta hacia una sala.
	2	El usuario PuestoMóvil selecciona una sala.
	3	El sistema manda una petición al servidor de coordenadas para obtener la posición (x,y) del usuario PuestoMóvil.
	4	El sistema recoge la posición (x,y) del usuario PuestoMóvil actual.
	5	El sistema recoge el nombre de la sala seleccionada.
	6	El sistema manda una petición al servidor de mapas para obtener la sala donde está el usuario PuestoMóvil.
	7	El sistema almacena la sala del usuario PuestoMóvil y la sala destino para el cálculo de rutas.
	8	Se realiza el caso de uso Calcular Ruta (UC-004)
Poscondición	Se muestra una imagen con la ruta y su información.	
Flujo alternativo	**Paso**	**Acción**
	5	Si el usuario PuestoMóvil no ha seleccionado una sala, se muestra un mensaje de error al usuario.
	8	Si la sala del usuario PuestoMóvil y la sala destino es la misma, se manda un mensaje de aviso al usuario informándole de la situación.

Tabla 4.12. Descripción del caso de uso Calcular ruta a una sala (UC-007)

UC-008	Calcular ruta predefinida	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario desee consultar una ruta predefinida del sistema y durante la realización del siguiente caso de uso: • Conocer siguiente información de la ruta (UC-009)	
Precondición	El usuario está conectado a la página de la aplicación.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario solicita al sistema comenzar el proceso de cálculo de ruta estática.
	2	Se realiza el caso de uso Ver rutas predefinidas (UC-010)
	3	El usuario selecciona una ruta predefinida.
	4	El sistema recoge el nombre de la ruta seleccionada.
	5	El sistema almacena el nombre de la ruta selecciona para el cálculo de rutas.
	6	Se realiza el caso de uso Calcular Ruta (UC-004)
Poscondición	Se muestra la ruta a seguir en forma de asistente	
Flujo alternativo	**Paso**	**Acción**
	4	Si el usuario no ha seleccionado una ruta, este caso de uso queda sin efecto.

Tabla 4.13. Descripción del caso de uso Calcular ruta predefina (UC-008)

UC-009	Conocer siguiente información de la ruta	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario desee consultar una ruta predefinida en forma de asistente.	
Precondición	El usuario está conectado a una página de la aplicación de cálculo de rutas estáticas.	
Secuencia normal	**Paso**	**Acción**
	1	Se realiza el caso de uso Calcular ruta predefinida (UC-008)
	2	El usuario selecciona la opción de conocer siguiente paso de la ruta predefinida mostrada en pantalla.
	3	El sistema consulta la información de la ruta para el siguiente paso.
	4	El sistema manda una petición al servidor de mapas para obtener las imágenes asociadas.
	5	El sistema muestra la información junto con imágenes, entre dos paradas importantes.
Poscondición	Se conoce la información de la ruta predefinida, cada dos paradas importantes.	
Flujo alternativo	**Paso**	**Acción**
	5	Si se ha llegado al final de la ruta se le indica al usuario.

Tabla 4.14. Descripción del caso de uso Conocer siguiente información de la ruta (UC-009)

UC-010	Ver rutas predefinidas	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso durante la realización del siguiente caso de uso: • Calcular ruta predefinida (UC-008)	
Precondición	El usuario está conectado a la página, y desea conocer que rutas predefinidas existen.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario solicita al sistema ver rutas importantes.
	2	El sistema recoge la petición solicitada.
	3	El sistema muestra la información solicitada en forma de listado.
Poscondición	El usuario puede consultar las rutas que puede visitar.	
Flujo alternativo	**Paso**	**Acción**
	3	Si el sistema no encuentra la información solicitada, muestra un mensaje de aviso al usuario.

Tabla 4.15. Descripción del caso de uso Ver rutas predefinidas (UC-010)

En las figuras siguientes se muestran los diagramas de actividad correspondientes a los casos de uso Consultar mi posición (UC-002), Consultar sala actual (UC-003), Calcular ruta (UC-004), Calcular ruta a un objeto (UC-005), Mostrar información de un pasillo (UC-006), Conocer siguiente información de la ruta (UC-009), que son los casos de uso que pueden resultar más difícil de comprender.

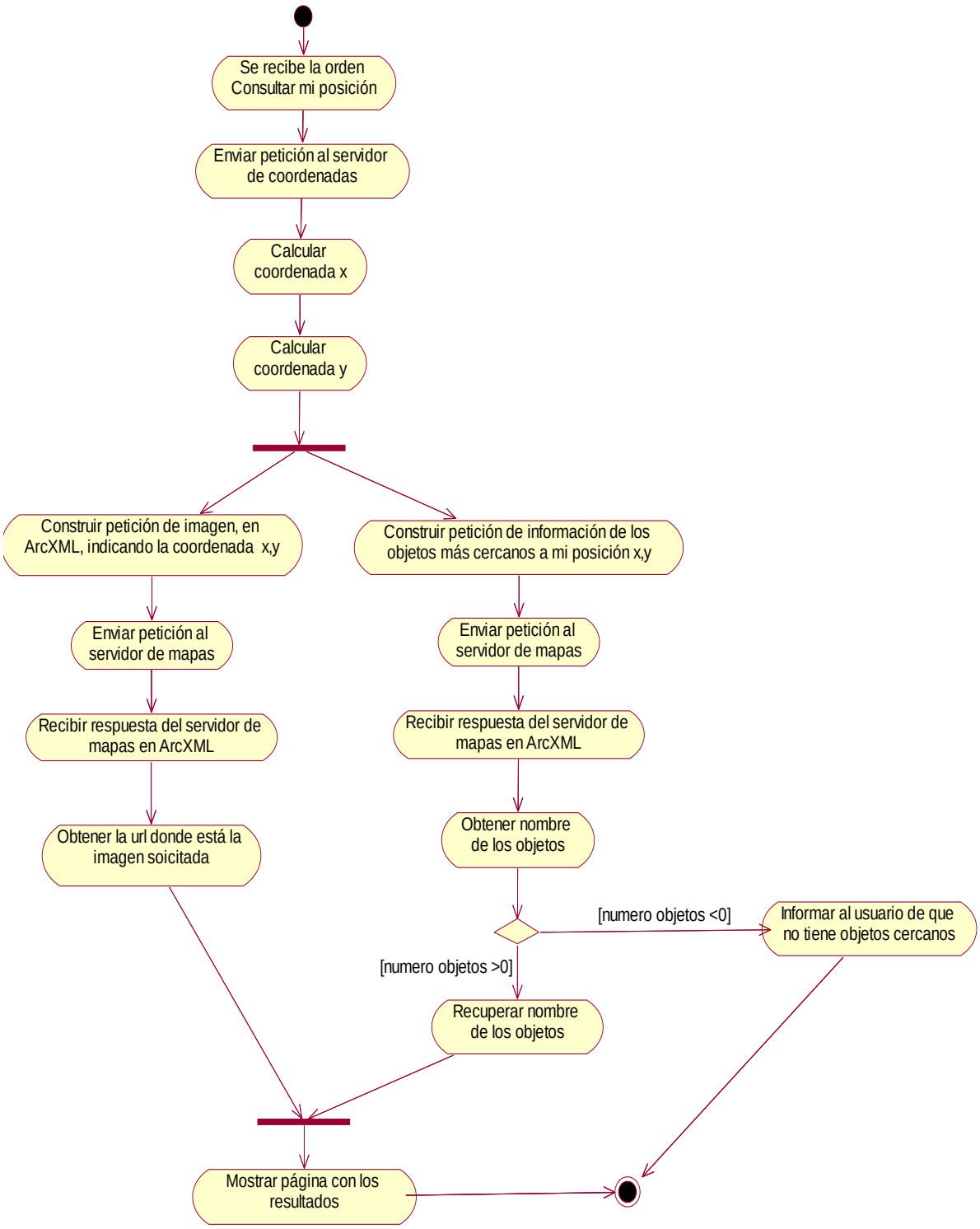


Figura 4.3. Diagrama de actividad correspondiente al caso de uso Consultar mi posición (UC-002)

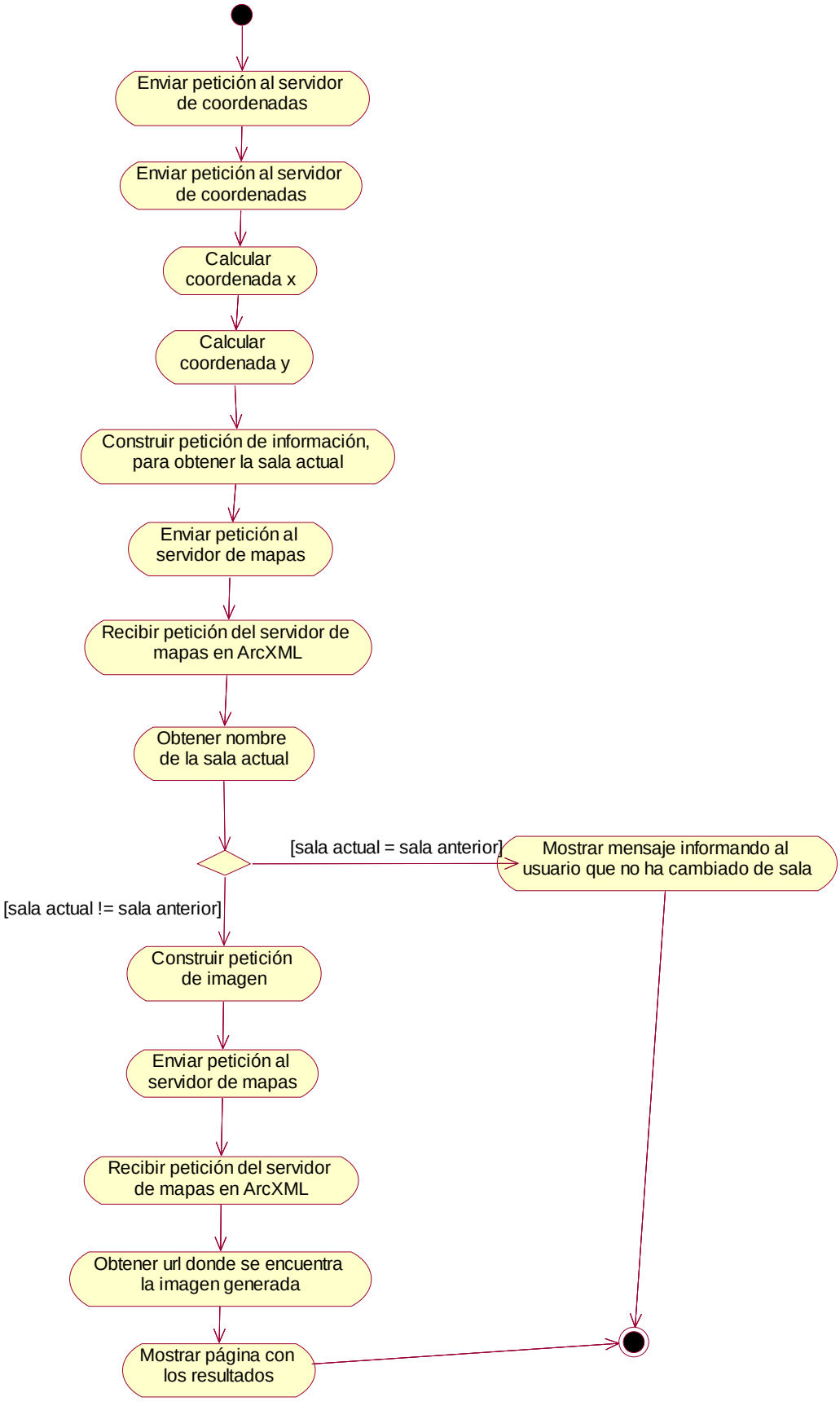


Figura 4.4. Diagrama de actividad correspondiente al caso de uso Consultar Sala Actual (UC-003)

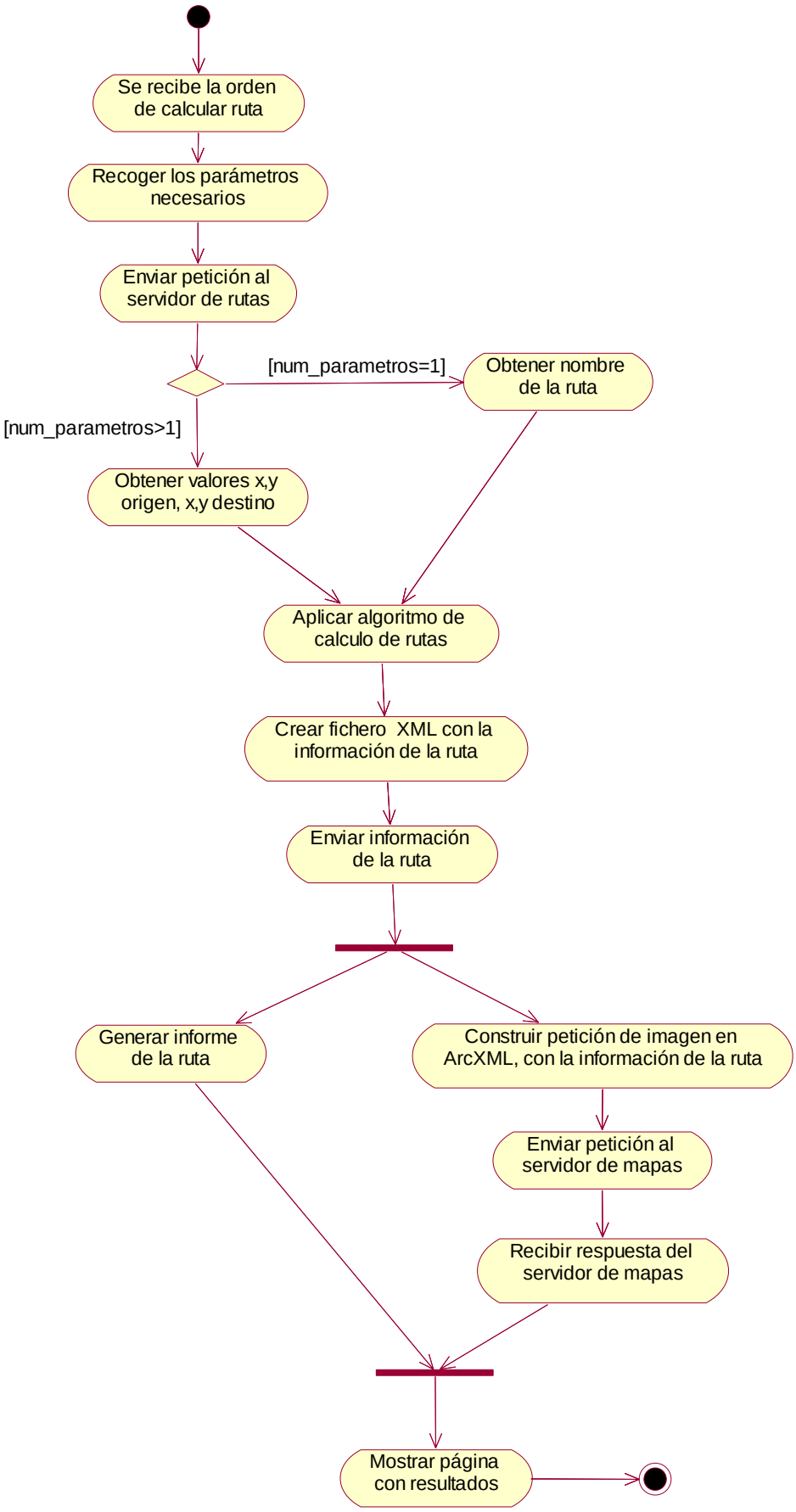


Figura 4.5. Diagrama de actividad correspondiente al caso de uso Calcular Ruta (UC-004)

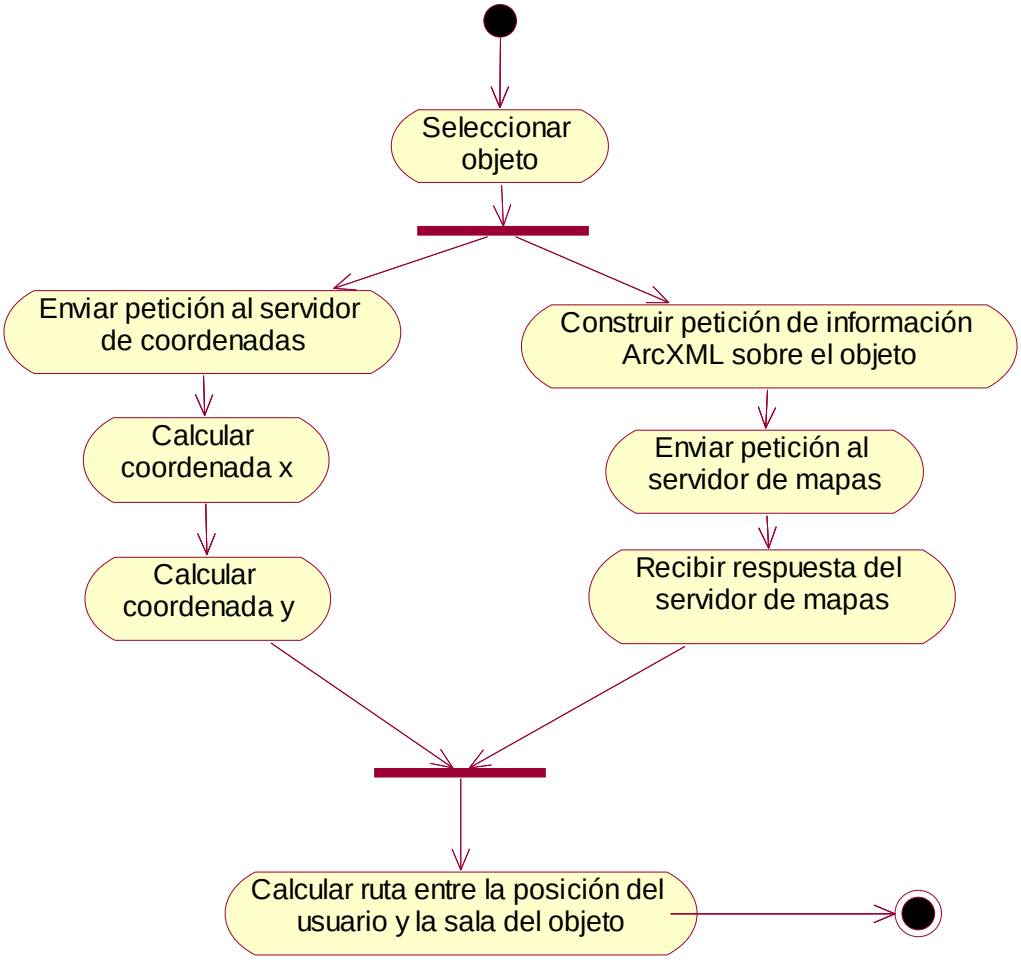


Figura 4.6. Diagrama de actividad correspondiente al caso de uso Calcular ruta a un objeto (UC-005)

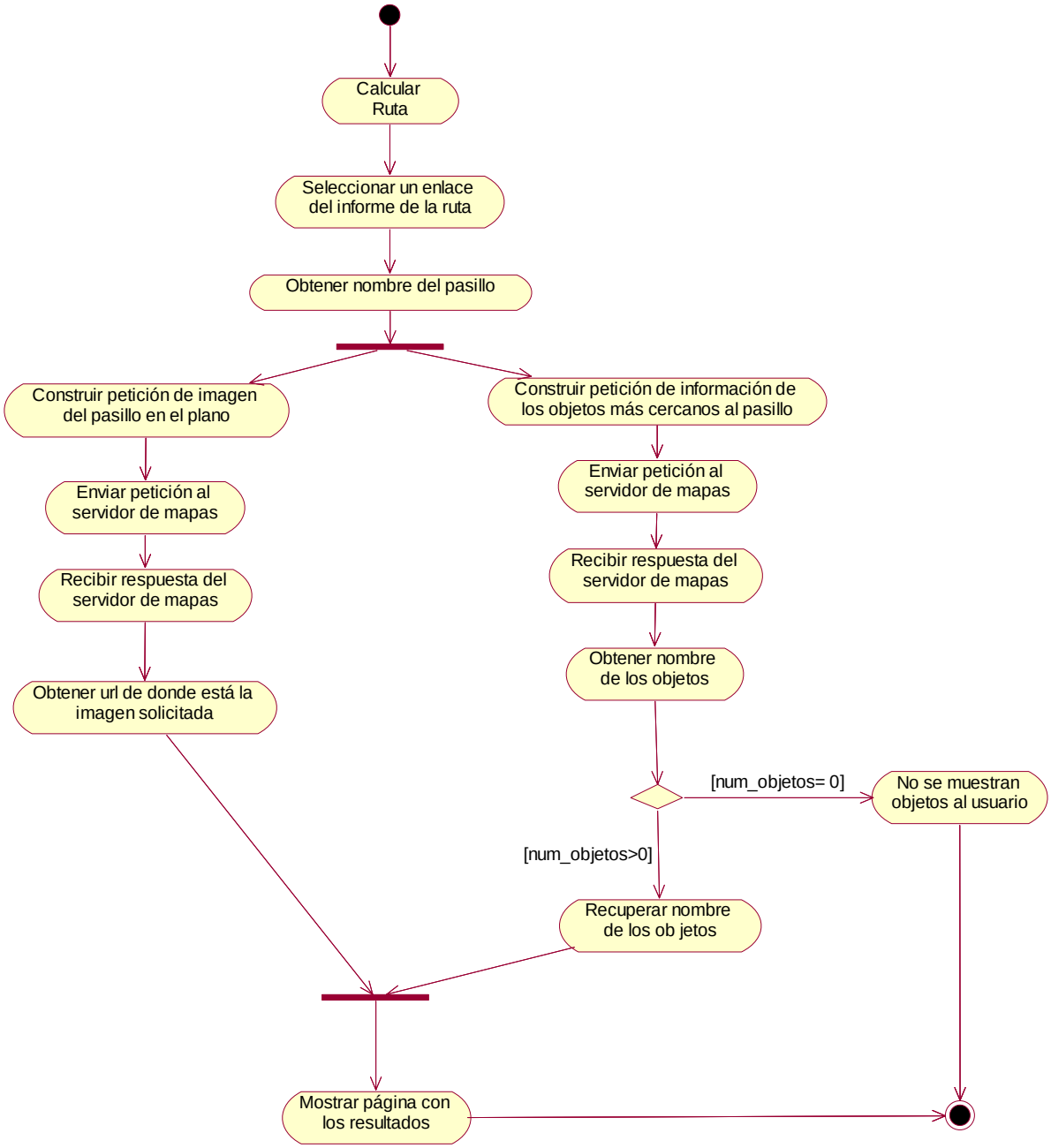


Figura 4.7. Diagrama de actividad correspondiente al caso de uso Mostrar información de un pasillo (UC-006)

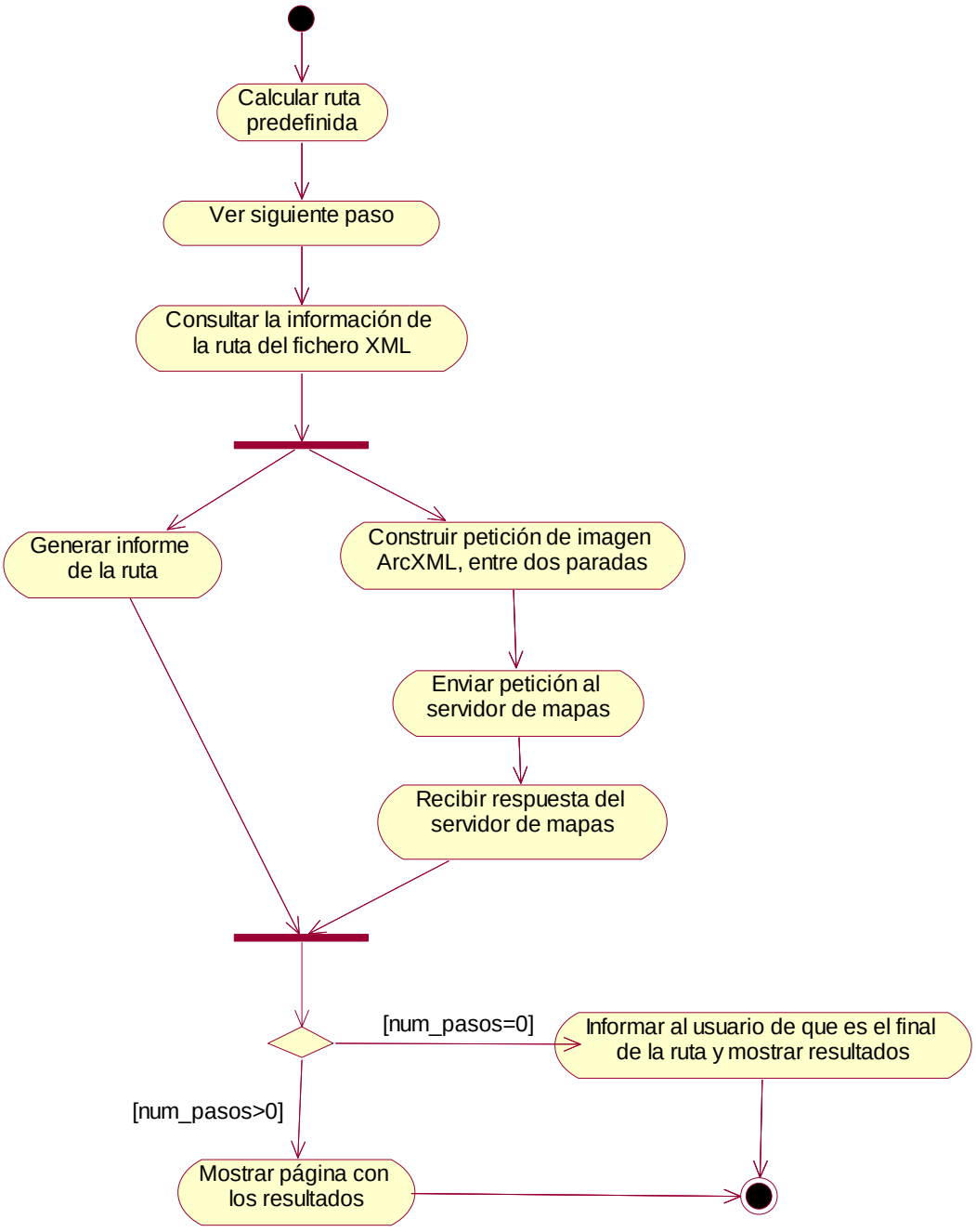


Figura 4.8. Diagrama de actividad correspondiente al caso de uso Conocer siguiente información de la ruta (UC-009)

4.6 Requisitos funcionales para dispositivo fijo

En la figura 4.9 se puede ver el diagrama de casos de uso para el caso particular de dispositivo fijo y desde la tabla 4.16 hasta la 4.27 se muestra las correspondientes descripciones para dichos casos de uso

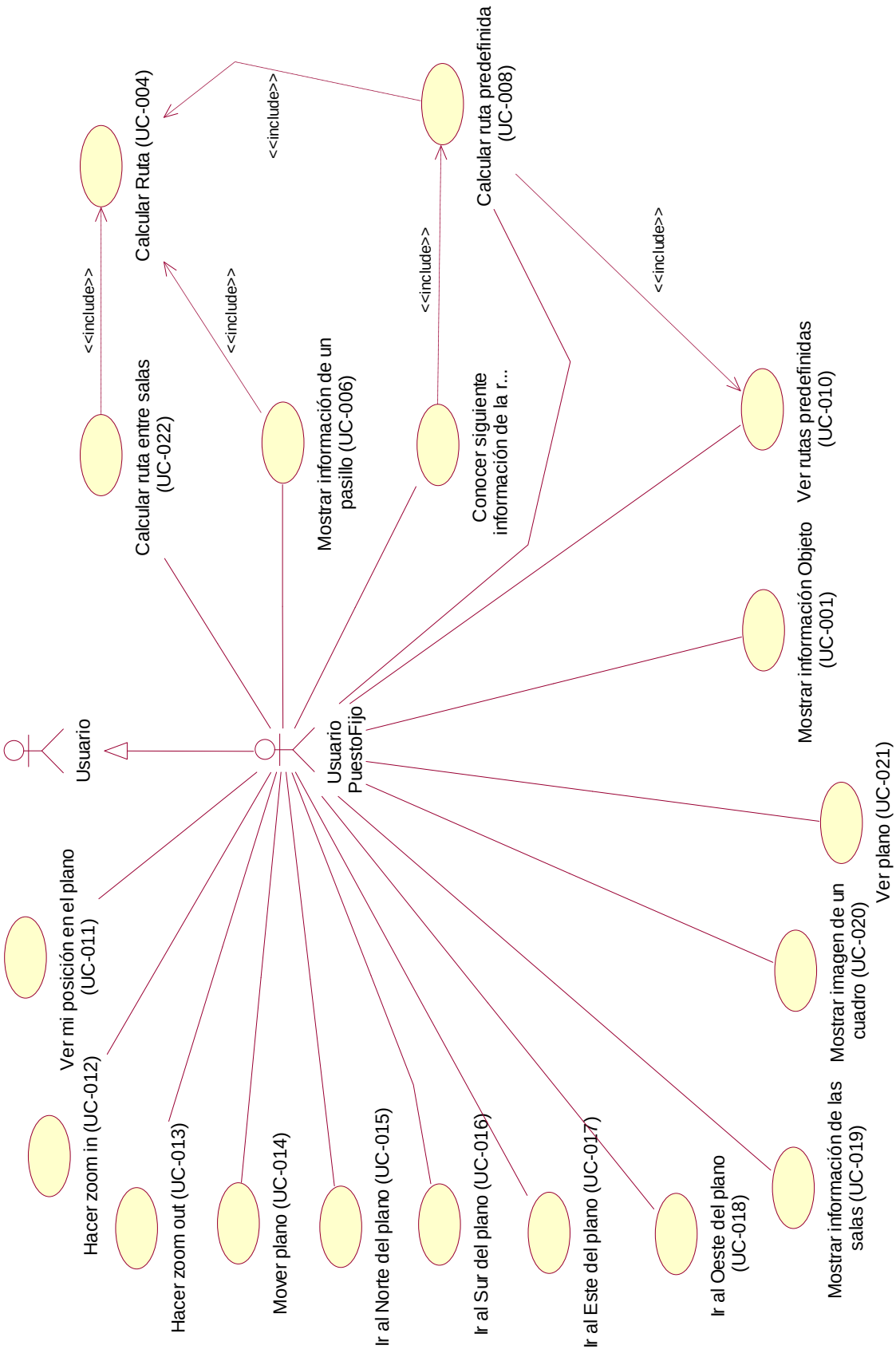


Figura 4.9. Diagrama de casos de uso para el puesto fijo

UC-011	Ver mi posición en el plano	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario PuestoFijo desee conocer la sala en la que se encuentra actualmente.	
Precondición	El sistema debe de tener almacenado la sala donde se encuentra el PuestoFijo.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario PuestoFijo solicita al sistema comenzar el proceso de ver posición actual en el plano.
	2	El sistema almacena la sala donde está el PuestoFijo.
	3	El sistema manda una petición al servidor de mapas para obtener la imagen del plano, indicando la sala actual.
	4	El sistema muestra al usuario PuestoFijo el plano, resaltando la sala actual donde se encuentra.
Poscondición	Se muestra una imagen del sistema indoor.	
Flujo alternativo	**Paso**	**Acción**
	2	Si el sistema no tiene almacenado donde está el PuestoFijo, se mostrará un mensaje de aviso al usuario.

Tabla 4.16. Descripción del caso de uso Ver mi posición en el plano (UC-011)

UC-012	Hacer zoom in	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario PuestoFijo desee ampliar la imagen del plano.	
Precondición	El usuario PuestoFijo está conectado a la página inicial	
Secuencia normal	**Paso**	**Acción**
	1	El usuario PuestoFijo solicita al sistema comenzar con el proceso de acercar plano.
	2	El usuario PuestoFijo selecciona con el ratón una parte del plano que quiere ampliar.
	3	El sistema recoge la posición (x,y) donde el usuario ha seleccionado.
	4	El sistema manda una petición al servidor de mapas, indicándole la zona del plano que quiere ampliar.
	5	El servidor de mapas le responde con una imagen.
	6	El sistema muestra al usuario PuestoFijo el plano ampliado, junto con los objetos que se pueden ver en esa zona.
Poscondición	El usuario PuestoFijo ha ampliado una zona del plano.	
Flujo alternativo	**Paso**	**Acción**
	2	Si el usuario PuestoFijo no selecciona una zona del plano, el caso de uso queda sin efecto.
	6	Si no hay objetos en esa zona del plano, sólo se mostrará el plano sin los objetos.

Tabla 4.17. Descripción del caso de uso Hacer zoom in (UC-012)

UC-013	Hacer zoom out	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario PuestoFijo desee ver una imagen más general del plano.	
Precondición	El usuario PuestoFijo está conectado a la página inicial.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario PuestoFijo solicita al sistema comenzar con el proceso de alejar plano.
	2	El usuario PuestoFijo selecciona una parte del plano.
	3	El sistema recoge la posición (x,y) que el usuario ha seleccionado con la ayuda del ratón.
	4	El sistema manda una petición al servidor de mapas, indicándole la zona del plano que quiere alejar.
	5	El servidor de mapas le responde con una imagen.
	6	El sistema muestra al usuario PuestoFijo el plano, junto con los objetos que se pueden ver en esa zona.
Poscondición	El usuario PuestoFijo ve una imagen más alejada del plano, de la mostrada actualmente.	
Flujo alternativo	**Paso**	**Acción**
	2	Si el usuario PuestoFijo no selecciona una zona del plano, el caso de uso queda sin efecto.
	6	Si no hay objetos en esa zona del plano, sólo se mostrará el plano sin los objetos.

Tabla 4.18. Descripción del caso de uso Hacer zoom out (UC-013)

UC-014	Mover plano	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario PuestoFijo desee mover el plano.	
Precondición	El usuario PuestoFijo está conectado a la página inicial.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario PuestoFijo solicita al sistema comenzar con el proceso de mover plano.
	2	El usuario PuestoFijo selecciona una zona del plano y arrastra hacia la posición donde se quiere desplazar.
	3	El sistema recoge la zona seleccionada por el usuario.
	4	El sistema manda una petición al servidor de mapas, indicándole la zona del plano que quiere mostrar.
	5	El servidor de mapas le responde con una imagen.
	6	El sistema muestra al usuario PuestoFijo el plano, junto con los objetos que se pueden ver en esa zona.
Poscondición	El usuario PuestoFijo ha movido el plano hacia una posición	
Flujo alternativo	**Paso**	**Acción**
	2	Si el usuario PuestoFijo no selecciona una zona del plano, el caso de uso queda sin efecto.
	6	Si no hay objetos en esa zona del plano, sólo se mostrará el plano sin los objetos.

Tabla 4.19. Descripción del caso de uso Mover plano (UC-014)

UC-015	Ir al Norte del plano	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario PuestoFijo desee ver la zona Norte del plano mostrado actualmente.	
Precondición	El usuario PuestoFijo está conectado a la página inicial	
Secuencia normal	**Paso**	**Acción**
	1	El usuario PuestoFijo solicita al sistema comenzar con el proceso de ver zona norte del plano.
	2	El sistema tiene almacenada la extensión del plano actual.
	3	El sistema manda una petición al servidor de mapas, indicándole la zona del plano que quiere mostrar, a partir de la zona actual.
	4	El servidor de mapas le responde con una imagen con esa extensión.
	5	El sistema muestra al usuario PuestoFijo el plano, junto con los objetos que se pueden ver en esa zona.
Poscondición	El usuario PuestoFijo ve la zona Norte del plano actual.	
Flujo alternativo	**Paso**	**Acción**
	5	Si no hay objetos en esa zona del plano, sólo se mostrará el plano sin los objetos.

Tabla 4.20. Descripción del caso e uso Ir al Norte del plano (UC-015)

UC-016	Ir al Sur del plano	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario PuestoFijo desee ver la zona Sur del plano mostrado actualmente.	
Precondición	El usuario PuestoFijo está conectado a la página inicial	
Secuencia normal	**Paso**	**Acción**
	1	El usuario PuestoFijo solicita al sistema comenzar con el proceso de ver zona sur del plano.
	2	El sistema tiene almacenada la extensión del plano actual.
	3	El sistema manda una petición al servidor de mapas, indicándole la zona del plano que quiere mostrar, a partir de la zona actual.
	4	El servidor de mapas le responde con una imagen con esa extensión.
	5	El sistema muestra al usuario PuestoFijo el plano, junto con los objetos que se pueden ver en esa zona.
Poscondición	El usuario PuestoFijo ve la zona Sur del plano actual.	
Flujo alternativo	**Paso**	**Acción**
	5	Si no hay objetos en esa zona del plano, sólo se mostrará el plano sin los objetos.

Tabla 4.21. Descripción del caso e uso Ir al Sur del plano (UC-016)

UC-017	Ir al Este del plano	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario PuestoFijo desee ver la zona Este del plano mostrado actualmente.	
Precondición	El usuario PuestoFijo está conectado a la página inicial	
Secuencia normal	**Paso**	**Acción**
	1	El usuario PuestoFijo solicita al sistema comenzar con el proceso de ver zona Este del plano.
	2	El sistema tiene almacenada la extensión del plano actual.
	3	El sistema manda una petición al servidor de mapas, indicándole la zona del plano que quiere mostrar, a partir de la zona actual.
	4	El servidor de mapas le responde con una imagen con esa extensión.
	5	El sistema muestra al usuario PuestoFijo el plano, junto con los objetos que se pueden ver en esa zona.
Poscondición	El usuario PuestoFijo ve la zona Este del plano actual.	
Flujo alternativo	**Paso**	**Acción**
	5	Si no hay objetos en esa zona del plano, sólo se mostrará el plano sin los objetos.

Tabla 4.22. Descripción del caso e uso Ir al Este del plano (UC-017)

UC-018	Ir al Oeste del plano	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario PuestoFijo desee ver la zona Oeste del plano mostrado actualmente.	
Precondición	El usuario PuestoFijo está conectado a la página inicial	
Secuencia normal	**Paso**	**Acción**
	1	El usuario PuestoFijo solicita al sistema comenzar con el proceso de ver zona Oeste del plano.
	2	El sistema tiene almacenada la extensión del plano actual.
	3	El sistema manda una petición al servidor de mapas, indicándole la zona del plano que quiere mostrar, a partir de la zona actual.
	4	El servidor de mapas le responde con una imagen con esa extensión.
	5	El sistema muestra al usuario PuestoFijo el plano, junto con los objetos que se pueden ver en esa zona.
Poscondición	El usuario PuestoFijo ve la zona Oeste del plano actual.	
Flujo alternativo	**Paso**	**Acción**
	5	Si no hay objetos en esa zona del plano, sólo se mostrará el plano sin los objetos.

Tabla 4.23. Descripción del caso e uso Ir al Norte del plano (UC-018)

UC-019	Mostrar información de las salas	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario PuestoFijo desee consultar la información de las salas.	
Precondición	El usuario PuestoFijo está conectado a la página inicial.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario PuestoFijo solicita al sistema comenzar con el proceso de identificación de Salas.
	2	El usuario PuestoFijo selecciona una Sala del plano.
	3	El sistema recoge la zona (x,y) seleccionada por el usuario PuestoFijo.
	4	El sistema manda una petición al servidor de mapas, preguntando por la sala seleccionada.
	5	El servidor de mapas le responde con toda la información de la Sala.
	6	El sistema muestra al usuario la información de la sala.
Poscondición	Se muestra la información de la sala.	
Flujo alternativo	**Paso**	**Acción**
	2	Si el usuario no ha seleccionado una sala, se muestra un mensaje de aviso al usuario.

Tabla 4.24. Descripción del caso de uso Mostrar información de las salas (UC-019)

UC-020	Mostrar imagen de un cuadro	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario PuestoFijo desee conocer la imagen de un objeto.	
Precondición	El usuario PuestoFijo está conectado a la página inicial.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario PuestoFijo solicita al sistema comenzar con el proceso de mostrar imagen de un objeto.
	2	El usuario PuestoFijo selecciona un objeto del plano
	3	El sistema recoge la zona x,y seleccionada por el usuario PuestoFijo.
	4	El sistema manda una petición al servidor de mapas, preguntándole por el objeto.
	5	El servidor de mapas le responde con el nombre del objeto y su imagen.
	6	El sistema muestra la imagen del objeto seleccionado por el usuario.
Poscondición	Se muestra la imagen del objeto.	
Flujo alternativo	**Paso**	**Acción**
	3	Si el usuario no ha seleccionado un objeto, se muestra un mensaje de aviso al usuario.

Tabla 4.25. Descripción del caso de uso Mostrar imagen de un cuadro (UC-020)

UC-021	Ver plano	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario PuestoFijo desee consultar el plano y realizar las consultas básicas.	
Precondición	El usuario no está en la página inicial.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario PuestoFijo solicita al sistema comenzar el proceso de ver plano completo.
	2	El sistema recoge la petición de ver plano.
	3	El sistema pide la página inicial.
	4	El sistema muestra la página inicial al usuario PuestoFijo
Poscondición	Se muestra la imagen del plano, junto con las opciones para realizar consultas básicas.	

Tabla 4.26. Descripción del caso de uso Ver plano (UC-021)

UC-022	Calcular ruta entre salas	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el usuario PuestoFijo desee conocer la ruta entre dos salas.	
Precondición	El usuario debe de estar conectado a la página de la aplicación.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario PuestoFijo solicita al sistema comenzar el proceso de calcular ruta entre salas.
	2	El usuario PuestoFijo selecciona la sala origen.
	3	El usuario PuestoFijo selecciona la sala destino.
	4	El sistema recoge el nombre de la sala origen seleccionado.
	5	El sistema recoge el nombre de la sala destino seleccionado.
	6	El sistema almacena la sala origen y la sala destino para el cálculo de rutas.
	7	Se realiza el caso de uso Calcular Ruta (UC-004)
Poscondición	Se muestra una imagen con la ruta y su información	
Flujo alternativo	**Paso**	**Acción**
	4	Si el usuario PuestoFijo no ha selecciona una sala origen, se muestra un mensaje de aviso al usuario.
	5	Si el usuario PuestoFijo no ha selecciona una sala destino, se muestra un mensaje de aviso al usuario.
	7	Si la sala origen y la sala destino es la misma, se manda un mensaje de aviso al usuario informándole de la situación, y a continuación el caso de uso queda sin efecto.

Tabla 4.27. Descripción del caso de Calcular ruta entre salas (UC-022)

En las figuras siguientes se muestran los diagramas de actividad correspondientes a los casos de uso Hacer *zoom in* (UC-012) y Calcular ruta entre salas (UC-022).

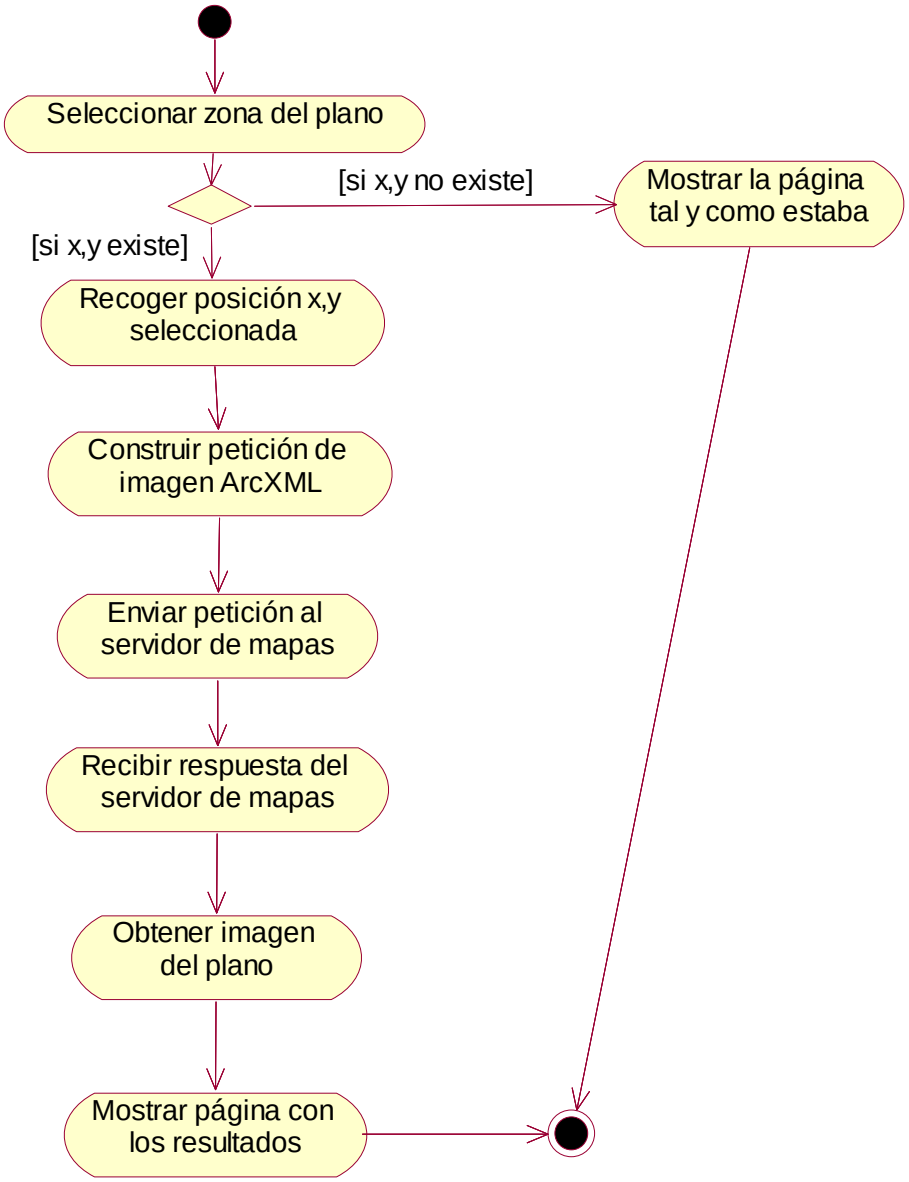


Figura 4.10. Diagrama de actividad correspondiente al caso de uso Hacer zoom in (UC-012)

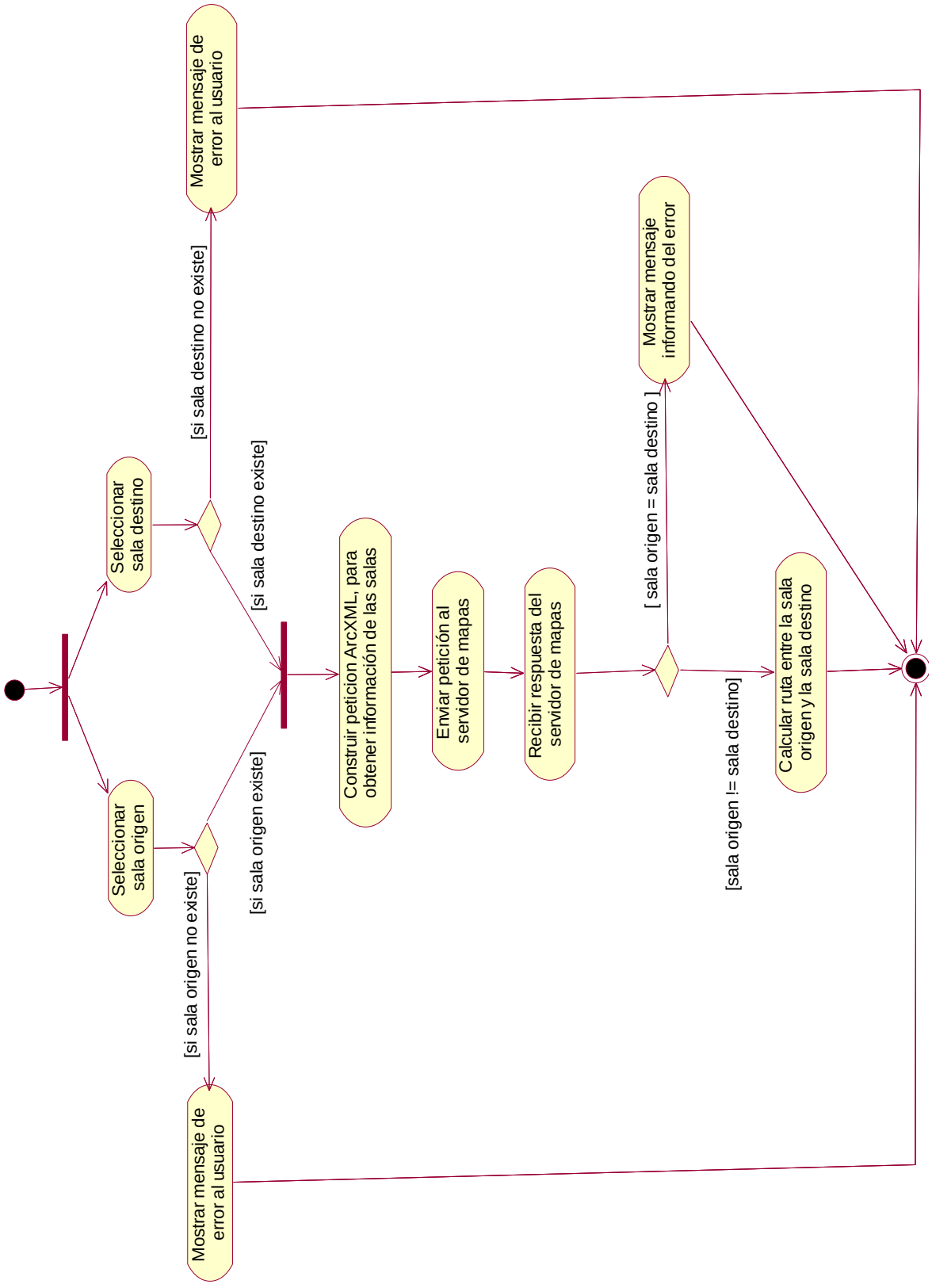


Figura 4.11. Diagrama de actividad correspondiente al caso de uso Calcular ruta entre salas (UC-022)

Una vez que ya tenemos definido, la funcionalidad de nuestro sistema, a continuación pasaremos a definir los requisitos no explícitamente funcionales.

4.7 Requisitos no explícitamente funcionales

Como requisito no explícitamente funcional se puede hablar de usabilidad, ya que este requisito se debe incluir en el proceso de desarrollo software desde el comienzo del mismo.

NFR-01	Usabilidad
Descripción	Requisito que aportará al sistema una mayor facilidad de uso y de aprendizaje para los usuarios.

Tabla 4.28. Requisito no funcional. Usabilidad (NFR-01)

Se elige un modelo de calidad basado en criterios ergonómicos y combinado con estándares internacionales (Tabla 3.3), a continuación se muestra cuales de estos criterios se han considerado:

- Criterios relacionados con la Understandability:
    - Legibility: todas las pantallas tienen títulos de la sección donde se encuentra, con un estilo de letra legible y centrada.
    - Promting: facilidad para hacer saber al usuario donde se encuentra, con títulos explicativos y texto informando con las distintas opciones a las que puede ir.
    - Helpfulness: ayudas contextuales en las cabeceras de todas las pantallas donde sea necesario, con una explicación del funcionamiento de la sección en la que se encuentra.
- Criterios relacionados con la Learnability:
    - El diseño de las pantallas seguido en todas las secciones es el mismo, con una clara separación entre los elementos de acción y de información, de manera que el usuario consiga aprender y recordar el funcionamiento del producto. La estructura de todas las ventanas será la siguiente:
        - Título: título de la aplicación.
        - Ayuda contextual: ayuda explicativa de la sección.

- Descripción de la consulta: Controles necesarios para describir la consulta solicitada, como imágenes, texto, etc.

- Botonera: en esta zona se colocan las distintas acciones a las que se puede acceder desde esta sección.

En la figura 4.12 se puede ver un esquema de la distribución de información en la PDA.

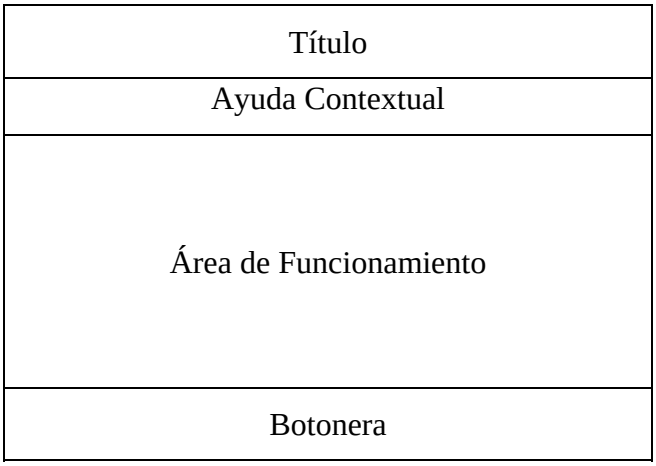


Figura 4.12. Patrón de diseño para las pantallas de la IU para PDA

- Density: densidad de la información mostrada, una de las características de los dispositivos móviles como la PDA es el reducido espacio de pantalla con el que se cuenta, es importante no aportar un exceso de información en pantalla, ya que sería ilegible y molesto.

- Minimal Action: la aplicación reduce el número de acciones mínimas para acometer tareas, por ejemplo, a la hora de realizar el cálculo de rutas el sistema coge la posición del usuario automáticamente.

- Criterios relacionados con la Understandability:

  - Error protection: gestión de errores, ante cualquier error en la aplicación como el acceso al servidor, la aplicación mostrará una descripción textual del mismo.

Una vez que la fase de análisis de requisitos está terminada, se tiene una serie de funcionalidades que se tiene que cumplir y una serie de pautas de calidad para conseguir una interfaz más usable.

4.8 DISEÑO DEL SISTEMA

Con la información obtenida en el apartado anterior, se diseñará un modelado conceptual del caso de estudio, mostrando un diagrama de clases.

4.8.1 Diagrama de clases

El diagrama de clases que va a tener el sistema se puede ver en la figura 4.13. La descripción de las clases se puede ver desde la tabla 4.29 a la tabla 4.42.

Propiedad	Descripción
Nombre:	Ruta
Descripción:	Representa la entidad Ruta
Responsabilidades:	Almacenar la información de una ruta.
Relaciones:	Relación con las clases Mapa, Red, Ruta Estática y Ruta Dinámica
Atributos:	• Imagen_Ruta • Leyenda • Información_Ruta
Operaciones:	• Ver_Objetos() • Hacer_Zoom_Ruta() • Ver_Posicion() • Ver_Ruta()

Tabla 4.29. Clase Ruta

Propiedad	Descripción
Nombre:	Ruta Estática
Descripción:	Representa la entidad Ruta Estática
Responsabilidades:	Almacenar la información de una ruta estática.
Relaciones:	Relación con la clase Ruta.
Atributos:	• Nombre Ruta • ZoomRuta • Plano • ListadoObjetosRuta
Operaciones:	• Seleccionar_Ruta() • Calcular_Ruta() • Ver_Siguiente_Paso() • Ver_Información()

Tabla 4.30. Clase Ruta Estática

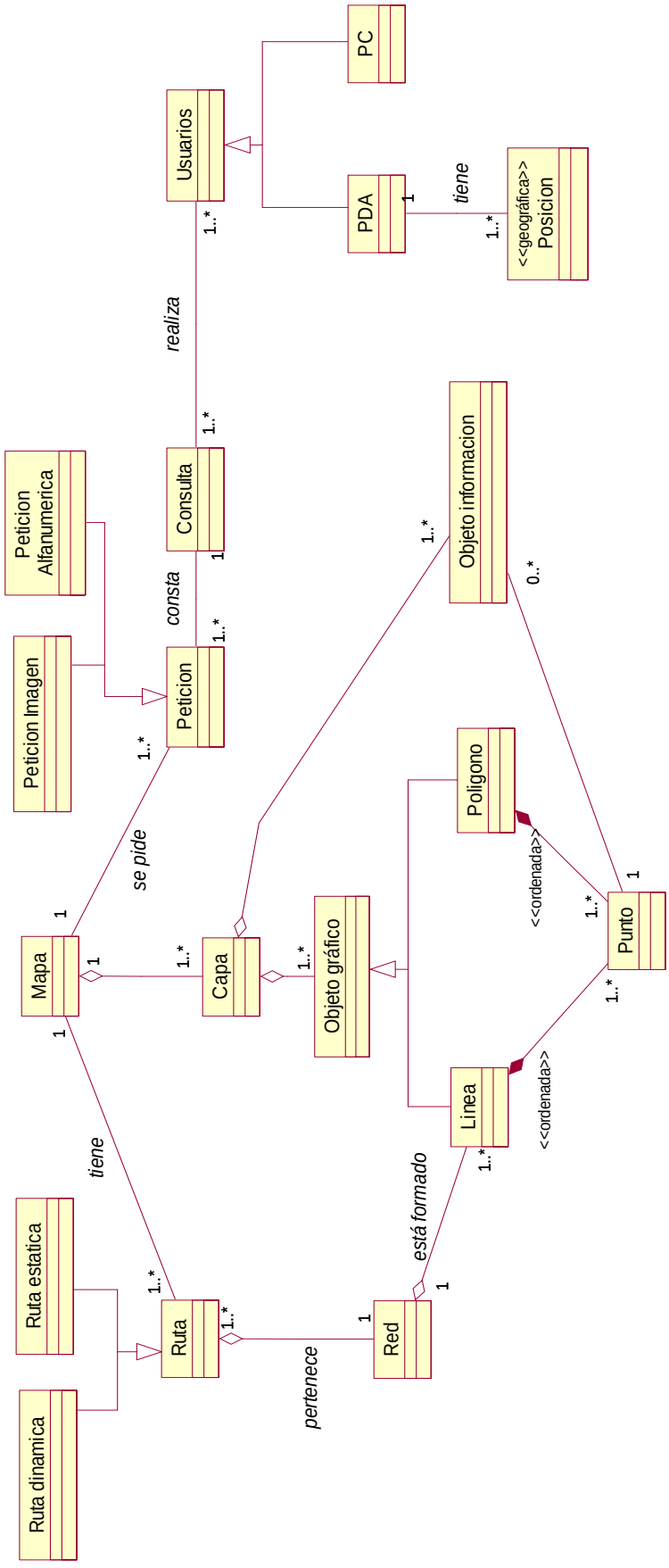


Figura 4.13. Diagrama de clases.

Propiedad	Descripción
Nombre:	Ruta Dinámica
Descripción:	Representa la entidad Ruta Dinámica
Responsabilidades:	Almacenar la información de una ruta dinámica.
Relaciones:	Relación con la clase Ruta
Atributos:	• Parada_Inicial • Parada_Final
Operaciones:	• Calcular_Ruta()

Tabla 4.31. Clase Ruta Dinámica

Propiedad	Descripción
Nombre:	Objeto Información
Descripción:	Representa la entidad Objeto Información
Responsabilidades:	Almacenar toda la información de los objetos
Relaciones:	Relación con las clases Capa y Punto.
Atributos:	• Título • Autor • Año • Dimensiones • Técnicas • ImagenObjeto • Plano
Operaciones:	• Ver_Informacion() • Ver_Posicion_Objeto() • Calcular_Ruta_Objeto()

Tabla 4.32. Clase Objeto Información

Propiedad	Descripción
Nombre:	Mapa
Descripción:	Representa la entidad Mapa.
Responsabilidades:	Almacena toda la información de un mapa completo.
Relaciones:	Relación con las clases Capa, Ruta y Petición.
Atributos:	• Nombre del mapa • Imagen • Información.
Operaciones:	• Obtener_imagen_mapa() • Obtener_informacion_mapa()

Tabla 4.33. Clase Mapa

Propiedad	Descripción
Nombre:	Capa
Descripción:	Representa la entidad Capa
Responsabilidades:	Almacena la información de las capas. Una capa está compuesta por objetos gráficos.
Relaciones:	Relación con la clase Mapa y Objeto gráfico.
Atributos:	• Nombre • Identificador
Operaciones:	• Consultar_Capa() • Ver_Imagen_Capa()

Tabla 4.34. Clase Capa

Propiedad	Descripción
Nombre:	Objeto Gráfico
Descripción:	Representa a la entidad Objeto Gráfico, que puede ser un polígono o línea.
Responsabilidades:	Almacena información sobre un Objeto gráfico (Línea, Punto o Polígono)
Relaciones:	Relación con las clases Capa, Línea y Polígono.
Atributos:	• Nombre • Tipo
Operaciones:	• Ver_Objeto_Gráfico()

Tabla 4.35. Clase Objeto Gráfico

Propiedad	Descripción
Nombre:	Polígono
Descripción:	Representa la entidad Polígono
Responsabilidades:	Almacena toda la información necesaria para formar un polígono.
Relaciones:	Relación con las clases Objeto Gráfico y Punto. Un Polígono se compone de Puntos.
Atributos:	• Valores_x • Valores_y
Operaciones:	• Obtener_Polígono()

Tabla 4.36. Clase Polígono

Propiedad	Descripción
Nombre:	Línea
Descripción:	Representa a la entidad Línea.
Responsabilidades:	Almacena toda información necesaria para formar una línea.
Relaciones:	Relación con las clases Objeto Gráfico y Punto. Una línea se compone de puntos.
Atributos:	• Valores_x • Valores_y
Operaciones:	• Obtener_Línea()

Tabla 4.37. Clase Línea

Propiedad	Descripción
Nombre:	Punto
Descripción:	Representa a la entidad Punto.
Responsabilidades:	Almacena toda la información de un punto.
Relaciones:	Relación con las clases Polígono, Línea y Objeto Información
Atributos:	• Valor_x • Valor_y
Operaciones:	• Obtener_x() • Obtener_y()

Tabla 4.38. Clase Punto

Propiedad	Descripción
Nombre:	Red
Descripción:	Representa a la entidad Red. Una red está formada por varias líneas.
Responsabilidades:	Almacena toda la información de la Red del sistema.
Relaciones:	Relación con las clases Ruta y Línea.
Atributos:	• Nombre
Operaciones:	• Obtener_red() • Consultar_red()

Tabla 4.39. Clase Red

Propiedad	Descripción
Nombre:	Usuarios
Descripción:	Representa a la entidad Usuarios
Responsabilidades:	Almacenar la información de los usuarios. Puede haber dos tipos de usuarios: PDA o PC
Relaciones:	Relación con las clases PDA, PC y consulta.
Atributos:	• Identificador
Operaciones:	• Consultar_Identificador()

Tabla 4.40. Clase Usuarios

Propiedad	Descripción
Nombre:	PDA
Descripción:	Representa a la entidad Usuario conectado a través de una PDA.
Responsabilidades:	Almacena la información de los usuarios conectados por un dispositivo móvil.
Relaciones:	Relación con las clases Usuarios y Posición,
Atributos:	• Coordenada x. • Coordenada y.
Operaciones:	• Realizar consultas() • Obtener_resultados()

Tabla 4.41. Clase PDA

Propiedad	Descripción
Nombre:	PC
Descripción:	Representa a la entidad Usuario conectado a través de un PC.
Responsabilidades:	Almacena la información de los usuarios conectados por un dispositivo fijo.
Relaciones:	Relación con la clase Usuarios.
Operaciones:	• Realizar_Consulta() • Obtener_resultados()

Tabla 4.42. Clase PC

Propiedad	Descripción
Nombre:	Posición
Descripción:	Representa a la entidad Posición.
Responsabilidades:	Almacena la posición de un usuario conectado a través de una PDA.
Relaciones:	Relación con la clase PDA.
Atributos:	• Coordenada.
Operaciones:	• Calcular_Posición() • Enviar_Posición()

Tabla 4.43. Clase Posición

Propiedad	Descripción
Nombre:	Consulta
Descripción:	Representa a la entidad Consulta.
Responsabilidades:	Almacena y procesa la consulta solicitada por el usuario.
Relaciones:	Relación con las clases Usuarios y Petición.
Atributos:	• Nombre de la consulta.
Operaciones:	• Recibir_Consulta() • Enviar_Consulta() • Procesar_Consulta() • Enviar_Resultados()

Tabla 4.44. Clase Consulta

Propiedad	Descripción
Nombre:	Petición
Descripción:	Representa a la entidad Petición.
Responsabilidades:	Almacena la información necesaria para procesar la petición. Que puede ser de dos tipos alfanumérica o de imagen.
Relaciones:	Relación con las clases Consulta, Mapa, Petición alfanumérica y Petición Imagen.
Atributos:	• Tipo de petición.
Operaciones:	• Realizar_petición() • Obtener_Resultados()

Tabla 4.45. Clase Petición

Propiedad	Descripción
Nombre:	Petición Imagen
Descripción:	Representa a la entidad Petición Imagen, para atender aquellas consultas que necesiten de una imagen para su resultado.
Responsabilidades:	Almacena la información necesaria para generar la petición de una imagen del mapa.
Relaciones:	Relación con la clase Petición.
Atributos:	• Datos_consulta.
Operaciones:	• Generar_Petición_Imagen() • Obtener_Imagen()

Tabla 4.46. Clase Petición Imagen

Propiedad	Descripción
Nombre:	Petición Alfanumérica
Descripción:	Representa a la entidad Petición Alfanumérica, para atender aquellas consultas que necesiten obtener información alfanumérica.
Responsabilidades:	Almacena la información necesaria para generar la petición de una consulta alfanumérica sobre el mapa.
Relaciones:	Relación con la clase Petición.
Atributos:	• Información_Consulta
Operaciones:	• Generar_Petición_Alfanumérica() • Obtener_Información()

Tabla 4.47. Clase Petición Alfanumérica

4.9 TECNOLOGÍA EMPLEADA

Atendiendo a la metodología de desarrollo de software empleada para llevar a cabo el proyecto, en este caso *eXtreme Programming (XP)*, una vez que se ha realizado la captura de requisitos y se tiene hecho el análisis funcional del sistema, se debe abordar la problemática de la tecnología a emplear para desarrollar el proyecto. Para ello, se ha realizado un estudio del estado del arte de las diferentes tecnologías que se deben emplear para realizar el proyecto, con el fin de encontrar las que mejor se adecuen a las necesidades del mismo. En este caso, el estudio se centra en los Servidores de Mapas en Internet y en los gestores de rutas que son las dos tecnologías que se aplicarán en el desarrollo del proyecto. Este estudio fue expuesto en capítulos anteriores (Capítulos 2 y 3) alterando la secuencia que propone la metodología utilizada, pero introduciendo, de esta forma al lector en la problemática planteada, y permitiéndole una mejor comprensión del presente documento.

En base al estudio realizado se ha optado por **Network Analyst** como tecnología a utilizar para el gestor de rutas y **ArcIMS** como Servidor de Mapas en Internet utilizado para poder lanzarle las consultas del usuario.

4.10 TAREAS DE ALTO NIVEL

A continuación se muestran diferentes tareas que se pueden realizar siguiendo los requisitos funcionales del sistema. Cada tarea se representa con el siguiente formato: una descripción de la tarea, parte de código ArcXML necesario y la descripción de la salida, junto con ejemplos de los valores obtenidos.

1 Nombre *Ver plano completo*	**Tipo** **Espacial**
Descripción	**Salida**
Se quiere obtener una visión general del sistema indoor.	Mapa del sistema indoor indicando el nombre de las salas.
Petición ArcXML	**Valores devueltos**
<?xml version=”1.0” encoding=”UTF-8”?> <ARCXML version=”1.1”> <REQUEST> <GET_IMAGE> <PROPERTIES> <LAYERLIST></LAYERLIST> **Listado de capas, activando las que se quieren mostrar** </PROPERTIES> <LAYER type=”featureclass” name=”**Nombre de las Capas**” id=”0”> **Características de la capa de Salas, indicando su nombre** </LAYER></GET_IMAGE> </REQUEST> </ARCXML>	 Imagen del sistema indoor inicial.

Tabla 4.48. Ver plano completo

2 Nombre *Salas del sistema indoor*	**Tipo** Alfanumérica
Descripción	**Salida**
Se quiere obtener todas las salas del sistema indoor.	Obtener el nombre de las salas que forman el sistema indoor en forma de lista.
Petición ArcXML	**Valores devueltos**
< ?xml version= »1.0 » ?> <ARCXML version= »1.1 »> <REQUEST> <GET_FEATURES > <LAYER id=”0” /> **Seleccionar el campo nombre de la capa Salas** <QUERY subfields=”NOMBRE” ></QUERY> </GET_FEATURES> </REQUEST></ARCXML>	Listado de todas las salas del sistema indoor. • SALA 1 • SALA 2 • …

Tabla 4.49. Salas del sistema indoor

3	**Nombre**	*Objetos del sistema indoor*	**Tipo**	Alfanumérica
Descripción			**Salida**	
Se quiere obtener todos los objetos del sistema indoor.			Obtener el nombre de todos los objetos del sistema indoor en una lista.	
Petición ArcXML			**Valores devueltos**	
<REQUEST> <GET_FEATURES outputmode=”xml” geometry=”false” envelope=”true” compact=”true”> <LAYER id=”2” /> **Seleccionar el campo Titulo de la capa de información más detallada** <QUERY subfields=”TITULO” > </QUERY> </GET_FEATURES> </REQUEST> </ARCXML>			Listado de todos los objetos del sistema indoor. • Arlequín • Cabeza de caballo • Retrato de Josette • …	

Tabla 4.50. Objetos del sistema indoor

4	**Nombre**	*Consulta sobre objetos por su nombre*	**Tipo**	Alfanumérica
Descripción			**Salida**	
Obtener información de un objeto identificado por su nombre seleccionado de una lista.			Obtener características del objeto, como título, autor, …	
Petición ArcXML			**Valores devueltos**	
< ?xml version= »1.0 » ?> <ARCXML version= »1.1 »> <REQUEST> <GET_FEATURES outputmode=”xml” geometry=”false” envelope=”true” compact=”true”> <LAYER id=”2” /> **Obtenemos del objeto seleccionado toda la información** <QUERY subfields=”#ALL#” where=”TITULO=’Arlequin’” > </QUERY> </GET_FEATURES> </REQUEST> </ARCXML>			Información textual del objeto, por ejemplo: • Titulo: La plaza Ravignan • Autor: Juan Gris • Fecha de creación: 1915 • Dimensiones: 116 x 89 cm • Técnica: Óleo sobre lienzo	

Tabla 4.51. Consulta sobre objetos por nombre

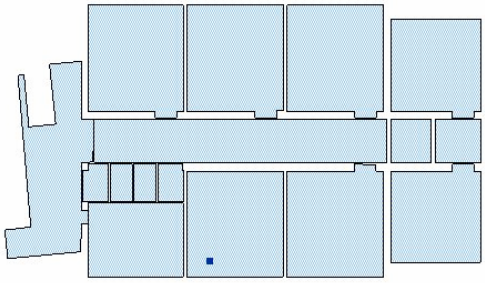
5	**Nombre** *Posición de objetos por nombre*	**Tipo**	Espacial
Descripción		**Salida**	
Obtener posición de un objeto identificado por su nombre seleccionado de una lista.		Obtener una imagen de su posición sobre el plano.	
Petición ArcXML		**Valores devueltos**	
< ?xml version= »1.0 » ?> <ARCXML version= »1.1 »> <REQUEST> <GET_IMAGE> <PROPERTIES> <LAYERLIST> **Listado de capas, activando las que se quieren mostrar** <LAYERDEF id=”2” visible=” ete” > **En la capa de objetos seleccionamos el objeto seleccionado**. <SPATIALQUERY where=”TITULO=’Arlequin’”/> </LAYERDEF> </LAYERLIST> </PROPERTIES> </GET_IMAGE> </REQUEST> </ARCXML>		 Mapa indicando de la capa de información más detallada, la posición donde se encuentra el objeto seleccionado.	

Tabla 4.52. Posición de objetos por nombre

6	**Nombre** *Consulta sobre los objetos más cercanos*	**Tipo**	Alfanumérica
Descripción		**Salida**	
Se está en una posición (x,y) y se quiere conocer los objetos más cercanos.		Obtener información de los objetos cercanos, identificados por su nombre.	
Petición ArcXML		**Valores devueltos**	
< ?xml version= »1.0 » ?> <ARCXML version= »1.1 »> <REQUEST> <GET_FEATURES outputmode=”xml” geometry=”false” envelope=”true” compact=”true”> <LAYER id=”2” /> <SPATIALQUERY> **Seleccionamos un entorno cerca de nuestra posición x,y, aproximadamente 1 metro.** <SPATIALFILTER relation=”area_intersection”> <ENVELOPE minx, miny, maxx, maxy /> </SPATIALFILTER> </SPATIALQUERY> </GET_FEATURES> </REQUEST> </ARCXML>		Se obtiene información textual de los objetos más cercanos a la posición actual, aproximadamente a un metro. Por ejemplo: • La mujer de azul • Muchacha de espaldas	

Tabla 4.53. Consulta sobre los objetos más cercanos

7	**Nombre**	*Mostrar posición de los objetos más cercanos*	**Tipo**	Espacial

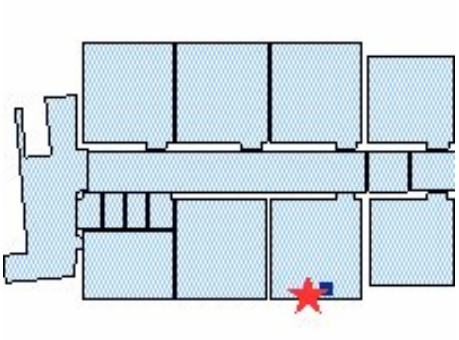
Descripción	**Salida**
Se está en una posición (x,y) y se quiere conocer los objetos más cercanos junto con la posición actual.	Obtener una imagen del mapa con la posición actual y los objetos más cercanos de las capas de información más detalladas.
Petición ArcXML	**Valores devueltos**
<?xml version=”1.0” encoding=”UTF-8”?> <ARCXML version=”1.1”> <REQUEST> <GET_IMAGE><PROPERTIES><LAYERLIST> **Listado de capas, activando las que se quieren mostrar** <LAYERDEF id=”2” visible=”true” > <SPATIALQUERY> **Seleccionamos un entorno cerca de nuestra posición x,y, aproximadamente 1 metro.** <SPATIALFILTER relation=”area_intersection”> <ENVELOPE minx, miny, maxx, maxy /> </SPATIALFILTER> </SPATIALQUERY></LAYERDEF></LAYERLIST> </PROPERTIES> <LAYER type=”acetate” visible=”true” name=”acetate” id=”acetate”> <SIMPLEMARKERSYMBOL color=”255,51,51” width=”20” type=”star”/> <MULTIPOINT> **Indicamos nuestra posición x,y obtenida del servidor de coordenadas** <POINT x, y /> </MULTIPOINT></OBJECT> </LAYER></GET_IMAGE></REQUEST></ARCXML>	 Mapa indicando la posición (x,y) actual y se muestra la capa de información más detalla, los objetos más cercanos.

Tabla 4.54. Mostrar posición de los objetos más cercanos

8	**Nombre**	*Informe de localización de objeto por nombre*	**Tipo**	Alfanumérica

Descripción	**Salida**
Obtener la ruta hacia un objeto identificado por su nombre seleccionado de una lista, desde la posición (x,y) actual.	Obtener información de la ruta que se debe seguir.
Petición XML	**Valores devueltos**
Esta petición se obtiene del fichero XML que devuelve el servidor de rutas.	Se muestra la información de la ruta que se debe seguir con las indicaciones de la red que forma el sistema indoor. • Depart Location 1 • Go East on PASILLO11 • Turn right on PASILLO2-3 • Arrive at Location 2

Tabla 4.55. Informe de localización de objeto por nombre

9	**Nombre**	*Informe de localización de sala por nombre (Puestos móviles)*	**Tipo**	Alfanumérica
Descripción			**Salida**	
Obtener la ruta hacia una sala identificada por su nombre seleccionada de una lista, desde la posición (x,y) actual.			Obtener información de la ruta que se debe seguir.	
Petición ArcXML			**Valores devueltos**	
Ver tarea 8			Ver valores devueltos de la tarea 8	

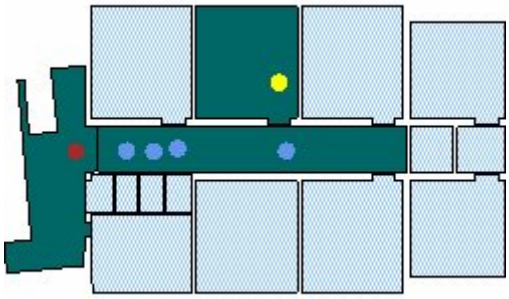
Tabla 4.56. Informe de Localización de sala por nombre (Puestos móviles)

10	**Nombre**	*Informe de localización de salas por nombre (Puestos fijos)*	**Tipo**	Alfanumérica
Descripción			**Salida**	
Obtener ruta entre dos salas identificadas por su nombre seleccionadas de una lista.			Obtener información de la ruta que se debe seguir.	
Petición XML			**Valores devueltos**	
Ver tarea 8			Ver valores devueltos de la tarea 8	

Tabla 4.57. Informe de localización de salas por nombre (Puestos fijos)

11	**Nombre**	*Consultar punto de la red que intersecciona con una sala.*	**Tipo**	Alfanumérica
Descripción			**Salida**	
Se pretende conocer el punto de la red que está contenido dentro de una sala, para así identificarlo como parada.			Se obtiene un punto (x,y), una parada final o para inicial, según lo considere el gestor de peticiones.	
Petición ArcXML			**Valores devueltos**	
< ?xml version= »1.0 » ?> <ARCXML version= »1.1 »> <REQUEST> <GET_FEATURES outputmode=”xml” > <LAYER id=”3” /> <SPATIALQUERY> **Obtenemos de la capa de puntos de red, el punto que está contenido en una sala determinada por un entorno.** <SPATIALFILTER relation=”area_intersection”> <ENVELOPE minx, miny, maxx, maxy/> </SPATIALFILTER> </SPATIALQUERY> </GET_FEATURES> </REQUEST> </ARCXML>			Se obtiene información alfanumérica, recogiendo el valor que interesa, que en este caso, es el valor (x,y) del punto de la red.	

Tabla 4.58. Consultar punto de la red que intersecciona con una sala.

<table>
<tr><th>12</th><th>Nombre</th><th>Localización de objeto por nombre</th><th>Tipo</th><th>Espacial</th></tr>
<tr><td colspan="3">Descripción</td><td colspan="2">Salida</td></tr>
<tr><td colspan="3">Obtener la ruta hacia un objeto identificado por su nombre seleccionado de una lista, desde la posición (x,y) del usuario.</td><td colspan="2">Obtener mapa con ruta desde la posición actual al objeto.</td></tr>
<tr><td colspan="3">Petición ArcXML</td><td colspan="2">Valores devueltos</td></tr>
<tr><td colspan="3"><?xml version=”1.0”?>
<ARCXML version=”1.1”>
<REQUEST><GET_IMAGE><PROPERTIES>
<LAYERLIST/>
Listado de capas, activando las que se quieren mostrar
<LEGEND display=”true”/>
Mostramos la leyenda para indicar que significado tiene cada representación en el mapa
</PROPERTIES>
<LAYER type=”featureclass” name=”Ruta” visible=”true” id=”0”>
<SPATIALQUERY>
<SPATIALFILTER relation=”area_intersection”>
<POLYLINE>
<PATH>
Puntos que forman la ruta, para resaltar las salas que intervienen en la ruta.
…….
</PATH></POLYLINE>
</SPATIALFILTER></SPATIALQUERY>
……
</LAYER>
<LAYER type=”acetate” visible=”true” name=”Ruta” id=”acetate”>
<SIMPLEMARKERSYMBOL color=”100,149,237” width=”10” />
<MULTIPOINT>
Indicamos los puntos que forman la ruta
…….
</MULTIPOINT></OBJECT></LAYER>
<LAYER type=”acetate” visible=”true” name=”Parada inicial” id=”acetate”>
<OBJECT units=”database”>
Indicamos con un color diferente el inicio de la ruta, nuestra posición x,y
……
<MULTIPOINT><POINT x, y />
</MULTIPOINT></OBJECT></LAYER>
<LAYER type=”acetate” visible=”true” name=”Parada final” id=”acetate”>
<OBJECT units=”database”>
Indicamos con un color diferente el final de la ruta, sitúado en la sala donde esta el objeto seleccionado
…….
<MULTIPOINT><POINT x, y /></MULTIPOINT>
</OBJECT></LAYER></GET_IMAGE>
</REQUEST></ARCXML></td><td colspan="2"> 
Mapa indicando el recorrido que se debe seguir para llegar a la sala donde se encuentra el objeto seleccionado.

Como se puede ver se distingue en 3 colores diferentes: la parada inicial, la parada final y la ruta. Para dejarlo más claro, se muestra una leyenda indicando el significado de cada color.
 
</td></tr>
</table>

Tabla 4.59. Localización de objeto por nombre

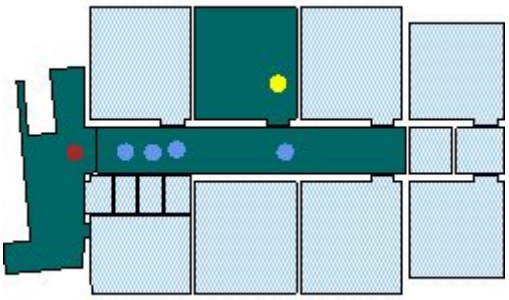
13	**Nombre**	*Localización de sala por nombre (Puestos móviles)*	**Tipo**	Espacial
Descripción			**Salida**	
Obtener la ruta hacia una sala identificada por su nombre seleccionada de una lista, desde la posición (x,y) actual.			Obtener mapa con ruta desde la posición actual a la sala destino.	
Petición ArcXML			**Valores devueltos**	
Esta tarea es similar, a la localización de objeto por nombre, ya que primero se realiza: "Obtener sala actual" "*Consultar punto de la red que intersecciona con la sala seleccionada."* La tarea ArcXML es similar al caso de *"Localización de objetos por nombre"*				

Tabla 4.60. Localización de sala por nombre (Puestos móviles)

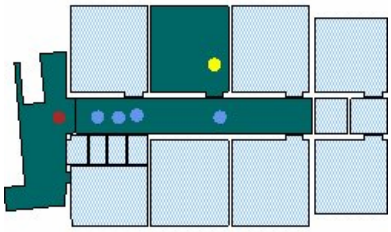
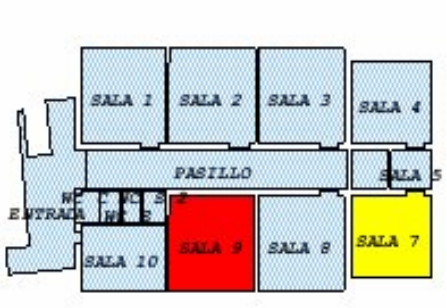
14	**Nombre**	*Localización de salas por nombre (Puestos fijos)*	**Tipo**	Espacial
Descripción			**Salida**	
Obtener ruta entre dos salas identificadas por su nombre seleccionadas de una lista.			Obtener mapa con ruta desde la sala origen a la sala destino.	
Petición ArcXML			**Valores devueltos**	
Esta tarea es similar a la anterior, se tendrá que realizar dos veces "*Consultar punto de la red que intersecciona con una sala."*, una por cada sala.				

Tabla 4.61. Localización de salas por nombre (Puestos fijos)

<table>
<tr><th>15</th><th>Nombre</th><td>Conocer cambio de mi posición</td><th>Tipo</th><td>Espacial</td></tr>
<tr><th colspan="3">Descripción</th><th colspan="2">Salida</th></tr>
<tr><td colspan="3">Se está en una posición (x,y) y se ha detectado un cambio con respecto a posición anterior. Por ejemplo: se ha cambiado de sala</td><td colspan="2">Obtener mapa de la nueva posición indicando la posición sobre el mismo, y la posición anterior, y zoom a la sala actual.</td></tr>
<tr><th colspan="3">Petición ArcXML</th><th colspan="2">Valores devueltos</th></tr>
<tr><td colspan="3">< ?xml version= »1.0 » ?>
<ARCXML version= »1.1 »>
<REQUEST>
<GET_IMAGE>
<PROPERTIES>
Mostramos una leyenda para dejar más clara la consulta
<LEGEND display=” ete”/>
<LAYERLIST>
Listado de capas, activando las que se quieren mostrar
</LAYERLIST>
</PROPERTIES>
<LAYER type=”featureclass” name=”Sala Anterior” visible=”true” id=”10”>
<DATASET fromlayer=”0” />
Seleccionamos la Sala anterior
<SPATIALQUERY where=”NOMBRE=’SALA 9 ‘” />
…….
</LAYER>
<LAYER type=”featureclass” name=”Sala Actual” visible=”true” id=”10”>
<DATASET fromlayer=”0” />
Seleccionamos la Sala actual
<SPATIALQUERY where=”NOMBRE=’SALA 7 ‘” />
…….
</LAYER>
<LAYER type=”featureclass” name=”” visible=”false” id=”0”>
Características de la capa de Salas, indicando su nombre
…….
</LAYER>
</GET_IMAGE>
</REQUEST>
</ARCXML></td><td colspan="2"> 

Mapa indicando la Sala Anterior y la Sala Actual donde se encuentra el usuario, junto con el nombre de cada una.

Como se puede ver en la leyenda se indica cual es la sala actual y cual fue la sala anterior.

<u>Leyenda</u>
 
</td></tr>
</table>

Tabla 4.62. Conocer cambio de posición actual

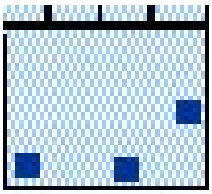
16	**Nombre**	*Zoom de la sala actual, cuando se ha cambiado de posición*	**Tipo**	Espacial
Descripción			**Salida**	
Se está en una posición (x,y) y se ha detectado un cambio con respecto a la posición anterior. Por ejemplo: se ha cambiado de sala			Obtener *zoom* de la sala actual, mostrando las capas más detalladas del sistema indoor.	
Petición ArcXML			**Valores devueltos**	
< ?xml version= »1.0 » ?> <ARCXML version= »1.1 »> <REQUEST> <GET_IMAGE> <PROPERTIES> **Entorno de la Sala Actual** <ENVELOPE minx, miny, maxx, maxy /> <LAYERLIST> **Listado de capas, activando las que se quieren mostrar** </LAYERLIST> </PROPERTIES> </GET_IMAGE> </REQUEST> </ARCXML>			 Ampliación de la Sala Actual, mostrando la capa de información más detallada.	

Tabla 4.63. Zoom de la sala actual, cuando he cambiado de posición

17	**Nombre**	*Consultar la sala de un objeto o de mi posición.*	**Tipo**	Alfanumérica
Descripción			**Salida**	
Se tiene un punto (x,y), bien sea un objeto o la posición actual del usuario y se desea conocer cual es la sala en la que se encuentra.			Obtener nombre de la Sala, y características almacenadas sobre ella.	
Petición ArcXML			**Valores devueltos**	
< ?xml version= »1.0 » ?> <ARCXML version= »1.1 »> <REQUEST> <GET_FEATURES outputmode=”xml” geometry=”false” envelope=”true” compact=”true”> <LAYER id=”0” /> <SPATIALQUERY> **Obtenemos de la capa de Salas, la sala que intersecciona con un entorno definido alrededor de mi posición x,y** <SPATIALFILTER relation=”area_intersection”> <ENVELOPE minx, miny, maxx, maxy/> </SPATIALFILTER> </SPATIALQUERY> </GET_FEATURES> </REQUEST> </ARCXML>			Se obtiene el nombre de la Sala, junto con otras características como entorno (minx, miny, maxx, maxy) que la forman. Ejemplo: • **Nombre**: Entrada • **Coordenadas**: minx=”124” miny=”119” maxx=”125” maxy=”133”	

Tabla 4.64. Consultar la sala de un objeto o de mi posición.

18 **Nombre**	*Información sobre recorridos predefinidos*	**Tipo**	Alfanumérica
Descripción		**Salida**	
Se pretende conocer la ruta de un recorrido definido de antemano. Ej: impresionistas, seleccionados,…		Obtener información de la ruta que se debe seguir.	
Petición XML		**Valores devueltos**	
Esta petición se obtiene del fichero XML recibido del servidor de rutas.		Se muestra la información de la ruta que se debe seguir con las indicaciones de la red que forma el sistema indoor. • Depart Location 1 • Go East on PASILLO14 • Turn right on PASILLO 6-7 • Arrive at Location 2	

Tabla 4.65. Información sobre recorridos predefinidos

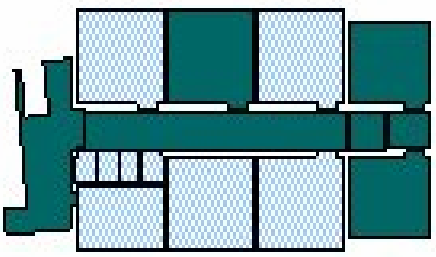
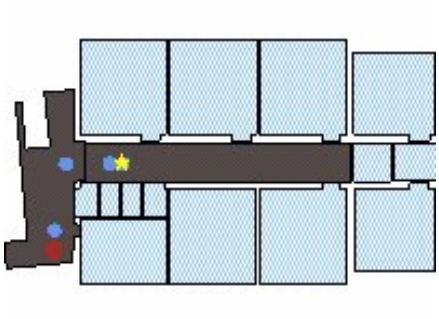
19 **Nombre**	*Consulta sobre recorridos*	**Tipo**	Espacial
Descripción		**Salida**	
Se pretende conocer las Salas que intervienen en una ruta de recorrido definido de antemano. Ej: impresionistas, seleccionados		Obtener mapa resaltando las salas que intervienen.	
Petición ArcXML		**Valores devueltos**	
<?xml version=”1.0”?> <ARCXML version=”1.1”> <REQUEST><GET_IMAGE><PROPERTIES> <LAYERLIST> **Listado de capas, activando las que se quieren mostrar** </LAYERLIST></PROPERTIES> <LAYER type=”featureclass” name=”**Salas con Ruta**” visible=”true” id=”0”> **Resaltamos las Salas que intervienen en la ruta con un color más oscuro** <DATASET fromlayer=”0” /> <SPATIALQUERY> <SPATIALFILTER relation=”area_intersection”> <POLYLINE> <PATH> **Indicamos los puntos que forman la ruta, obtenidos del servidor de rutas** **…….** </PATH></POLYLINE></SPATIALFILTER> </SPATIALQUERY> **…….** </LAYER></GET_IMAGE></REQUEST> </ARCXML>		 Mapa resaltando las salas que intervienen de la ruta seleccionada, en un color más oscuro.	

Tabla 4.66. Consulta sobre recorridos, mostrando las salas que intervienen.

<table>
<tr><th colspan="2">20 Nombre Consulta sobre recorridos Tipo Espacial</th></tr>
<tr><td>Descripción</td><td>Salida</td></tr>
<tr><td>Se pretende conocer la ruta de un recorrido definido de antemano.
Ej: impresionistas, seleccionados</td><td>Obtener mapa que se debe seguir, indicando la ruta cada dos paradas.</td></tr>
<tr><td>Petición ArcXML</td><td>Valores devueltos</td></tr>
<tr><td><?xml version=”1.0”?>
<ARCXML version=”1.1”>
<REQUEST><GET_IMAGE><PROPERTIES>
<LAYERLIST></LAYERLIST>
Listado de capas, activando las que se quieren mostrar
<LEGEND display=”true”/></PROPERTIES>
Mostramos la leyenda para indicar que significado tiene cada representación en el mapa
<LAYER type=”featureclass” name=”Ruta” visible=”true” id=”0”>
<SPATIALQUERY>
<SPATIALFILTER relation=”area_intersection”>
<POLYLINE><PATH>
<POINT x, y />
Hacemos la intersección de la ruta con la capa de Salas para resaltar las salas que intervienen en un color más oscuro
<POINT x, y />
</PATH></POLYLINE></SPATIALFILTER>
</SPATIALQUERY><SIMPLERENDERER>
<SIMPLEPOLYGONSYMBOL fillcolor=”0,102,102” />
</SIMPLERENDERER></LAYER>
<LAYER type=”acetate” visible=”true” name=”Ruta” id=”acetate”>
<SIMPLEMARKERSYMBOL color=”100,149,237” width=”8” />
<MULTIPOINT></MULTIPOINT>
Indicamos los puntos que forman la ruta, obtenidos del servidor de rutas
</OBJECT></LAYER>
<LAYER type=”acetate” visible=”true” name=”Parada inicial” id=”acetate”>
<OBJECT units=”database”>
<SIMPLEMARKERSYMBOL color=”165,042,042” width=”10” />
Indicamos con un color diferente el inicio de la ruta
<MULTIPOINT><POINT x, y /></MULTIPOINT>
</OBJECT></LAYER>
<LAYER type=”acetate” visible=”true” name=”Parada final” id=”acetate”>
<OBJECT units=”database”>
<SIMPLEMARKERSYMBOL color=”255,255,000” width=”10” type=”star”/>
Indicamos con un color diferente el final de la ruta
<MULTIPOINT><POINT x, y /></MULTIPOINT>
</OBJECT></LAYER>
</GET_IMAGE></REQUEST></ARCXML></td><td> 
Mapa indicando el recorrido que se debe seguir para ir recorriendo en este caso los objetos clasificados como impresionistas, se muestra la ruta entre dos paradas.

Como se puede ver se distingue en 3 colores diferentes: la parada inicial, la parada final y la ruta. Para una mejor comprensión, se muestra una leyenda indicando el significado de cada color.
 
</td></tr>
</table>

Tabla 4.67. Consulta sobre recorridos, mostrando la ruta

21	**Nombre**	*Consulta sobre información de las rutas*	**Tipo**	Alfanumérica

Descripción	**Salida**
Se pretende conocer los objetos por los que se irá pasando en la ruta obtenida.	Obtener objetos identificados por su nombre en forma de listado, de una parte de la información de la ruta. Por ejemplo: Al norte del pasillo 11.
Petición ArcXML	**Valores devueltos**
< ?xml version= »1.0 » ?> <ARCXML version= »1.1 »> <REQUEST> <GET_FEATURES outputmode=”xml” geometry=”false” envelope=”true” compact=”true”> <LAYER id=”2” /> <SPATIALQUERY> <SPATIALFILTER relation=”envelope_intersection”> **Obtenemos los objetos de este pasillo** <ENVELOPE minx, miny, maxx, maxy /> </SPATIALFILTER> </SPATIALQUERY> </GET_FEATURES> </REQUEST> </ARCXML>	Se obtiene los objetos que se van visitando: • El carnaval de Arlequín. • El pájaro migratorio. • Interior holandés.

Tabla 4.68. Consulta sobre información de las rutas

22	**Nombre**	*Zoom sobre información de las rutas*	**Tipo**	Espacial

Descripción	**Salida**
Se pretende conocer la zona del plano por la que se va pasando en la ruta obtenida.	Obtener *zoom* de la esa zona del plano.
Petición ArcXML	**Valores devueltos**
< ?xml version= »1.0 » ?> <ARCXML version= »1.1 »> <REQUEST> <GET_IMAGE><PROPERTIES> **Zoom a esta zona del pasillo** <ENVELOPE minx, miny, maxx, maxy/> <LAYERLIST> **Listado de capas, activando las que se quieren mostrar** </LAYERLIST></PROPERTIES> <LAYER type=”featureclass” name=”Nombre” visible=”true” id=”0”> <GROUPRENDERER> <SIMPLELABELRENDERER field=”NOMBRE”> **Indicamos de la capa Salas el nombre de las Salas que pertenecen a este pasillo** </SIMPLELABELRENDERER></GROUPRENDERER> </LAYER></GET_IMAGE></REQUEST></ARCXML>	 Se obtiene imagen ampliada de esa zona del pasillo, mostrándose el nombre de las salas que pertenecen ese pasillo.

Tabla 4.69. Zoom sobre información de las rutas

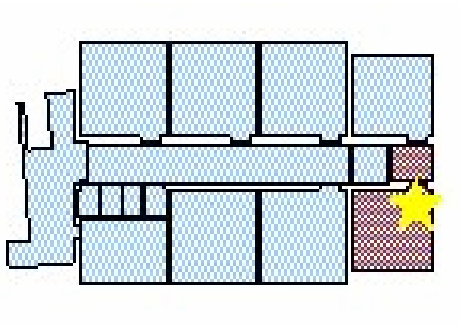
<table>
<tr><th>23</th><th>Nombre</th><th>Consulta espacial sobre información de las rutas</th><th>Tipo</th><th>Espacial</th></tr>
<tr><td colspan="2">Descripción</td><td colspan="3">Salida</td></tr>
<tr><td colspan="2">Se pretende conocer la zona del plano por la que se va pasando en la ruta obtenida.</td><td colspan="3">Obtener imagen completa del plano indicando posición actual del usuario en esa zona.</td></tr>
<tr><td colspan="2">Petición ArcXML</td><td colspan="3">Valores devueltos</td></tr>
<tr><td colspan="2"><?xml version=”1.0”?>
<ARCXML version=”1.1”>
<REQUEST><GET_IMAGE><PROPERTIES>
<LAYERLIST>
Listado de capas, activando las que se quieren mostrar
</LAYERLIST>
</PROPERTIES>
<LAYER type=”featureclass” name=”select layer” visible=”true” id=”0”>
<DATASET fromlayer=”0” />
<SPATIALQUERY><SPATIALFILTER relation=”area_intersection”>
Hacemos la intersección de esa zona de la red con la capa Salas para indicar con otro color la zona que interviene
<ENVELOPE minx, miny, maxx, maxy />
</SPATIALFILTER>
</SPATIALQUERY>
…….
</LAYER>
<LAYER type=”acetate” visible=”true” name=”Mi Posición” id=”acetate”>
<OBJECT units=”database”>
<SIMPLEMARKERSYMBOL color=”255,255,0” width=”20” type=”star”/>
Indicamos nuestra posición sobre el pasillo, que la obtenemos del servidor de coordenadas
<MULTIPOINT><POINT x, y />
</MULTIPOINT></OBJECT></LAYER></GET_IMAGE></REQUEST></ARCXML></td><td colspan="3"> 
Mapa completo resaltando en un color diferente la zona del pasillo que forma parte de la ruta, y con una estrella la posición si se encontrara en el pasillo seleccionado.</td></tr>
</table>

Tabla 4.70. Consulta espacial sobre información de las rutas

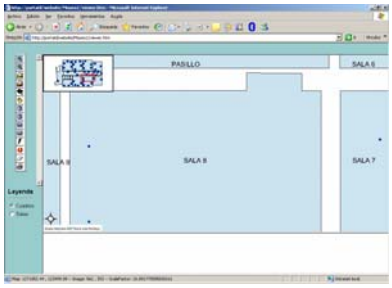
24	**Nombre**	*Zoom in al plano (Puestos fijos)*	**Tipo**	Espacial
Descripción			**Salida**	
Se quiere ver una zona del plano ampliada.			Ampliar la imagen del mapa, mostrándose de forma ampliada la zona donde se ha hecho clic con el ratón.	
Petición ArcXML			**Valores devueltos**	
< ?xml version= »1.0 » ?> <ARCXML version= »1.1 »> <REQUEST> <GET_IMAGE> <PROPERTIES> **Entorno de la zona que queremos hacer zoom in** <ENVELOPE minx, miny, maxx, maxy/> <LAYERLIST> **Listado de capas, activando las que se quieren mostrar** </LAYERLIST> </PROPERTIES> </GET_IMAGE> </REQUEST> </ARCXML>			 Imagen ampliada de la zona donde se ha pinchado con el ratón.	

Tabla 4.71. Zoom in al plano (Puestos fijos)

25	**Nombre**	*Zoom out al plano (Puestos fijos)*	**Tipo**	Espacial
Descripción			**Salida**	
Se quiere ver el mapa con sus capas de información menos detalladas.			Obtener mapa mostrando las capas de información menos detalladas, es decir, mostrando las salas, junto con su nombre.	
Petición ArcXML			**Valores devueltos**	
< ?xml version= »1.0 » ?> <ARCXML version= »1.1 »> <REQUEST> <GET_IMAGE> <PROPERTIES> **Entorno de la zona que queremos hacer zoom out** <ENVELOPE minx, miny, maxx, maxy/> <LAYERLIST> **Listado de capas, activando las que se quieren mostrar** </LAYERLIST> </PROPERTIES> </GET_IMAGE> </REQUEST> </ARCXML>			 Imagen alejada de la zona donde se ha pinchado con el ratón.	

Tabla 4.72. Zoom out al plano (Puestos fijos)

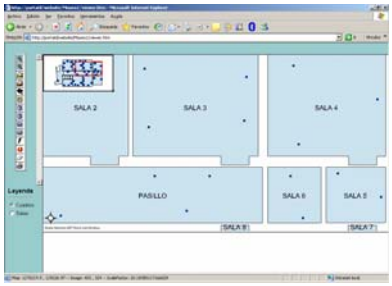
26 **Nombre** *Mover plano*	**Tipo** Espacial
Descripción	**Salida**
Se quiere mover la imagen del mapa libremente, hacia arriba, abajo, izquierda o derecha, sin hacer zoom	Obtener mapa según la posición que movamos el ratón.
Petición ArcXML	**Valores devueltos**
< ?xml version= »1.0 » ?> <ARCXML version= »1.1 »> <REQUEST> <GET_IMAGE> <PROPERTIES> **Entorno de la zona que queremos hacia donde hemos movido el plano** <ENVELOPE minx, miny, maxx, maxy/> <LAYERLIST> **Listado de capas, activando las que se quieren mostrar** </LAYERLIST> </PROPERTIES> </GET_IMAGE> </REQUEST> </ARCXML>	 Vista del sistema indoor alrededor de la zona donde se ha desplazado el ratón.

Tabla 4.73. Mover plano

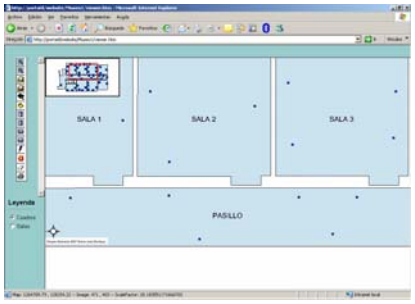
27 **Nombre** *Ir al Norte*	**Tipo** Espacial
Descripción	**Salida**
Según la imagen del mapa disponible actualmente, mover la imagen hacia el Norte	Obtener mapa mostrando los objetos hacia el Norte
Petición ArcXML	**Valores devueltos**
< ?xml version= »1.0 » ?> <ARCXML version= »1.1 »> <REQUEST> <GET_IMAGE> <PROPERTIES> **Entorno de la zona que se muestra al Norte** <ENVELOPE minx, miny, maxx, maxy/> <LAYERLIST> **Listado de capas, activando las que se quieren mostrar** </LAYERLIST> </PROPERTIES> </GET_IMAGE> </REQUEST> </ARCXML>	 Imagen del sistema indoor, hacia el Norte

Tabla 4.74. Ir al Norte

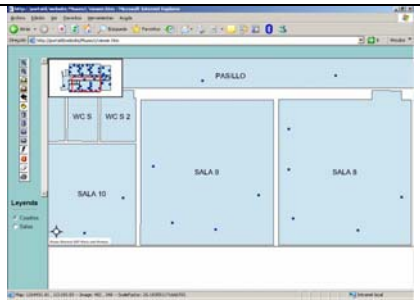
28 **Nombre** *Ir al Sur*	**Tipo** Espacial
Descripción	**Salida**
Según la imagen del mapa disponible actualmente, mover la imagen hacia el Sur	Obtener mapa mostrando los objetos hacia el Sur
Petición ArcXML	**Valores devueltos**
< ?xml version= »1.0 » ?> <ARCXML version= »1.1 »> <REQUEST> <GET_IMAGE> <PROPERTIES> **Entorno de la zona que se muestra al Sur** <ENVELOPE minx, miny, maxx, maxy/> <LAYERLIST> **Listado de capas, activando las que se quieren mostrar** </LAYERLIST> </PROPERTIES> </GET_IMAGE> </REQUEST> </ARCXML>	 Imagen del sistema indoor, hacia el Sur.

Tabla 4.75. Ir al Sur

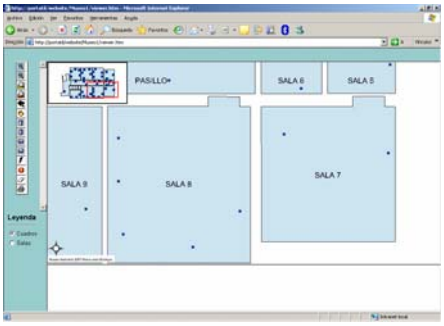
29 **Nombre** *Ir al Este*	**Tipo** Espacial
Descripción	**Salida**
Según la imagen del mapa disponible actualmente, mover la imagen hacia el Este	Obtener mapa mostrando los objetos hacia el Este
Petición ArcXML	**Valores devueltos**
< ?xml version= »1.0 » ?> <ARCXML version= »1.1 »> <REQUEST> <GET_IMAGE> <PROPERTIES> **Entorno de la zona que se muestra al Este** <ENVELOPE minx, miny, maxx, maxy/> <LAYERLIST> **Listado de capas, activando las que se quieren mostrar** </LAYERLIST> </PROPERTIES> </GET_IMAGE> </REQUEST> </ARCXML>	 Imagen del sistema indoor, hacia el Este

Tabla 4.76. Ir al Este

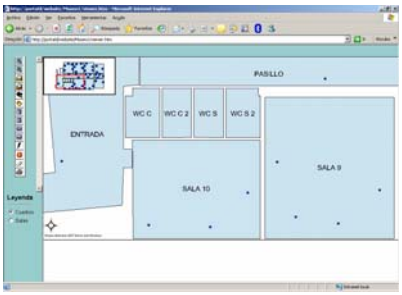
30 **Nombre** *Ir al Oeste*	**Tipo** Espacial
Descripción	**Salida**
Según la imagen del mapa disponible actualmente, mover la imagen hacia el Oeste	Obtener mapa mostrando los objetos hacia el Oeste
Petición ArcXML	**Valores devueltos**
< ?xml version= »1.0 » ?> <ARCXML version= »1.1 »> <REQUEST> <GET_IMAGE> <PROPERTIES> **Entorno de la zona que se muestra al Oeste** <ENVELOPE minx, miny, maxx, maxy/> <LAYERLIST> **Listado de capas, activando las que se quieren mostrar** </LAYERLIST> </PROPERTIES> </GET_IMAGE> </REQUEST> </ARCXML>	 Imagen del sistema indoor, hacia el Oeste.

Tabla 4.77. Ir al Oeste

31 **Nombre** *Mostrar información de un objeto*	**Tipo** Alfanumérica
Descripción	**Salida**
Se desea obtener la información de un objeto haciendo clic sobre el plano actual.	Obtener características del objeto, como título, autor, …
Petición ArcXML	**Valores devueltos**
< ?xml version= »1.0 » ?> <ARCXML version= »1.1 »> <REQUEST> <GET_FEATURES outputmode=”xml”> <LAYER id=”2” /> **Seleccionamos la capa de objetos** <SPATIALQUERY > <SPATIALFILTER relation=”area_intersection”> **Entorno al punto donde se ha hecho click** <ENVELOPE minx, miny, maxx, maxy/> </SPATIALFILTER> </SPATIALQUERY> </GET_FEATURES> </REQUEST> </ARCXML>	Obtenemos información alfanumérica del objeto seleccionado, por ejemplo: • **Titulo**: Muchacha en la ventana • **Año**: 1925 • **Autor**: Salvador Dalí • **Técnica**: Óleo sobre cartón piedra. **……**

Tabla 4.78. Mostrar información de un objeto

32	**Nombre** *Mostrar información de una sala*	**Tipo**	Alfanumérica
Descripción		**Salida**	
Se desea obtener la información de una sala haciendo clic sobre el plano		Obtener características de la sala seleccionada.	
Petición ArcXML		**Valores devueltos**	
< ?xml version= »1.0 » ?> <ARCXML version= »1.1 »> <REQUEST> <GET_FEATURES outputmode=”xml”> <LAYER id=”2” /> **Seleccionamos la capa Salas** <SPATIALQUERY > <SPATIALFILTER relation=”area_intersection”> **Entorno al punto donde se ha hecho click** <ENVELOPE minx, miny, maxx, maxy/> </SPATIALFILTER> </SPATIALQUERY> </GET_FEATURES> </REQUEST> </ARCXML>		Obtenemos información alfanumérica de la sala seleccionada. Por ejemplo: • **Nombre**: Sala 10	

Tabla 4.79. Mostrar información de una sala

33	**Nombre** *Mostrar imagen de un cuadro.*	**Tipo**	Alfanumérica
Descripción		**Salida**	
Se desea obtener la imagen de un cuadro haciendo clic sobre el plano.		Obtener imagen del objeto seleccionado.	
Petición ArcXML		**Valores devueltos**	
< ?xml version= »1.0 » ?> <ARCXML version= »1.1 »> <REQUEST> <GET_FEATURES outputmode=”xml”> <LAYER id=”2” /> **Seleccionamos la capa de objetos** <SPATIALQUERY subfields=”**Titulo**” > <SPATIALFILTER relation=”area_intersection”> **Entorno al punto donde se ha hecho click** <ENVELOPE minx, miny, maxx, maxy/> </SPATIALFILTER> </SPATIALQUERY> </GET_FEATURES> </REQUEST> </ARCXML>		Con la petición ArcXML obtenemos el título del objeto seleccionado, y con este título el gestor de peticiones muestra la imagen del cuadro asociada. 	

Tabla 4.80. Mostrar imagen de un cuadro.

34	**Nombre** *Conocer objetos de la sala actual*	**Tipo**	Alfanumérica
Descripción		**Salida**	
Se desea consultar los objetos de la sala actual.		Mostrar información de los objetos de la sala actual.	
Petición ArcXML		**Valores devueltos**	
< ?xml version= »1.0 » ?> <ARCXML version= »1.1 »> <REQUEST> <GET_FEATURES outputmode=”xml” geometry=”false” envelope=”true” compact=”true”> <LAYER id=”2” /> <SPATIALQUERY> <SPATIALFILTER relation=”envelope_intersection”> **Realizamos la intersección con el entorno de la sala actual, e indicamos que se muestre las características de la capa de información más detallada de esta Sala.** <ENVELOPE minx, miny, maxx, maxy/> </SPATIALFILTER> </SPATIALQUERY> </GET_FEATURES> </REQUEST> </ARCXML>		Información textual de los objetos de la sala actual. Por ejemplo de la Sala 9 • Muchacha en la ventana • Arlequín • Retrato de Luis Buñuel • Muchacha de espaldas	

Tabla 4.81. Conocer objetos de la sala actual

Capítulo 5. ARQUITECTURA DEL SISTEMA

"Sólo el conocimiento que llega desde dentro
es el verdadero conocimiento"

Sócrates (470a.c - 390a.c.)
Filósofo griego.

En el capítulo actual se detalla la arquitectura del sistema junto con unos ejemplos para mostrar la forma en que interactúan los diferentes componentes. Se comenzará dando una visión general de la arquitectura, para posteriormente detallar cada uno de los módulos que forman la solución. Posteriormente, se proporcionará dos ejemplos de funcionamiento y finalmente se hará una descripción de la tecnología utilizada para la realización del proyecto.

5.1 INTRODUCCIÓN

Ya realizado el análisis del sistema en el Capítulo 4, en este Capítulo se definirá la arquitectura del sistema. En primera lugar, se mostrará una visión general del sistema y progresivamente se profundizará en aquellos elementos que la componen. Además se expondrán una serie de ejemplos con la finalidad de proporcionar una mejor comprensión de los módulos desarrollados.

La estructura del Capítulo queda como sigue. En la sección 5.2 se muestra el diseño de la arquitectura, junto con una descripción general de sus componentes. En la sección 5.3 se expone un conjunto de ejemplos de funcionamiento que ilustrarán la interacción entre cada uno de los componentes que forman la arquitectura. Finalmente, en la sección 5.4 se explica la tecnología empleada para desarrollar los módulos, junto con una sencilla descripción.

5.2 DISEÑO DE LA ARQUITECTURA DEL SISTEMA

La arquitectura del sistema se puede dividir en 3 partes: cliente, proceso y almacenamiento. En la parte del cliente se encuentran todas las páginas web que se mostrarán al usuario, en la parte de proceso se tiene toda la lógica necesaria para el procesamiento de las peticiones de ruta, coordenada y datos. Mientras que en la parte de almacenamiento se encuentran todos los datos que el sistema necesita para su correcto funcionamiento. En la figura 5.1 se puede ver la arquitectura del sistema.

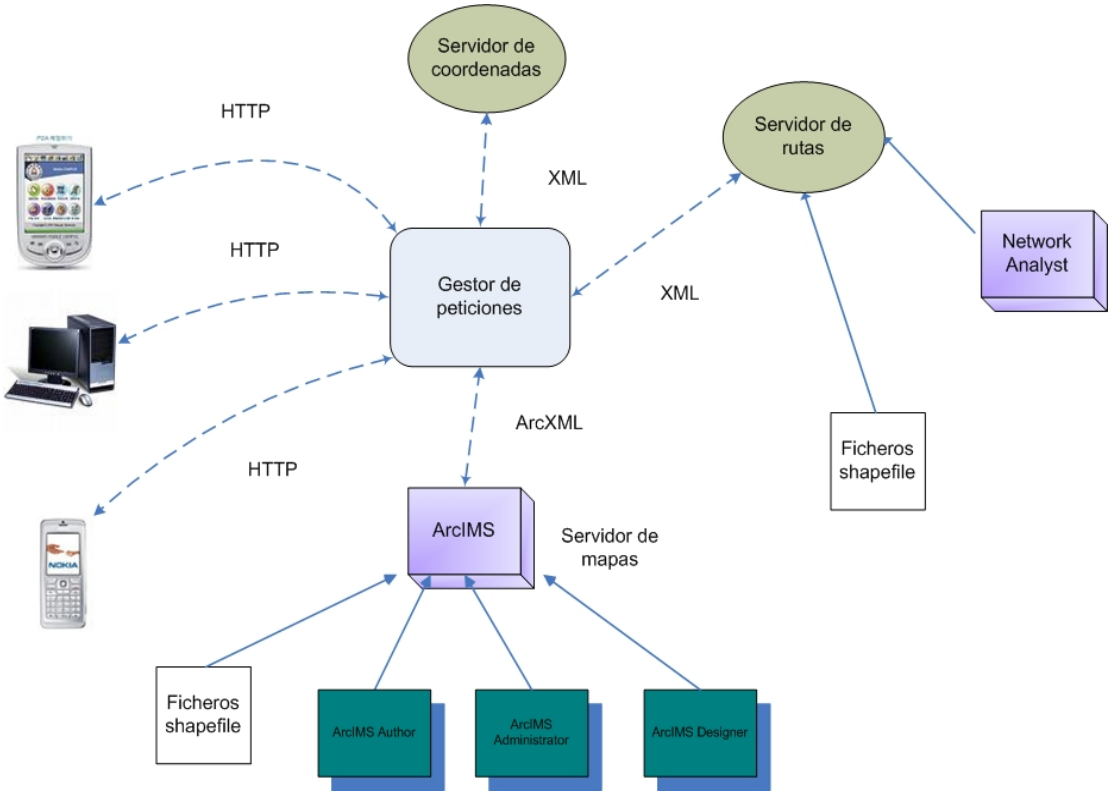


Figura 5.1 Arquitectura del sistema.

Como se muestra en la figura 5.1, el **núcleo principal** de la arquitectura es el **Gestor de peticiones**. Él es el encargado de atender las peticiones de consulta solicitadas por los puestos fijos (**PC**) o móviles (**PDA o teléfono móvil**) y devolver los resultados con la interfaz deseada, siguiendo unas pautas de usabilidad. Las peticiones son enviadas por los puestos, tanto móviles como fijos, siguiendo el protocolo HTTP.

Como se comentó en el Capítulo 2, el servidor de mapas utilizado para el desarrollo del proyecto ha sido **ArcIMS** (los motivos de su elección se encuentran detallados en la sección 2.1.9), Por lo tanto, ArcIMS será el encargado de gestionar la información tanto espacial como alfanumérica sobre el sistema indoor.

ArcIMS tiene cuatro elementos necesarios para realizar su labor: shapefiles, ArcIMS Author, ArcIMS Administrator y ArcIMS Designer.

Tal y como se comentó en el Capítulo 2, un **shapefile** es un formato vectorial de almacenamiento digital donde se guarda la localización de los elementos geográficos y los atributos asociados a ellos.

Por otro lado, ArcIMS Author, ArcIMS Administrator y ArcIMS Designer son los componentes encargados de poner en marcha el servidor de mapas a través de unos asistentes fáciles de utilizar. ArcIMS Author es un asistente para la generación del fichero de configuración del servicio de mapas, ArcIMS Administrator es un asistente para la administración de los servicios publicados y de los servidores espaciales. Y ArcIMS Designer es un asistente para el diseño del sitio WEB que permite definir la funcionalidad a la que tendrá acceso el cliente.

La comunicación entre el gestor de peticiones y servidor de mapas se realiza a través del protocolo **ArcXML**. Este lenguaje será la piedra angular sobre la que se soliciten las peticiones al servidor de mapas y se expresen los resultados que éste devuelva.

El **servidor de rutas** será el encargado de calcular las rutas solicitadas y enviarlas al gestor de peticiones. Este servidor utiliza la tecnología de **Network Analyst**. Tal y como se comentó en el Capítulo 3**,** este componente calcula rutas en redes multimodales. Entre las funcionalidades que ofrecen están, por ejemplo, la generación de la ruta más eficiente, localización de las ubicaciones más próximas, generación de áreas de servicio basadas en tiempos de viaje, cálculo de matriz de origen-destino y listado de informe de direcciones. Este servidor utilizará ficheros *Shapefiles* para obtener la información necesaria para realizar el cálculo de rutas.

El componente más sencillo y no por ello el menos importante es el **servidor de coordenadas,** que trata de simular un sistema de localización. Este servidor dará la posición del dispositivo móvil y se podrá reemplazar en un futuro por cualquier sistema de localización.

Por último, indicar que la arquitectura está diseñada tanto para dispositivos móviles (PDA o teléfonos móviles) como fijos (PC). Manteniendo un principio de abstracción de la arquitectura, solamente es necesario modificar el diseño de las interfaces para dispositivos móviles o fijos, dejando la lógica del sistema igual para los dos entornos.

En las siguientes secciones se detallarán cada uno de los módulos esbozados en este punto.

5.2.1 Gestor de peticiones

Este módulo es el **núcleo principal** de la arquitectura, su objetivo será atender las consultas solicitadas por los puestos (fijos o móviles) y devolver los resultados con la interfaz deseada. Para llevar a cabo este objetivo, el gestor de peticiones debe comunicarse con el servidor de coordenadas, el servidor de rutas y el servidor de mapas.

Por un lado, la comunicación con el servidor de coordenadas se realizará únicamente cuando sea necesario conocer la posición del puesto móvil para realizar una consulta. Por otro lado, la comunicación con el servidor de rutas será más detallada en la sección 5.2.2.

Las funciones a detallar del gestor de peticiones son: cómo el gestor de peticiones analiza las consultas recibidas del usuario, cómo se comunica con el servidor de mapas y cómo obtiene una respuesta acorde con la consulta solicitada, mostrando al usuario su respuesta en forma de interfaz.

Como se ha comentado en la sección anterior, para establecer la comunicación el gestor de peticiones con servidor de mapas (ArcIMS) se utiliza un lenguaje de comunicación conocido como **ArcXML**. Con este lenguaje se indicará el tipo de consulta que se hará sobre el sistema indoor (imágenes o alfanumérica) y una vez recibida la respuesta, (recibida también en formato ArcXML) el gestor de peticiones recogerá aquellos parámetros necesarios para mostrar la respuesta al usuario.

ArcXML es un lenguaje basado en XML por lo que es independiente de la plataforma, se pueden hacer tanto consultas espaciales como alfanuméricas, de las que se pueden obtener tanto datos como imágenes, añadir capas dinámicamente para determinadas consultas, etc.

Para poder establecerse la comunicación entre el gestor de peticiones y el servidor de mapas, la aplicación web situada en el gestor de peticiones debe de tener configurados una serie de parámetros: (i) nombre del servicio que se ha configurado con la herramienta ArcIMS Administrator (ii) nombre del servidor de mapas, y (iii) puerto por el que el servidor de mapas recibe las peticiones. Una vez que se tienen los parámetros configurados, el gestor de peticiones está preparado para poder comunicarse con el servidor de mapas, y atender las peticiones del usuario.

Cada vez que el usuario hace una petición de consulta al gestor de peticiones éste la analiza y dependiendo del tipo de la consulta solicitada se tiene:

- Si el gestor de peticiones necesita conocer la posición del puesto móvil para continuar con la consulta, enviará una petición al servidor de coordenadas, para

obtener la posición (x,y) donde se encuentra el puesto móvil. Y posteriormente, podrá generar la petición al servidor de mapas o al servidor de rutas, dependiendo de lo que se vaya a consultar.

- Si la consulta solicitada necesita calcular rutas, enviará una petición al servidor de rutas. Este servidor responde con la ruta mediante un XML al gestor de peticiones, el cual generará, por un lado, una petición ArcXML con la que se obtendrá una imagen que representará la ruta a seguir. Por otro lado, generará una respuesta en forma de interfaz al usuario con los pasos a seguir para llegar al destino solicitado.

Una vez que el gestor de peticiones tiene toda la información necesaria para realizar tanto la consulta espacial como alfanumérica, realizará los siguientes pasos:

- **Establecerá la conexión** con el servidor de mapas.
- Generará una **petición** ArcXML que se enviará al servidor de mapas.
- El servidor de mapas procesará la petición y contestará con una **respuesta** ArcXML al gestor de peticiones.
- Finalmente, cuando el gestor de peticiones ha obtenido la respuesta del servidor de mapas, la reordenará mostrando sólo aquellos campos que le interesen para devolver la respuesta en forma de interfaz al usuario.

Cuando se haya completado todo este proceso, se mostrará al usuario la repuesta obtenida.

5.2.2 Servidor de rutas

Una vez entendido el papel que desempeña el módulo anterior, se estudiarán las diferentes consultas que pueden ocurrir entre el gestor de peticiones y el servidor de rutas. Este servidor de rutas sólo se va a utilizar cuando la petición solicitada (del puesto fijo o del puesto móvil) requiera del cálculo de una ruta en el plano.

El objetivo del servidor de rutas es realizar el cálculo de la ruta más corta, entre un origen y un destino de forma dinámica (**rutas dinámicas**). Además, puede calcular una serie de rutas ya definidas de antemano (**rutas estáticas**) dentro de un sistema indoor.

Para el cálculo de la ruta el servidor necesitará una serie de parámetros: (i) una red definida sobre el sistema indoor, (ii) una serie de paradas, (iii) y un parámetro indicando que se quiere obtener la ruta más corta, el valor que se considera para este

parámetro es la longitud de la red. Estos parámetros se vieron de forma más detallada en la sección 3.2.5.4.

Por tanto, se han diferenciado dos tipos de rutas: **rutas estáticas** y **rutas dinámicas**. Se define ruta estática como la ruta más corta entre una serie de *paradas* ya definidas de antemano. Y ruta dinámica como la ruta más corta entre un origen y un destino cualesquiera.

Lo primero que se debe hacer para poder ver una ruta, ya sea estática o dinámica, es conectarse a la página del sistema indoor, es decir, se manda una petición al gestor de peticiones, a través del protocolo HTTP, solicitándole una página web. El gestor de peticiones la procesa y responde con una página HTML.

Una vez que el usuario está conectado a la página del visor, podrá realizar el cálculo de rutas. Será a partir de este punto donde el gestor de rutas entrará en acción, pudiéndose calcular dos tipos de rutas:

- Ruta entre un origen y un destino, en este caso se tratará de una ruta dinámica.
- Ruta predefinida, en este caso se tratará de una ruta estática.

5.2.2.1 Rutas dinámicas

Se entiende por rutas dinámicas aquellas que dado un punto origen y un punto destino cualquiera, se obtiene el camino más corto entre ambos puntos. En este caso se dan dos posibilidades según el tipo de dispositivo desde el que se realiza la petición, puesto fijo (PC) o puesto móvil (PDA,…).

Si el usuario se conecta a través de un **puesto fijo**, se tiene que indicar tanto el origen como el destino de la ruta. Mientras que si el usuario se conecta a través de un **puesto móvil** sólo se tiene que indicar el destino de la ruta, ya que el gestor de peticiones será el encargado de realizar una consulta al servidor de coordenadas para obtener la posición origen, que es la posición en la que se encuentra el usuario en el momento de hacer la consulta.

Una vez que el gestor de peticiones tiene la posición origen y la posición destino, envía una consulta al servidor de rutas para que realice los cálculos necesarios. Esta comunicación se realiza a través de un *"proxy"* que hereda toda la funcionalidad necesaria para las comunicaciones HTTP y SOAP entre el servicio web y el gestor de peticiones.

A continuación, el servidor de rutas sitúa los dos puntos origen y destino en una capa, que contiene toda la lógica de la red, y por último, carga la red para calcular la ruta siguiendo la condición impuesta de la ruta más corta.

Cuando el servidor de rutas ha terminado de realizar su cálculo, crea un fichero XML que contiene toda la información necesaria, paso a paso, para llevar a cabo la ruta: parada inicial, parada final, diferentes paradas intermedias, y la longitud de la ruta total.

El servidor de rutas envía este fichero XML al gestor de peticiones, utilizando HTTP como protocolo de transporte y XML como codificación de datos. El gestor de peticiones procesa el XML obtenido y muestra la información al usuario en forma de página web.

5.2.2.2 Rutas estáticas

Rutas estáticas son todas aquellas que se encuentran definidas a priori en el sistema. En este tipo de rutas tanto el punto origen como el punto destino son fijos, definiéndose estos puntos en el momento en que se especifica la ruta.

Cuando el usuario selecciona una de las rutas predefinidas, se manda una petición HTTP al gestor de peticiones, indicando el nombre de la ruta que se quiere conocer. Una vez que el gestor de peticiones recibe este nombre, envía una consulta al servidor de rutas indicándole el **nombre de la ruta** que debe calcular. Esta comunicación como se comentó, se realiza a través de un *"proxy"*, que hereda toda la funcionalidad necesaria para las comunicaciones HTTP y SOAP entre el servicio web y el gestor de peticiones.

Para este tipo de consultas, el servidor de rutas utilizará como paradas los puntos que se han fijado en un fichero que define la ruta.

Una vez que el servidor ha terminado de realizar el cálculo de la ruta, creará un fichero XML que contiene toda la información necesaria, paso a paso, para realizar la ruta: parada inicial, parada final, diferentes paradas intermedias, y la longitud de la ruta total. El fichero XML indicará una ruta a seguir cada dos paradas.

El servidor de rutas envía este fichero XML al gestor de peticiones, utilizando HTTP como protocolo de transporte y XML como codificación de datos. El gestor de peticiones procesa el XML obtenido, y muestra la información al usuario en forma de página web.

En el punto 5.9 se expondrán una serie de consultas para comprender mejor la comunicación entre estos componentes.

5.2.3 Servidor de coordenadas

El servidor de coordenadas sólo se va a utilizar cuando las peticiones solicitadas provengan de un dispositivo móvil, y el gestor de peticiones necesite conocer su posición para procesar la consulta.

Este servidor calcula una posición (x,y) aleatoria dentro de unos límites predefinidos. Posteriormente, este sistema se podrá reemplazar en un futuro por cualquier sistema de localización *indoor*.

Como alternativas al servidor de coordenadas podemos citar el caso de RFID (Identificación por Radiofrecuencia), que es un método de almacenamiento y recuperación de datos remotos que usa dispositivos denominados **etiquetas**. Una etiqueta RFID es un dispositivo pequeño, como una pegatina, que puede ser adherida a cualquier dispositivo. Estas etiquetas contienen antenas que les permiten recibir y responder a peticiones por radiofrecuencia desde un emisor-receptor [WRFID].

Otras alternativas pueden ser las siguientes: UWB O WI-Fi.

Para la comunicación entre el gestor de peticiones y el servidor de coordenadas, se utiliza HTTP como protocolo de transporte y XML como codificación de datos.

5.2.4 ArcIMS Author, ArcIMS Administrator y ArcIMS Designer

El objetivo del servidor de mapas es ofrecer la información alfanumérica del sistema indoor al gestor de peticiones. Para realizar este objetivo se deben de seguir unos pasos de configuración iniciales.

Para llevar a cabo la configuración inicial se han utilizado dos herramientas: **ArcIMS Author y ArcIMS Administrator**. Una información más detallada de estas dos herramientas se puede consultar en la sección 2.1.7.3.

En primer lugar, se utiliza ArcIMS Author que permite crear el **fichero de configuración** del mapa (fichero *.axl*)**.** Este fichero, contiene las rutas de los *shapefles* necesarios para configurar tanto el sistema indoor como las propiedades del mapa (color de las capas, forma de las capas, etc.). Los detalles de este fichero son dados en la sección 3.1.7, donde se explica el lenguaje ArcXML en más detalle.

A continuación, con el fichero .axl definido anteriormente se crea y se pone en ejecución, mediante ArcIMS Administrator, un servicio de mapas que representa al sistema indoor y será sobre el que se realizarán las diferentes consultas solicitadas.

Finalmente, se diseñará una interfaz web, que permitirá realizar consultas ArcXML sobre el servicio de mapas definido con anterioridad.

Para diseñar una interfaz web ArcIMS proporciona una herramienta conocida como **ArcIMS Designer**, que se explicó en la sección 5.2.4, donde es posible crear una serie de páginas HTML con una funcionalidad básica sobre el sistema indoor. Esta herramienta se ha utilizado para la página inicial mostrada en el visor web en el caso de un puesto fijo. En la figura 5.2, a la izquierda se muestra la interfaz generada a través de ArcIMS Designer y a la derecha, la interfaz que ha quedado tras las modificaciones.

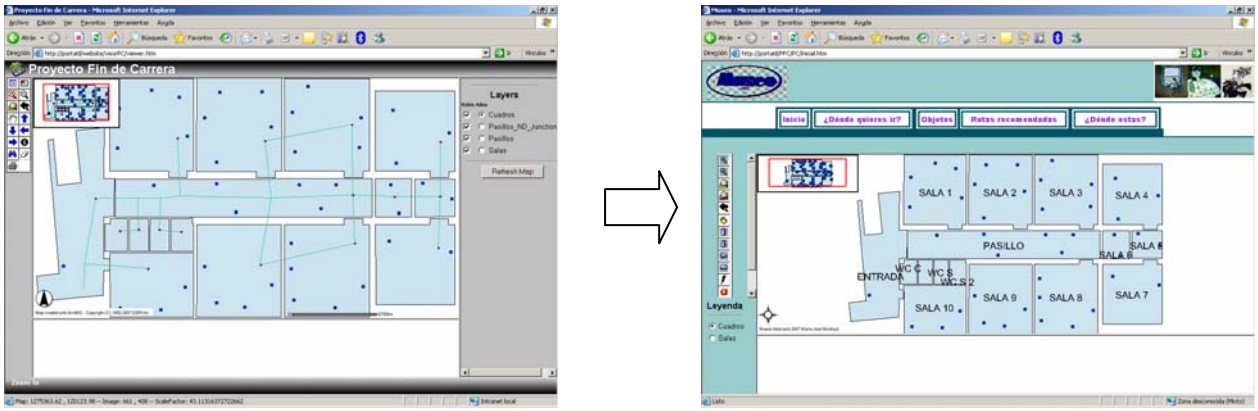


Figura 5.2. Página web con ArcIMS Designer

Para desarrollar el resto de interfaces necesarias para atender las consultas, se ha utilizado la tecnología **.NET**, tanto para la interfaz de los puestos móviles como para los puestos fijos, teniendo en cuenta la usabilidad, ya que es sin duda un requisito imprescindible para tener una solución viable.

5.3 EJEMPLOS DE CONSULTAS

Hasta aquí se tiene la definición de la arquitectura del proyecto, por tanto, ahora se verá dos ejemplos de consultas para tener una mejor comprensión de cada uno de los módulos explicados en la sección anterior y ver como interactúan entre ellos.

Se pueden distinguir claramente dos tipos de consultas: aquellas con las que se pretende obtener una **imagen** del sistema indoor, y aquellas cuya finalidad es obtener **información alfanumérica** del sistema indoor. Para el primer caso, su formato será .jpg y para el segundo caso, el gestor de peticiones se encargará de procesar la respuesta y se la mostrará al usuario de forma adecuada.

Para tener una mejor comprensión de las consultas se define un sistema indoor que está formado por salas que contienen información. Esta información puede ser cuadros, esculturas,… Se pueden realizar consultas espaciales y cálculo de rutas sobre este sistema.

Las dos consultas que se van a describir son: *Calcular ruta a un objeto, Conocer la posición actual del usuario y los objetos más cercanos*. Estas consultas se realizarán desde un dispositivo móvil como puede ser una PDA. Estas consultas se detallan en la sección 5.3.1 y 5.3.2, respectivamente.

5.3.1 Calcular ruta a un objeto desde un dispositivo móvil

Este tipo de consulta solo tendrá lugar cuando se trate de un usuario conectado a través de un dispositivo móvil (PDA). Para realizar esta consulta será necesaria la intervención de todos los módulos de los que consta la arquitectura de la solución. Los pasos que se van a llevar a cabo son los siguientes:

1. En un primer lugar, el usuario se conecta a la página inicial a través del protocolo HTTP.

2. El gestor de peticiones envía una página inicial siguiendo el protocolo HTTP.

3. Una vez que el puesto móvil recibe la respuesta, podrá realizar consultas al gestor de peticiones a través de la interfaz, para obtener: información alfanumérica o imágenes asociadas al plano, así como rutas, información de objetos,… Para este ejemplo, el usuario solicita calcular la ruta a seguir hacia un objeto.

4. El gestor de peticiones recibe la consulta, y comienza a procesarla

    - En primer lugar, por tratarse de una consulta desde un puesto móvil, se envía una petición al servidor de coordenadas para obtener la posición (x,y) del usuario.

    - A continuación, el servidor de coordenadas envía la posición (x,y) al gestor de peticiones.

5. Ahora, el gestor de peticiones deberá conocer la sala donde se encuentra el usuario y la sala donde se encuentra el objeto. Para ello:

    - Envía una petición ArcXML al servidor de mapas para obtener la sala donde se encuentra el usuario, y otra petición ArcXML para obtener la sala donde se encuentra el objeto.

    - El servidor de mapas responde a las dos consultas anteriores, indicando el nombre de la sala del usuario y el nombre de la sala del objeto.

- Si las dos salas son diferentes, entonces el gestor de peticiones envía dos peticiones ArcXML al servidor de mapas para obtener un punto (x,y) de la sala del usuario que pertenezca a la red, y la otra para obtener un punto (x,y) de la sala del objeto que pertenezca a la red. Si las dos salas son la misma, no se continúa con los pasos siguientes, entonces el gestor de peticiones envía una respuesta al usuario en forma de interfaz, indicándole que el objeto se encuentra en su misma sala.

- Finalmente, el servidor de mapas responde con dos peticiones ArcXML indicando un punto (x,y) en cada una de ellas. A partir de aquí, el gestor de peticiones tendrá la información necesaria para continuar con la consulta.

6. Ahora, el gestor de peticiones envía una consulta al servidor de rutas indicando el punto (x,y) origen (posición cercana del usuario) y el punto (x,y) destino (posición cercana al objeto).

7. El servidor de rutas procesa la petición, y construye la ruta más corta entre dos puntos. Genera un fichero XML con la información de la ruta, que se lo enviará al gestor de peticiones.

8. El gestor de peticiones recibe el XML y lo procesa. A continuación, con los puntos (x,y) que forman la ruta y con el texto asociado a estos puntos se genera un informe y una imagen.

    - Para generar la imagen de la ruta, el gestor de peticiones envía una petición ArcXML al servidor de mapas indicando las posiciones (x,y) que forman la ruta.

    - A continuación, el servidor de mapas envía una respuesta ArcXML con la imagen generada.

9. Finalmente, el gestor de peticiones muestra al usuario en forma de interfaz, siguiendo unos criterios de usabilidad, la imagen con la ruta a seguir y un informe de los pasos a seguir.

En la figura 5.3 se muestra un diagrama de actividad con los pasos que se tienen que llevar a cabo para realizar la consulta descrita.

5.3.2 Conocer la posición actual y los objetos más cercanos

Este tipo de consulta solo tendrá lugar cuando un usuario está conectado a través de un dispositivo móvil. Para realizar esta consulta no será necesaria la intervención del

módulo servidor de rutas, ya que no se necesita del cálculo de rutas para mostrar la respuesta al usuario. Aquí si que se realizará una consulta imagen y una consulta alfanumérica, con lo que quedara más claro estos tipos de consultas al servidor de mapas. Los pasos que se llevarán a cabo son los siguientes:

1. El usuario se conecta a la página inicial con el protocolo HTTP.

2. El gestor de peticiones envía una página inicial siguiendo el protocolo HTTP.

3. Una vez que el puesto móvil recibe la respuesta, podrá realizar consultas al gestor de peticiones a través de la interfaz, para obtener: información alfanumérica o imágenes asociadas al plano, así como rutas, información de objetos, etc. Para este ejemplo, el usuario solicita conocer la posición actual y ver la información de los objetos más cercanos.

4. El gestor de peticiones recibe la consulta, y comienza a procesarla.

5. A continuación, por tratarse de una consulta desde un puesto móvil, se envía una petición al servidor de coordenadas para obtener la posición del usuario.

6. El servidor de coordenadas envía la posición (x,y) al gestor de peticiones.

7. Ahora, en el gestor de peticiones se almacena la posición (x,y) del usuario y se genera la petición ArcXML con este punto (x,y). Esta petición será de tipo imagen, para mostrar la posición del usuario en el plano, junto con los objetos más cercanos. El gestor de peticiones envía esta petición al servidor de mapas.

8. Además, el gestor de peticiones generará otra petición ArcXML pero de tipo alfanumérico para obtener la información de los objetos.

9. A continuación, el servidor de mapas responde a la petición de imagen, con una respuesta ArcXML al gestor de peticiones.

10. Y después, el servidor de mapas responde al gestor de peticiones, con otra respuesta ArcXML con la información alfanumérica, indicando el nombre de los objetos más cercanos a la posición.

11. Finalmente, el gestor de peticiones muestra al usuario en forma de interfaz, siguiendo unos criterios de usabilidad, la imagen y un listado con los objetos más cercanos.

En la figura 5.4 se muestra un diagrama de actividad con los pasos que se tienen que llevar a cabo para realizar la consulta descrita.

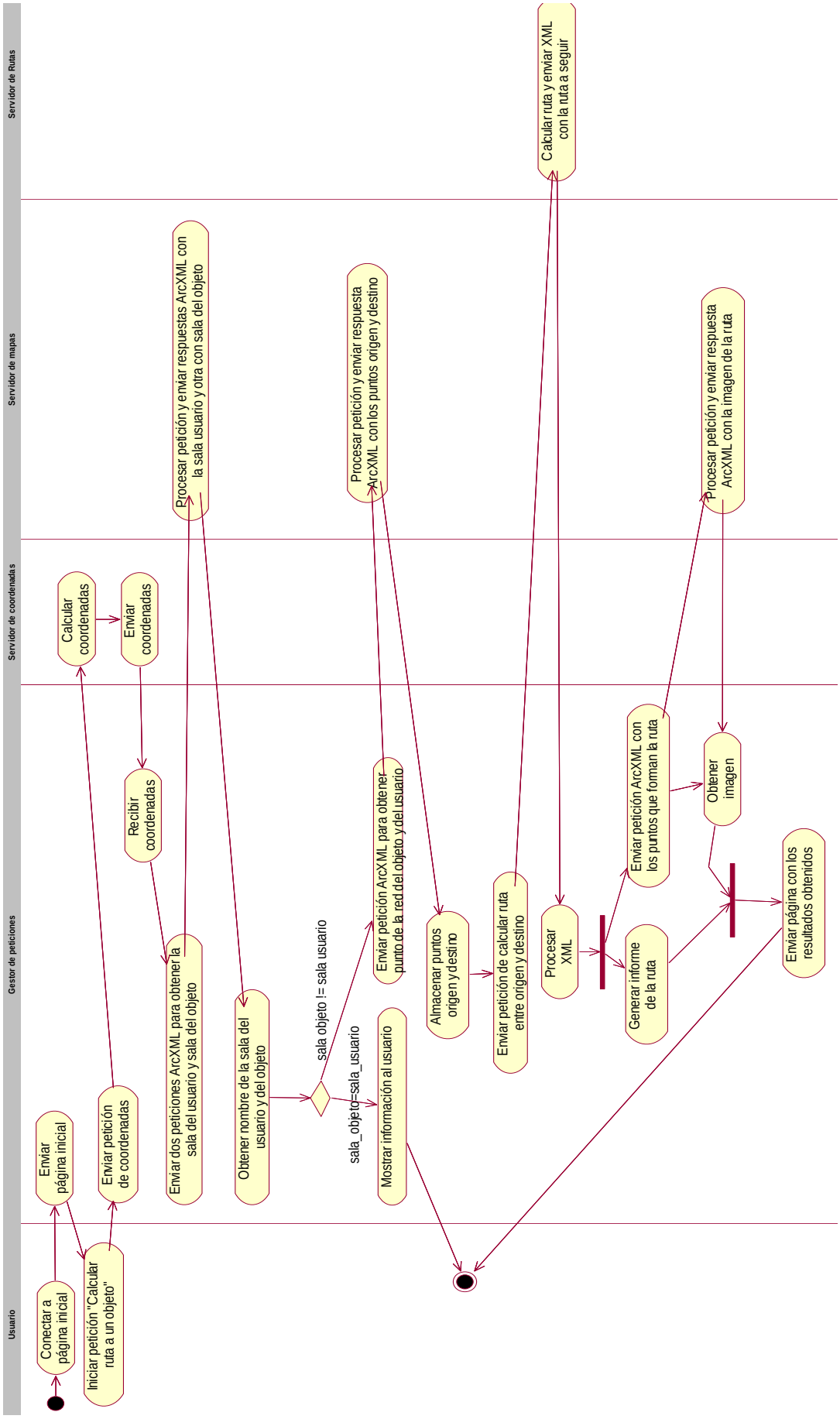


Figura 5.3. Consulta “Calcular ruta a un objeto”

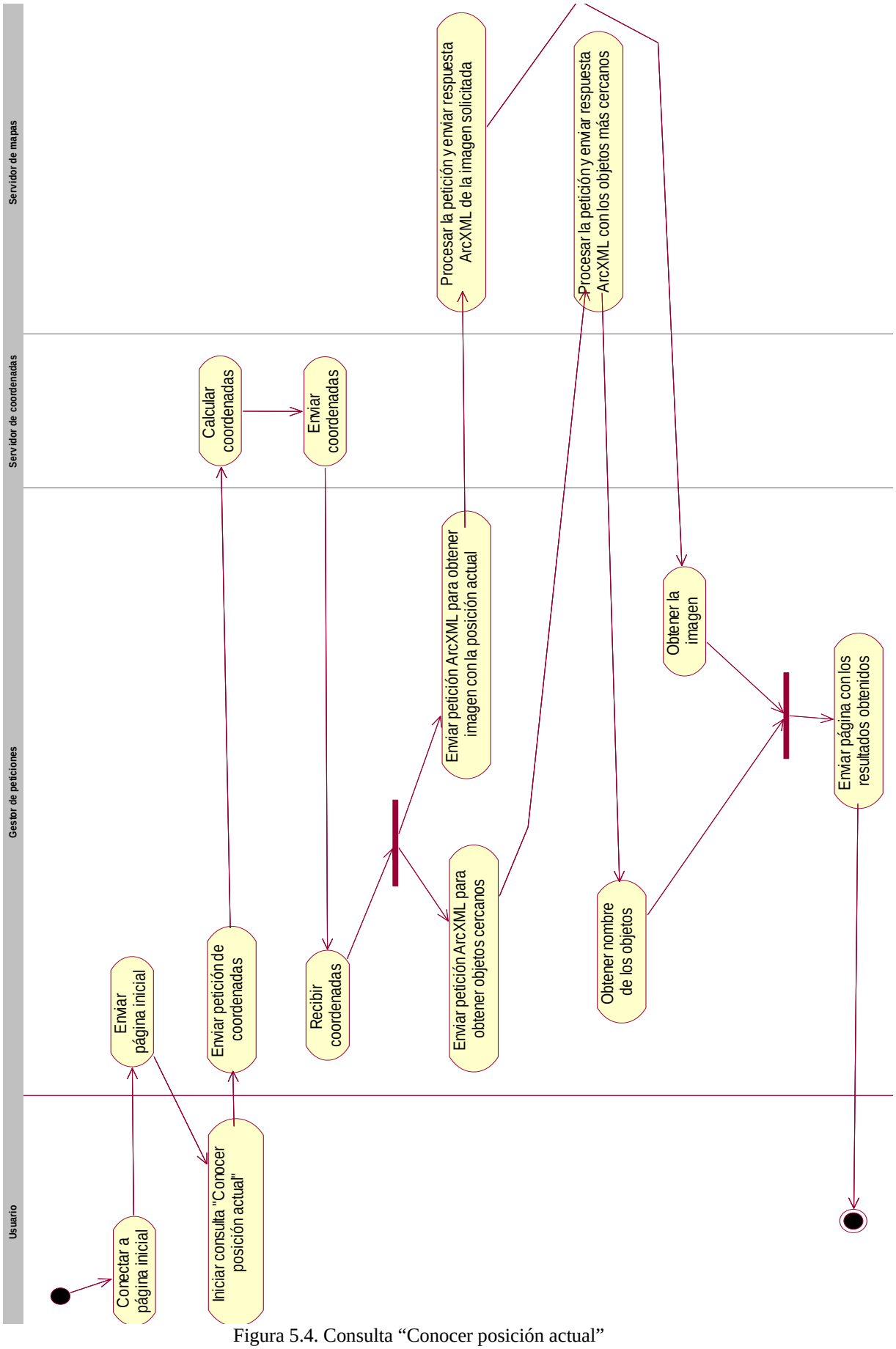


Figura 5.4. Consulta “Conocer posición actual”

5.4 TECNOLOGÍA EMPLEADA

Una vez entendido todo el funcionamiento de cada uno de los módulos de los que consta la solución se expone la tecnología empleada en cada uno de ellos.

Dado que uno de los principales requisitos del proyecto es que la aplicación desarrollada se ejecute sobre PDA, hace que se seleccione .NET como tecnología a emplear, ya que .**NET Compact Framework** proporcionado por Microsoft, está pensado para SmartPhone y Pocket PC.

Por ello, se ha utilizado .NET para cada uno de los módulos que se ha desarrollado, tanto para el gestor de peticiones, como para el servidor de rutas, así como para el servidor de coordenadas.

Otra de las causas, de por qué se ha utilizado la tecnología .NET, es porque para desarrollar el servidor de rutas se necesita un software conocido como **ArcObject**. Este software posee un API muy cómodo de utilizar para su extensión de Network Analyst para la tecnología .NET. ArcObject se explicará con más detalle en la sección 5.4.4.

Para el gestor de peticiones se decidió emplear dicha tecnología, por lo que se ha comentado anteriormente, ya que esta plataforma permite desarrollar aplicaciones para PDA, con la ventaja de facilitar, con mínimas modificaciones, su migración a teléfonos móviles en un futuro.

Hay que indicar que para comunicar el gestor de peticiones y el servidor de mapas se utiliza uno de los conectores que se explicaron en la sección 3.1.3.4, que se conoce con el nombre de **“enlace .NET”**.

Para el servidor de coordenadas se utiliza esta tecnología, por estar ya familiarizado con el entorno y el lenguaje de programación. Este servidor de coordenadas se trata de un módulo independiente, y se podrá sustituir en un futuro por cualquier sistema de localización.

Para desarrollar toda la solución se ha utilizado un IDE de uso comercial **Visual Studio 2003**, entorno de programación repleto de herramientas que contiene toda la funcionalidad necesaria para la creación de proyectos en .NET grandes y pequeños. Es posible crear, incluso, proyectos que combinan de forma homogénea módulos de lenguajes diferentes. La característica más notable del IDE es su soporte de los nuevos lenguajes .NET. Una descripción más extensa sobre el IDE Visual Studio se muestra en la sección 5.4.2.

Tanto para desarrollar el servidor de coordenadas como el servidor de rutas se ha utilizado **servicios Web XML en .NET**, ya que un servicio web aporta gran independencia entre la aplicación que usa el servicio Web y el propio servicio. De esta forma, los cambios a lo largo del tiempo en uno no deben afectar al otro. Esta flexibilidad es cada vez más importante, dado que la tendencia a construir grandes aplicaciones a partir de componentes distribuidos más pequeños es cada día más acusada. Una descripción más detallada de los servicios web se muestra en la sección 5.4.3.

Hay que señalar, que el servidor de coordenadas, el servidor de rutas y el servidor de mapas necesitan de la existencia de **IIS** como servidor web.

Una vez que ha quedado completamente definida la tecnología empleada para la total implementación del proyecto, hay que indicar que el servidor de mapas necesita para su funcionamiento, de la existencia de un servidor web y de un conector de *servlets*. Por tanto, se ha seleccionado **IIS** (Internet Information Server) como servidor web y **Apache Tomcat** como conector de servelts. Para obtener más información de estos dos componentes consultar la sección 5.4.5 y 5.4.6, respectivamente.

5.4.1 .NET

.NET es un proyecto de Microsoft para crear una nueva plataforma de desarrollo de software con énfasis en transparencia de redes, con independencia de plataforma y que permita un rápido desarrollo de aplicaciones. Basado en esta plataforma, Microsoft intenta desarrollar una estrategia horizontal que integre todos sus productos, desde el Sistema Operativo hasta las herramientas de mercado [SWXML].

.NET podría considerarse una respuesta de Microsoft al creciente mercado de los negocios en entornos Web, como competencia a la plataforma Java de Sun Microsystems.

A largo plazo Microsoft pretende reemplazar la API Win32 o Windows API con la plataforma .NET. Esto debido a que la API Win32 o Windows API fue desarrollada sobre la marcha, careciendo de documentación detallada, uniformidad y cohesión entre sus distintos componentes, provocando múltiples problemas en el desarrollo de aplicaciones para el sistema operativo Windows. La plataforma .NET pretende solventar la mayoría de estos problemas proveyendo un conjunto único y expandible con facilidad, de bloques interconectados, diseñados de forma uniforme y bien documentados, que permitan a los desarrolladores tener a mano todo lo que necesitan para producir aplicaciones sólidas.

Con esta plataforma Microsoft incursiona de lleno en el campo de los Servicios Web y establece el XML como norma en el transporte de información en sus productos y lo promociona como tal, en los sistemas desarrollados utilizando sus herramientas.

.NET intenta ofrecer una manera rápida y económica pero a la vez segura y robusta de desarrollar aplicaciones, o como la misma plataforma las denomina, soluciones, permitiendo a su vez una integración más rápida y ágil entre empresas y un acceso más simple y universal a todo tipo de información desde cualquier tipo de dispositivo. En la figura 5.5 se muestra un esquema de lo que Microsoft llama .**NET Framework,** se define .NET Framework de la siguiente forma:

- Como plataforma de desarrollo de los .Net Smart Devices que forma parte de la iniciativa .NET de Microsoft. El .NET Framework lleva el código manejado y los XML Web Services a los dispositivos móviles inteligentes, facilita la ejecución de aplicaciones seguras en dispositivos como PDAs y teléfonos móviles entre otros. Otra de las ventajas que ofrece es que permite el desarrollo de aplicaciones en los lenguajes Visual Basic .NET y C#, entre otros.

  El .NET Framework es el marco donde van a correr las aplicaciones, en lugar de directamente sobre el sistema operativo del dispositivo. Los dos elementos principales del .NET Compact Framework son: el Common Language Runtime (CLR) y .NET CF Class Library.

  - **Common Language Runtime** (CLR): Es el motor de ejecución de las aplicaciones .NET, lo que en Java sería la maquina virtual Java. Proporciona un entorno de ejecución de código administrado. El código administrado es el código desarrollado con un compilador de lenguaje orientado al tiempo de ejecución. La administración de código hace referencia a la administración de memoria, subprocesos, seguridad así como la verificación y compilación del código. El código administrado o código intermedio recibe el nombre de MSIL (Microsoft Intermediate Language), ha de ser convertido a código nativo de la plataforma en particular en tiempo de ejecución. Este proceso se hace mediante un compilador JIT (Just-In-Time) que lleva incorporado el CLR.

  - **Class Library**: La librería de clases es un rico conjunto de clases, interfaces y tipos que simplifican y optimizan el desarrollo de aplicaciones .NET. Esta estrechamente relacionada con el CLR. Permite llevar a cabo tareas como el diseño de interfaces, el uso de XML, acceso a BBDD, administración de subprocesos, E/S, etc.

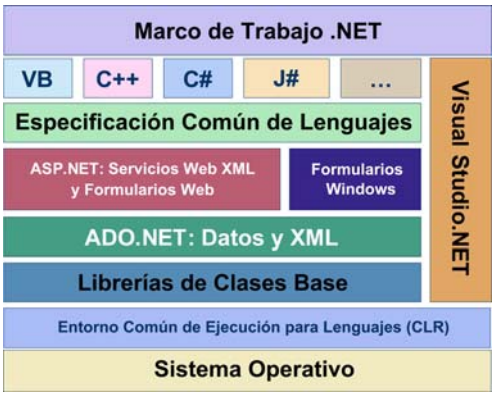


Figura 5.5. .NET Framework

5.4.2 Visual Studio 2003

Visual Studio 2003 ha sido la herramienta utilizada para desarrollar el gestor de peticiones, servidor de rutas y el servidor de coordenadas. Fue desarrollado por Microsoft a partir de 2002. Es para el sistema operativo Microsoft Windows y está pensado, principal pero no exclusivamente, para desarrollar para plataformas Win32 [WVS2003].

Algunas características destacables al IDE se encuentran en que la interfaz es más limpia y tiene una mayor cohesión. También es más personalizable con ventanas informativas de estado que automáticamente se ocultan cuando no se usan. Todas las versiones de Visual Studio, también su predecesora Visual C++, incluyen un depurador integrado en el entorno de edición.

La característica más notable del IDE es su soporte de los nuevos lenguajes .NET. Los programas desarrollados en esos lenguajes no se compilan a código máquina ejecutable (como por ejemplo hace C++) sino que son compilados a algo llamado CIL. Cuando los programas ejecutan la aplicación CIL, ésta es compilada en ese momento al código de máquina apropiado para la plataforma en la que se está ejecutando. Mediante este método, Microsoft espera poder soportar varias implementaciones de sus sistemas operativos Windows (como Windows CE). Los programas compilados a CIL pueden ejecutarse sólo en plataformas que tengan una implementación de .NET framework. Es posible ejecutar programas CIL en Linux o en Mac OS X utilizando algunas implementaciones .NET que no pertenecen a Microsoft.

Como lenguaje utilizado para el desarrollo del proyecto se ha utilizado **Visual Basic .NET.**

Visual Basic.NET (**VB.NET**) es una versión de Visual Basic enfocada al desarrollo de aplicaciones .NET. El lenguaje de programación es Visual Basic, que apareció en el año 1991 como una evolución del QuickBasic que fabricaba Microsoft.

Es un lenguaje de programación orientado a objetos (POO), y como novedades más importantes en la versión .NET, podemos citar la posibilidad de definir ámbitos de tipo, clases que pueden derivarse de otras mediante herencia, sobrecarga de métodos, nuevo control estructurado de excepciones o la creación de aplicaciones con múltiples hilos de ejecución, además de contar con la extensa librería de .NET, con la que es posible desarrollar tanto Windows Applications y Web Forms, así como un extenso número de clientes para bases de datos. Gracias a estas mejoras en lo que vendría siendo Visual Basic 7.0 los programadores de este lenguaje pueden desarrollar aplicaciones más robustas que en el pasado con una base sólida orientada a objetos.

Otras de sus características más importantes son:

- Diseño de controles de usuario para aplicaciones Windows y Web.
- Programación de bibliotecas de clase.
- Envío de datos a través de documentos XML.
- Generación de informes basados en Crystal Reports a partir de información obtenida de orígenes de datos (archivos de texto, bases, etc.)

En fin, una amplia gama de características nuevas que permiten diseñar aplicaciones escalables en pequeñas inversiones de tiempo.

5.4.3 Servicios web XML

Una de las razones por las que se ha utilizado los servicios web para desarrollar el servidor de rutas y el servidor de coordenadas, es por la gran independencia que aportan entre la aplicación que usa el servicio Web y el propio servicio.

Los servicios Web XML son los bloques de construcción básicos en la transición al proceso distribuido en Internet. Los estándares abiertos y el foco en la comunicación y colaboración entre las personas y aplicaciones han creado un entorno donde los servicios Web XML se están convirtiendo en la plataforma para la integración de aplicaciones. Las aplicaciones se construyen utilizando múltiples servicios Web XML desde diversas fuentes que trabajan conjuntamente con independencia de dónde residen o cómo hayan sido implementadas [WSWXML].

En la figura 5.6 podemos ver un esquema gráfico de los siguientes aspectos de un Servicio Web XML:

- Los Servicios Web XML exponen funcionalidad útil a los usuarios Web mediante un protocolo Web estándar. En la mayoría de casos, el protocolo utilizado es Simple Object Access Protocol (SOAP).

- Los Servicios Web XML proporcionan un modo de describir sus interfaces con suficiente detalle para permitir a un usuario construir una aplicación cliente para hablar con ellos. Esta descripción se proporciona generalmente en un documento XML que responde al nombre de documento Web Services Description Language (WSDL).

- Los Servicios Web XML se registran de modo que los potenciales usuarios puedan encontrarlos. Esto se realiza mediante Universal Discovery Description and Integration (UDDI).

Finalmente, podemos concluir diciendo que una de las ventajas principales de la arquitectura de los servicios Web XML es que permite a los programas escritos en diferentes lenguajes sobre diferentes plataformas, comunicarse entre sí de un modo basado en estándares, y es por ello por lo que se ha hecho uso de servicios web en el proyecto.

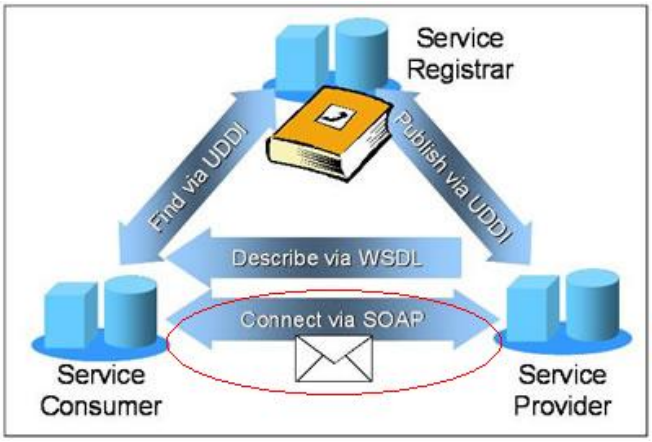


Figura 5.6. Esquema de Servicio Web XML

5.4.4 ArcObject

ArcObjects es un conjunto de componentes de software con funcionalidad GIS e interfaces programables, mediante los cuales ha sido creada cada uno de las aplicaciones de los clientes **ArcGIS Desktop** (ArcMap y ArcCatalog). Se presentan

como una colección de componentes ordenados dentro de un Modelo de Objetos [WNA].

La tecnología ArcObjects cumple con las especificaciones COM (Component Object Model), y su empleo permite desarrollar nuevas herramientas y funciones, o crear flujos de trabajo para ArcGIS Desktop (ArcView, ArcEditor y ArcInfo). También es posible, a través de desarrollos más avanzados, generar aplicaciones independientes que cumplan una funcionalidad concreta, así como añadir clases de elementos personalizadas para extender el modelo de datos de ArcGIS.

Todas las personalizaciones realizadas directamente con ArcObjects, se llevan a cabo a través de Visual Basic para aplicaciones (VBA) o lenguajes de programación que cumplen con las especificaciones COM, como Visual Basic, Visual C++ o Delphi, esta es una de las razones de por qué se ha utilizado Visual Basic .NET para el desarrollo de la parte del servidor de rutas. Para acceder al potencial de ArcObjects, es necesario tener instalada una licencia de ArcView, ArcEditor o ArcInfo.

Además de las personalizaciones básicas que es posible realizar sin ningún desarrollo, en el interfaz de usuario de ArcCatalog y ArcMap, la tecnología ArcObjects permite personalizaciones más avanzadas que pueden agruparse en dos categorías:

- **Personalización mediante VBA:** El empleo de ArcObjects a través de VBA embebido en ArcGIS, permite añadir menús y herramientas personalizadas, así como flujos de trabajo al entorno de trabajo de ArcGIS. La combinación ArcObjects/VBA es una buena opción cuando se quiere desarrollar aplicaciones que se ejecutan en el entorno de ArcGIS Desktop.

- **Empleo directo de ArcObjects**: A través de lenguajes COM como Visual Basic, Visual C++ o Delphi. Este desarrollo permite a programadores ampliar el modelo de datos de la geodatabase con elementos personalizados, y crear módulos de software reutilizables.

5.4.5 Internet Information Server (IIS)

Internet Information Services (**o Server**), IIS, es una serie de servicios para los ordenadores que funcionan con Windows. Originalmente era parte del *Option Pack* para Windows NT. Luego fue integrado en otros sistemas operativos de Microsoft destinados a ofrecer servicios, como Windows 2000 o Windows Server 2003. Windows XP Profesional incluye una versión limitada de IIS. Los servicios que ofrece son: FTP, SMTP, NNTP y HTTP/HTTPS [WIIS].

Este servicio convierte a un computador en un servidor de Internet o Intranet, es decir, que en las computadoras que tienen este servicio instalado se pueden publicar páginas web, tanto local como remotamente. Este servicio debe de estar instalado en los puestos donde este ejecutándose el servidor de rutas, el servidor de coordenadas, el gestor de peticiones y el servidor de mapas.

5.4.6 Apache Tomcat

Apache Tomcat es un contenedor *servlets* que usa la implementación de referencia oficial de Java *Servlets* y *JavaServer Pages* (JSP). Tomcat ha sido desarrollado bajo un entorno abierto y colaborativo, y trabaja bajo licencia de Apache Software [WTOM].

Apache Tomcat se debe tener instalado en el puesto de trabajo donde se encuentre alojado el servidor de mapas (ArcIMS), ya que lo necesita para su funcionamiento.

Capítulo 6. CASO DE ESTUDIO

"La inteligencia consiste no sólo en el conocimiento, sino también en la destreza de aplicar los conocimientos en la práctica"

Sócrates (470a.c - 390a.c.)
Filósofo griego.

En este capítulo se expondrán una serie de casos prácticos para comprender mejor la finalidad y la forma de utilizar la aplicación, teniendo en cuenta el estudio realizado en los capítulos anteriores.

6.1 INTRODUCCIÓN

En el Capítulo 5 se ha descrito la arquitectura de la solución y su funcionamiento, mientras que en el capítulo actual se expondrán una serie de ejemplos prácticos para que quede más claro el funcionamiento de la aplicación.

Estos ejemplos consistirán en, (i) consultar la posición actual del usuario conectado a la aplicación a través de una PDA, (ii) calcular la ruta hacia un objeto seleccionado de una lista, (iii) calcular un recorrido de las rutas recomendadas, y por último (iv) obtener información a través de la aplicación PC de un objeto y una sala.

Todas las pantallas siguen la estructura que se detalla en la figura 6.1: en la parte superior se muestra siempre visible el título de la aplicación y se insertará una ayuda contextual en caso de que sea necesaria. En medio de la pantalla se ofrece la descripción de la consulta solicitada, y en la parte inferior se tiene la botonera con las acciones posibles que pueden realizarse desde esa ventana.

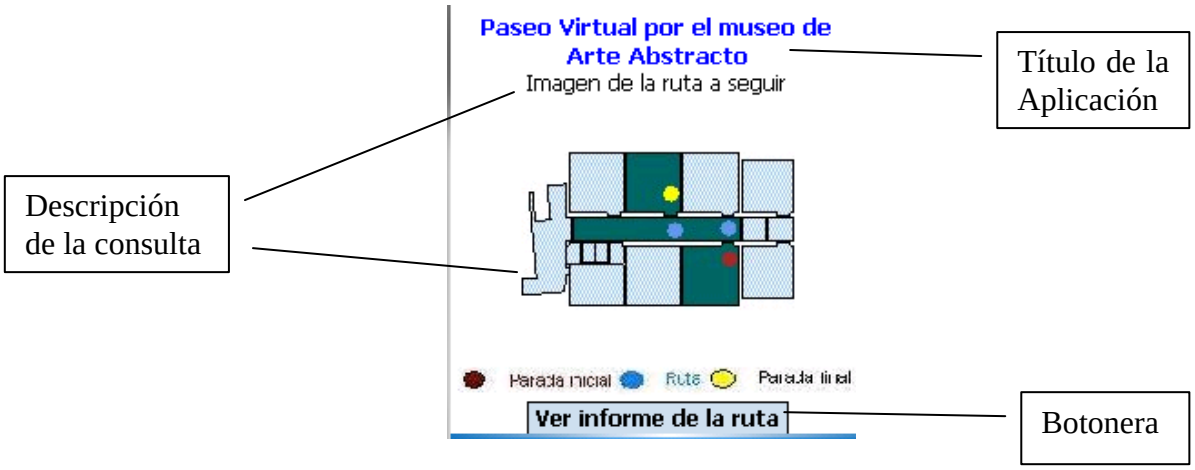


Figura 6.1. Estructura de las pantallas

6.2 INFORMACIÓN DEL SISTEMA

El caso de estudio sobre el que se aplican los conceptos vistos en los capítulos anteriores consiste en una aplicación para un museo de una sola planta, que consta de salas, las cuales tienen cuadros y obras distribuidos por cualquier parte de la sala. Cada una de las salas tienen un nombre por ejemplo: Pasillo, Entrada, Sala 1,…

De los cuadros y obras se tiene información como son: nombre, imagen, autor,… A estos cuadros y obras se les llama en la aplicación objetos, y se trata de objetos de arte abstracto, en concreto obras del Museo Nacional Centro de Arte Reina Sofía [WMAARS], y Museo de Arte Abstracto Español de Cuenca [WMAC].

Para permitir el cálculo de rutas se han definido unos pasillos, que mostrarán al usuario por donde debe desplazarse. Estos pasillos representan el cambio de una sala a otra para seguir la ruta calculada. De esta estructura de pasillos definida sólo se almacena su nombre, como por ejemplo, PASILLO11D.

En las secciones siguientes, se exponen con más detalle los cuatro casos de estudio que se van a realizar.

6.3 POSICIÓN ACTUAL

Esta consulta detallará la posición actual del usuario conectado a la página de la aplicación web, a través de una PDA. También se mostrará los objetos más cercanos a su posición, indicando su nombre en un listado y mostrándolos en la imagen. Se han considerado objetos cercanos, aquellos que se encuentren a una distancia menor de 1 metro de la posición actual.

Para poder comenzar la consulta el usuario debe conectarse a la página inicial de la aplicación web. El aspecto de esta página se puede ver en la figura 6.2.

Figura 6.2. Página inicial de la aplicación PDA

Una vez que el usuario se encuentra conectado a la página inicial de aplicación, se muestra una visión general del sistema indoor. A continuación, se debe pulsar sobre el botón *Hacer consultas*, y se mostrará una serie de botones que indicarán las diferentes acciones que se pueden hacer sobre el sistema indoor (ver figura 6.3). Como en este caso la consulta que se quiere realizar es conocer la posición actual, el usuario deberá pulsar sobre el botón *Posición Actual*.

Figura 6.3. Página para iniciar consultas

Si la consulta se ha resuelto con éxito, se mostrará una página similar a la de figura 6.4, donde se puede ver una imagen y un listado. La imagen indica la posición actual del usuario en forma de estrella y los objetos más cercanos en forma de puntos. Y en el listado se indica el nombre de los cuadros más cercanos. Como se puede ver en la figura 6.4, para este ejemplo, *La ventana del pintor* y el *Retrato de Josette* son los cuadros más cercanos. También se informa del nombre de la sala actual, *Sala 1*.

El usuario puede consultar la leyenda para entender mejor la simbología utilizada, como se puede ver en la figura 6.5.

Si el usuario no tiene cuadros cercanos sólo se mostrará la imagen indicando su posición y se le informará de la sala en la que se encuentra. Como se puede ver en la figura 6.7.

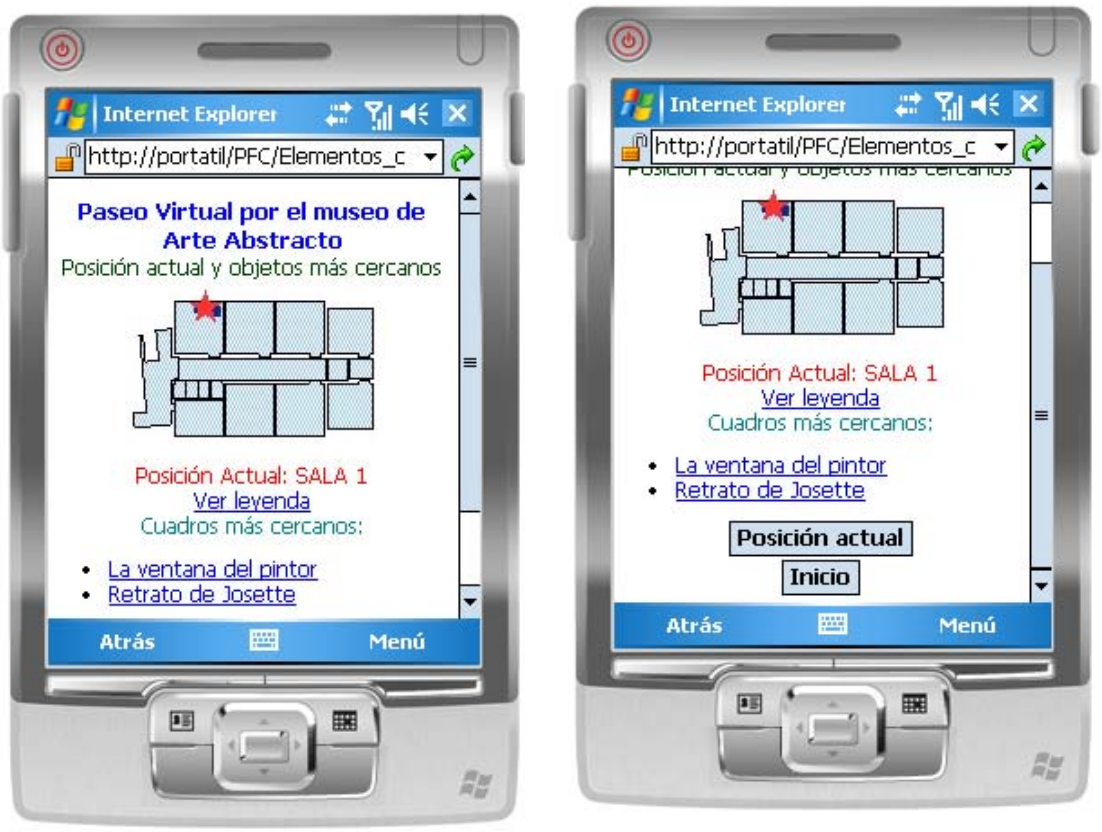


Figura 6.4. Posición Actual y cuadros cercanos

Figura 6.5. Leyenda de posición actual

Finalmente, desde la página anterior, figura 6.4, se puede visitar los enlaces para consultar la información de cada uno de los cuadros obtenidos. En la figura 6.6 se puede ver el resultado de visitar el cuadro *La ventana del pintor*.

Figura 6.6 La ventana del pintor

Figura 6.7. Posición Actual, sin cuadros cercanos

En la figura 6.7, al igual que en la figura 6.4, se tiene un botón para poder consultar la posición actual en cualquier momento dentro del sistema indoor. Si se quiere volver a la página inicial se pulsará el botón *Inicio.*

6.4 CALCULAR RUTA A UN OBJETO

Esta consulta mostrará la ruta a seguir desde la posición actual del usuario hasta el objeto seleccionado. El usuario estará conectado a la página de la aplicación web, a través de una PDA.

Para poder comenzar la consulta el usuario debe conectarse a la página inicial de la aplicación web. El aspecto de esta página se puede ver en la figura 6.2. A continuación, se debe pulsar sobre el botón *Hacer consultas*, y se mostrará una serie de botones que indicarán las diferentes consultas que se pueden hacer sobre el sistema indoor, como se mostró en la figura 6.3. En este caso, la consulta que se quiere realizar es sobre algún objeto, por tanto, el usuario deberá pulsar sobre el botón *Objetos*.

Al iniciar la consulta aparecerá una página similar a la de la figura 6.8. Desde esta página se seleccionará un objeto, y se podrá realizar las siguientes acciones: (i) consultar información del objeto y (ii) calcular ruta al objeto.

Figura 6.8. Página con acciones sobre objetos

El usuario seleccionará un objeto de la lista desplegable y pulsará sobre el botón *Ruta al objeto*. Para el ejemplo que se muestra en la figura 6.9, el usuario selecciona el cuadro *Muchacha de espaldas*.

Si el objeto seleccionado se encuentra en la misma sala que el usuario, se informará al usuario de que el objeto está en la sala actual, como se puede ver en la figura 6.9. Si el objeto se encuentra en otra sala diferente a la del usuario, se mostrará la ruta a seguir como se puede ver en la figura 6.10.

En la figura 6.10 se muestra la imagen con la ruta a seguir para llegar al objeto seleccionado, en este caso, se ha seleccionado el cuadro *El hombre invisible*. Se muestra una leyenda para indicar la posición inicial donde está el usuario, la ruta a seguir y la posición final donde se encuentra el objeto. Para ver un informe más detallado de la ruta se pulsará el botón *Mostrar informe de la ruta*.

Figura 6.9. Ruta al objeto que está en la misma sala que el usuario.

Figura 6.10. Imagen de la ruta al objeto

En la figura 6.11, se muestra el informe de la ruta a seguir y la longitud de la ruta. Se podrá visitar los enlaces del informe para ver una descripción más detallada de cada caso.

Figura 6.11. Informe de la ruta al objeto

Estos enlaces se clasifican en 2 categorías, los que representan a la posición inicial o posición final y los que representan alguna indicación a seguir.

En el primer caso, se mostrará, por un lado, una visión general del sistema indoor indicando con un punto la posición actual según el texto del enlace visitado y por otro lado, un listado con los objetos que se pueden consultar en esta sala. En la figura 6.12 se puede ver el resultado de visitar un enlace de este tipo.

En el segundo caso, se mostrará una imagen ampliada de esa zona del plano y una vista general del sistema indoor indicando con una estrella la posición. Dependiendo de si se encuentran objetos por el camino se mostrará un botón que permitirá ver el nombre de dichos objetos. En la figura 6.16 se puede ver el resultado de visitar un enlace de este tipo.

Figura 6.12. Posición inicial y Posición final

Si se quiere volver al informe de la ruta se tiene la opción *Atrás*, y si se desea volver a la página inicial se pulsará el botón *Inicio*. Si se pulsa sobre uno de los enlaces de los objetos aparecerá su información, como se puede ver en la figura 6.13a y 6.13b. Desde esta página se puede ver la imagen del cuadro pulsando sobre el botón *Ver imagen*, como se puede ver en la figura 6.14a y 6.14b.

Figura 6.13a. Información del cuadro *La mujer de azul*

Figura 6.14a Imagen del cuadro *La mujer de azul*

Figura 6.13b. Información del cuadro *El hombre invisible*

Figura 6.14b. Imagen del cuadro *El hombre invisible*

Si del informe de la ruta se visita un enlace diferente de los descritos anteriormente, se mostrará una página similar a la de la figura 6.16. Por un lado, se muestra una imagen ampliada de esa zona del plano, mostrando los objetos que se encuentran al paso, y por otro lado, se muestra una imagen indicando la zona del sistema indoor a la que hace referencia mediante el nombre del enlace, destacando en forma de estrella la posición exacta a la que hace referencia el enlace.

Como se ha comentado anteriormente, si se pulsa sobre un enlace de alguno de los pasos a seguir para llegar al destino, se podrán dar dos casos: (i) que no se encuentren objetos al paso o (ii) que se vayan encontrando objetos al paso.

Si no se encuentran objetos al paso, se mostrará una pantalla similar a la de la figura 6.15, donde no se tendrá la opción de *Ver objetos cercanos*, ya que en esa zona no se encuentran objetos.

Figura 6.15. Información del PASILLO2-3

Si se han encontrado objetos al paso, entonces aparecerá el botón *Ver objetos cercanos*, desde el que se podrá acceder para conocer de que objetos se trata, como se puede ver en la figura 6.16.

Figura 6.16. Información del *PASILLO11D*

Una vez que el usuario ha pulsado el botón *Ver objetos cercanos*, aparecerá una página similar a la figura 6.17. Por un lado, se muestra una imagen ampliada de esa zona del plano y por otro lado, un listado con el nombre de los objetos que se encuentran al paso.

Si se pulsa sobre un enlace de los objetos, aparecerá la información del objeto en cuestión, y pulsando el botón *Ver imagen*, se mostrará su imagen. En las figuras 6.18 y 6.19 se muestran la información del cuadro *El carnaval del Arlequín* y la imagen del cuadro, respectivamente.

Figura 6.17. Cuadros del *PASILLO11D*

Figura 6.18. Información del cuadro *El carnaval del Arlequín*

Figura 6.19. Imagen del cuadro *El carnaval del Arlequín*

6.5 RECORRIDO DE LOS IMPRESIONISTAS MÁS DESTACADOS

Esta consulta mostrará la ruta a seguir para conocer los impresionistas más destacados. El usuario estará conectado a la página de la aplicación web a través de una PDA.

Para poder comenzar la consulta el usuario debe conectarse a la página inicial de la aplicación web. El aspecto de esta página se mostró en la figura 6.2. A continuación, se debe pulsar sobre el botón *Hacer consultas*, y se mostrará una serie de botones que indicarán las diferentes consultas que se pueden hacer sobre el sistema indoor, como se mostró en la figura 6.3. En este caso, la consulta será sobre una ruta recomendada, por tanto, se pulsará el botón *Rutas recomendadas*.

Aparecerá una página similar a la de la figura 6.20, donde se muestra los diferentes recorridos que se pueden hacer sobre el sistema indoor en forma de asistente. Para este ejemplo, se pulsará el enlace *Impresionistas más destacados*.

A continuación, aparecerá una página indicando la longitud de la ruta que se ha consultado, y una imagen resaltando en un color más oscuro las salas que se van a visitar. Como acciones posibles en esta página se tiene: un botón *Mostrar informe* para mostrar los pasos a seguir, y un botón *Inicio*, para volver a la página inicial. En la figura 6.21 se puede ver el resultado descrito.

Figura 6.20. Listado de rutas recomendadas

Figura 6.21. Página principal de la ruta de los impresionistas

Si el usuario ha seleccionado la opción de *Mostrar informe*, aparecerá una imagen similar a la de la figura 6.22, donde se muestra en primer lugar una imagen con la ruta hasta la primera parada, y en segundo lugar, un listado con las indicaciones a seguir. Como acciones posibles se tiene un botón *Siguiente*, que mostrará la ruta a seguir hacia la siguiente parada.

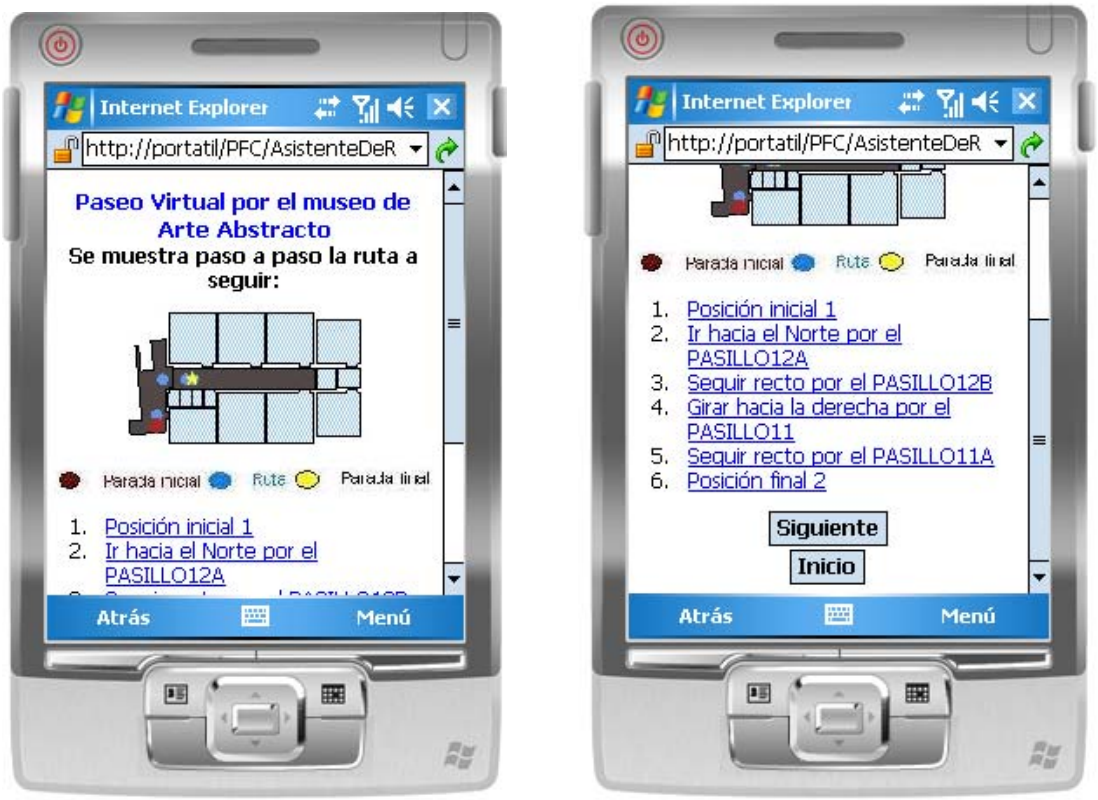


Figura 6.22. Ruta de impresionistas paso 1

Si el usuario desea conocer el siguiente paso de la ruta aparecerá una pantalla similar a la figura 6.23, donde se muestra al igual que en el caso anterior, una imagen indicando la ruta a seguir y un informe en forma de listado para detallar las indicaciones.

Figura 6.23. Ruta de impresionistas paso 2

Figura 6.24. Parada final de la ruta de impresionistas

Cuando se llega a la última parada del recorrido, el botón *Siguiente* desaparece, entonces las acciones posibles que se tendrán serán: volver a la página inicial con el botón *Inicio*, o visitar un enlace del informe de la ruta, como se puede ver en la figura 6.24.

Los enlaces del informe de la ruta se pueden visitar. Estos enlaces se clasifican en 2 categorías: (i) los que representan a la posición inicial o posición final, y (ii) los que representan alguna indicación a seguir.

En el primer caso se mostrará, por un lado, una visión general del sistema indoor indicando con un punto la posición actual según el texto del enlace visitado y por otro lado, un listado con los cuadros clasificados por impresionistas. En la figura 6.25 se pueden ver los resultados de visitar un enlace para este caso.

En el segundo caso, se mostrará una imagen zoom a esa zona del plano y una vista general del sistema indoor indicando con una estrella la posición, dependiendo de si se encuentran objetos por el camino se mostrará un botón para ver de qué objetos se trata. En la figura 6.26 se puede ver el resultado de visitar un enlace para este caso.

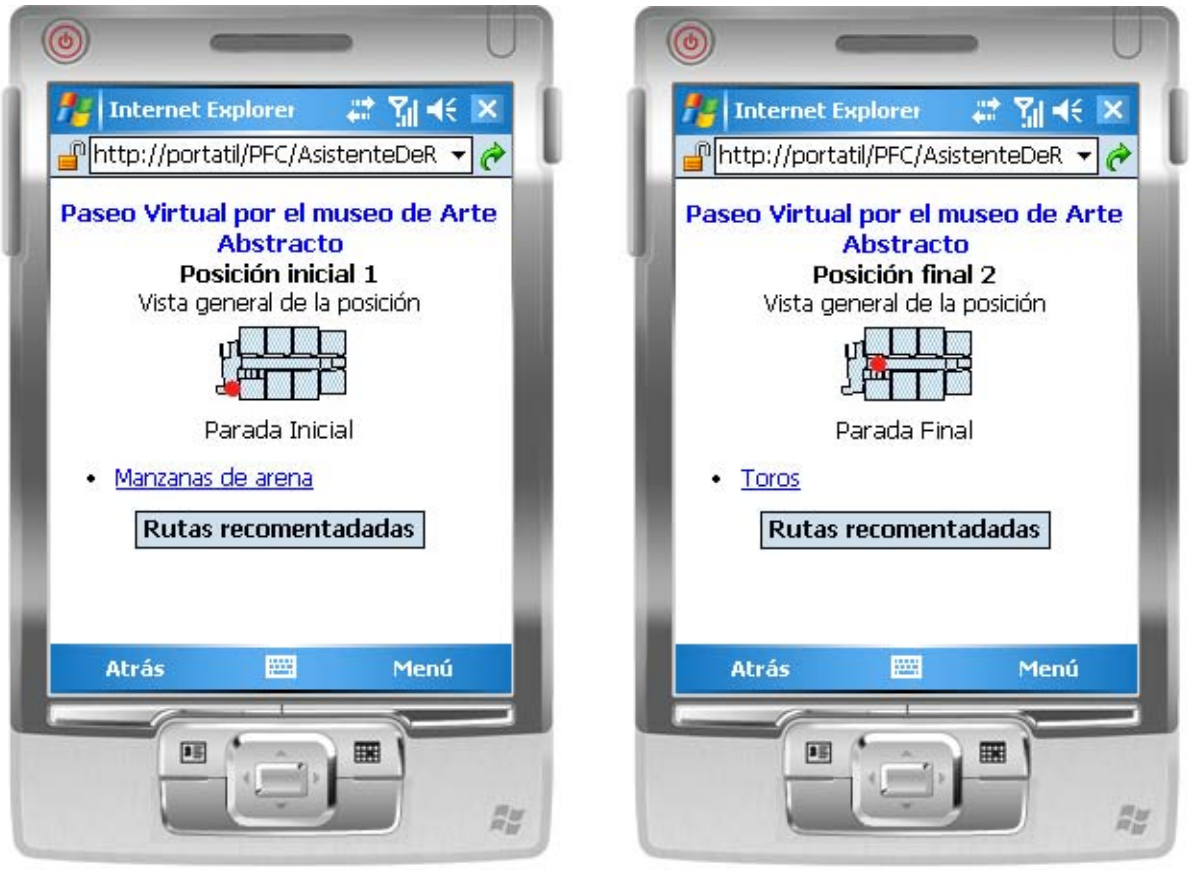


Figura 6.25. Posición inicial y posición final de ruta de impresionistas.

Como se puede ver en la figura 6.25, se mostrará en forma de listado los objetos que se encuentran al paso, pero sólo se mostrará los cuadros clasificados como impresionistas. Si se visita el enlace del cuadro aparecerá su información y un botón para ver su imagen.

Para los enlaces que representan las indicaciones a seguir se pueden dar dos casos: (i) aquellos que no se encuentran objetos en el camino, y (ii) aquellos que si se encuentran objetos en el camino. Para el primer caso, la pantalla que aparecerá será similar a la de la figura 6.26, y para el según caso se muestra en la figura 6.27.

Figura 6.26. Indicaciones intermedias sin objetos.

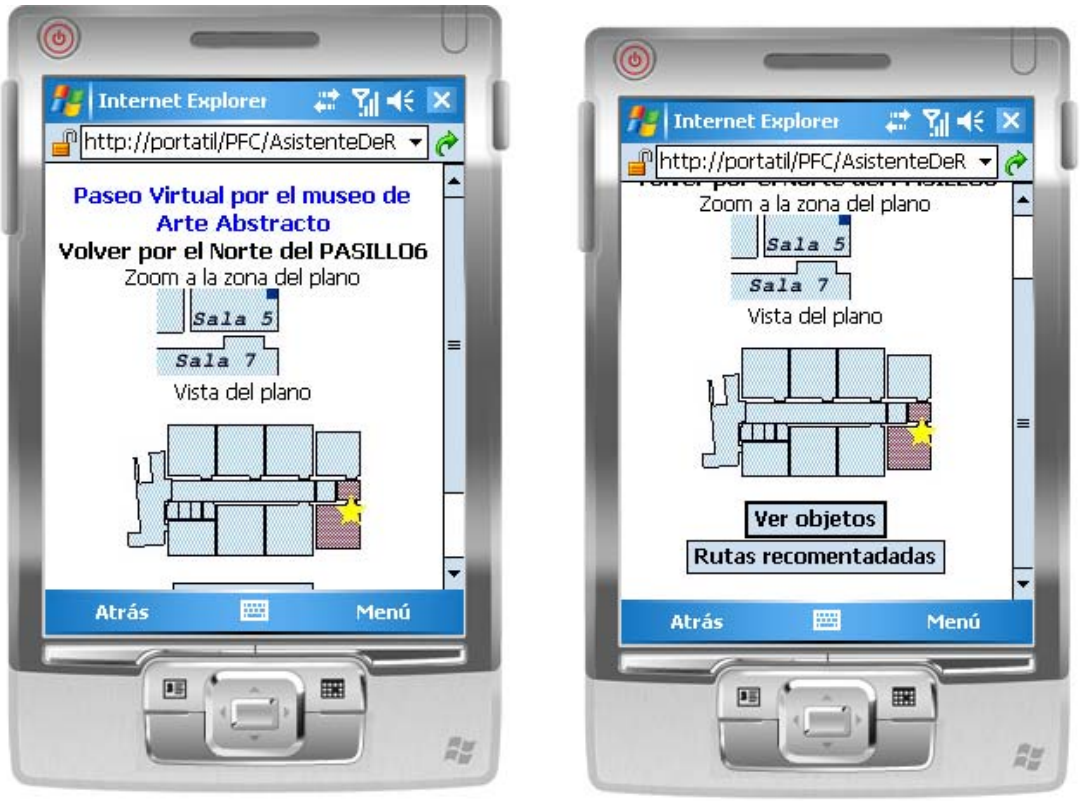


Figura 6.27. Indicaciones intermedias con objetos.

Figura 6.28. Objetos de una posición intermedia

Figura 6.29. Última parada del recorrido

Como acciones asociadas a la pantalla de la figura 6.27 se tendrá: ver objetos que se encuentran en el camino o volver a la página inicial, donde se elegirá otro recorrido de rutas recomendadas. Para ver los objetos, el usuario deberá pulsar el botón *Ver objetos*, entonces se mostrará una imagen similar a la de la figura 6.28, con una imagen ampliada de esa zona del plano y un listado con el nombre de los objetos, que en este caso son impresionistas.

Si se va avanzando en el recorrido con el botón *Siguiente*, se llegará a la posición final, que en el caso de los impresionistas, se mostrará la pantalla de la figura 6.29.

Como acciones asociadas a la pantalla de la figura 6.29, se tiene: volver a consultar otro recorrido de rutas recomendadas, o ver la información del objeto. Si se visita el enlace del objeto se mostrará su información. En este ejemplo se trata del *El peine del viento I*, como se puede ver en la figura 6.30.

Si el usuario desea conocer la imagen del objeto pulsará el botón *Ver imagen* y se mostrará una pantalla similar a la de la figura 6.31. Sino el usuario podrá regresar a la pantalla anterior con la opción *Atrás* o pulsar el botón *Volver a Rutas recomendadas* para consultar otro recorrido.

Figura 6.30. Información del objeto *El peine de viento I*

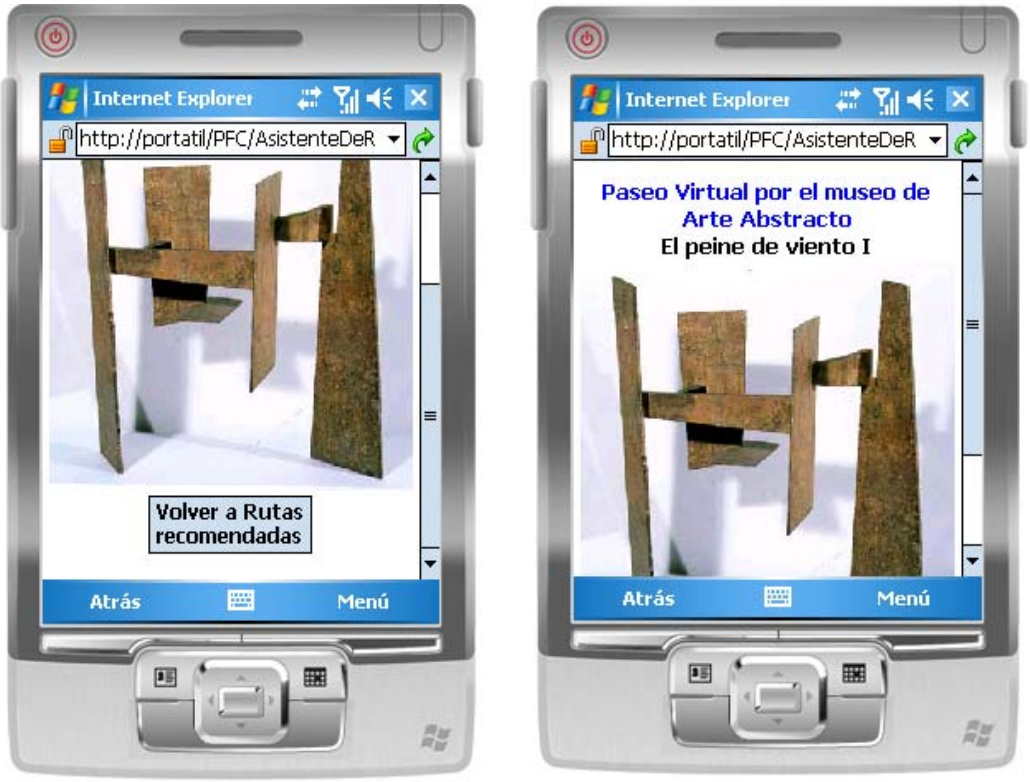


Figura 6.31. Imagen del objeto *El peine de viento I*

6.6 IDENTIFICACIÓN

Esta consulta mostrará la información de un objeto o de una sala pulsando en la pantalla con la ayuda del ratón. Para ello, el usuario estará conectado a la página de la aplicación web a través de un PC.

Para poder comenzar la consulta el usuario debe conectarse a la página inicial de la aplicación web. El aspecto de esta página se puede ver en la figura 6.32.

Aparece una página con un índice en forma de pestañas donde se podrán realizar diferentes consultas. En la página inicial se muestra un visor web sobre el que se podrán realizar consultas básicas sobre el sistema indoor, como por ejemplo: *zoom a una sala*, *ir al norte del plano*,… Para el ejemplo en cuestión, el usuario deberá pulsar el botón *Identificar* que se encuentra en la barra de iconos de la izquierda.

A continuación, se tendrá dos posibilidades: identificar un objeto para obtener su información, o bien, identificar una sala para obtener su información.

Para el primer caso, se deberá asegurar que la capa cuadros que se muestra en la leyenda se encuentra seleccionada. Para el segundo caso la capa que se deberá seleccionar es la de salas.

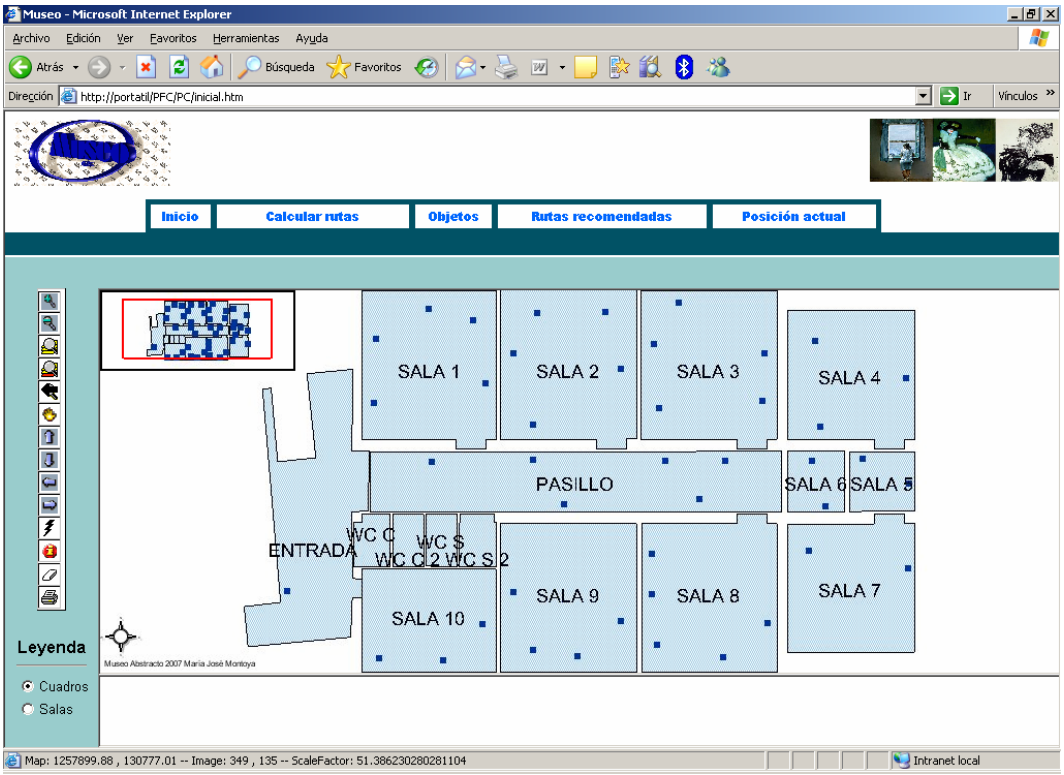


Figura 6.32. Página inicial para PC

Si el usuario ha seleccionado un cuadro, se mostrará su información en la parte inferior de la pantalla, como se puede ver en la figura 6.33. Aparecerá la información, y un enlace a la imagen del cuadro en cuestión.

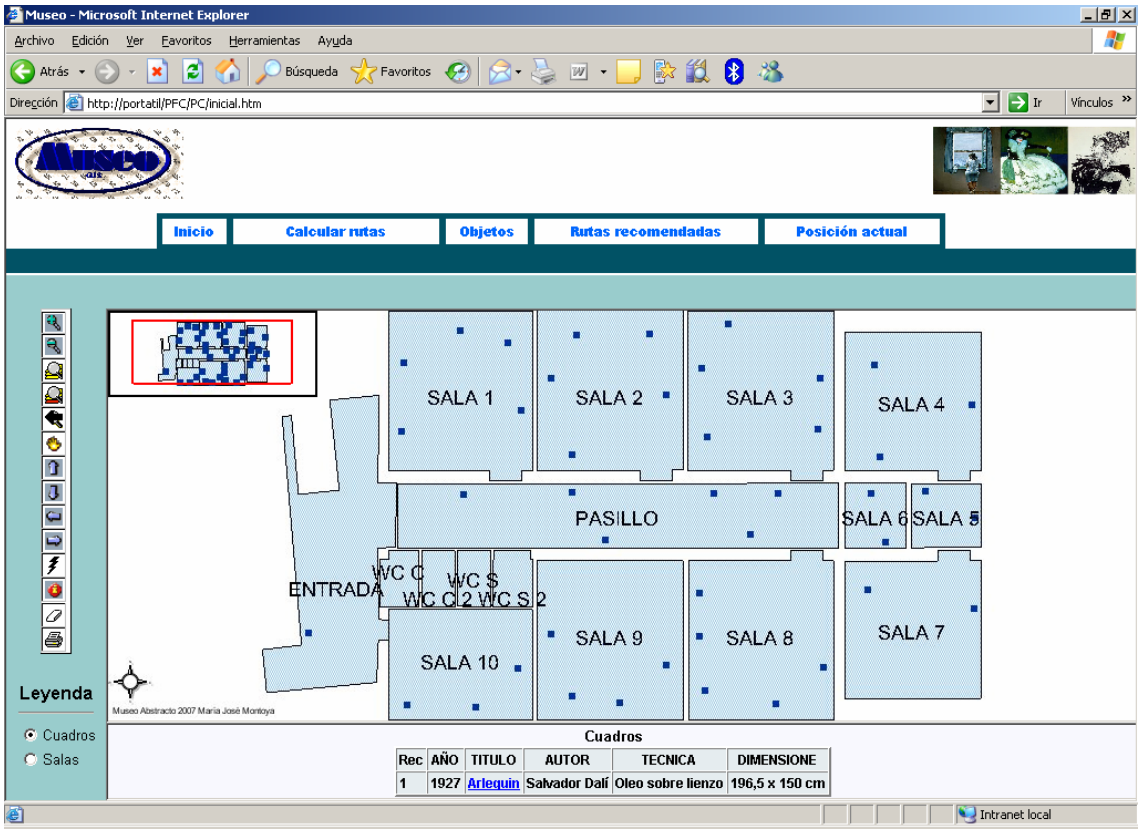


Figura 6.33. Información de un objeto en PC.

Si el usuario visita el enlace del cuadro se mostrará una página con la imagen, como se muestra en la figura 6.34.

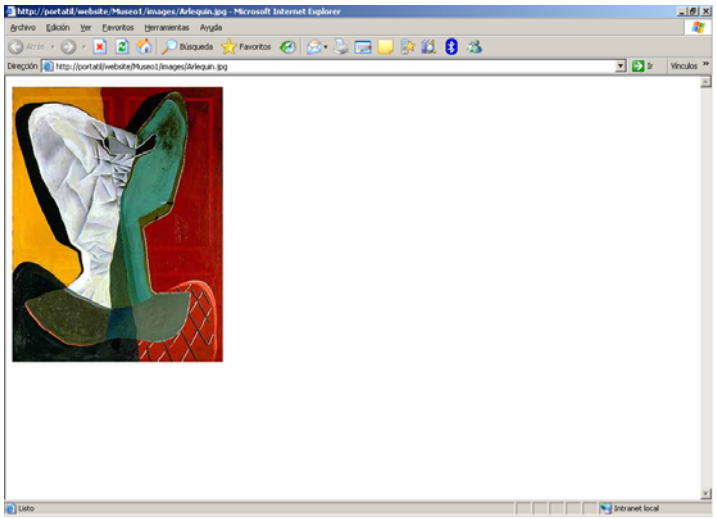


Figura 6.34. Imagen del cuadro *Arlequín*

Si el usuario selecciona la capa *Salas*, y a continuación pulsa sobre una sala, aparecerá su información, como se muestra en la figura 6.35.

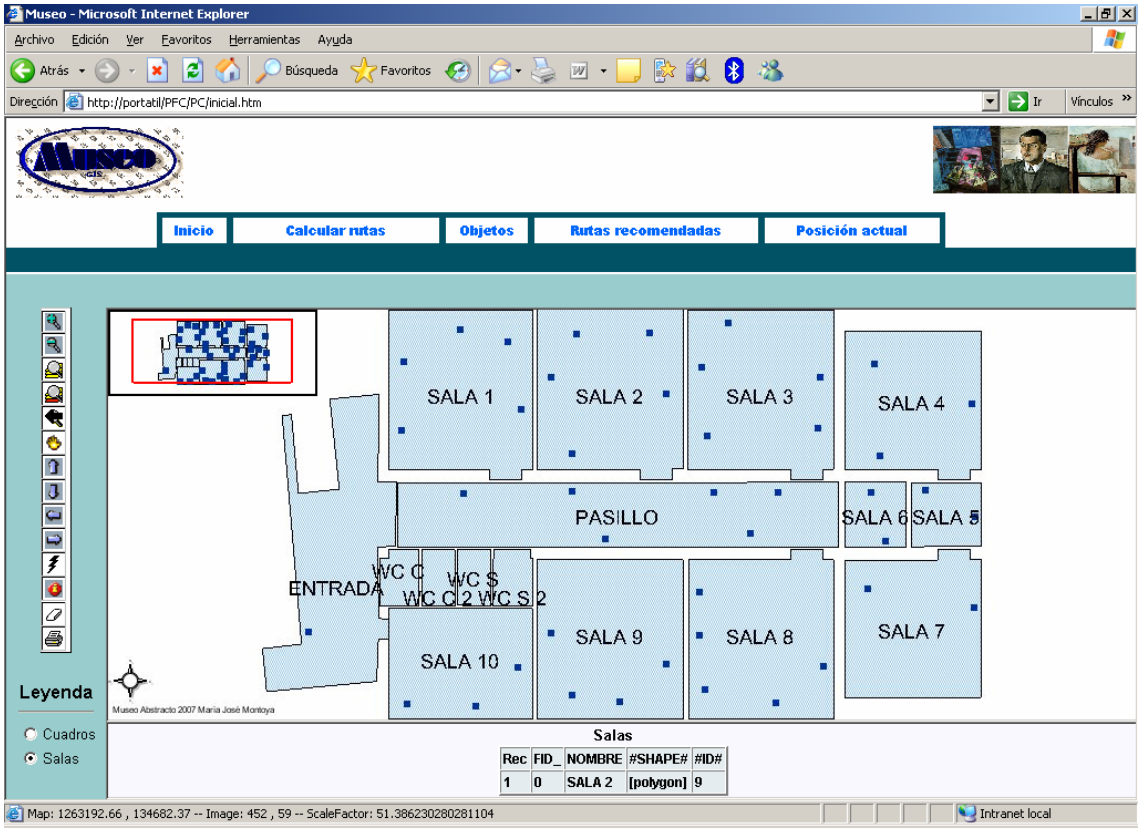


Figura 6.35. Información de la *Sala 2*

Una vez expuesto varios casos de estudio sobre el proyecto desarrollado, se pasará a concluir el trabajo realizado, así como exponer una serie de trabajos futuros que pueden llevarse a cabo.

Capítulo 7. CONCLUSIONES Y TRABAJOS FUTUROS

"La vida es el arte de sacar conclusiones suficientes a partir de datos insuficientes"

Samuel Burlet (1612 - 1680)
Poeta inglés

7.1 CONCLUSIONES DEL TRABAJO

Como se detalló en el Capítulo 1 de esta memoria, el objetivo de este proyecto fin de carrera ha sido:

Estudiar, diseñar e implementar una arquitectura software que, basada en sistemas de localización indoor, permita la ejecución de funcionalidades básicas en este tipo de sistemas como son: cálculo de rutas en interiores, conocer la posición actual de un usuario, obtener información de un objeto, conocer los objetos más cercanos al usuario, etc.

Este objetivo ha sido satisfecho completamente durante el desarrollo, aunque no ha estado exento de dificultades. La solución de esas dificultades (mediante adaptaciones, matizaciones, concreciones, etc.) ha aportado nuevos requisitos y puntos de vista lo cual a su vez han enriquecido el proyecto.

A lo largo del proyecto se ha estudiado la forma de ejecutar consultas de localización en ambientes indoor, con el fin de permitir al usuario realizar consultas espaciales destinadas a interrogar elementos de interés situados en interiores.

Para conseguir el objetivo, se ha comenzado realizando un estudio de diferentes servidores de mapas, seleccionando el que se ha deducido que resultaría mejor para el desarrollo del proyecto. Como se vio en el Capítulo 2 el servidor de mapas elegido para

el desarrollo ha sido ArcIMS por ser el software comercial que cuenta con mayores funcionalidades en comparación con los servidores de mapas de software libre y porque es el líder indiscutible dentro de las preferencias de los usuarios a nivel mundial.

Además, se ha presentado un estudio sobre diferentes alternativas para desarrollar el cálculo de rutas. Ésta es una parte importante de la arquitectura que se ha estudiado en detalle en el Capítulo 5. Se ha realizado una comparativa de las dos alternativas posibles y se ha seleccionado *Network Analyst* como mejor solución para realizar el cálculo de rutas. En resumen, el principal motivo para su elección ha sido que Network Analyst cuenta para su desarrollo de algoritmos suficientes para obtener la ruta más corta entre un origen y un destino.

Tomando como base los estudios realizados, siguiendo una metodología ágil, se han capturado los requisitos del sistema de localización. En base a estos requisitos se han obtenido un conjunto de consultas cuyo objetivo es servir de conjunto *básico* para sistemas de localización indoor. Las consultas realizadas han sido:

- Obtener una visión general del sistema indoor.
- Obtener un listado con todas las salas del sistema indoor.
- Obtener un listado con todos los objetos (cuadros) del sistema indoor.
- Obtener información de un objeto identificado por su nombre seleccionado de una lista.
- Obtener posición de un objeto identificado por su nombre seleccionado de una lista.
- Conocer el nombre de los objetos más cercanos a una posición (x,y).
- Mostrar la posición de los objetos más cercanos a una posición (x,y), en forma de imagen.
- Obtener la ruta hacia un objeto desde la posición (x,y) del usuario.
- Obtener la ruta hacia una sala desde la posición (x,y) del usuario.
- Obtener ruta entre dos salas seleccionadas de un listado.
- Conocer el cambio de la posición actual. Es decir, se está en una posición (x,y) y se ha detectado un cambio con respecto a la posición anterior, por ejemplo, se ha cambiado de sala.
- Zoom de la sala actual. Es decir, obtener imagen de la sala actual de forma ampliada.

- Consultar el nombre de la sala donde se encuentra un objeto.
- Consultar el nombre de la sala actual donde se encuentra el usuario.
- Conocer la ruta de un recorrido definido de antemano, tanto el informe indicando los pasos que se deben seguir como la imagen indicando la ruta a seguir. Por ejemplo: Recorrido de los impresionistas más destacados.
- Consultar las salas que intervienen en un recorrido definido de antemano: mostrar las salas resaltándolas en un color diferente.
- Conocer los objetos que se irán visitando según la ruta calculada.
- Zoom a un punto de la ruta. Es decir, ampliar la imagen de un punto en concreto de la ruta.
- Conocer la zona del plano de la ruta obtenida, mostrando una imagen del plano, y resaltando las salas que se están visitando actualmente.
- Mover plano. Es decir, desplazar el plano hacia un lado con ayuda del ratón. Funcionalidad exclusiva de los dispositivos fijos.
- Ir al Norte, Sur, Este u Oeste. Funcionalidad disponible únicamente en dispositivos fijos.
- Mostrar información de un objeto.
- Mostrar información de una sala.
- Mostrar imagen de un objeto.
- Obtener un listado con los objetos de la sala actual.

Es importante destacar que dentro de los requisitos de la arquitectura está el no depender de ningún sistema de localización para su funcionamiento. La arquitectura es independiente del hardware para localización utilizado (RFID, Zigbee, UWB, etc).

Teniendo por tanto los requisitos de nuestro sistema especificados en el Capítulo 4, se ha abordado en el Capítulo 5 la arquitectura de nuestro sistema de localización. Esta arquitectura es una arquitectura web en dónde el peso de la aplicación recae en el servidor, y relega a los dispositivos a meros clientes. En resumen, la parte de servidor de la arquitectura permite:

- **Establecer la conexión** con el servidor de mapas por Internet (ArcIMS).
- **Conocer la posición** del usuario conectado a través del dispositivo móvil. Esto sólo es posible a través de la aplicación para dispositivos móviles, estableciendo la comunicación con el simulador del sistema de localización.

- Establecer conexión con el sistema de cálculo de rutas, para **obtener la ruta** que desea conocer el usuario.

- **Mostrar la respuesta** al usuario.

Un aspecto a destacar en el desarrollo de todo el proyecto es la utilización de tecnologías punteras en la confección de aplicaciones web, de amplia aceptación en entornos profesionales. Se ha utilizado para la implementación de todo el proyecto la herramienta Visual Studio 2003, en concreto su lenguaje Visual Basic .NET. Utilizando ArcIMS como servidor para publicar los mapas y Network Analyst como tecnología utilizada para el cálculo de rutas.

La utilización de estas dos últimas tecnologías (ArcIMS y Network Analyst) ha supuesto un gran esfuerzo ya que nunca se habían utilizado durante los estudios de Ingeniería Informática, y para llevar a cabo el proyecto se ha tenido que contar con una fase de aprendizaje previa, una etapa de autoformación con la ayuda de SITESA.

El caso de estudio sobre el que se ha basado el proyecto ha sido para un museo, en concreto, para un museo de "Arte Abstracto", sobre el que se podrá realizar las diferentes consultas implementadas como son: *consultar la información de los objetos más cercanos, calcular la ruta a un objeto, conocer la posición actual, etc.*

A las dificultades encontradas en este proyecto hay que sumarle la recogida de información para un mayor entendimiento de las consultas, como es la información para los cuadros (autor, año, dimensiones, imágenes, etc). Toda esta información se ha recogido de la página web del Museo de Arte Abstracto Reina Sofía [WMAARS] y del Museo Abstracto de Cuenca [WMAC].

Al diseñar e implementar un producto software pensado para dispositivos que presentan dimensiones reducidas y funcionalidad limitada, como es el caso de PDA, se han ido encontrando varios problemas:

- Problemas para realizar las consultas sobre el servidor ArcIMS, ya que esta tecnología no está totalmente avanzada en el campo de los dispositivos móviles.

- Integración de la tecnología NetWork Analyst a la implementación del producto, al tratarse de una tecnología independiente de ArcIMS.

- Organización de la información en la PDA (imágenes y texto obtenidos como resultado de la consulta).

- Y dificultad a la hora de programar funcionalidad básica que requiera la necesidad de recoger posiciones concretas sobre la pantalla de la PDA, debido a la pequeña precisión que tiene el “lápiz” utilizado para seleccionar.

En conclusión los objetivos definidos al principio de esta memoria han sido alcanzados en su totalidad al conseguir:

- Tener un estudio del arte de los principales servidores de mapas por Internet que existen actualmente, obteniendo como conclusión que ArcIMS es el líder indiscutible dentro de las preferencias de los usuarios a nivel mundial que han desarrollado Aplicaciones de Mapas para Internet.

- Tener un estudio del arte de dos tecnologías para cálculo de rutas, obteniendo como conclusión que Network Analyst posee varios algoritmos para el cálculo de rutas ideales para obtener la ruta más corta, rutas alternativas, etc.

- Profundizar en las características tanto de ArcIMS como de Network Analyst, con el fin de tener un conocimiento mayor sobre éste servidor de mapas y la tecnología del cálculo de rutas.

- Realizar un sistema mediador que permita ejecutar consultas espaciales destinadas a interrogar elementos de interés situados en interiores.

- Desarrollar una interfaz de usuario considerando de forma empírica la necesidad de tener presentes requisitos no explícitamente funcionales relacionados con la usabilidad.

Toda la infraestructura necesaria para el desarrollo del proyecto la ha aportado **SITESA**, empresa especializada en las Tecnologías de la Información. Por tanto, es justo mencionarla en este documento y agradecerla el apoyo prestado, sin su ayuda no hubiese sido posible la realización de este trabajo.

7.2 TRABAJOS FUTUROS

El trabajo desarrollado, aunque plantea una solución factible al objetivo propuesto, es susceptible de mejoras que lo completen.

Una posible mejora del trabajo propuesto sería la extensión de la funcionalidad del sistema para el cálculo de rutas, permitiendo obtener no sólo la ruta más corta como se ha contemplado en el proyecto si no obtener diferentes rutas alternativas para llegar al destino solicitado.

Una segunda mejora, es sustituir el simulador del sistema de localización, por un sistema de localización real, como pueden ser: RFID, UWB, Zigbee, etc. De esta manera se podrán solventar problemas reales que pueden surgir al utilizar esa tecnología y que con el simulador no pueden ser dimensionados.

Otra posible mejora, es añadir más funcionalidad a la interfaz de usuario en el caso de un dispositivo móvil, permitiendo que desde una PDA sean posibles todas las funcionalidades que posee la aplicación de un dispositivo fijo, como es la identificación de un objeto, *zoom in* y *zoom out* al plano mostrado, etc.

Y por último, una mejora de la interfaz de usuario, así como de las imágenes mostradas. Como puede ser: indicar con unas imágenes más creativas el cambio de sala del usuario, mostrar la información de la ruta junto con sus símbolos adecuados, etc.

Anexo I. METODOLOGÍAS ÁGILES

El desarrollo de software no es una tarea fácil. Prueba de ello es que existen numerosas propuestas metodológicas que inciden en distintas dimensiones del proceso de desarrollo. Por una parte tenemos aquellas propuestas más tradicionales que se centran especialmente en el control del proceso, estableciendo rigurosamente las actividades involucradas, los artefactos que se deben producir, y las herramientas y notaciones que se usarán. Estas propuestas han demostrado ser efectivas y necesarias en un gran número de proyectos, pero también han presentado problemas en otros muchos.

Una posible mejora es incluir en los procesos de desarrollo más actividades, más artefactos y más restricciones, basándose en los puntos débiles detectados. Sin embargo, el resultado final sería un proceso de desarrollo más complejo que puede incluso limitar la propia habilidad del equipo para llevar a cabo el proyecto.

Otra aproximación es centrarse en otras dimensiones, como por ejemplo el factor humano o el producto software. Esta es la filosofía de las metodologías ágiles, las cuales dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas. Este enfoque está mostrando su efectividad en proyectos con requisitos muy cambiantes y cuando se exige reducir drásticamente los tiempos de desarrollo pero manteniendo una alta calidad.

Las metodologías ágiles están revolucionando la manera de producir software, y a la vez generando un amplio debate entre sus seguidores y quienes por escepticismo o convencimiento no las ven como alternativa para las metodologías tradicionales.

En este anexo se presenta resumidamente el contexto en el que surgen las metodologías ágiles, sus valores, principios y comparación con las metodologías tradicionales. Además se describen brevemente las principales propuestas, especialmente **Programación Extrema** (***eXtreme Programming,* XP**) la metodología ágil más popular en la actualidad.

I.I INTRODUCCIÓN

En las dos últimas décadas las notaciones de modelado y posteriormente las herramientas pretendieron ser las *balas de plata* para el éxito en el desarrollo de software, sin embargo, las expectativas no fueron satisfechas. Esto se debe en gran parte a que otro importante elemento, la metodología de desarrollo, había sido postergado. De nada sirven buenas notaciones y herramientas si no se proveen directivas para su aplicación [CLP].

Así, esta década ha comenzado con un creciente interés en metodologías de desarrollo. Hasta hace poco el proceso de desarrollo llevaba asociada un marcado énfasis en el control del proceso mediante una rigurosa definición de roles, actividades y artefactos, incluyendo modelado y documentación detallada. Este esquema *tradicional* para abordar el desarrollo de software ha demostrado ser efectivo y necesario en proyectos de gran tamaño (respecto a tiempo y recursos), donde por lo general se exige un alto grado de ceremonia en el proceso. Sin embargo, este enfoque no resulta ser el más adecuado para muchos de los proyectos actuales donde el entorno del sistema es muy cambiante, y en donde se exige reducir drásticamente los tiempos de desarrollo, pero manteniendo una alta calidad.

Ante las dificultades para utilizar metodologías tradicionales con estas restricciones de tiempo y flexibilidad, muchos equipos de desarrollo se resignan a prescindir del buen hacer de la ingeniería del software, asumiendo el riesgo que ello conlleva. En este escenario, las metodologías ágiles emergen como una posible respuesta para llenar ese vacío metodológico.

Por estar especialmente orientadas para proyectos pequeños, las metodologías ágiles constituyen una solución a medida para ese entorno, aportando una elevada simplificación que a pesar de ello no renuncia a las prácticas esenciales para asegurar la calidad del producto.

Las metodologías ágiles son sin duda uno de los temas recientes en ingeniería de software que están acaparando gran interés. Prueba de ello es que se están haciendo un espacio destacado en la mayoría de conferencias y *workshops* celebrados en los últimos años. Es tal su impacto que actualmente existen 4 conferencias internacionales de alto nivel y específicas sobre el tema[2].

Además, ya es un área con cabida en prestigiosas revistas internacionales. En la comunidad de la ingeniería del software, se está viviendo con intensidad un debate abierto entre los partidarios de las metodologías tradicionales (referidas peyorativamente como *metodologías pesadas*) y aquellos que apoyan las ideas emanadas del *Manifiesto Ágil*[3].

[2] XP Agile Universe: http://www.agileuniverse.com. Conference on eXtreme Programming and Agile Processes in Software Engineering: http://www.xp2004.org. Agile Development Conference (EEUU): http://www.agiledevelopmentconference.com. Agile Development Conference (Australia): http://www.softed.com/adc2003/

[3] http://www.agilemanifesto.org

La curiosidad que siente la mayor parte de ingenieros de software, profesores, e incluso alumnos, sobre las metodologías ágiles hace prever una fuerte proyección industrial. Por un lado, para muchos equipos de desarrollo el uso de metodologías tradicionales les resulta muy lejano a su forma de trabajo actual considerando las dificultades de su introducción e inversión asociada en formación y herramientas. Por otro, las características de los proyectos para los cuales las metodologías ágiles han sido especialmente pensadas se ajustan a un amplio rango de proyectos industriales de desarrollo de software; aquellos en los cuales los equipos de desarrollo son pequeños, con plazos reducidos, requisitos volátiles, y/o basados en nuevas tecnologías.

El anexo está organizado como sigue. En la sección I.II se introducen las principales características de las metodologías ágiles, recogidas en el **Manifiesto ágil** y se hace una comparación con las tradicionales. La sección I.III se centra en **eXtreme Programming** (**XP**), presentando sus características particulares, el proceso que se sigue y las prácticas que propone. Finalmente, en la sección I.IV se citan otros métodos ágiles, enumerándose sus principales características.

I.II METODOLOGÍAS ÁGILES

En febrero de 2001, tras una reunión celebrada en et (EEUU), nace el término *ágil* aplicado al desarrollo de software. En esta reunión participan un grupo de 17 expertos de la industria del software, incluyendo algunos de los creadores o impulsores de metodologías de software. Su objetivo fue esbozar los valores y principios que deberían permitir a los equipos desarrollar software rápidamente y respondiendo a los cambios que puedan surgir a lo largo del proyecto. Se pretendía ofrecer una alternativa a los procesos de desarrollo de software tradicionales, caracterizados por ser rígidos y dirigidos por la documentación que se genera en cada una de las actividades desarrolladas.

Tras esta reunión se creó **The Agile Alliance**[4], una organización, sin ánimo de lucro, dedicada a promover los conceptos relacionados con el desarrollo ágil de software y ayudar a las organizaciones para que adopten dichos conceptos. El punto de partida fue el **Manifiesto Ágil**, un documento que resume la filosofía .ágil.

[4] http://www.agilealliance.com

I.II.I El Manifiesto Ágil

Según el Manifiesto se valora:

- **Al individuo y las interacciones del equipo de desarrollo sobre el proceso y las herramientas**. La gente es el principal factor de éxito de un proyecto software. Es más importante construir un buen equipo que construir el entorno. Muchas veces se comete el error de construir primero el entorno y esperar que el equipo se adapte automáticamente. Es mejor crear el equipo y que éste configure su propio entorno de desarrollo en base a sus necesidades.

- **Desarrollar software que funciona más que conseguir una buena documentación**. La regla a seguir es *no producir documentos a menos que sean necesarios de forma inmediata para tomar una decisión importante*. Estos documentos deben ser cortos y centrarse en lo fundamental.

- **La colaboración con el cliente más que la negociación de un contrato**. Se propone que exista una interacción constante entre el cliente y el equipo de desarrollo. Esta colaboración entre ambos será la que marque la marcha del proyecto y asegure su éxito.

- **Responder a los cambios más que seguir estrictamente un plan**. La habilidad de responder a los cambios que puedan surgir a los largo del proyecto (cambios en los requisitos, en la tecnología, en el equipo, etc.) determina también el éxito o fracaso del mismo. Por lo tanto, la planificación no debe ser estricta sino flexible y abierta.

Los valores anteriores inspiran los doce principios del manifiesto. Son características que diferencian un proceso ágil de uno tradicional. Los dos primeros principios son generales y resumen gran parte del espíritu ágil. El resto tienen que ver con el proceso a seguir y con el equipo de desarrollo, en cuanto a metas a seguir y organización del mismo. Los principios son:

1. La prioridad es **satisfacer al cliente** mediante tempranas y continuas entregas de software que le aporte un valor.

2. **Dar la bienvenida a los cambios**. Se capturan los cambios para que el cliente tenga una ventaja competitiva.

3. **Entregar frecuentemente software que funcione** desde un par de semanas a un par de meses, con el menor intervalo de tiempo posible entre entregas.

4. **La gente del negocio y los desarrolladores deben trabajar juntos** a lo largo del proyecto.

5. **Construir el proyecto en torno a individuos motivados**. Darles el entorno y el apoyo que necesitan y confiar en ellos para conseguir finalizar el trabajo.

6. El **diálogo cara a cara** es el método más eficiente y efectivo para comunicar información dentro de un equipo de desarrollo.

7. El **software que funciona es la medida principal** de progreso.

8. **Los procesos ágiles promueven un desarrollo sostenible**. Los promotores, desarrolladores y usuarios deberían ser capaces de mantener una paz constante.

9. La **atención continua, la calidad técnica y el buen diseño** mejoran la agilidad.

10. La **simplicidad** es esencial.

11. Las mejores arquitecturas, requisitos y diseños surgen de los **equipos organizados por sí mismos**.

12. En intervalos regulares, **el equipo reflexiona respecto a cómo llegar a ser más efectivo**, y según esto ajusta su comportamiento.

I.II.II Comparación

La Tabla I.1 recoge esquemáticamente las principales diferencias de las metodologías ágiles con respecto a las tradicionales (*no ágiles*). Estas diferencias afectan no sólo al proceso en sí, sino también al contexto del equipo así como a su organización.

Metodologías Ágiles	Metodologías Tradicionales
Basadas en heurísticas provenientes de prácticas de producción de código	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo
Especialmente preparados para cambios durante el proyecto	Cierta resistencia a los cambios
Impuestas internamente (por el equipo)	Impuestas externamente
Proceso menos controlado, con pocos principios	Proceso mucho más controlado, con numerosas políticas/normas
No existe contrato tradicional o al menos es bastante flexible	Existe un contrato prefijado
El cliente es parte del equipo de desarrollo	El cliente interactúa con el equipo de desarrollo mediante reuniones
Grupos pequeños (<10 integrantes) y trabajando en el mismo sitio	Grupos grandes y posiblemente distribuidos
Pocos artefactos	Más artefactos
Pocos roles	Más roles
Menos énfasis en la arquitectura del software	La arquitectura del software es esencial y se expresa mediante modelos

Tabla I.1. Comparativa entre las metodologías ágiles y metodologías tradicionales.

I.III PROGRAMACIÓN EXTREMA (EXTREME PROGRAMMING, XP)

XP[5] [Beck2000] es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en la realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico.

Los principios y prácticas son de sentido común pero llevadas al extremo, de ahí proviene su nombre. Kent Beck, el padre de XP, describe la filosofía de XP en [Beck2000] sin cubrir los detalles técnicos y de implantación de las prácticas. Posteriormente, otras publicaciones de experiencias se han encargado de dicha tarea. A continuación se presentarán las características esenciales de XP organizadas en los tres apartados siguientes: (i) historias de usuario, (ii) roles, (iii) proceso y (iv) prácticas.

[5] http://www.extremeprogramming.org, http://www.xprogramming.com, http://www.c2.com/cgi/wiki?ExtremeProgramming

I.III.I Las Historias de Usuario

Son la técnica utilizada para especificar los requisitos del software. Se trata de tarjetas de papel en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las historias de usuario es muy dinámico y flexible. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas.

Beck en su libro [Beck2000] presenta un ejemplo de ficha (*customer eter and task card*) en la cual pueden reconocerse los siguientes contenidos: fecha, tipo de actividad (nueva, corrección, mejora), prueba funcional, número de historia, prioridad técnica y del cliente, referencia a otra historia previa, riesgo, estimación técnica, descripción, notas y una lista de seguimiento con la fecha, estado, cosas por terminar y comentarios.

A efectos de planificación, las historias pueden ser de una a tres semanas de tiempo de programación (para no superar el tamaño de una iteración). Las historias de usuario son descompuestas en tareas de programación (*task card*) y asignadas a los programadores para ser implementadas durante una iteración.

I.III.II Roles XP

Los roles de acuerdo con la propuesta original de Beck son:

- **Programador:** el programador escribe las pruebas unitarias y produce el código del sistema.

- **Cliente:** escribe las historias de usuario y las pruebas funcionales para validar su implementación. Además, asigna la prioridad a las historias de usuario y decide cuáles se implementan en cada iteración centrándose en aportar mayor valor al negocio.

- **Encargado de pruebas (*Tester*):** ayuda al cliente a escribir las pruebas funcionales. Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas.

- **Encargado de seguimiento (*Tracker*):** proporciona realimentación al equipo. Verifica el grado de acierto entre las estimaciones realizadas y el tiempo real

dedicado, para mejorar futuras estimaciones. Realiza el seguimiento del progreso de cada iteración.

- **Entrenador (*Coach*):** es responsable del proceso global. Debe proveer guías al equipo de forma que se apliquen las prácticas XP y se siga el proceso correctamente.

- **Consultor:** es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto, en el que puedan surgir problemas.

- **Gestor (*Big boss*):** es el vínculo entre clientes y programadores, ayuda a que el equipo trabaje de forma efectiva creando las condiciones adecuadas. Su labor esencial es de coordinación.

I.III.III Proceso XP

El ciclo de desarrollo consiste (a grandes rasgos) en los siguientes pasos:

1. El cliente define el valor de negocio a implementar.

2. El programador estima el esfuerzo necesario para su implementación.

3. El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.

4. El programador construye ese valor de negocio.

5. Vuelve al paso 1.

En todas las iteraciones de este ciclo tanto el cliente como el programador aprenden. No se debe presionar al programador a realizar más trabajo que el estimado, ya que se perderá calidad en el software o no se cumplirán los plazos. De la misma forma el cliente tiene la obligación de manejar el ámbito de entrega del producto, para asegurarse que el sistema tenga el mayor valor de negocio posible con cada iteración.

El ciclo de vida ideal de XP consiste de seis fases [Beck2000]: (i) Exploración, (ii) Planificación de la entrega (*Release*), (iii) Iteraciones, (iv) Producción, (v) Mantenimiento y (vi) Muerte del Proyecto.

I.III.IV Prácticas XP

La principal suposición que se realiza en XP es la posibilidad de disminuir la mítica curva exponencial del costo del cambio a lo largo del proyecto, lo suficiente para que el diseño evolutivo funcione. Esto se consigue gracias a las tecnologías disponibles para ayudar en el desarrollo de software y a la aplicación disciplinada de las siguientes prácticas:

- **El juego de la planificación:** hay una comunicación frecuente entre el cliente y los programadores. El equipo técnico realiza una estimación del esfuerzo requerido para la implementación de las historias de usuario y los clientes deciden sobre el ámbito y tiempo de las entregas de cada iteración.

- **Entregas pequeñas:** producir rápidamente versiones del sistema que sean operativas, aunque no cuenten con toda la funcionalidad del sistema. Esta versión ya constituye un resultado de valor para el negocio. Una entrega no debería tardar más de 3 meses.

- **Metáfora:** el sistema es definido mediante una metáfora o un conjunto de metáforas compartidas por el cliente y el equipo de desarrollo. Una metáfora es una historia compartida que describe cómo debería funcionar el sistema (conjunto de nombres que actúen como vocabulario para hablar sobre el dominio del problema, ayudando a la nomenclatura de clases y métodos del sistema).

- **Diseño simple:** se debe diseñar la solución más simple que pueda funcionar y ser implementada en un momento determinado del proyecto.

- **Pruebas:** la producción de código está dirigida por las pruebas unitarias. Éstas son establecidas por el cliente antes de escribirse el código y son ejecutadas constantemente ante cada modificación del sistema.

- **Refactorización (*Refactoring*):** es una actividad constante de reestructuración del código con el objetivo de remover duplicación de código, mejorar su legibilidad, simplificarlo y hacerlo más flexible para facilitar los posteriores cambios. Se mejora la estructura interna del código sin alterar su comportamiento externo [PP2003].

- **Programación en parejas:** toda la producción de código debe realizarse con trabajo en parejas de programadores. Esto conlleva ventajas implícitas (menor tasa de errores, mejor diseño, mayor satisfacción de los programadores, etc.).

- **Propiedad colectiva del código:** cualquier programador puede cambiar cualquier parte del código en cualquier momento.

- **Integración continua:** cada pieza de código es integrada en el sistema una vez que esté lista. Así, el sistema puede llegar a ser integrado y construido varias veces en un mismo día.

- **40 horas por semana:** se debe trabajar un máximo de 40 horas por semana. No se trabajan horas extras en dos semanas seguidas. Si esto ocurre, probablemente está ocurriendo un problema que debe corregirse. El trabajo extra desmotiva al equipo.

- **Cliente in-situ:** el cliente tiene que estar presente y disponible todo el tiempo para el equipo. Éste es uno de los principales factores de éxito del proyecto XP. El cliente conduce constantemente el trabajo hacia lo que aportará mayor valor de negocio y los programadores pueden resolver de manera inmediata cualquier duda asociada. La comunicación oral es más efectiva que la escrita.

- **Estándares de programación:** XP enfatiza que la comunicación de los programadores es a través del código, con lo cual es indispensable que se sigan ciertos estándares de programación para mantener el código legible.

El mayor beneficio de las prácticas se consigue con su aplicación conjunta y equilibrada puesto que se apoyan unas en otras. Esto se ilustra en la figura I.1 (obtenida de [Beck2000]), donde una línea entre dos prácticas significa que las dos prácticas se refuerzan entre sí. La mayoría de las prácticas propuestas por XP no son novedosas sino que de alguna forma ya habían sido propuestas en ingeniería del software e incluso demostrado su valor en la práctica (ver [ASRW2002] para un análisis histórico de ideas y prácticas que sirven como antecedentes a las utilizadas por las metodologías ágiles). El mérito de XP es integrarlas de una forma efectiva y complementarlas con otras ideas desde la perspectiva del negocio, los valores humanos y el trabajo en equipo.

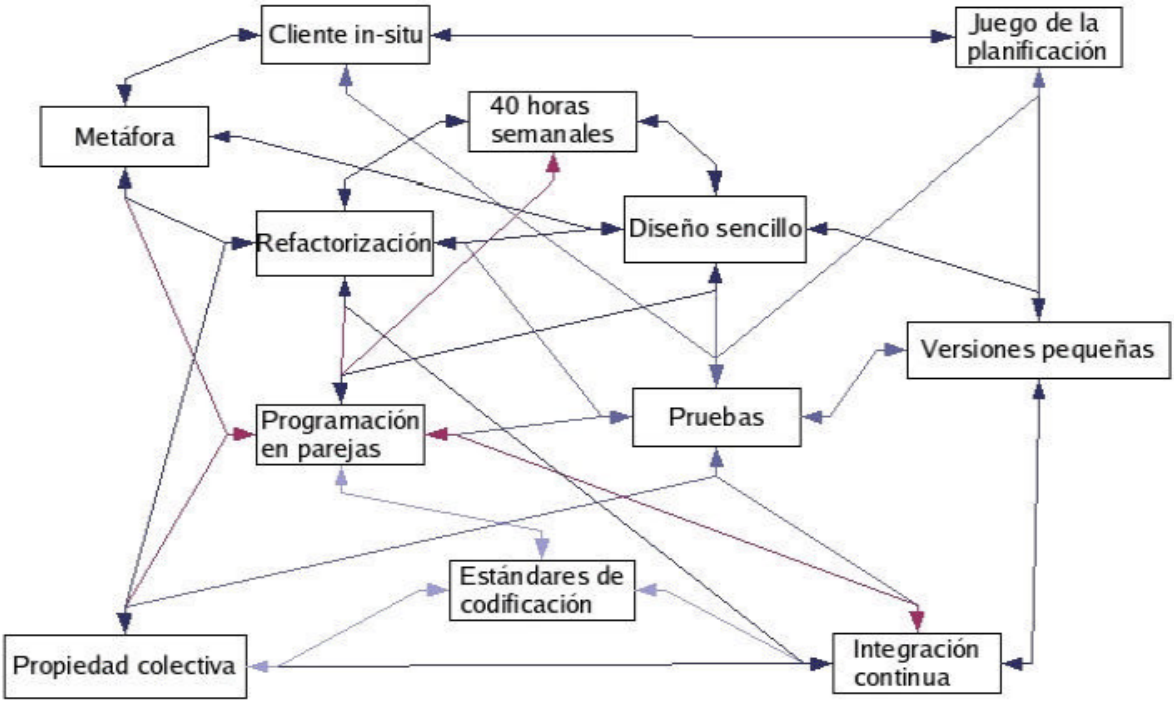


Figura I.1. Las prácticas se refuerzan entre sí.

I.IV OTRAS METODOLOGÍAS ÁGILES

Aunque los creadores e impulsores de las metodologías ágiles más populares han suscrito el manifiesto ágil y coinciden con los principios enunciados anteriormente, cada metodología tiene características propias y hace hincapié en algunos aspectos más específicos. A continuación se resumen otras metodologías ágiles. La mayoría de ellas ya estaban siendo utilizadas con éxito en proyectos reales pero les faltaba una mayor difusión y reconocimiento.

- **SCRUM[6] [SBM2001]:** desarrollada por Ken Schwaber, Jeff Sutherland y Mike Beedle. Define un marco para la gestión de proyectos, que se ha utilizado con éxito durante los últimos 10 años. Está especialmente indicada para proyectos con un rápido cambio de requisitos. Sus principales características se pueden resumir en dos. El desarrollo de software se realiza mediante iteraciones, denominadas *sprints*, con una duración de 30 días. El resultado de cada *sprint* es un incremento ejecutable que se muestra al cliente. La segunda característica importante son las reuniones a lo largo del proyecto, entre ellas destaca la reunión diaria de 15 minutos del equipo de desarrollo para coordinación e integración.

[6] http://www.controlchaos.com

- **Crystal Methodologies** [7] **[FBB1999]:** se trata de un conjunto de metodologías para el desarrollo de software caracterizadas por estar centradas en las personas que componen el equipo y la reducción al máximo del número de artefactos producidos. Han sido desarrolladas por Alistair Cockburn. El desarrollo de software se considera un juego cooperativo de invención y comunicación, limitado por los recursos a utilizar. El equipo de desarrollo es un factor clave, por lo que se deben invertir esfuerzos en mejorar sus habilidades y destrezas, así como tener políticas de trabajo en equipo definidas. Estas políticas dependerán del tamaño del equipo, estableciéndose una clasificación por colores, por ejemplo *Crystal Clear* (3 a 8 miembros) y *Crystal Orange* (25 a 50 miembros).

- **Dynamic Systems Development Method**[8] **(DSDM) [Stapleton1997]:** define el marco para desarrollar un proceso de producción de software. Nace en 1994 con el objetivo de crear una metodología RAD unificada. Sus principales características son: es un proceso iterativo e incremental y el equipo de desarrollo y el usuario trabajan juntos. Propone cinco fases: (i) estudio de la viabilidad, (ii) estudio del negocio, (iii) modelado funcional, (iv) diseño y construcción, y finalmente (v) implementación. Las tres últimas son iterativas, además de existir realimentación a todas las fases.

- **Adaptive Software Development**[9] **(ASD) [HO2000]:** su impulsor es Jim Highsmith. Sus principales características son: iterativo, orientado a los componentes software más que a las tareas y tolerante a los cambios. El ciclo de vida que propone tiene tres fases esenciales: (i) especulación, (ii) colaboración y (iii) aprendizaje. En la primera de ellas se inicia el proyecto y se planifican las características del software; en la segunda se desarrollan las características y finalmente en la tercera se revisa su calidad, y se entrega al cliente. La revisión de los componentes sirve para aprender de los errores y volver a iniciar el ciclo de desarrollo.

- **Feature-Driven Development**[10] **(FDD) [CLD1999]:** define un proceso iterativo que consta de 5 pasos. Las iteraciones son cortas (hasta 2 semanas). Se centra en las fases de diseño e implementación del sistema partiendo de una lista de características que debe reunir el software. Sus impulsores son Jeff De Luca y Peter Coad.

---

[7] http://www.crystalmethodologies.org/
[8] http://www.dsdm.org
[9] http://www.adaptivesd.com
[10] http://www.featuredrivendevelopment.com

- **Lean Development**[11] **(LD) [PP2003]:** definida por Bob Charettes a partir de su experiencia en proyectos con la industria japonesa del automóvil en los años 80 y utilizada en numerosos proyectos de telecomunicaciones en Europa. En LD, los cambios se consideran riesgos, pero si se manejan adecuadamente se pueden convertir en oportunidades que mejoren la productividad del cliente. Su principal característica es introducir un mecanismo para implementar dichos cambios.

I.V SELECCIÓN DE LA METODOLOGÍA ADECUADA

No existe una metodología universal para hacer frente con éxito a cualquier proyecto de desarrollo de software. Toda metodología debe ser adaptada al contexto del proyecto (recursos técnicos y humanos, tiempo de desarrollo, tipo de sistema, etc.).

Históricamente, las metodologías tradicionales han intentado abordar la mayor cantidad de situaciones de contexto del proyecto, exigiendo un esfuerzo considerable para ser adaptadas, sobre todo en proyectos pequeños y con requisitos muy cambiantes. Mientras que las metodologías ágiles ofrecen una solución casi a medida para una gran cantidad de proyectos que tienen estas características.

Una de las cualidades más destacables en una metodología ágil es su sencillez, tanto en su aprendizaje como en su aplicación, reduciéndose así los costos de implantación en un equipo de desarrollo. Esto ha llevado hacia un interés creciente en las metodologías ágiles. Sin embargo, hay que tener presente una serie de inconvenientes y restricciones para su aplicación, tales como: están dirigidas a equipos pequeños o medianos (Beck sugiere que el tamaño de los equipos se limite de 3 a 20 como máximo, otros dicen no más de 10 participantes), el entorno físico debe ser un ambiente que permita la comunicación y colaboración entre todos los miembros del equipo durante todo el tiempo, cualquier resistencia del cliente o del equipo de desarrollo hacia las prácticas y principios puede llevar al proceso al fracaso (el clima de trabajo, la colaboración y la relación contractual son claves), el uso de tecnologías que no tengan un ciclo rápido de realimentación o que no soporten fácilmente el cambio, etc.

Falta aún un cuerpo de conocimiento consensuado respecto de los aspectos teóricos y prácticos de la utilización de metodologías ágiles, así como una mayor consolidación de los resultados de aplicación. La actividad de investigación está

[11] http://www.poppendieck.com

orientada hacia líneas tales como: métricas y evaluación del proceso, herramientas específicas para apoyar prácticas ágiles, aspectos humanos y de trabajo en equipo. Entre estos esfuerzos destacan proyectos como **NAME**[12] (*Network for Agile Methodologies Experience*). Aunque en la actualidad ya existen libros asociados a cada una de las metodologías ágiles existentes y también abundante información en Internet, es XP la metodología que resalta por contar con la mayor cantidad de información disponible y es con diferencia la más popular.

[12] http://www.name.case.unibz.it/

Anexo II. GLOSARIO DE TÉRMINOS

Aplicación CGI: CGI, o Common Gateway Interface, permite la interacción entre un visitante y su página. Una aplicación CGI es un programa hecho en un lenguaje de programación que prácticamente hace todo lo que el programador quiera. Puede ser un motor de búsqueda, un carrito de compras, un contador, un juego, un tablón de expresión, etc. Hay miles de aplicaciones CGI en el Internet, muchas están disponibles gratuitamente.

ArcEditor: es un potente cliente GIS con el que editar y gestionar información geográfica. Es parte de la familia de productos de ArcGIS, e incluye toda la funcionalidad de ArcView añadiendo una serie de herramientas para crear, editar y asegurar la calidad de los datos.

ArcGIS Desktop: constituyen un conjunto escalable de productos que permiten al usuario generar, importar, editar, consultar, cartografiar, analizar y publicar información geográfica. (ArcMap, Network Analyst, etc)

ArcGIS Publisher: es la extensión integrable con ArcView, ArcEditor y ArcInfo que permite la conversión de documentos de mapa (MXDs) en documentos de mapa publicables (PMFs), para su visualización posterior a través de la aplicación gratuita de ESRI, ArcReader.

ArcIMS: es una aplicación muy potente, escalable y basada en estándares que permite, de manera rápida y sencilla, diseñar y gestionar servicios de cartografía en Internet

ArcInfo: es el SIG más completo que existe. Posee toda la funcionalidad de ArcView y ArcEditor además de incluir herramientas avanzadas de análisis espacial, tratamiento de datos y cartografía de calidad.

ArcMap: es una herramienta de ESRI que permite visualizar, explorar, consultar datos, así como capacidad para crear y editar datos geográficos y alfanuméricos.

ArcObjects: es un conjunto de componentes de software con funcionalidad GIS e interfaces programables, mediante los cuales ha sido creada cada uno de las aplicaciones de los clientes **ArcGIS Desktop**. Se presentan como una colección de componentes ordenados dentro de un Modelo de Objetos

ArcReader: es una aplicación gratuita con la que los usuarios pueden visualizar, explorar e imprimir mapas y globos de manera sencilla.

ArcView: es una herramienta GIS con la que se puede visualizar, analizar, crear y gestionar información geográfica.

ArcXML: es un lenguaje basado en XML por lo que es independiente de la plataforma, se pueden hacer tanto consultas espaciales como alfanuméricas, de las que se pueden obtener tanto datos como imágenes, añadir capas dinámicamente para determinadas consultas, etc.

Capa: es una colección de geometrías que tienen el mismo conjunto de atributos.

Dataset: es un fichero que contiene relación lógica de unos datos con otros, para poder establecer relaciones como puede ser cálculo de rutas. Por ejemplo, la red utilizada en todo el sistema indoor (intersecciones que se pueden encontrar, caminos que se pueden tomar, etc.)

Feature: se compone de una o mas coordenadas *(x,y)* y una serie de atributos alfanuméricos.

GIS (Geographical Information System): estas siglas son las correspondientes a SIG en inglés.

IMS: es un Servidor de Mapas por Internet, es un SIG a través de Internet.

Licencia GNU: es una licencia creada por la Free Software Foundation a mediados de los 80, y está orientada principalmente a proteger la libre distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es software libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios.

Línea: tipo geométrico formado por uno o más pares de puntos que definen segmentos de líneas.

Metadatos: son datos que describen otros datos. En general, un grupo de metadatos se refiere a un grupo de datos, llamado *recurso*.

Metodología ágil: metodología que da mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas, agilizando el desarrollo de software.

Network Analyst: es la extensión de ArcGIS Desktop que resuelve problemas de rutas

con redes multimodales como la generación de la ruta más eficiente, localización de las ubicaciones más próximas, generación de áreas de servicio basadas en tiempos de viaje, cálculo de matriz de origen-destino y listado de informe de direcciones.

OGC (OpenGeospatial Consortium): organización voluntaria, internacional y sin ánimo de lucro encargada de desarrollar estándares para servicios geoespaciales y basados en localización.

Polígono: tipo geométrico compuesto de secuencias de líneas rectas conectadas que forman una superficie cerrada, lo cual implica que los polígonos poseen un área.

Punto: tipo geométrico compuesto de dos coordenadas, *x* e *y*, a menudo se corresponden con longitud y latitud.

Raster: es un área espacial dividida en celdas regulares (generalmente en cuadrícula pero no necesariamente), en las que cada una de las cuales presentan unos atributos o valor (altitud, reflectancia, etc.) que por lo general son almacenados en una base de datos.

Shapefile: es un formato vectorial de almacenamiento digital donde se guarda la localización de los elementos geográficos y los atributos asociados a ellos. Es una capa guarda en local.

SIG (Sistema de Información Geográfica): es un modelo informatizado del mundo real, descrito en un sistema de referencia ligado a la Tierra, establecido para satisfacer unas de las necesidades de información específicas respondiendo a un conjunto de preguntas concreto

Sistema espacial: sistema hardware, software y procedimientos elaborados para facilitar la obtención, gestión, manipulación, análisis, modelado, representación y salida de datos espacialmente referenciados para resolver problemas complejos de planificación y gestión.

TrueType: es un formato de fuente tipográfica estándar desarrollada inicialmente por Apple Computer a finales de los 1980s como competidor de la fuente *Type 1* de Adobe, usada en PostScript. La principal fortaleza de TrueType fue que ofrecía a los diseñadores de fuentes un gran grado de control sobre la forma que sus fuentes se mostraban a diferentes tamaños.

Usabilidad: es un término que indica facilidad de uso, entendimiento y aprendizaje.

WMS: servicio Web Map Service definido por el OGC produce mapas de datos espaciales referidos de forma dinámica a partir de información geográfica.

XP: eXtreme Programming es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo.

BIBLIOGRAFÍA

[ArcSDE] ESRI (2005). “ArcGIS 9: Understanding ArcSDE”

[ASRW2002] Abrahamsson, P., Salo, O., Ronkainen, J., Warsta, J. (2002). Agile software development methods Review and analysis. VTT Publications

[Beck2000] Beck, K. (2000). Extreme Programming Explained. Embrace Change., Pearson Education. Traducido al español como: .Una explicación de la programación extrema. Aceptar el cambio., Addison Wesley.

[BKM1991] Bevan,N; Kirakowsky,J; Maissel,J. (1991). ”What is Usability”. Proccedings of 4th Intl. Conference on HCI (September 1991).

[BM1994] Bevan,N.; Macleod,M. (1994). Usability measurement in context. Natinal Physical Laboratory, Teddington, Middlesex, UK, Behaviour and Information Thechnology, 13, 132-145.

[BT2005] Bricker, Tim (2005). “Using the ArcIMS .Net Link”. ESRI Educational Services

[CLD1999] Coad P., Lefebvre E., De Luca J. (1999). Java Modeling In Color With UML: Enterprise Components and Process. Prentice Hall.

[CLP] Canós, José H. Letelier, Patricio y Penadés, Mª Carmen. Metodologías ágiles en el desarrollo del software. Universidad politécnica de Valencia.

[CM1986] CEBRIÁN, J.A. y MARK, D. (1986): Sistemas de Información Geográfica II. Gestión y perspectivas de desarrollo. Estudios geográficos, nº 188; Madrid.

[ESRI1] ESRI. “ArcIMS Administration”

[ESRI2] ESRI. “Customizing ArcIMS using ArcXML”

[ESRI2004] ESRI (2004) “ArcIMS Architecture and Functionality”

[ESRI2005] ESRI (2005) “Hierarchical Routes in ArcGIS Network Analyst”

[ESRI3] ESRI. “Customizing ArcIMS using HTML/JavaScript”

[ESRI4] ESRI. “Getting Started with ArcIMS”

[ESRI5] “ESRI Shapefile Technical Description”

[ESRI6] ESRI. “Introduction to ArcIMS”

[FBB1999] Fowler, M., Beck, K., Brant, J. (1999).Refactoring: Improving the Design of Existing Code.. Addison-Wesley.

[HO2000] Highsmith J., Orr K. (2000).Adaptive Software Development: A Collaborative Approach to Managing Complex Systems. Dorset House.

[ISO9126] ISO/IEC Standar, ISO-9126 Software Product Evaluation – Quality Characteristic and Guidelines for Their Use. 1991

[Montero2005] Montero, M. Integración de calidad y experiencia en el desarrollo de interfaces de usuario dirigida por modelos. Tesis doctoral realizada en la Universidad Politécnica de Castilla la Mancha, 2005.

[MT2005] Mitchell, Tyler. (2005) Web Mapping Ilustrated

[NCGIA] NCGIA (1990): Core Curriculum. Tres volúmenes: I. Introduction to GIS; II. Technical issues in GIS; III. Application issues in GIS, National Center for Geographic Information and Analysis/University of California, Santa Bárbara (California).

[NJ1993] Nielsen, J. (1993). “Usability Engineering.” Academic Press Professional, Boston, MA.

[OSW] Oracle Spatial
http://www.oracle.com/database/spatial.html

[PJ1994] Preece, J. (1994). “Human-computer interaction”. Academic Press

Professional, Boston, MA.

[PP2003] Poppendieck M., Poppendieck T. (2003). Lean Software Development: An Agile Toolkit for Software Development Managers. Addison Wesley.

[QW2001] Quesenbery, W. (2001). "What Does Usability Mean: Looking Beyond 'Ease of Use'" Proceedings of the 48th Annual Conference, Society for Technical Communication. Disponible en **http://www.wqusability.com/articles/more-than-ease-of-use.html**

[RJ1995] Redish, J. (1995). Are we really a post-usability? ACM SIGDOC Asterisk Journal of Computer Documentation, vol 19 (1), pags 18-24.

[RP1993] RODRÍGUEZ PASCUAL, A. (1993): "Proposición de una definición profunda de SIG". Los Sistemas de Información Geográfica en el umbral del siglo XXI. 2Q Congreso de la Asociación Española de Sistemas de Información Geográfica. Junio 2-3 y 4. Madrid.

[SBM2001] Schwaber K., Beedle M., Martin R.C. (2001). Agile Software Development with SCRUM.. Prentice Hall.

[Stapleton1997] Stapleton J. (1997). Dsdm Dynamic Systems Development Method: The Method in Practice. Addison-Wesley.

[SWXML] Cauldwell, P. Charla,R. Chopra V. Damschen, G. Dix, C. Hong, T. Servicios Web XML. Anaya.

[W3C2006] W3C (2006). Extensible Markup Language (XML) 1.0 Fourth Edition.

[WAGIS] **http://www.esri.com/software/arcgis/index.html**

[WAIMS] Documentación sobre ArcIMS y ArcXML. **http://edndoc.esri.com/arcims/9.1/**

[WESRI] Página oficial ESRI España. **www.esri-es.com**

[WGT] Web de tecnología.
http://www.gacetatecnologica.com/files/File/pdf/gaceta8_en_baja.pdf

[WIIS] Definición de IIS (Internet Information Server)
http://es.wikipedia.org/wiki/Internet_Information_Services

[WMAARS] Museo de Arte Abstracto Reina Sofía (Madrid)
http://www.museoreinasofia.es/portada/portada.php

[WMAC] Museo Abstracto de Cuenca
http://www.march.es/arte/cuenca//

[WMI] Mapping Interactivo. (2006) Revista Internacional de Ciencias de la Tierra. **www.mappinginteractivo.com**

[WMS] Página oficial de MapServer. **http://mapserver.gis.umn.edu/**

[WNA] Documentación sobre Network Analyst.
http://edndoc.esri.com/arcobjects/9.1/

[WOGC] Web oficial de OGC. **http://www.opengeospatial.org**

[WRFID] Definición de sistema de localización RFID.
http://es.wikipedia.org/wiki/RFID

[WSGT] The Worldwide Source for Geospatial Technology
http://www.directionsmag.com/article.php?articleid=414&trv=1

[WSITESA] Página oficial SITESA.
http://www.sitesa.com

[WSMI] CIAT ”Servicios de Mapas por Internet”
http://gisweb.ciat.cgiar.org/SIG/esp/tecnologias-esri.htm

[WSWXML] Definición de Servicios Web XML.
http://es.wikipedia.org/wiki/Servicio_Web

[WTOM] Web del servidor de aplicaciones Apache Tomcat.
http://jakarta.apache.org/tomcat/

[WVS2003] Definición y características de Visual Studio.
http://www.microsoft.com/spanish/msdn/latam/vstudio/