Servei municipal d'emergències

Memòria

Autor: Jordi Vilalta i Oliu
Consultor: Anna Muñoz i Bollas

Projecte de fi de carrera
Enginyeria d'Informàtica
Curs 2010-11, 1r semestre

Resum

Es presenta un sistema de gestió d'emergències dins d'un àmbit municipal on es pretén donar una resposta eficaç a les emergències que es puguin donar en un determinat municipi com, en el nostre cas, Barcelona.

Distribuïdes pel municipi es troben un conjunt de unitats de resposta (ambulàncies, policia i bombers) i un conjunt d'hospitals. Enfront a un accident, ocorregut dins el municipi, s'informa al sistema del lloc del incident i de l'hospital a on caldrà traslladar a l'accidentat. El sistema calcularà les rutes més òptimes desde les diferents unitats de resposta que es troben més properes al lloc del incident fins al mateix lloc on ha ocorregut el incident. També, calcularà la ruta desde el mateix lloc del incident fins a l'hospital que hagi estat sel.leccionat. Posteriorment, es procedirà a visualitzar les diferents rutes òptimes calculades.

La implementació d'aquest projecte s'ha fet, de forma exclusiva, utilitzant únicament eines de programari lliure.

Índex

Índex de Figures

1. Introducció

1.1 Context del Treball

Avui en dia es pot trobar en el mercat una gran quantitat i varietat de programari de sistemes d'informació Geogràfica (SIG). La oferta és molt variada i tard o d'hora caldrà escollir una o altre opció. Una de les primeres valoracions que s'ha fer és si es decideix escollir un programari lliure o no. En aquest projecte, s'optarà per desenvolupar-lo en tota la seva totalitat mitjançant software lliure.

Aquesta decisió afectarà de forma clara en el context de treball donats els avantatges i inconvenients d'aquest tipus de software respecte al propietari. Alguns d'ells són:

Avantatges:

- Existeixen aplicacions per a totes les plataformes (Linux, Windows, Mac OS, etc).
- El preu de les aplicacions és molt menor i la majoria de les vegades són gratuïtes.
- Llibertat de còpia, modificació, d'ús i de redistribució.
- Facilitat a l'hora de traduir una aplicació en varis idiomes.
- L'usuari no depèn de l'autor del programari.

Inconvenients:

- Algunes aplicacions en determinats entorns (sobretot en Linux) poden arribar a ser difícils d'instal·lar.
- És un producte sense cap garantia per part de l'autor.
- Menor compatibilitat amb el maquinari.

Aquests avantatges i inconvenients es gaudiran i sofriran durant l'el.laboració de tot projecte. Aquest projecte també pot ser vist com una integració de diferents components o peces de programari lliure. De forma resumida, es pot dir que el projecte està composat dels següents processos o etapes:

- S'extrauran les dades de *OpenStreetMaps*.
- Es crearà la topologia amb *osm2pgrouting*.
- Es desaran les dades a la base de dades *PostgreSQL*.
- Es Calcularan les rutes mitjançant *pgrouting*.
- Es farà servir el servidor de mapes *MS4W*.
- S'implantarà una interfície d'usuari amigable perquè l'usuari pugui interactuar amb la cartografia base i es presentaran aquestes dades en un portal web mitjançant *OpenLayers*.

L'àrea que s'ha triat per aquest treball correspon a la àrea metropolitana de Barcelona. El motiu ha estat la diversitat i les grans possibilitats que ofereix la capital catalana. A més, el fet d'escollir una zona d'aquestes característiques ajudarà a explorar i explotar les possibilitats que ofereix el SIG implementat.

1.2 Objectius

Amb la realització d'aquest projecte, s'espera implementar un sistema, amb programari ja existent, que permeti una gestió òptima de les emergències en un àmbit municipal. De la mateixa manera, també s'espera assolir els següents objectius generals i específics:

1.2.1 Objectius generals

- Comprendre els conceptes teòrics bàsics sobre els que sustenta tot SIG, els conceptes de la tecnologia SIG i la seva metodologia.
- Conèixer l'estructura dels diferents tipus de dades amb què treballa un SIG.
- Conèixer els sistemes d'emmagatzemament estàndards (ràster i vectorial) i ser capaços d'ubicar la informació en les coordenades que correspongui.
- Trobar, generar i manipular dades geogràfiques.
- Saber plantejar un projecte SIG.
- Conèixer les operacions d'anàlisi espacial i transformacions en el SIG analitzat.

1.2.2 Objectius específics

- Determinar l'estat actual de l'ecosistema d'aplicacions lliures, i en particular les que corresponen als servidors de mapes.
- Treballar amb dades en format *OpenStreetMap* (OSM).
- Conèixer les funcions de càlcul de rutes de què disposa pgRouting: Dijkstra, TSP problems, etc. Com a programari de càlcul de rutes es treballarà amb *pgRouting*.
- Implementar un visor de la ruta obtinguda amb *OpenLayers* com a client GIS.
- Utilitzar *UMN Mapserver* com a servidor de mapes.
- Treballar amb bases de dades geogràfiques *PostGIS/PostgreSQL.*

1.3 Metodologia aplicada

Durant l'execució del projecte es seguirà una metodologia determinada. Aquesta metodologia anirà enfocada a assolir els objectius definit dins del pla de treball i es pot resumir en els següents punts:

- Estudi teòric dels components i conceptes que formen la base de tot SIG. El material del que es disposarà per a tal efecte serà el material didàctic de l'assignatura juntament amb la bibliografia recomanada i la consulta de la pàgina web de les diferents institucions més importants de l'àrea.
- Instal·lació del diferent programari a utilitzar així com dels diferents entorns necessaris per dur a terme les operacions amb les dades obtingudes. També s'hi inclouran diferents tipus de proves i tests per tal d'habituar-se a l'entorn que es farà servir.
- Tria de les diferents fonts de dades que es faran servir en el projecte. Valoració de les diferents possibilitats existents actualment i l'elecció de l'opció es cregui més òptima per al cas que es tracta.
- Aplicació i comparació dels diferents algorismes de càlcul de rutes sobre les dades obtingudes en el punt anterior.
- Implantació d'una interfície que permeti preguntar sobre el contingut de la base de dades, definir punts de localització d'incidents i mostrar la ruta calculada. Finalment, presentar els resultats obtinguts en un entorn web.

1.4 Planificació del projecte

A fi i efecte d'organitzar i planificar el projecte, s'ha dut a terme un pla de treball consistent en la descripció de les diferents etapes de les que està composat el projecte.

Per a cada una de les diferents etapes, s'ha fet una descomposició de diferents tasques que el composen així com la seva temporització aproximada de les mateixes. També s'ha representat un diagrama de Gantt per tal de veure les dependències entre les diferents tasques i com aquestes afecten a la globalitat del projecte.

A grans trets, s'ha dividit el projecte en quatre grans blocs clarament definits. Aquests blocs es corresponen amb cadascun dels quatre lliuraments programats. Aquests són:

- Planificació del projecte.
- Preparació del entorn de treball i càrrega de les dades.
- Càlcul de rutes i disseny i programació de la interfície de l'usuari.
- Confecció de la presentació i lliurament final de la memòria.

1.5 La Memòria

Aquesta memòria està estructurada en varis apartats o capítols:

- Introducció: està composat de la descripció del context de treball i els objectius programats, la metodologia que hem aplicat i la planificació del projecte.
- Fonaments teòrics: Aquest capítol pretén donar una base teòrica al lector per a una major comprensió dels SIG. Està estructurat en una introducció, un apartat de geodèsia i cartografia i, finalment, una referència a les bases de dades espacials.
- Configuració de l'entorn de treball: Es detalla la instal·lació dels diferents programaris que intervenen en cada punt; així com, les possibles incidències que han aparegut i com s'han solventat.
- Càrrega de dades: Es detalla el procés de càrrega de dades i la seva adaptació per tal de dotar-les de topologia.
- Càlcul de rutes: S'especifica com s'han fet el càlcul d'aquestes rutes fent servir pgRouting.
- Implantació de la interfície d'usuari: S'utilitzarà per al càlcul de rutes sobre dades d'OSM.
- Conclusions: en aquest darrer capítol s'inclouen unes conclusions del treball realitzat i una previsió de l'evolució futura dels SIG.

2. Fonaments Teòrics

Aquest capítol introductori permetrà al lector comprendre que és un sistema d'informació geogràfica (en endavant, SIG), així els altres conceptes teòrics sobre els que es basa tot SIG: la Geodèsia i la Cartografia. Per acabar, es veurà com es guarda, informàticament, tota aquesta informació.

2.1 Sistemes d'Informació Geogràfica

Un Sistema d'informació Geogràfica el formen tot un conjunt de maquinari, programari i dades geogràfiques de manera que hom pugui ser capaç de capturar, emmagatzemar, manipular i analitzar aquestes dades obtingudes per tal de resoldre problemes complexos de planificació i de gestió. Aquesta informació que es fa servir es diu que és *geogràficament referenciada*. Això significa que, per localitzar un objecte a l'espai (representat amb punts, àrees, volums, etc), s'hauran d'establir un sistema de coordenades i utilitzar el s'anomena un datum (conceptes que es veurà més endavant).

En quan a les aplicacions, els SIG es fan servir per a investigacions científiques, arqueologia, planificació urbana, màrqueting, logística, etc.

2.2 Geodesia i Cartografia

La **geodèsia** (*del grec geo=terra, desios=divisió*) és la ciència que estudia la figura i les dimensions de la terra. Es farà incidència en un dels seus objectius: estudiar el posicionament sobre la superfície terrestre.

Per determinar la posició, però, es necessitarà definir una sèrie de coordenades. Els més utilitzats són: el sistema de coordenades geogràfiques, el sistema de coordenades cartesià i el sistema de coordenades projectades.

Un cop es tenen definits els sistemes de coordenades, interessarà que un punt que està contingut sobre una superfície esfèrica (la terra) pugui representar-se sobre una superfície plana (un planell). Hi ha molts tipus de projeccions i cadascuna d'elles té algun tipus de distorsió. Les projeccions es classifiquen en: conformes, equivalents, equidistants, azimutals i de compromís. L'objectiu és escollir una projecció centrada de forma que asseguri una mínima distorsió per a una determinada zona.

De totes les projeccions, una de les més utilitzades és la de Mercator, més coneguda com a *UTM* (Universal Transversa Mercator). UTM és una projecció cilíndrica dissenyada per a cartografiar objectes continguts en un sol fus. El mètode defineix fins a seixanta projeccions standard diferents. Es gira el cilindre fins a 60 vegades amb separacions de 6 graus per cobrir el total dels 360 graus terrestres. De tal manera que, per a cartografiar qualsevol punt sobre la terra, n'hi ha prou amb triar la línia central del fus UTM més proper al punt.

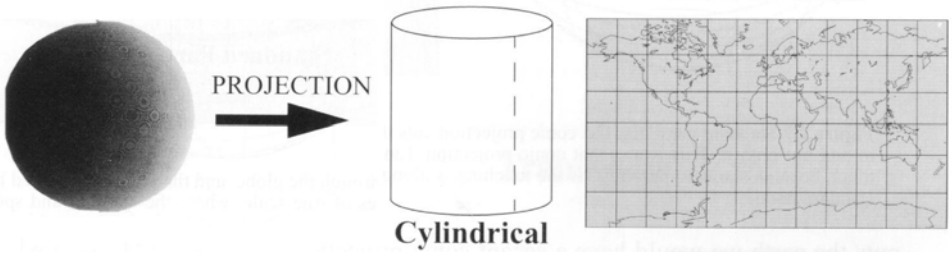

Figura 2.1. Projecció Cilíndrica

Un cop determinades aquestes posicions, farà falta una manera per definir de forma única la localització dels objectes en la superfície de la terra: la georeferenciació.

La forma de la terra és molt irregular i complexa, està condicionada per molts factors que varien amb el temps. Es necessita, però, un mètode que asseguri l'objectiu de la *georeferenciació*. Per això, es necessitarà una forma simplificada de la seva forma per tal de poder-hi treballar. Els models més aproximats són el geoide i l'el·lipsoide.

Un *geoide* és una superfície on la gravetat (Figura 2.2) és la mateixa en tots els llocs i que correspon amb el nivell mitja dels oceans. Es prendrà aquest nivell mig del mar com un punt de referència a partir del qual es descriuran les variacions de l'altura. El geoide, però, és una representació massa complexa desde el punt de vista matemàtic i necessita ser simplificada.

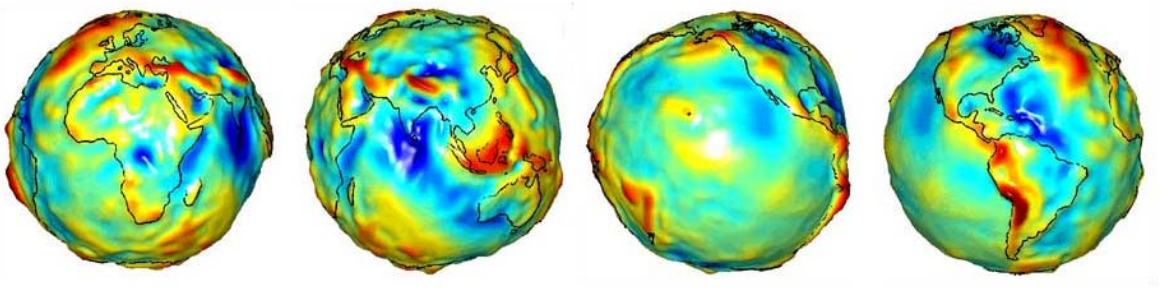

Figura 2.2. Mapa de gravetat terrestre

Una simplificació, i que s'ajusta millor a la forma de la terra, la configura l'*el·lipsoide*. En el globus terraqui podem tenir definits infinitat d'el·lipsoides (Figura 2.3). Depenent de la superfície terrestre que interessi, es pot definir l'el·lipsoide en particular que millor s'hi adapti. Concretament a la península ibèrica disposa del Internacional de Hayford de 1924 i el World Geodetic Datum de 1984 (*WGS84*).

Un cop establerta la ubicació bidimensional, ara sorgeix la necessitat d'ubicar de forma tridimensional l'el·lipsoide triat respecte un geoide. La ubicació serà un punt de tangència comú entre les superfícies del geoide i del el·lipsoide; a aquest punt s'anomena *punt fonamental*. És en aquest punt fonamental on l'el·lipsoide és tangent al geoide. Un datum geodèsic és un el·lipsoide (definit pels seus paràmetres) que defineix les dimensions i la forma de la terra; hi ha dos tipus de datum: el *vertical*, que permet el càlcul d'altures i l'*horitzontal*, que defineix la relació entre la terra física i les coordenades horitzontals.

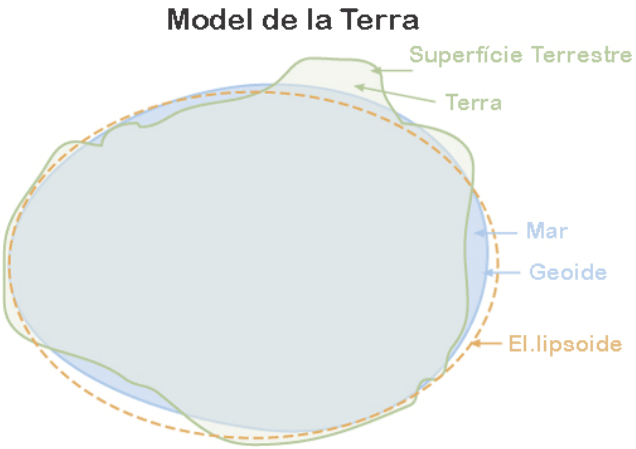

Figura 2.3. Model de la Terra

Quan s'expressen unes coordenades, aquestes han d'anar referides a un determinat datum.

Aquests datums han de materialitzar-se en el terreny de forma física perquè es poden agafar com a punts desde on començar a mesurar una posició. Un conjunt de vèrtexs geodèsics (Figura 2.4) formen el que es coneix com a *xarxes geodèsiques*.

La resta de punts es determinen per triangulació. Depenent de la separació dels vèrtex, aquests formen xarxes d'un o un altre ordre. També es poden expressar les coordenades donades en un datum origen a un datum destí; per això, és necessari aplicar-hi transformacions numèriques a les coordenades; les més comunes són les de Molodenski i les de Bursa-Wolf.

Figura 2.4. Vèrtex Geodèsic

La **cartografia** (*del grec chartis=mapa, grafos=escrit*) és la transmissió d'informació fent servir un sistema de símbols. Un cas particular són els mapes. Cada mapa respon a un propòsit i la informació que conté va en funció del seu objectiu. Es classifica la cartografia oficial en: bàsica, temàtica i derivada.

- En la *cartografia bàsica* es mostren els elements vinculats a la forma de la terra. Alguns exemples típics són: els mapes topogràfics, els mapes d'imatge i les cartes nàutiques.
- La *cartografia temàtica* es centra en la informació d'un tema concret. En són exemples: els mapes geològics, els mapes de transit, els mapes sísmics, etc.
- La *cartografia derivada* és la resultant de processos d'addició d'informació topogràfica a partir d'informació ja existent.

La proporció entre els elements representats en un mapa i la realitat s'anomena *escala*. Els mapes amb escales grans mostren més de detall que les escales petites però cobreixen un menor espai de terreny. Aspectes relacionats amb el mapa són: el grau de detall del mapa, la resolució del mapa i la precisió del mapa. Un mapa perquè estigui ben creat, ha de seguir dues etapes ben definides: la definició de l'objectiu del mapa i la sel.lecció del mètode d'el.laboració.

Els mapes, tot i estar fets sobre paper, també representen la tercera dimensió. La forma que tenen de fer-ho és mitjançant corbes de nivell, ombrejats del terreny i cotes. Un cop es té aquesta representació de la tercera dimensió es poden fer càlculs com, per exemple, determinar el pendent d'un determinat terreny.

Els SIG tracten informació molt diversa; generalment es classifica en: *informació ràster* (imatges de satèl·lit, fotografies aèries, ortofotomapes) o *informació vectorial* (digitalització de mapes, dades obtingudes per GPS, etc).

Les dades vectorials solen representar-se amb punts, línies i polígons. Aquests, proporcionen una major precisió gràfica i maneguen millor la topologia. Les dades ràster solen representar-se amb píxels i representen millor els fenòmens espacials continus, també faciliten la integració de les dades i maneguen millor les operacions de càlcul i actualització.

2.3 Bases de Dades Geogràfiques

La informació de totes aquestes dades es guarden en unes bases de dades específiques: les bases de dades amb capacitats espacials.

Com s'ha vist anteriorment, hi han dos tipus de dades: *vectorials* i *ràster*. En les dades vectorials, les dades a emmagatzemar seran objectes puntuals, lineals o d'àrea. Els seus elements: nodes, arestes i cares són elements bàsics de la topologia, imprescindibles per poder indicar una posició sobre el territori.

El magatzematge d'aquestes *geometries vectorials* crea un gran nombre de taules i de registres que s'han d'emmagatzemar, això es tradueix en un costós manteniment i en una execució lenta de les consultes. Relacionat amb l'execució d'aquestes consultes, hi ha associats la creació de índexs que permeten accelerar-les. Aquests índexs, es coneixen com a índexs espacials, alguns exemples d'ells són: el QuadTree i els R-Trees.

De la mateixa manera, també es creen relacions espacials i de filtrat i operacions especifiques per aquests tipus de dades que fan referència a les propietats de les formes geomètriques quan es sotmeten a processos d'escalat o de projecció. Entre les propietats topològiques més importants hi figuren les disjuncions, interseccions, inclusions, l'ordre, la connectivitat, la proximitat i l'adjacència entre els diferents elements geomètrics. El fet de dotar de topologia a una base de dades implica el poder realitzar càlculs complexes amb menor cost computacional.

Apart de les dades en sí i de les seves operacions, també es necessitarà saber què representa una geometria dins d'un territori; és per això que es requerirà el que es coneix com informació geogràfica de recolzament. O sigui que, un SIG, al connectar-se a una font de dades, ha de ser capaç d'extreure capes, sistemes de coordenades i els camps on es guarda la geometria.

Aquesta geometria, es pot guardar en varis formats d'emmagatzement. Alguns d'ells són: AutoCAD, ESRI Shapefiles, GML, ArcSDE, PostGIS, Oracle Spatial, etc

Les *dades ràster* tenen menys diversitat i es emmagatzemen de forma similar a altres formats. Per fer-nos una idea, es pot fer una analogia amb un objecte cobert amb cel·les, a les quals posteriorment s'hi assignaran valors. El ràster, és un format amb menys precisió que el vectorial i està limitat pel tamany de les cel·les.

Alguns exemples ràster són les imatges georeferenciades (com ortofotoimatges o imatges satèl·lit) i les cobertures (on el valor que pren una cel·la representa el valor d'un variable ubicada a sobre el territori).

Les dades ràster s'agrupen en bandes i capes. Normalment coincideixen, però sovint, a dins d'una banda poden coexistir varies capes.

Alguns format de emmagatzament ràster són: arxius world, GeoTiff, MrSID, ECW i el Georaster.

3. Configuració de l'entorn de treball

En aquest apartat, es definirà l'entorn de treball que utilitzarem, la seva configuració i el programari que es farà servir en el desenvolupament del SIG. Cal recordem que, segons els requeriments de l'enunciat, s'utilitzarà únicament programari lliure.

Una de les dificultats inicials, i no prevista en el pla de treball inicial, ha estat la referida a la problemàtica de la càrrega de les dades. Les dades provinents a través d'una exportació a partir de OpenStreetMaps (OSM) era convenient que s'exportessin cap a una base de dades amb PostgreSQL/postGIS amb una estructura i una topologia determinades de manera que permetés, d'aquesta manera, implementar tota la funcionalitat requerida i poder aplicar els algorismes de càlcul de rutes.

El programari que permetia la creació d'aquesta topologia, osm2pgrouting[1], sols funcionava amb entorns Linux. Aquest fet, ha propiciat la necessitat de la creació sobre Linux Ubuntu[2] d'una base de dades amb PostgreSQL/postGIS equivalent a la creada, inicialment, en entorn Windows.

Posteriorment, s'ha dut a terme l'execució del programari de conversió (osm2pgrouting) i una posterior migració de les dades de la Base de Dades desde Ubuntu cap a Windows XP.

Això, ha suposat una modificació del pla de treball projectat inicialment. Aquestes noves tasques que s'han generat, han afectat als programaris: PostgreSQL, PostGIS, pgRouting i osm2pgrouting. Apart, ha suposat la instal.lació del sistema operatiu Linux Ubuntu 10.4 en un maquinari.

Abans d'entrar en detall, primerament es mostrarà un diagrama on s'aprecia clarament la interacció entre els diferents components. És un esquema adaptat del document "*infraestructura de datos espaciales de España*[3]" del consell superior geogràfic.

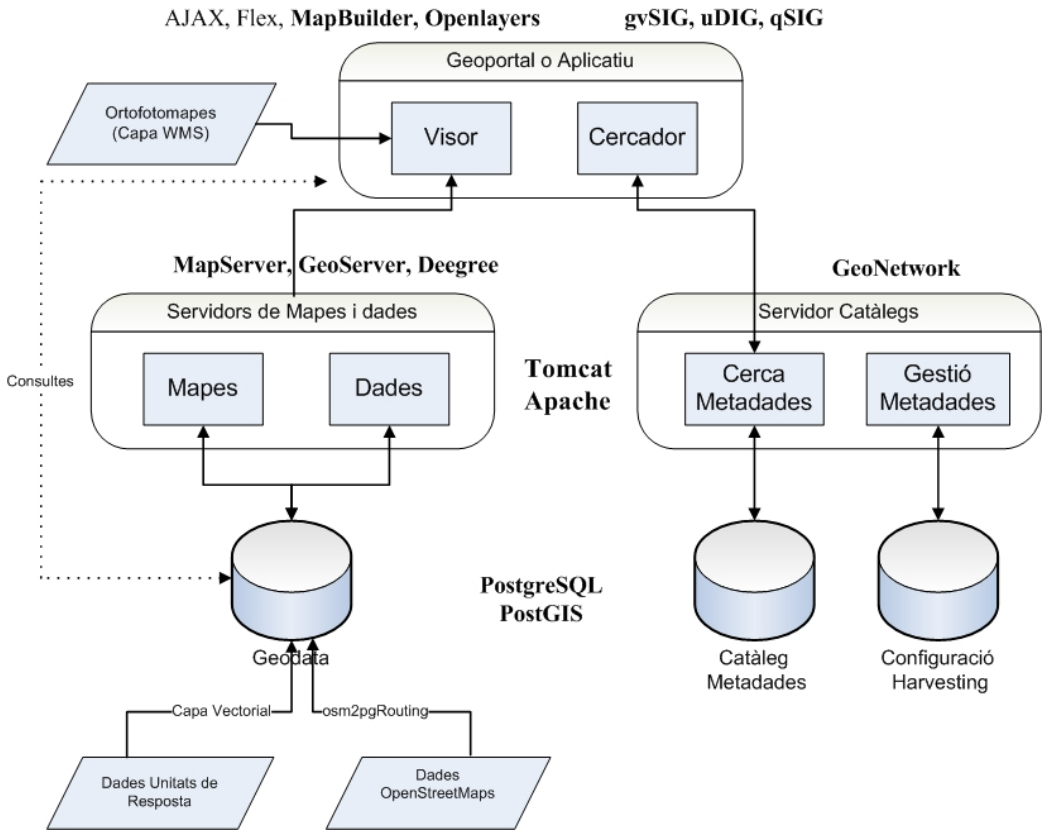

Figura 3.1. Diagrama d'interacció entre els diferents components

3.1 MapServer

MapServer[4] és un entorn de desenvolupament de codi obert per a la creació d'aplicacions SIG tant en Internet com en una Intranet. És una plataforma on es poden publicar dades espacials i mapes interactius en un entorn web. En realitat, es tracta d'un programa CGI (Common Gateway Interface) que s'instal·la en un servidor web i que rep peticions per la generació i publicació de mapes. A més, l'aplicatiu, també permet la consulta i l'anàlisi de la informació geogràfica.

Les peticions es realitzen mitjançant el navegador i a través de Mapfiles (.map). En aquests fitxers la informació està organitzada per capes i fa referència a com s'accedirà a la informació geogràfica. En aquest fitxer també hi estarà inclòs el format del mapa de sortida (gràfic o alfanumèric).

MapServer, suporta els estàndards i les especificacions definides per l'Open Geospatial Consortium (OGC), un consorci que agrupa més de 350 organitzacions públiques i privades. Entre les especificacions més importants, destaquen:

- GML (Geography Markup Language): Llenguatge de Marcat Geogràfic.
- KML (Keyhole Markup Language) : Llenguatge de marcat basat en XML per representar dades geogràfiques en tres dimensions.
- WFS (Web Feature Service) : Proporciona informació relativa a l'entitat emmagatzemada en capes vectorials.
- WMS (Web Map Service) : Es tracta d'un servei de mapes en la web que produeix mapes en format imatge sota demanda i que permet ser visualitzats per un navegador web.

Algunes d'aquestes especificacions es faran servir en el projecte perquè serviran per a la cerca d'altres fonts de dades, tal i com demana l'enunciat.

En aquest projecte s'utilitzarà MapServer (MS4W) versió 5.2.1 en un servidor web Apache versió 2.2.10. Per a la instal.lació de MapServer sols ha calgut seguir els passos del instal·lador i especificar-li com a port Apache el 9000 per tal d'evitar problemes posteriors de bloquejos per part de Windows. Per tal de comprovar el correcte funcionament del servidor, s'ha visualitzat la pantalla de benvinguda del servidor a través d'un navegador, tot fent: *http://localhost:900*0.

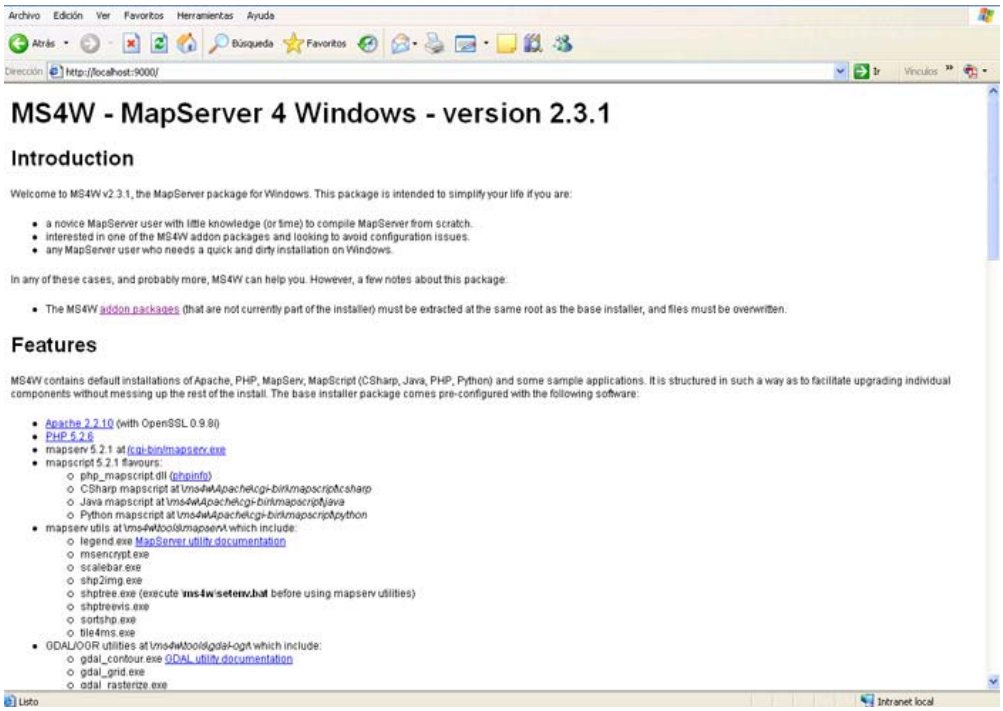

Figura 3.2. Pantalla de presentació de MapServer 2.3.1

3.2 Ubuntu

En una màquina en paral·lel, s'ha procedit a instal·lar el sistema operatiu Ubuntu versió 10.4 per a 32 bits. Prèviament, s'ha creat un CD instal·lable amb el sistema operatiu descarregat i, posteriorment mitjançant el CD, s'ha procedit a instal·lar el sistema operatiu en l'esmentada màquina. La instal·lació ha transcorregut sense incidències remarcables.

Posteriorment a la instal·lació del sistema operatiu, s'ha realitzat les pertinents actualitzacions disponibles fins al moment de realitzar aquest projecte.

3.3 PostgreSQL

PostgreSQL[5], és un sistema de gestió de bases de dades relacionals orientat a objectes de codi lliure. Té com a principals característiques:

- *Alta concurrència*: Disposa d'un sistema anomenat MVCC (MultiVersion Concurrency Control) que permet un accés simultani a les dades d'una taula, on cada usuari té la seva pròpia vista. Això elimina l'ús de bloquejos explícits per taules o per files típics d'altres bases de dades.

- *Gran varietat de tipus natius*: Suporta una àmplia varietat de tipus de dades: nombres de variada precisió, figures geomètriques, adreces IP, adreces MAC i blocs d'adreces, matrius, texts amb llargades il·limitades, etc.

Altres característiques són: claus foranes, disparadors (*triggers*) integrats a dins de la base de dades, vistes, herència de taules i gran varietat de funcions. Existeixen blocs de codi, escrits en gran diversitat de llenguatges, que s'executen en el servidor i que li proporcionen funcionalitats addicionals.

Al voltant seu hi ha un gran nombre d'utilitats que potencien el treball amb PostgreSQL. Algunes d'aquestes eines són: PostGIS (extensió de suport d'objectes geogràfics), PgCluster (eina de replicació) i PgAdmin3 (eina d'administració). Algunes d'elles, es veuran més endavant en detall.

Tornant a la instal.lació del programari, tal i com comentàvem al inici d'aquest apartat, el fet de tenir que treballar amb Linux, ha fet que la tasca d'instal·lació de postgreSQL s'hagi dividit en dues subtasques.

3.3.1 PostgreSQL sobre Ubuntu 10.4

La versió de PostgreSQL que s'ha fet servir ha estat la 8.4 i la instal·lació s'ha realitzat a partir de *Synaptic*, de la mateixa manera, i fent servir la mateixa eina, s'ha instal·lat pgadmin3 amb la seva versió 1.10.3. La instal·lació s'ha transcorregut sense cap incidència.

3.3.2 PostgreSQL sobre Windows XP

La versió de PostgreSQL per a Windows que s'ha utilitzat ha estat la 8.4.4-1. L'arxiu ha estat un programa autoinstal·lable que demanava a on s'instal·laria el programa, a on s'ubicarien les dades, el port i la contrasenya que faria servir.

Per acabar, a través de *Stack Builder* s'ha instal·lat el PostGIS, versió 1.5.1 que es descriu seguidament.

3.4 PostGIS

PostGIS[6] és una extensió del SGBD PostgreSQL que ofereix suport d'objectes geogràfics. PostGIS ha estat desenvolupat per l'empresa canadenca Refraction Research, especialitzada en productes Open Source, i es publica sota la llicència pública general de GNU. PosGIS segueix l'especificació Simple Features for SQL de l'OGC[7,8] i, a més, inclou:

- Diferents tipus de geometries per punts, polígons, multipunts, multipolígons i col·leccions de geometries.
- Predicats espacials per determinar diferents interaccions entre geometries.
- Operadors espacials per determinar mesures geoespacials com àrees, distàncies, longituds i perímetres.
- Operadors espacials per determinar operacions geoespacials com unions, diferències, etc.
- Indexació espacial de dades mitjançant estructures GiST (Generalized Search Tree), R-tree-overGiST, etc.

Agafant com a referència documentació específica sobre el tema[9], es farà una descripció de les dades que contenen les principals taules

- Taula *spacial_ref_sys*: Conté els indicadors numèrics i les descripcions textuals dels sistemes de referència. És una taula on es llisten més de 3000 sistemes de referència espacial, amb els detalls de la projecció i de transformació entre ells. Concretament, en cada camp s'hi guarda:
    - *srid*: Nombre enter que identifica unívocament al sistema de referència espacial en la base de Dades.
    - *auth_name*: Nom del estàndard per a aquest sistema de referència.
    - *auth_srid*: Conté un nombre assignat pel responsable.
    - *srtext*: Identificador del sistema tal i com ve definit l'estàndard que s'indica al camp *auth_name*.
    - *proj4text*: Cadena amb la definició de les coordenades de la llibreria Proj4 per a un sistema d'identificació espacial (SRID) donat.

- Taula *geometry_columns*: Taula que conté la descripció de totes les columnes geomètriques de les diferents taules de la base de dades. Guarda un índex de les taules que contenen algun tipus de camp amb geometria. Concretament, en cada camp s'hi guarda:
    - *f_table_catalog*, f_table_schema, f_table_name: el nom complert de la taula que conté la geometria.
    - *f_geometry_column*: Nom de la columna geomètrica en la taula de les característiques, aquest camp emmagatzema la geometria.
    - *coord_dimension*: Dimensió espacial (2, 3 o 4 dimensions) de la columna de la geometria.
    - *srid*: Identificador del sistema de referència espacial que fa servir la geometria de la taula.
    - *type*: Tipus de l'objecte espacial (punt, polígon, etc).

També s'han inclòs la llibreria geomètrica OSGEOS4W[10] i la llibreria de projeccions cartogràfiques Proj 4.4.6[11] Després de descriure els camps i les seves característiques generals, es detallarà la instal·lació d'aquest programari en ambdós entorns:

	srid [PK] int	auth_name character va	auth_srid integer	srtext character varying(2048)	proj4text character varying(2048)
1	2000	EPSG	2000	PROJCS["Anguilla 1957 / British West Indies Grid",GEOGCS["A	+proj=tmerc +lat_0=0 +lon_0=-62 +k=0.9995000000000001 +x_0=40
2	2001	EPSG	2001	PROJCS["Antigua 1943 / British West Indies Grid",GEOGCS["A	+proj=tmerc +lat_0=0 +lon_0=-62 +k=0.9995000000000001 +x_0=40
3	2002	EPSG	2002	PROJCS["Dominica 1945 / British West Indies Grid",GEOGCS["D	+proj=tmerc +lat_0=0 +lon_0=-62 +k=0.9995000000000001 +x_0=40
4	2003	EPSG	2003	PROJCS["Grenada 1953 / British West Indies Grid",GEOGCS["G	+proj=tmerc +lat_0=0 +lon_0=-62 +k=0.9995000000000001 +x_0=40
5	2004	EPSG	2004	PROJCS["Montserrat 1958 / British West Indies Grid",GEOGCS	+proj=tmerc +lat_0=0 +lon_0=-62 +k=0.9995000000000001 +x_0=40
6	2005	EPSG	2005	PROJCS["St. Kitts 1955 / British West Indies Grid",GEOGCS["St	+proj=tmerc +lat_0=0 +lon_0=-62 +k=0.9995000000000001 +x_0=40
7	2006	EPSG	2006	PROJCS["St. Lucia 1955 / British West Indies Grid",GEOGCS["S	+proj=tmerc +lat_0=0 +lon_0=-62 +k=0.9995000000000001 +x_0=40
8	2007	EPSG	2007	PROJCS["St. Vincent 45 / British West Indies Grid",GEOGCS["S	+proj=tmerc +lat_0=0 +lon_0=-62 +k=0.9995000000000001 +x_0=40
9	2008	EPSG	2008	PROJCS["NAD27(CGQ77) / SCoPQ zone 2",GEOGCS["NAD27(C	+proj=tmerc +lat_0=0 +lon_0=-55.5 +k=0.9999 +x_0=304800 +y_0=
10	2009	EPSG	2009	PROJCS["NAD27(CGQ77) / SCoPQ zone 3",GEOGCS["NAD27(C	+proj=tmerc +lat_0=0 +lon_0=-58.5 +k=0.9999 +x_0=304800 +y_0=

Figura 3.3. Contingut parcial de la taula 'spacial_ref_sys'

	oid	f_table_catal [PK] characte	f_table_schei [PK] characte	f_table_name [PK] characte	f_geometry_ [PK] characte	coord_dimen: integer	srid integer	type character varying(30)
1	21198	''	public	unit_resposta	the_geom	2	4326	POINT
2	28950	''	public	unit_resposta_a	the_geom	2	4326	POINT
3	21228	''	public	unit_resposta_b	the_geom	2	4326	POINT
4	21243	''	public	unit_resposta_h	the_geom	2	4326	POINT
5	21213	''	public	unit_resposta_p	the_geom	2	4326	POINT
6	21037	''	public	vertices_tmp	the_geom	2	4326	POINT
7	21036	''	public	ways	the_geom	2	4326	MULTILINESTRING

Figura 3.4. Contingut de la taula 'geometry_columns'

3.4.1 PostGis sobre Ubuntu 10.4

Sobre Ubuntu, s'ha hagut de procedir, de forma separada, a la instal·lació de PostGIS per una banda i, per l'altre, de les eines de desenvolupament i compilació build-essentials.

Degut a les dependències que té PostGIS, s'ha hagut d'instal·lar les llibreries de les quals en depèn: build-essentials 11.4, libgeos-dev 3.1 i proj 4.6.

3.4.2 PostGis sobre Windows XP

En entorn Windows i tal i com hem comentat anteriorment, la instal.lació s'ha dut a terme a través del Stack Builder. S'ha fet servir la versió PostGis 1.5.1 (sobre PostgreSQL 8.4). Durant la instal·lació, s'ha generat també una plantilla de base de dades que inclou les extensions PostGIS (*template_postgis*).

3.5 PgRouting

Per al càlcul de rutes, s'utilitzarà l'extensió pgRouting[12] . Aquesta extensió de la base de dades Postgis aportarà la funcionalitat necessària per a poder fer el càlcul de rutes de un mínim cost entre dos nodes. El càlcul es farà a través d'algoritmes eficients, com el de Dijkstra, A* o Shooting Star i s'aplicaran sobre les consultes que es vagin fent sobre la base de dades PostGIS.

3.5.1 PgRouting sobre Ubuntu 10.4

La instal·lació de pgRouting (versió 1.03) sobre Ubuntu, ha estat el punt més crític en quan a incidències sorgides. Per a la seva compilació, s'ha hagut d'instal·lar la llibreria *cmake*, així com, tenir en compte les dependències de pgRouting cap a les llibreries: *libboost-graph-dev* (per als algoritmes shortest-path), *gaul-devel* (per al TSP) i, finalment, *libcgal3* i *ligcgal-dev* (per al driving distance).

Les versions que han estat instal·lades han estat:

- cmake 2.6.4-1ubuntu2 (a partir de Synaptic).
- libboost-graph-dev 1.38 (també a partir de Synaptic)
- gaul-devel 0.1850-0 contingut de l'adreça de descàrrega[13].
- libcgal3 i ligcgal-dev 3.4-4ubuntu1 (a partir de Synaptic).

Durant el procés de generació dels arxius *make*, al afegir-hi TSP i DD, s'han produït una sèrie d'errades de compilació, que s'han solventat seguint les indicacions descrites a la plana web de pgRouting[14,15], i fent les següents modificacions:

1) S'ha afegit en els següents arxius la instrucció *#include "catalog/pg_type.h":*

```
./core/src/dijkstra.c
./core/src/astar.c
./core/src/shooting_star
./extra/driving_distance/src/alpha.c
./extra/driving_distance/src/drivedist.c
```

2) i la instrucció *#include "stdio.h*"en l'arxiu:

```
./src/XMLPArser.cp
```

3) Un cop compilat i instal·lat pgRouting (revisió 356), s'ha creat una plantilla de base de dades amb la funcionalitat de pgRouting la qual s'ha denominat *template_routing* creada a partir de *template_postgis* i executant els següents scripts sql:

```
./usr/share/postlbs/routing_core.sql
./usr/share/postlbs/routing_core_wrappers.sql
./usr/share/postlbs/routing_topology.sql
./usr/share/postlbs/routing_tsp.sql
./usr/share/postlbs/routing_tsp_wrappers.sql
```

3.5.2 PgRouting sobre Windows XP

Pel que fa al pgRouting, s'ha instal·lat la versió 1.0.3. Per instal·lar-lo, s'ha descomprimit l'arxiu zip obtingut i s'ha afegit el contingut de les carpetes *lib* i *share* a dins de les carpetes corresponents dins del directori d'instal·lació de PostgreSQL (per defecte, C:\Archivos de programa\PostgreSQL\8.4).

Tot seguit, s'ha procedit a crear una plantilla de base de dades *template_routing* a partir de la plantilla *template_postgis*. Per acabar, s'han afegit les funcionalitats de pgRouting, tot executant les consultes:

Aquestes consultes estaven ubicades en el subdirectori *\share\contrib* del directori d'instal·lació de PostgreSQL.

```
Routing_core.sql
Routing_core_wrappers.sql
Routing_topology.sql
```

3.6 OpenLayers

OpenLayers[16] és una llibreria de JavaScript idònia per a la visualització de mapes en entorn web. Com a diferents mètodes d'accés a les dades geogràfiques, la llibreria implementa, entre altres, els protocols WMS i WFS.

Això, permet incorporar mapes procedents de diverses fonts a una pàgina web, aportant funcionalitats de navegació i de selecció de capes.

La instal·lació d'OpenLayers (versió 2.10) s'ha fet de forma exclusiva sobre Windows XP. Per a la seva instal·lació sols ha calgut descomprimir i copiar la carpeta obtinguda a sobre de la de MapServer: MS4W.

3.7 Eines complementàries

En aquest apartat es descriuen les eines complementàries que s'han fet servir:

- S'han testejat els següents SIGs d'escriptori: gvSIG[17] uDIG[18] qGIS[19]. Aquests SIGs permeten un treball més còmode amb les dades geogràfiques de la base de dades postGIS i amb els servidors WMS. El seu entorn gràfic permet la creació de capes de forma visual. Finalment, s'ha optat per utilitzar qSIG.

- Firebug[20]: Extensió del Mozilla Firefox per a la depuració del llenguatge Javascript. La seva instal·lació (versió 1.5.2) té com a requisit previ la instal·lació de Firefox, del qual s'ha instal·lat la versió 3.6.2.

- NotePAD++[21]: Editor de text, de codi font lliure, versió 5.8.2.

4. Obtenció i tractament de les dades

En aquest apartat, es descriu com i d'on s'han obtingut les dades geogràfiques necessàries per la realització del projecte i de quina forma s'han emmagatzemat en la base de dades pel posterior càlcul de rutes.

L'àrea geogràfica que es tractarà en el projecte correspon a la ciutat de Barcelona. El motiu d'escollir aquest municipi ha estat les grans possibilitats que ofereix una ciutat de les dimensions de Barcelona.

Per tal de realitzar els diferents requeriments demanats pel projecte, primer serà necessari disposar de les dades dels carrers de la ciutat així com de les localitzacions de les diferents unitats de resposta urgents de que disposi la ciutat. Aquestes dades, s'hauran d'obtenir de tal forma que es puguin introduir posteriorment en una base de dades espacial sobre la que es realitzaran consultes i càlculs.

Inicialment, s'ha contemplat l'obtenció de les dades del municipi de més d'una font. El motiu ha estat el fet de poder disposar de criteri per tal de determinar la opció que millor s'adapta a la nostra problemàtica.

De les dues opcions que s'han contemplat: Cartociudad[22] i OpenStreetMaps[23]; finalment, tot i ser els dos uns bons mètodes, s'optarà per la segona opció. El motiu de l'elecció ha estat purament pràctic. Cartociudad és el resultat de la integració i homogeneïtzació de les dades aportades per a diferents organismes públics (INE, IGN, Societat de Correus i telègrafs, per esmentar-ne uns quants). Això ha donat lloc a un sistema de informació geogràfica d'àmbit nacional. Segons la seva documentació està orientat a obtenir informació orientada al cens, parcel·lària i postal. En canvi, OpenStreetMaps està orientat a un propòsit més general.

4.1 Recerca i extracció de dades d'OpenStreetMaps

OpenStreetMaps és un projecte col·laboratiu per tal de crear i proporcionar dades geogràfiques lliures. Els diferents punts de resposta es seleccionaran i referenciaran de forma manual mitjançant la generació d'una capa amb el programari *qSIG*. Posteriorment, en l'entorn d'usuari final, com veurem, s'utilitzaran altres fonts de dades cartogràfiques per tal de dotar d'un aspecte més complet al mapa final resultant.

Inicialment, es farà una exportació de les dades de l'àrea desitjada, tot connectant-nos a l'adreça d'OpenStreetMaps referenciada anteriorment i sel.leccionant l'àrea d'interès. En el nostre cas, s'exportarà una àrea compresa entre les latituds *41,3671 i 41,449* i les longituds *2,0994* i *2,2329* (figura 4-1), coordenades geogràfiques expressades en graus decimals.

En fer l'exportació de les dades, es triarà l'opció *OpenStreetMap XML Data*. Aquesta opció permetrà fer una exportació de l'àrea sel.leccionada en format XML (eXtensible Markup Language). Finalment, s'obtindrà un arxiu amb extensió *osm* que serà, a través del que, més endavant, es farà la càrrega a la base de dades.

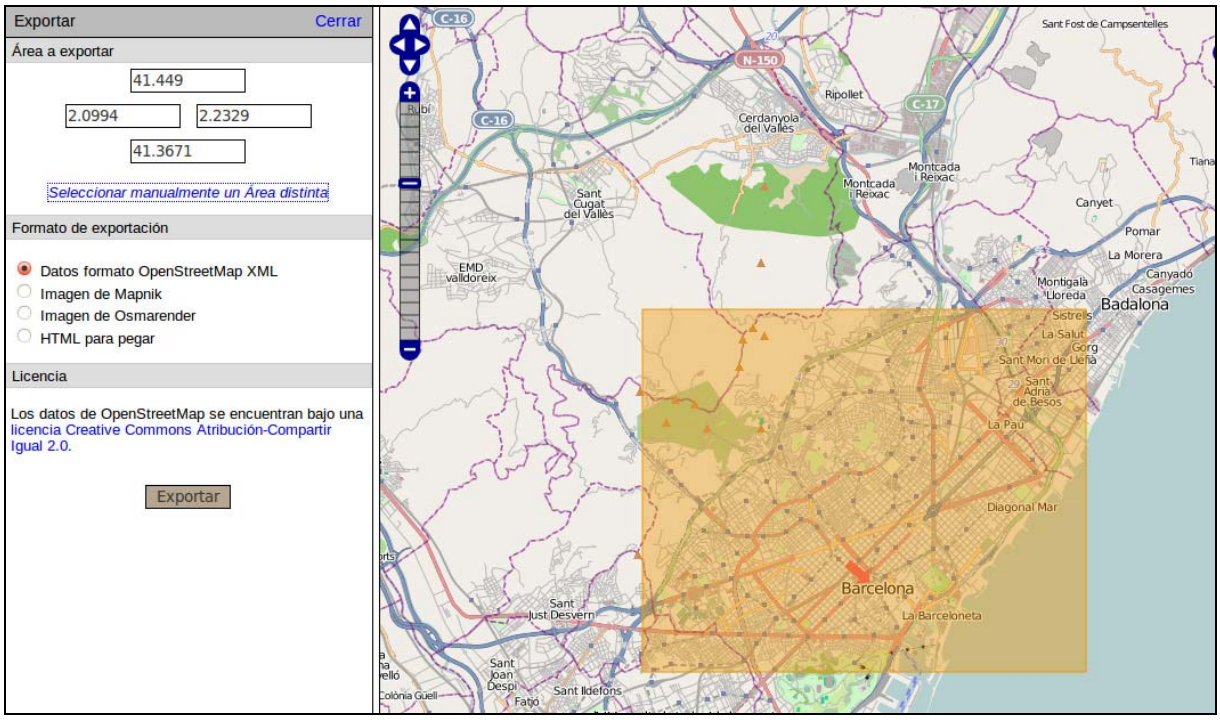

Figura 4.1. Àrea de la ciutat de Barcelona exportada d'OpenStreetMaps.

4.2 Creació de la base de dades espacial PostGIS

Per crear la base de dades espacial PostgreSQL / PostGIS s'ha creat una plantilla de base de dades PostgreSQL sobre la que s'ha afegit la funcionalitat PostGIS i pgRouting per, respectivament, donar suport d'objectes geomètrics i poder utilitzar els seus algoritmes de càlcul de rutes. Posteriorment es crearà la base de dades de treball (pgis_BCN) fent servir aquesta mateixa plantilla.

Un cop afegits els requeriments necessaris i solventades les incidències que han anat sorgint, el procediment ha estat el següent:

1) Connexió i autentificació com a superusuari:

```
sudo su postgres
```

2) Es crearà la base de dades PostgreSQL (que servirà de plantilla) i activem el llenguatge PL/pgSQL en aquesta base de dades:

```
createdb  -E UTF8 pgis_BD
createlang  plpgsql pgis_BD
```

3) Afegir la funcionalitat postGIS:

```
psql  -f  /usr/share/postgresql/8.4/contrib/postgis.sql pgis_BD
psql  -f  /usr/share/postgresql/8.4/contrib/spatial_ref_sys.sql  pgis_BD
```

4) Agregar funcionalitat bàsica de pgRouting:

```
psql  -f  /usr/share/postlbs/routing_core.sql pgis_BD
psql  -f  /usr/share/postlbs/routing_core_wrappers.sql pgis_BD
psql  -f  /usr/share/postlbs/routing_topology.sql pgis_BD
```

5) Incorporar funcionalitat TSP de pgRouting:

```
psql  -f  /usr/share/postlbs/routing_tsp.sql pgis_BD
psql  -f  /usr/share/postlbs/routing_tsp_wrappers.sql pgis_BD
```

6) Crear la base de dades espacial de treball pgis_BCN a partir de la plantilla pgis_BD:

```
createdb  -T pgis_BD pgis_BCN
```

4.3 Carrega de dades a la Base de Dades

Un cop creada la base de dades, es farà la càrrega d'aquestes dades. Es començarà per fer la càrrega de les dades dels carrers de Barcelona i, seguidament, es farà el mateix amb les diferents dades de les unitats de resposta.

4.3.1 Inserció de les dades genèriques del Municipi

En la recerca que hem dut a terme, hem trobat molts mètodes per fer aquesta càrrega genèrica de dades. Inicialment, s'han considerat diferents opcions:

- *osm2pgsql*[24]: Aquesta utilitat permet convertir dades en format OpenStreetMaps (OSM) i carregar-les a PostgreSQL. Té com a principal desavantatge no poder treballar de forma directa amb fitxers OSM sinó que ho fa a través de la eina de renderització de mapes *Mapnik*. Uns altres inconvenients són la gran quantitat de memòria que consumeix en la seva execució en sistemes de 32 bits i en les taules que crea durant el procés (planet_osm_line, planet_osm_point, planet_osm_polygon i planet_osm_roads) no es genera la topologia que es necessitarà posteriorment.

- *osm2pgr*[25]: És una eina escrita amb Java sense dependències com la majoria de les seves predecessores. Com a sortida genera un fitxer amb SQL i crea una sola taula (*osm_topo*). Com a principals desavantatges destaquen el caire experimental de l'eina i el fet que tampoc generarà la topologia necessària.

- *OpenGeo suite*[26]: Aquesta suite permet fer una importació de dades, però sols amb format de dades shape file (shp). Les dades s'han hagut d'extreure d'un programari que permetés fer una captura de dades en arxius Shape: *OSMTranslator*[27]. El programa generarà tres taules (lines, points i regions) degut a que el propi format shape sols accepta un sol tipus de geometria en cada arxiu shp. Aquesta limitació ha estat un factor de pes perquè fos desestimat.

- *osm2pgrouting*[28]: Eina que permet crear sofisticades funcions de càlcul de cost i la topologia necessària per aquesta mena de càlculs que necessitem. Al fer la importació de l'arxiu OSM amb les dades del municipi, la utilitat crea, en la base de dades, totes les estructures necessàries: taules, classes i els diferents tipus de dades. La utilitat utilitza un arxiu XML de configuració per triar els tipus de vies i totes les classes a importar.

Després de la valoració de les diferents opcions per a la càrrega de dades, s'ha considerat adient escollir *osm2pgrouting* per la seva eficàcia en crear la topologia de manera adequada al seu ús per pgRouting. Com ja s'ha comentat, el fet d'escollir aquesta eina representarà una problemàtica especial: la no disponibilitat del programari osm2pgrouting per a sistemes operatius Windows.

Donada la potència de l'eina a l'hora de crear la topologia de les dades, indispensable per aplicar funcions de càlcul de costos, suposarà un canvi en la planificació del projecte al tenir que instal·lar un entorn paral·lel amb Ubuntu.

El procediment, per a la càrrega de les dades, ha estat mitjançant l'execució de l'ordre:

```
./osm2pgrouting -file ./bcn.osm -conf ./mapconfig.xml -dbname pgis_BCN -user postgres –passwd parker33
```

D'aquesta ordre cal comentar el fitxer de configuració *mapconfig.xml*. Aquest fitxer conté la informació dels tipus de vies que s'han d'inserir a la base de dades; osm2pgrouting n'incorpora un per defecte i ha estat aquest mateix el que s'ha utilitzat.

Durant el procés, s'ha generat un log en l'execució, un extracte del qual, és el que es mostra tot seguit:

```
connection success
Trying to load config file mapconfig.xml
Trying to parse config
SE for <configuration>
SE for <type>
SE for <class>
class name = motorway
class id = 101
class id = 101 name = motorway added to type name=highway
SE for <class>
class name = motorway_link
class id = 102
class id = 102 name = motorway_link added to type name=highway
SE for <class>
class name = motorway_junction
class id = 103
class id = 103 name = motorway_junction added to type name=highway

.....

Edge 52513561 is oneway
We need a way of type highway and class residential
Edge 4750730 is oneway
We need a way of type highway and class residential
We need a way of type highway and class pedestrian
We need a way of type highway and class path

.....

Split ways
NOTICE: CREATE TABLE / PRIMARY KEY creará el índice implícito «nodes_pkey» para la tabla «nodes»
Nodes table created
NOTICE: CREATE TABLE / PRIMARY KEY creará el índice implícito «ways_pkey» para la tabla «ways»
Ways table created
Types table created
Classes table created
create topology
NOTICE: CREATE TABLE creará una secuencia implícita «vertices_tmp_id_seq» para la columna serial «vertices_tmp.id»
CONTEXT: sentencia SQL: «CREATE TABLE vertices_tmp (id serial)»
PL/pgSQL function "assign_vertex_id" line 14 at sentencia EXECUTE
##########################
size of streets: 5391
size of splitted ways : 16321
finished
postgres@jordi-desktop:/usr/share/pgrouting/osm2pgrouting$
```

Després de l'execució del programa, s'han generat aquest conjunt de taules, seqüències i vistes:

Taules	Seqüències	Vistes
classes geometry_columns spatial_ref_sys types nodes vertices_tmp ways	vertices_tmp_id_seq	geography_columns

Més endavant, es farà una descripció detallada de la informació que contenen cadascun dels camps de les taules.

Un cop carregada la base de dades a qGIS, s'obtindrà la representació de la taula ways i de vertices_tmp, representada cada una en capes diferents. Per tal de millorar la imatge i dotar-la d'un aspecte més realista s'han utilitzat ortofotografies de la ciutat subministrades a través del servei web de mapes del Institut Cartogràfic de Catalunya[29] (ICC).

4.3.2 Migració de les dades desde Ubuntu cap a Windows.

Un cop creades les taules i inserides les dades per part del programari osm2pgrouting en la base de dades que se li ha especificat, es procedirà a la migració d'aquestes taules i de les seves dades cap a Windows XP.

Per fer-ho es recorrerà a dues eines pròpies de PostgreSQL[30]: *pg_dump* i *pg_restore*. Per fer-ho s'haurà de fer servir el superusuari del sistema.

- **pg_dump:** És una utilitat que permetrà fer una còpia de la base de dades; la forma que té de fer-ho és creant un fitxer de text editable que contindrà ordres SQL (DDL) per a la posterior creació de la base de dades en la destinació.

En la ordre, es poden observar els següents paràmetres:

```
pg_dump -U postgres -C -f C:\Temp\pgis_BCN_copia   pgis_BCN
```

-C crea una base de dades nova
-f indica que seguidament se li especificarà la ubicació del fitxer de la còpia

Per tant, amb aquesta ordre el que farà serà crear una nova base de dades a partir de la base de dades pgis_BCN i el fitxer que permetrà fer tot això serà el pgis_BCN_copia i estarà ubicat a C:\temp.

Posteriorment, s'agafarà aquest fitxer que haurà creat i es migrarà cap a entorn Windows XP. Desde allà es farà la restauració.

- **pg_restore:** Amb aquesta utilitat es podrà restaurar la còpia de la base de dades feta anteriorment. Per fer-ho cal obrir una sessió amb símbol del sistema i executar-la:

En la ordre, es pot observar els següents paràmetres:

```
pg_restore -l C:\Temp\ pgis_BCN_copia
```

-l indica el fitxer a restaurar

Aquesta ordre el que farà serà executar el fitxer creat amb anterioritat. Aquest fitxer ja contindrà totes les ordres per a la creació de la base de dades i per la inserció de les corresponents dades.

De forma alternativa, també es pot fer desde *Admin III,* obrint la base de dades a la que interessi posar-hi les dades i obrir una nova consulta SQL amb el contingut del fitxer generat anteriorment. El resultat serà el mateix.

Un cop carregades de nou les dades en entorn Windows, es procedirà a fer-ne una primera mostra a través de qGIS. El procediment ha estat el següent:

Inicialment, s'afegeix a qGIS una capa que tindrà origen en PostGIS. Per fer-ho, es crearà una connexió a la base de dades i s'escolliran les taules: *vertices_tmp* i *ways*.

Aquest procés crearà dues capes, una amb cada informació. També es podran veure de forma separada o conjunta. Així, en la figura 4.2 es podrà apreciar la capa de la taula ways i en la figura 4.3 la taula ways més la vertices_tmp.

Ara, a través de qGIS s'afegirà una capa WMS. Per obtenir-la, caldrà una connexió al servei web de mapes del ICC mitjançant la URL del servidor proporcionat[31].

Seguidament, sols s'ha hagut de triar entre les diferents opcions per a carregar la ortofotografia que ha semblat més adient, seguidament s'ha aplicat a aquesta nova capa un cert grau de transparència per tal de que els demés elements també fossin visibles.

Seguidament, es mostra una evolució del procés. Es pot observar en les dues imatges inferiors, les vies amb una ortofotografia i, amb les vies, una ortofotografia i la ubicació de les diferents unitats de resposta.

Figura 4.2. Vies de la ciutat.

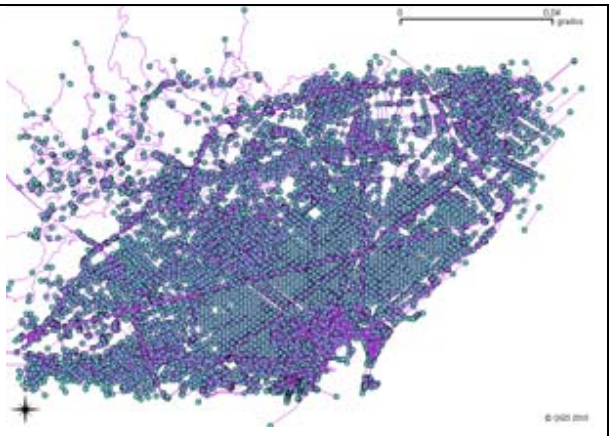

Figura 4.3. Vies i nodes de la ciutat.

Figura 4.4. Vies i ortofotografia de la ciutat.

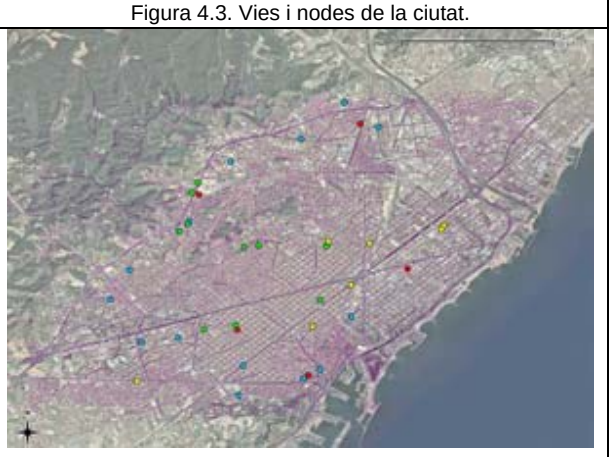

Figura 4.5. Vies, ortofotografies i unitats de resposta.

4.3.3 Inserció de les unitats de resposta del Municipi

Seguidament, es detallarà el procés per a la inserció de les diferents unitats de resposta. La sel.lecció de les unitats s'ha fet de la forma més *realista i variada* possible. Han estat un total de 36 unitats de resposta establertes de la forma següent: 7 ambulàncies, 5 bombers, 10 hospitals i 14 policia.

Hi ha hagut una tasca laboriosa i acurada per tal de ubicar de forma correcta les diferents unitats. Aquesta riquesa de dades, permetrà fer posteriorment altres estudis paral·lels i detallats en el pla de treball. La forma d'ubicar-les ha estat la següent: S'ha fet la cerca de les diferents unitats a google maps[32], seguidament, s'ha passat a ubicar el punt a qGIS fent-lo coincidir amb un node proper existent. Per a cada una d'elles, com a informació addicional; a tall d'exemple, sols s'ha informat del seu telèfon.

S'han distribuït les diferents unitats en taules diferents per tal d'assignar-les-hi la capa corresponent. D'aquesta manera es feia la imatge molt més entenedora.

Per a la senyalització de les unitats, s'han fer servir rodones de diferents colors enlloc d'icones (Figura 4-6). El motiu es que aquestes permeten una major visibilitat sense necessitat de fer ampliacions. De la mateixa manera s'han triat colors primaris amb tons clars per tal que el contrast fos més gran; també, s'ha considerat un tamany adequat dels icones de tal manera que tinguessin una proporció adequada amb les imatges del fotoortografia de la ciutat.

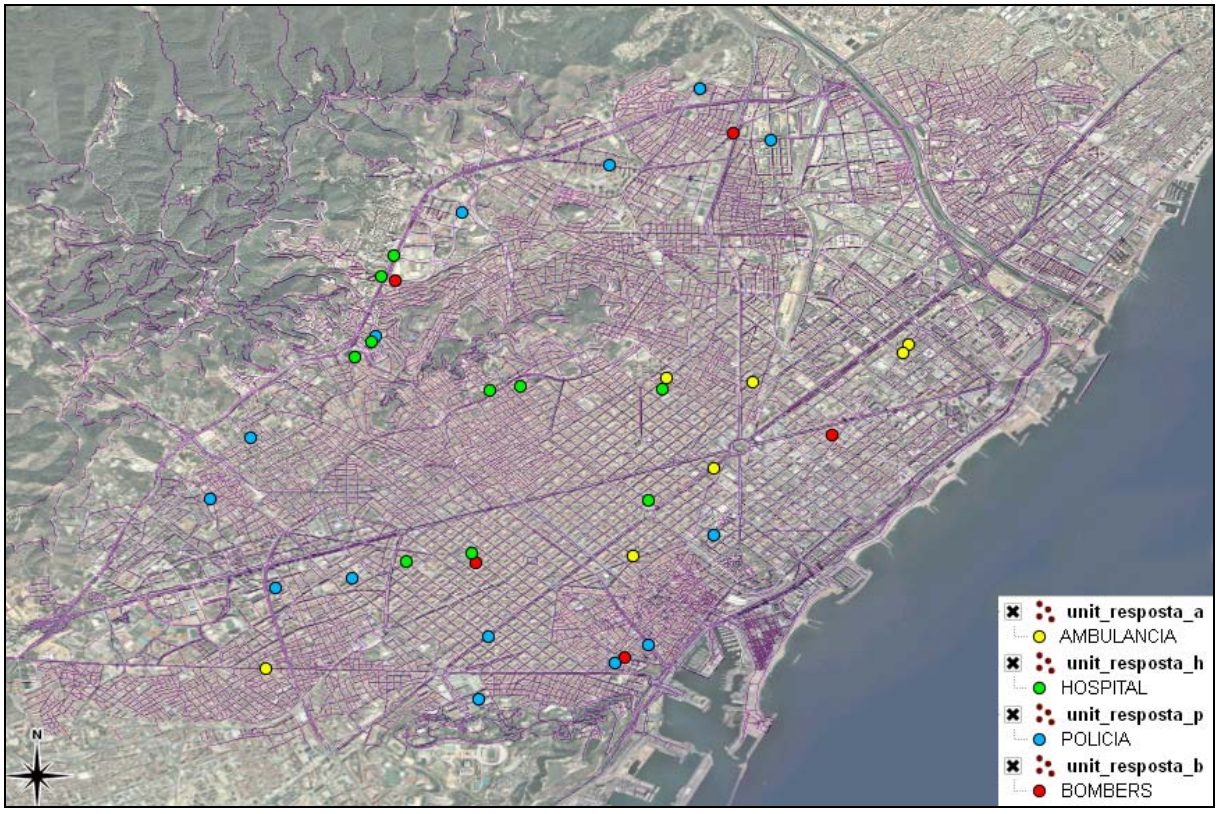

Figura 4.6. Unitats de resposta de Barcelona

A nivell de base de dades, s'ha comentat que cada capa es guardava en una taula diferent, aquest és el contingut de les diferents taules:

Ambulàncies:

id [PK]	name text	lat double precis	lon double precis	the_geom geometry	descr text
1	Ambulancias Tonas	41.400765	2.183712	0101000020E6100000[illegible]	Telf: 932 32 30 30
2	Aresa	41.38961	2.17347	0101000020E6100000[illegible]	Telf: 902 25 62 56
3	Ambulancias Barcelona S.A.	41.411528	2.188499	0101000020E6100000[illegible]	Telf: 932 47 35 35
4	Ambulancias Domingo	41.415237	2.207387	0101000020E6100000[illegible]	Telf: 933 14 44 44
5	Ambulancias Lázaro	41.412038	2.17773	0101000020E6100000[illegible]	Telf: 934 55 10 00
6	Ambulancias Cataluña	41.375516	2.127229	0101000020E6100000[illegible]	Telf: 934 22 88 88
7	Oxigen Salud S.A.	41.41629	2.20817	0101000020E6100000[illegible]	Telf: 934 98 76 82

Figura 4.7. Contingut de la taula 'unit_resposta_a'

Policia:

id [PK]	name text	lat double precis	lon double precis	the_geom geometry	descr text
1	Comissaria Mossos d'esquadra - Les Corts	41.386807	2.130195	0101000020E6100000[illegible]	Telf: 933 00 22 96
2	Comissaria Mossos d'esquadra Sarria - Sant Gervasi	41.4045	2.125309	0101000020E6100000[illegible]	Telf: 933 06 23 10
3	Comissaria Mossos d'esquadra - Gran Via Corts Catalanes	41.37946	2.15531	0101000020E6100000[illegible]	Telf: 934 24 26 88
4	Comissaria Mossos d'esquadra - Ciutat Vella	41.37618	2.17128	0101000020E6100000[illegible]	Telf: 933 06 23 00
5	Comissaria Mossos d'esquadra - Nou Barris	41.44855	2.18196	0101000020E6100000[illegible]	Telf: 932 43 74 00
6	Comissaria Mossos d'esquadra - Sant Andreu	41.44206	2.1908	0101000020E6100000[illegible]	Telf: 932 76 13 26
7	Guarda Urbana - Ciutat Vella	41.378497	2.175449	0101000020E6100000[illegible]	Telf: 932 56 24 30
8	Guarda Urbana - Sants Montjuic	41.37158	2.15404	0101000020E6100000[illegible]	Telf: 932 91 50 08
9	Guarda Urbana - Sarria Sant Gervasi	41.3969003	2.1203205	0101000020E6100000[illegible]	Telf: 932 91 43 53
10	Guarda Urbana - Nou Barris	41.438824	2.170492	0101000020E6100000[illegible]	Telf: 932 91 48 46
11	Guarda Urbana - Gracia	41.417262	2.161174	0101000020E6100000[illegible]	Telf: 932 37 30 44
12	Guarda Urbana - L'Eixample	41.3922198	2.1835607	0101000020E6100000[illegible]	Telf: 932 91 50 92
13	Guarda Urbana - Les Corts	41.385655	2.128558	0101000020E6100000[illegible]	Telf: 932 91 50 92
14	Guarda Urbana - Horta-Guinardo	41.432857	2.151887	0101000020E6100000[illegible]	Telf: 932 74 98 40

Figura 4.8. Contingut de la taula 'unit_resposta_p'

Hospitals:

id [PK]	name text	lat double precis	lon double precis	the_geom geometry	descr text
1	Hospital Universitari Sagrat Cor	41.388994	2.144909	0101000020E6100000[illegible]	Telf: 933 22 11 11
2	Hospital Clinic i Provincial De Barcelona	41.38999	2.153088	0101000020E6100000[illegible]	Telf: 932 27 54 00
3	Hospital Quiron	41.4147	2.138462	0101000020E6100000[illegible]	Telf: 932 55 40 00
4	Hospital Sant Rafael	41.424882	2.141777	0101000020E6100000[illegible]	Telf: 934 28 95 41
5	Hospital General Vall d'Hebrón	41.427401	2.14343	0101000020E6100000[illegible]	Telf: 932 74 61 00
6	Hospital de l'esperança	41.410593	2.15548	0101000020E6100000[illegible]	Telf: 933 67 41 00
7	Hospital Dos De Maig	41.4107	2.177194	0101000020E6100000[illegible]	Telf: 935 07 27 00
8	Hospital De Nens De Barcelona	41.396629	2.17542	0101000020E6100000[illegible]	Telf: 932 31 05 12
9	Hospital Pere Virgili	41.416665	2.140504	0101000020E6100000[illegible]	Telf: 932 59 40 00
10	Hospital Evangelico	41.411112	2.159265	0101000020E6100000[illegible]	Telf: 932 85 99 55

Figura 4.9. Contingut de la taula 'unit_resposta_h'

Bombers:

id [PK]	name text	lat double precis	lon double precis	the_geom geometry	descr text
1	PARC DE BOMBERS EIXAMPLE - CENTRAL	41.388968	2.154023	0101000020E6100000[illegible]	Telf: 932915335
2	PARC DE BOMBERS DE LLEVANT	41.404844	2.190558	0101000020E6100000[illegible]	Telf: 932915335
3	PARC DE BOMBERS DRASSANES	41.376866	2.172403	0101000020E6100000[illegible]	Telf: 932915335
4	PARC DE BOMBERS DE LA VALL D'HEBRON	41.424339	2.143554	0101000020E6100000[illegible]	Telf: 932915335
5	PARC DE BOMBERS DE SANT ANDREU	41.44293	2.186	0101000020E6100000[illegible]	Telf: 932915335

Figura 4.10. Contingut de la taula 'unit_resposta_b'

Durant la creació de les taules corresponents a les diferents unitats de resposta, s'ha buscat documentació específica[33], [34] que servirà pel apartat següent de càlcul de rutes.

4.4 Definició del model de dades

Un cop s'ha vist com s'han inserit les diferents unitats de resposta, es descriurà la informació continguda en les taules creades mitjançant l'eina d'importació vista abans, *osm2pgrouting*. Les taules spatial_ref_sys i geometry_columns, taules pròpies de postGIS, ja s'han descrit amb anterioritat, ara es descriuran la resta. Les altres taules que resten contenen les dades geogràfiques de treball importades desde OpenStreetMaps, corresponents a la ciutat de Barcelona, són:

- *classes* i *types*: Taules que descriuen els tipus de vies de la ciutat i permeten classificar-les segons els seus usos.

- *nodes*: Conté la relació de tots els nodes del mapa, cadascun associat a un identificador, una longitud i una latitud. Venen expressades en graus decimals.

- *vertices_tmp*: conté la relació de tots dels nodes com a element geomètric, amb un identificador únic per a cadascun d'ells i una columna geomètrica amb objectes del tipus punt (POINT).

- *ways*: Taula que conté els carrers i vies del mapa. Cadascun té un identificador únic, la classe de via, el nom, la longitud, les coordenades d'inici i final (expressades com a latitud i longitud) el cost associat (pel càlcul de rutes), una columna geomètrica de tipus polilínia (MULTILINESTRING), i finalment els nodes d'origen i fi identificats pel identificador d'aquests a la taula vertices_tmp.

Aquestes dues darreres taules són les úniques d'interès per a la topologia. En la darrera taula, ways, depenent del algorisme que es faci servir, emprearà uns o altres camps. Així, més endavant pel càlcul optimitzat de rutes, es veurà que:

- L'algorisme de *Dijikstra* farà servir els camps: gid, source, target, cost i reverse_cost.
- L'algorisme A-*star* utilitzarà, a més: x1, y1, x2, y2.
- Per l'algorisme *Shotting Star* s'haurà d'afegir: rule, to_cost.

5. Càlcul de Rutes

5.1 Introducció

Seguidament, es descriurà en detall el càlcul de rutes amb els algorismes esmentats en l'apartat anterior. Inicialment, es farà una descripció general comentant-ne les seves necessitats específiques en quan a les dades i es realitzarà una comparativa entre ells. Finalment, es calcularà una mateixa ruta per a tots ells. En la descripció es farà servir documentació[35,36] existent sobre el tema.

Abans d'entrar a comentar els diferents algorismes, primerament s'haurà de definir què s'entén com a camí més curt. Generalment, es pot definir el camí més òptim basant-se en multitud de criteris. El més senzill, però, és escollir el criteri de la ruta que passa pel menor nombre de nodes, també s'haurà de tenir en compte diversos condicionants o restriccions com poden ser les direccions de circulació dels carrers, els girs en les vies que estiguin prohibits i el no permetre l'accés a determinades zones del municipi.

Recordem que *pgrouting* era una extensió per a bases de dades espacials PostgreSQL/PostGIS que proporcionava un conjunt de funcionalitats per al càlcul de rutes. La forma que té de fer-ho es mitjançant la combinació de les diferents funcions que porta incorporades amb consultes SQL a la base de dades. Es Calcularan les diferents rutes mitjançant tres algorismes:

- **Algorisme de Dijkstra**: Aquest algorisme, també conegut com a algoritme de camins mínims, fou creat al 1959 pel informàtic holandès Edsger Dijkstra. És un algorisme per a la determinació del camí més curt desde un vèrtex origen cap a la resta de vèrtexs en un graf dirigit, simple, connexa i ponderat positivament en les seves arestes.

  El funcionament és el següent: l'algorisme va explorant tots els camins més curts que troba que surten del vèrtex origen i que duen cap als demés vèrtexs. Quan s'obté el camí més curt desde el vèrtex origen cap a la resta de vèrtexs del graf, l'algorisme s'atura.

  Sense fer servir cap cua de prioritat, l'algorisme fa $O(n^2)$ operacions, com a màxim, per tal de determinar la longitud del camí més curt entre dos vèrtex qualsevol.

- **Algorisme A-Star**: Creat al 1968, és una extensió de l'algorisme de Dijkstra. Aconsegueix una gran millora respecte al temps d'execució mitjançant l'ús d'una funció heurística (h(n)) d'avaluació. Aquesta funció serveix com a estimació del cost del camí més econòmic desde un determinat node fins al node objectiu. La cerca sempre escollirà el node que té un valor més baix en la funció heurística.

  L'algorisme, segueix una trajectòria del cost més petit conegut tot mantenint una cua de prioritats amb les alternatives possibles de cada moment. Si en algun punt troba un segment amb un cost menor, abandona la trajectòria de major cost i pren la trajectòria amb cost més petit. El procés continua fins que s'arriba a l'objectiu.

  La complexitat de l'algorisme va en funció de la funció heurística que es fa servir. Per tant, aquesta es pot expressar com a: O(logh*(x)), on 'h' és la funció heurística.

- **Algorisme Shooting-Star**: Fins ara, en els dos algorismes anteriors, es determinava el càlcul de vèrtex a vèrtex. Ara es determinarà la ruta òptima entre dues arestes. La ruta que es calcularà tornarà com a punt de partida el node del inici de la línia. Fins ara calculàvem el cost en funció de la seva llargada sense tenir en compte res més. Fent servir aquest algorisme es poden afegir restriccions a nivell de via (way) que permetrà controlar sentits de circulació, girs prohibits, etc.

5.2 Preparació de les dades

Un cop feta la importació de les dades cap a la nostre base de dades, caldrà que assegurar-nos que també s'hi hagi creat la corresponent topologia. Aquesta consisteix en assegurar l'existència de connexions entre l'origen i el destí de cadascun dels identificadors.

Un cop s'han importat les dades mitjançant osm2pgrouting, serà el mateix programa qui faci aquestes assignacions de forma automàtica. De totes formes, el propi pgrouting disposa de la funció *assign_vertex_id()* amb la que verifica la topologia tot assignant identificadors a cada aresta.

La seva sintaxi és:

```
assign_vertex_id('taula arestes', float tolerancia, 'columna de la geometria', 'gid');
```

Com veiem, en la funció s'hi especificarà la taula de les arestes (ways), la tolerància que hi volem aplicar, la columna geomètrica (the_geom) i la columna que conté l'índex espacial de la taula (gid).

En cas de que, pel que fos, no es disposés de la topologia ja creada, s'hauria d'executar aquesta funció de la següent manera: *assign_vertex_id('ways', 0.00001, 'the_geom', 'gid')*

Aquesta funció assignaria a cada origen i a cada destí una connexió, a més, també té en compte els vèrtexs de la vora amb una certa tolerància. La *tolerància* que hi apliquem dependrà de la projecció de les dades que s'hagi fet, normalment es sol expressar en graus o en metres. Aquesta tolerància farà que assigni el mateix identificador a vèrtex separats per una distància menor que la tolerància que se li expressi. Així, si es disposa d'una bona qualitat de les dades per a una determinada localitat n'hi haurà prou en escollir una tolerància petita (per exemple 0.00001 graus).

Depenent de la quantitat de les dades que haguem importat i per tal d'accelerar-ne les consultes, convindrà que hi afegim *índexs*. Els índexs que es definiran faran referència a les columnes origen i destí i a la columna geomètrica. Aquest darrer índex s'hi afegirà un índex tipus GiST (arbre de cerca generalitzada), que serà un tipus d'indexació especifica per aquest tipus de dades.

GiST (Generalized Search Tree) suposa un clar exemple del d'extensibilitat en el context de les bases de dades. És el tipus d'índex espacial que s'usa a PostGIS. Es tracta d'una generalització dels arbres B+ que permet disposar d'una càrrega balancejada amb independència de les dades que continguin. Els GiST es fan servir per implementar un ampli rang d'índexs com arbres B+, arbres R, etc.

Per a la creació dels tres indexs que faran falta les instruccions serien:

```
CREATE INDEX source_idx ON ways(source);
CREATE INDEX target_idx ON ways(target);
CREATE INDEX geom_idx ON ways USING GIST(the_geom GIST_GEOMETRY_OPS);
```

Un cop creats els índexs i comprovada la topologia, ja es podrà aplicar els diferents algorismes, es començarà per l'algorisme de Dijkstra.

5.3 Algorisme de Dijkstra

L'algorisme de Dijkstra fa una crida a la funció *shortest_path*, passant-li cinc paràmetres:

- Una sentència SQL que ve referida a la taula 'ways' (text).
- El identificador del vèrtex de l'origen (enter).
- El identificador del vèrtex del destí (enter).
- Informar-lo de si el graf és dirigit o no (booleà).
- Si disposa d'un cost reversible o no (booleà).

La seva sintaxi complerta és:

```
shortest_path (sentència_sql, vèrtex_origen, vèrtex_destí, es_dirigit, te_cost_invers);
```

La crida a aquesta funció serà part de la crida que fa una altre sentència SQL de nivell més extern. El conjunt d'aquesta darrera crida retornarà el conjunt de segments correlatius, amb un cost associat, que s'hauran de seguir per anar desde el vèrtex origen fins al vèrtex destí.

Anem a veure un cas pràctic amb tota mena de detall:

En la suposició que hagi ocorregut un accident en un punt del carrer 'Salvador Aulet' de ciutat vella i volem traslladar l'accidentat cap a 'l'hospital Clínic Provincial' en ple eixample.

- El primer que es necessitarà serà, mitjançant les eines informatives de que disposa qGIS, trobar els identificadors dels nodes de l'origen del accident i del destí al que es vol anar, en aquest supòsit seran, respectivament, 4783 i 9576.

- Seguidament, s'informarà a la funció de si el graf és dirigit o no i si disposa de cost invers. Se suposa que en ambdós casos li indiquem 'fals'.

Per tant en aquest cas, la sentència complerta serà:

```
SELECT * FROM shortest_path('SELECT gid as id, source, target, length as cost FROM ways',4783, 9576, false, false);
```

Aquesta sentència retornarà el conjunt de trams que formen el recorregut desde el vèrtex inicial fins al final amb el seu cost associat. Aquest n'és un resum:

En la taula del costat, que conté un resum del resultat, es pot apreciar el node origen i el node destí, respectivament, en primera i en darrera posició.

Per a obtenir més informació, per exemple el cost total resultat, el que es farà serà modificar la sentència SQL anterior i s'hi aplicarà una funció de suma:

vertex_id	edge_id	cost
4783	5538	0.0546201220962796
4782	5537	0.0213651939838433
4781	5536	0.0268780129383792
4780	5535	0.0471724093054659
3004	5534	0.0333791551286295
……		
9770	15900	0.110404595841492
9576	-1	0

Figura 5.1. Resum resultat aplicació Dijkstra

```
SELECT sum(cost) FROM shortest_path('SELECT gid as id, source, target, length as cost FROM ways',4783, 9576, false, false);
```

A l'aplicar aquesta nova funció donarà un resultat de 2.89292472014671.

Per a obtenir més funcionalitat, es farà servir el que es coneix com a *funcions wrapper*. Aquestes funcions el que fan és, mitjançant la crida a una altre funció, una presentació més elaborada dels resultats de les rutines base de càlcul de rutes. Aquestes funcions permeten oferir variacions de l'ordre i dels resultats obtinguts. El propòsit general és el de simplificar la funcionalitat que ofereix pgrouting.

S'utilitzaran aquestes funcions wrapper per tal de mostrar la ruta creada d'una forma més gràfica. D'aquesta manera, s'obtindran les arestes com a elements geomètrics ideals perquè puguin mostrar-se gràficament.

La forma de fer-ho serà mitjançant les funcions *dijkstra_sp*, *dijkstra_sp_delta*. Aquestes funcions utilitzen la funció *shortest_path()* sobre la taula on hi ha les dades (en el nostre cas la taula 'ways') especificant-hi els nodes origen i destí. La seva sintaxi és:

- *dijkstra_sp*: Aquesta és la funció wrapper que s'utilitza per 'defecte'. La funció té tres paràmetres: la taula, el node origen i el node destí.

```
SELECT the_geom FROM dijkstra_sp('ways', 4783, 9576);
```

Com a retorn, s'obtindran un conjunt de resultats amb la geometria:

0105000020E610000001000000010200000002000000 1CF0F961845001401B82E3326EB144405C3F582140510140EFED4C5766B14440
0105000020E610000001000000010200000002000000 B0F3250AE35D014000BF901F3BB14440EBD44B42D85F0140B207A40E3CB14440
0105000020E610000001000000010200000002000000 EBD44B42D85F0140B207A40E3CB14440A9A44E401361014051DF8D603EB14440
……
0105000020E610000001000000010200000002000000 932FB2AE1B500140D14DAC9D72B144403FABCC94D64F01405A7336D373B14440

- *dijkstra_sp_delta*: Aquesta funció és molt més focalitzada ja que limita la cerca en l'àrea limitada pels nodes origen i destí més una distància addicional especificada en el nou paràmetre que rep (bbox_buffer).

```
SELECT the_geom FROM dijkstra_sp_delta('ways', 4783, 9576, 0.1);
```

Aquesta variació està pensada per reduir el temps de resposta. El temps de resposta, consisteix de 2 parts principals (càrrega de dades i càlcul del resultat). La primera part depèn del tamany de les dades. Reduint aquest tamany s'aconsegueix accelerar aquesta part. El increment és la distància entre el punt d'origen i destí i el seu voltant. A aquest paràmetre s'anomena delta.

Cal assignar un valor de delta lo suficientment gran perquè les principals vies del municipi estiguin incloses en les dades a agafar.

Altres variacions de les funcions que hem vist, són: dijkstra_sp_directed, dijkstra_sp_delta_directed.

Un cop aplicades les funcions descrites anteriorment, per tal de visualitzar-ne gràficament el resultat es crearà una nova taula (*dijksta_result*) on s'hi posaran les dades de la consulta.

Expressant-ho amb tot un conjunt de sentències, s'optimitzarà el procés de tal manera que, per a cada càlcul, sols se li haurà de indicar el node origen i el node destí.

```
DROP TABLE IF EXISTS dijkstra_result;
CREATE TABLE dijkstra_result(gid int4) with oids;
SELECT AddGeometryColumn('dijkstra_result', 'the_geom', '4326', 'MULTILINESTRING', 2);
INSERT INTO dijkstra_result(the_geom)
SELECT the_geom FROM dijkstra_sp('ways', 4783, 9576);
```

Com es pot veure, en el procés primer es crearà la taula i la columna geomètrica per emmagatzemar les vies del resultat, seguidament se li indicarà el sistema de referència i s'omplirà la columna geomètrica amb el resultat que s'obtingui.

Cal observar que, al afegir-hi la geometria, se li indicarà el sistema de referència que es farà servir. Aquí s'utilitzarà el Sistema Geodèsic Mundial del 1984 (WGS84). D'aquí que el paràmetre referent a el identificador del sistema de referència espacial prengui com a valor 4326.

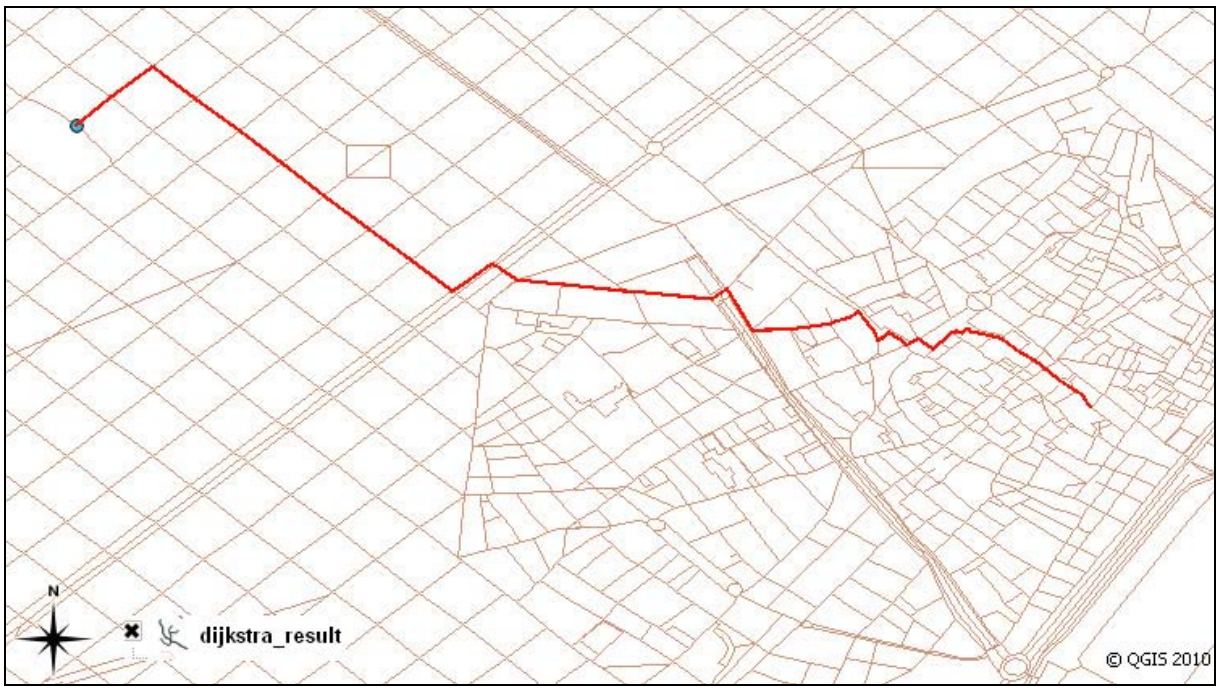

Figura 5.2. Ruta més curta entre els nodes 4783 i 9576 aplicant Dijkstra

5.4 Algorisme A*

L'algorisme A* crida a la funció *shortest_path_astar*, passant-li cinc paràmetres:

- Una sentència SQL que ve referida a la taula 'ways' (text).
- El identificador del vèrtex de l'origen (enter).
- El identificador del vèrtex del destí (enter).
- Informar-lo de si el graf és dirigit o no (booleà).
- Si disposa d'un cost reversible o no (booleà).

La seva sintaxi complerta és:

```
shortest_path_astar (sentència_sql, vèrtex_origen, vèrtex_destí, es_dirigit, te_cost_invers);
```

Com en la funció shortest_path, la crida a aquesta nova funció serà alhora part de la crida que farà a una altre sentència SQL de nivell més extern.

Considerant els mateixos nodes que en el cas anterior: 4783 i 9576, la sentencia complerta serà:

```
SELECT * FROM shortest_path_astar ('SELECT gid as id, source, target, length as cost, x1, y1, x2, y2 FROM ways',4783, 9576, false, false);
```

Aquesta sentència, com en el cas anterior, també retornarà el conjunt de trams que formen el recorregut desde el vèrtex inicial fins al final amb el seu cost associat. Aquest n'és un resum:

A la taula del costat, que conté un resum del resultat, es pot apreciar el node origen i el node destí, respectivament, en primera i en darrera posició.

Per a obtenir més informació, per exemple el cost total resultat, es modificarà la sentència SQL anterior i s'hi aplicarà una funció de suma:

vertex_id	edge_id	cost
4783	5538	0.0546201220962796
4782	5537	0.0213651939838433
4781	5536	0.0268780129383792
4780	5535	0.0471724093054659
3004	5534	0.0333791551286295
……		
9770	15900	0.110404595841492
9576	-1	0

Figura 5.3. Resum resultat aplicació A*

```
SELECT sum(cost) FROM shortest_path_astar('SELECT gid as id, source, target, length as cost, x1, y1, x2, y2 FROM ways',4783, 9576, false, false);
```

Com es pot observar en la sintaxi de la funció, es necessitaran columnes per a la longitud i la latitud (x1, y1, x2, y2) i s'hauran d'emplenar amb els corresponents valors.

Per posar-hi els valors que corresponen caldrà completar-les, si és el cas, calculant-ne els seus corresponents valors. Es farà mitjançant les instruccions:

```
UPDATE ways SET x1 = x (PointN(the_geom, 1));
UPDATE ways SET y1 = y (PointN(the_geom, 1));
UPDATE ways SET x2 = x (PointN(the_geom, NumPoints(the_geom)));
UPDATE ways SET x1 = x (PointN(the_geom, 1));
UPDATE ways SET y2 = y (PointN(the_geom, NumPoints(the_geom)));
```

Mantenint la funció heurística constant (en cas contrari s'hauria de tornar a recompilar pgRouting) aquesta nova funció donarà un resultat de 2.89292472014671. Que és el mateix valor que l'obtingut per l'algorisme anterior.

Seguidament, s'aplicarà la corresponent funció wrapper per l'algorisme A*. Cal comentar que per aquest algorisme, a diferència de Dijkstra, sols hi ha la funció wrapper disponible per la funció delta.

```
SELECT the_geom FROM astar_sp_delta('ways', 4783, 9576, 0.1);
```

Com a retorn de la crida de la funció s'obtindrà un conjunt de resultats amb la geometria:

0105000020E6100000010000000102000000020000001CF0F961845001401B82E3326EB144405C3F582140510140EFED4C5766B14440
0105000020E6100000010000000102000000020000000B0F3250AE35D014000BF901F3BB14440EBD44B42D85F0140B207A40E3CB14440
0105000020E6100000010000000102000000020000000EBD44B42D85F0140B207A40E3CB14440A9A44E401361014051DF8D603EB14440
……
0105000020E6100000010000000102000000020000000932FB2AE1B500140D14DAC9D72B144403FABCC94D64F01405A7336D373B14440

De forma anàloga a l'algorisme de dijkstra, un cop aplicades les funcions descrites anteriorment, per tal de visualitzar gràficament el resultat, es crearà una nova taula (*astar_result*) on s'hi posaran les dades de la consulta.

S'optimitzarà el procés de tal manera que, sols s'haurà de canviar els nodes origen i destí i visualitzar-ne el resultat.

```
DROP TABLE IF EXISTS astar_result;
CREATE TABLE astar_result(gid int4) with oids;
SELECT AddGeometryColumn('astar_result', 'the_geom', '4326', 'MULTILINESTRING', 2);
INSERT INTO astar_result(the_geom)
SELECT the_geom FROM astar_sp_delta('ways', 4783, 9576, 0.1);
```

En el procés primerament es crearà la taula i la columna geomètrica per emmagatzemar les vies del resultat, seguidament se li indicarà el sistema de referència i s'omplirà la columna geomètrica amb el resultat que s'obtingui.

Aquí també s'indicarà com a sistema de referència el Sistema Geodèsic Mundial del 1984 (WGS84).

El resultat que es pot apreciar és exactament el mateix que el calculat en l'apartat anterior:

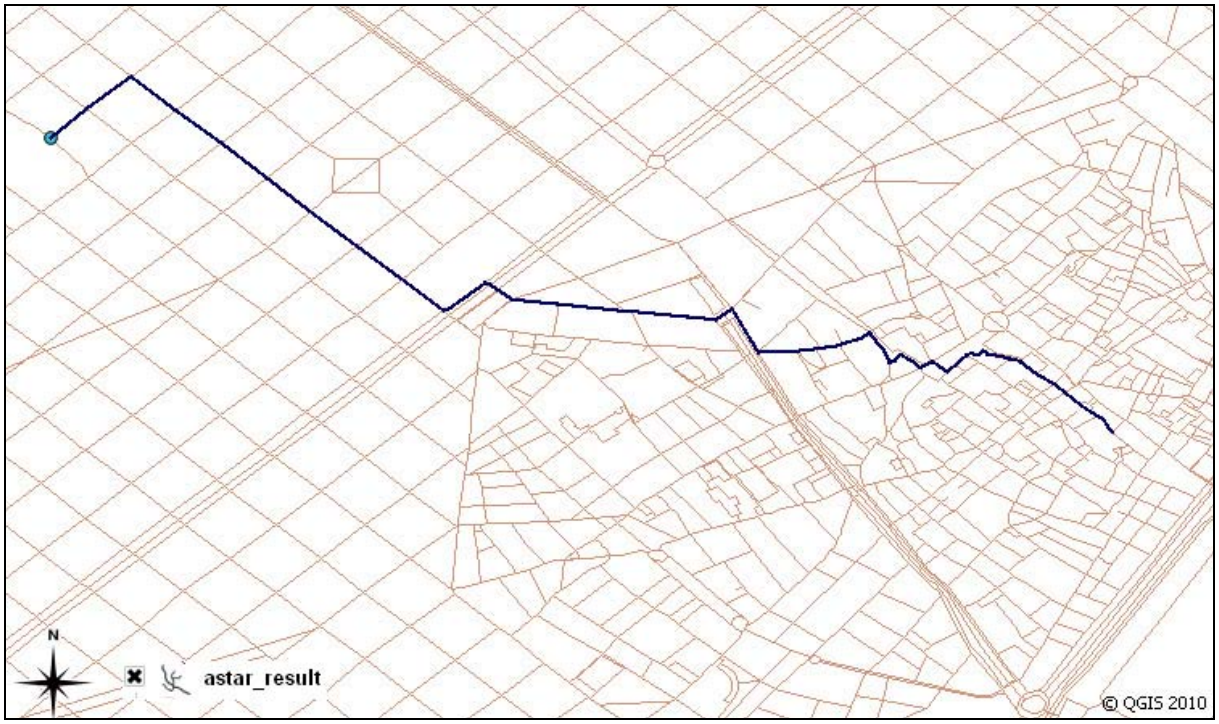

Figura 5.4. Ruta més curta entre els nodes 4783 i 9576 aplicant A*

5.5 Algorisme Shooting-Star

L'algorisme Shooting-Star crida a la funció *shortest_path_shooting_star*, passant-li cinc paràmetres:

- Una sentència SQL que ve referida a la taula 'ways' (text).
- El identificador del vèrtex de l'origen (enter).
- El identificador del vèrtex del destí (enter).
- Informar-lo de si el graf és dirigit o no (booleà).
- Si disposa d'un cost reversible o no (booleà).

La seva sintaxi és:

```
shortest_path_shooting_star (sentència_sql, vèrtex_origen, vèrtex_destí, es_dirigit, te_cost_invers);
```

Com passa amb shortest_path i shortest_path_astar, la crida a aquesta funció serà alhora part de la crida que farà a una altre sentència SQL de nivell més extern.

Ara, a diferència d'abans, l'algorisme no calcularà del vèrtex origen al vèrtex destí sinó del cantó origen al cantó destí (edge). La columna vertex_id contindrà el vèrtex d'inici d'un cantó per la columna edge_id.

Per tant, si ara considerant els vèrtexs d'origen i destí: 5538, 15900, la sentencia complerta serà:

```
SELECT * FROM shortest_path_shooting_star ('SELECT gid as id, source, target, length as cost, x1, y1, x2, y2, rule, to_cost FROM ways',5538, 15900, false, false);
```

Aquesta sentència, com en els casos anteriors, també retornarà el conjunt de trams que formen el recorregut desde el cantó inicial fins al final amb el seu cost associat. Aquest n'és un resum:

A la taula del costat, que conté un resum del resultat, es pot apreciar el cantó origen i el cantó destí, respectivament, en primera i en darrera posició.

Per a obtenir més informació, per exemple el cost total resultat, el que es farà serà modificar la sentència SQL anterior i s'hi aplicarà una funció de suma:

vertex_id	edge_id	cost
7924	**5538**	0.0546201220962796
7926	8194	0.0436542881904481
11252	11668	0.0783080282246671
15346	11669	0.00809624716133584
15348	12670	0.0357561963138756
……		
9916	15901	0.129191427796658
19391	**15900**	0.110404595841492

Figura 5.5. Resum resultat aplicació Shooting-Star

```
SELECT sum(cost) FROM shortest_path_shooting_star ('SELECT gid as id, source, target, length as cost, x1, y1, x2, y2, rule, to_cost FROM ways',5538, 15900, false, false);
```

Aquesta nova funció donarà un resultat de 3.01049627311481. Aquest cost serà superior al obtingut pels dos mètodes anteriors.

També, de forma anàloga, s'aplicarà la funció wrapper per l'algorisme Shooting-Star:

```
SELECT the_geom FROM shootingstar_sp('ways', 5538, 15900, 0.1, 'length', true, true);
```

Com a retorn de la crida de la funció s'obtindrà un conjunt de resultats amb la geometria:

0105000020E6100000010000000102000000020000001CF0F961845001401B82E3326EB144405C3F582140510140EFED4C5766B14440
0105000020E6100000010000000102000000020000000B0F3250AE35D014000BF901F3BB14440EBD44B42D85F0140B207A40E3CB14440
0105000020E6100000010000000102000000020000000EBD44B42D85F0140B207A40E3CB14440A9A44E401361014051DF8D603EB14440
……
0105000020E6100000010000000102000000020000000932FB2AE1B500140D14DAC9D72B144403FABCC94D64F01405A7336D373B14440

De forma anàloga als dos algorismes anteriors, un cop aplicades les funcions descrites anteriorment, per tal de visualitzar gràficament el resultat, es crearà una nova taula (*shooting_star_result*) on s'hi posaran les dades de la consulta.

Com abans, es farà un procés de tal manera que, sols s'hagi de canviar els cantons origen i destí.

```
DROP TABLE IF EXISTS shooting_star_result;
CREATE TABLE shooting_star_result(gid int4) with oids;
SELECT AddGeometryColumn('shooting_star_result', 'the_geom', '4326', 'MULTILINESTRING',
2);
INSERT INTO shooting_star_result(the_geom)
SELECT the_geom FROM shootingstar_sp('ways', 5538, 15900, 0.1, 'length', true, true);
```

En el procés primerament es crearà la taula i la columna geomètrica per emmagatzemar-hi les vies del resultat, seguidament se li indicarà el sistema de referència i s'omplirà la columna geomètrica amb el resultat que s'obtingui.

Aquí també se li indicarà com a sistema de referència el Sistema Geodèsic Mundial del 1984 (WGS84).

El resultat que es pot apreciar és exactament el mateix que el calculat en l'apartat anterior:

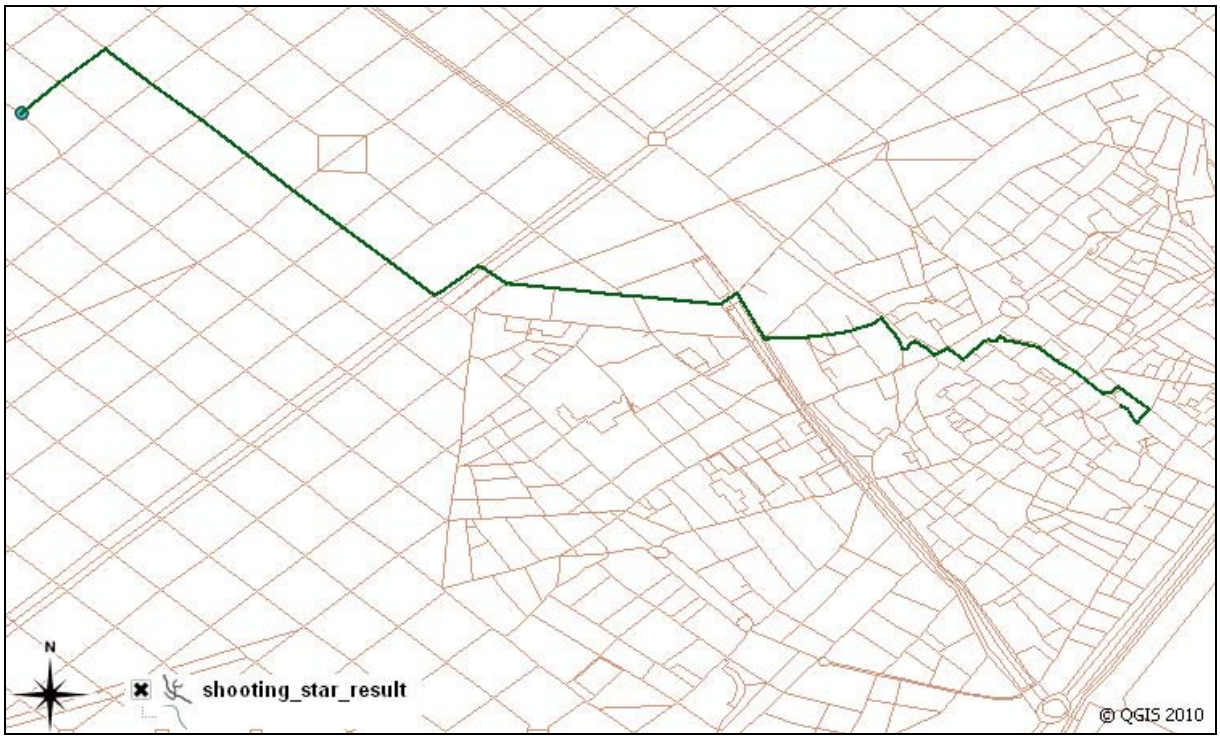

Figura 5.6. Ruta més curta entre els vèrtexs 5538, 15900 aplicant Shooting-Star

5.6 Comparativa entre algorismes

Una vegada s'ha vist en detall cada algorisme per separat, es farà una comparativa de la seva eficiència.

Els dos primers algorismes, el de Dijkstra i el A*, fan servir els vèrtexs per fer les cerques i retornen un resultat semblant. El primer algorisme, el de Dijkstra, s'ha vist que retorna resultats òptims i fiables i l'algorisme de A*, si fem servir un quadre delimitador amb una delta lo suficientment grossa, també retorna un resultat igualment eficient.

En canvi, el darrer algorisme, el Shooting-Star, fa servir les arestes de la topologia per fer els càlculs pel que canvia totalment el mètode en que fa la cerca. Aquest mètode retorna un cost més elevat que els dos anteriors.

En successives proves s'ha pogut apreciar una superior eficiència del algorisme A* respecte el de Dijkstra; també, conceptualment, al suposar A* una millora respecte al de Dijkstra es pot concloure que serà molt més eficient en la majoria dels casos. D'aquí que es faci servir l'algorisme A* sempre que es pugui.

6. Entorn web amb OpenLayers

En aquest capítol es descriurà com integrar tots els components vistos fins el moment. La integració es farà mitjançant la llibreria OpenLayers on es presentarà la imatge, de forma dinàmica, del municipi en entorn web.

6.1. OpenLayers

OpenLayers és una llibreria de JavaScript de codi obert que permetrà mostrar mapes interactius en els navegadors web. A més, OpenLayers ofereix una API per accedir a diferents fonts d'informació cartogràfica en la xarxa: Web Map Services (WMS), Web Features Services (WFS), etc.

El funcionament de OpenLayers es basa en el contingut d'un objecte *map* que guarda tota la informació utilitzada: la projecció que es farà servir, el format en que es mostrarà per pantalla, les dimensions del mapa, les ampliacions, les etiquetes de llegenda, etc.

La informació es troba dividida per capes (layers) i, per a cadascuna d'elles, s'especifica l'origen i com s'ha de llegir aquesta informació. Els tipus de capes que suporta OpenLayers és molt variat. En aquest projecte s'han fet servir capes tipus WMS (per a la connexió amb MapServer) i capes tipus OSM (per a la connexió amb OpenStreetMaps).

Es en el fitxer HTML de la pàgina principal on es col·loca l'script corresponent que farà la crida a l'objecte map amb les corresponents capes. De forma dinàmica, OpenLayers permetrà la gestió d'aquestes capes depenent de les necessitats del moment.

6.1.1 Creació de l'Objecte MAP

Per a la creació de l'objecte MAP es farà servir com ajuda el client pesat gvSIG i de la seva extensió de publicació[37].

La versió del client que es farà servir serà la 1.1.1

El motiu de fer servir una de les versions anteriors a l'actual ha estat el fet de poder garantir la compatibilitat entre el programari i la seva extensió de publicació (figura 6.1).

Figura 6.1. Extensió publicació

El procés de publicació generarà una primera versió del fitxer MAP que necessitarem. Aquesta versió es faran les modificacions que facin falta per tal d'adaptar-la a les nostres necessitats.

Tot seguit, es descriurà el procés que s'ha seguit per a la generació del fitxer MAP a través de gvSIG:

1) Inicialment es començarà *creant una vista* amb les dades que es volen acabar publicant.

Per a la creació de la vista, caldrà especificar el sistema de referència que es farà servir i, una vegada oberta, s'aniran carregant les capes que es necessitin.

En cada capa que s'especifiqui es triarà entre les diferents taules de la BD que es necessitarà i s'hi aplicarà la projecció definida al començament (Figures 6.2 i 6.3).

Una vegada carregada la informació, es podrà modificar l'ordre i les propietats de les capes.

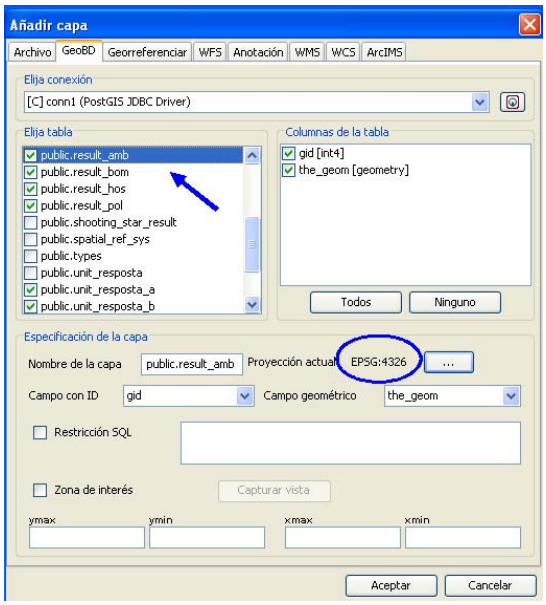

Figura 6.2 Tria de capes

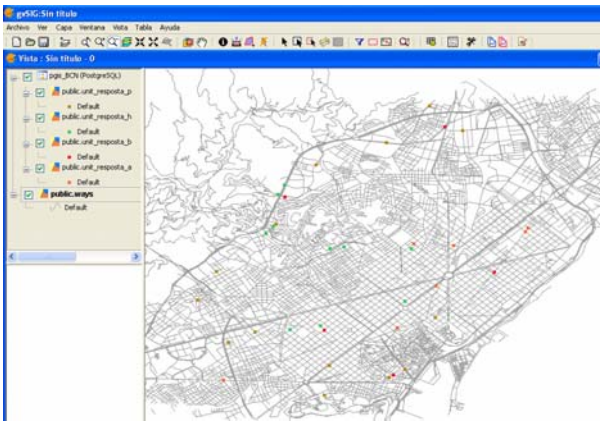
Figura 6.3 Vista integrada de capes

Una vegada creada la vista amb les dades que es desitgen incorporar en el fitxer MAP, es procedirà a la creació del propi fitxer.

2) Per a la creació del fitxer MAP es triarà l'opció de *Publicar* (Figura 6.1).

En l'opció de publicar es procedirà d'una forma molt semblant a la creació d'una nova vista. Quan es creï la publicació i s'obri, un assistent serà l'encarregat de guiar a l'usuari pas a pas a través del procés.

2.1) Inicialment s'especificarà la URL del servidor, el servidor i el servei (figura 6.4).

2.2) Seguidament, es definirà el fitxer de configuració. Es tracta d'on anirà ubicat el fitxer MAP (carpeta htdocs) (figura 6.5).

Si s'activen les opcions avançades, es podrà activar la depuració, indicar on es volen guardar els fitxers temporals que es generin. Finalment, s'especifica el directori de les dades.

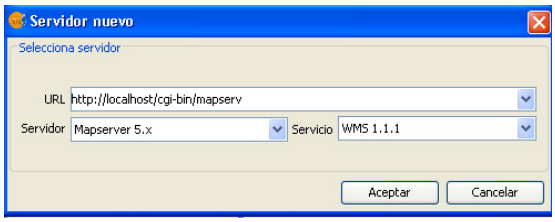

Figura 6.4 - paràmetres servidor

Cal notar que les opcions avançades és un paràmetre opcional. En cas de no indicar-se, els fitxers temporals es situaran en el mateix lloc on el directori a on es genera el mapfile.

Seguidament s'especificarà el títol que se li voldrà donar al servei, una descripció d'ell i l'adreça URL d'aquest (figura 6.6).

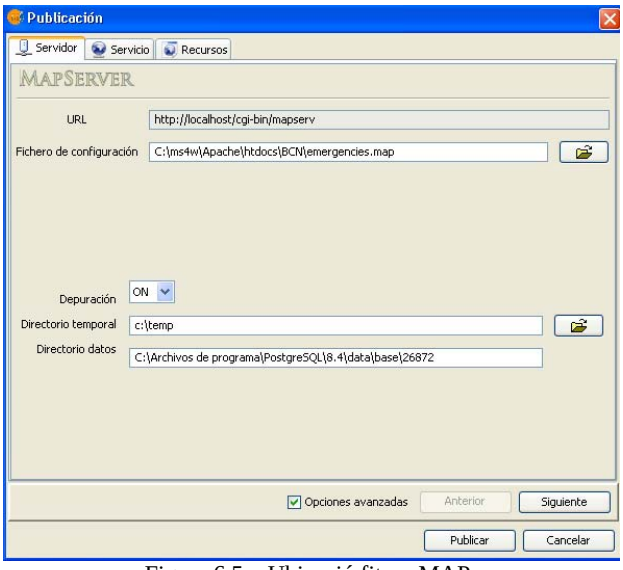

Figura 6.5 – Ubicació fitxer MAP

Finalment, s'afegiran els recursos que es volen publicar (figura 6.7). En aquest cas, s'afegiran les capes que es publicaran.

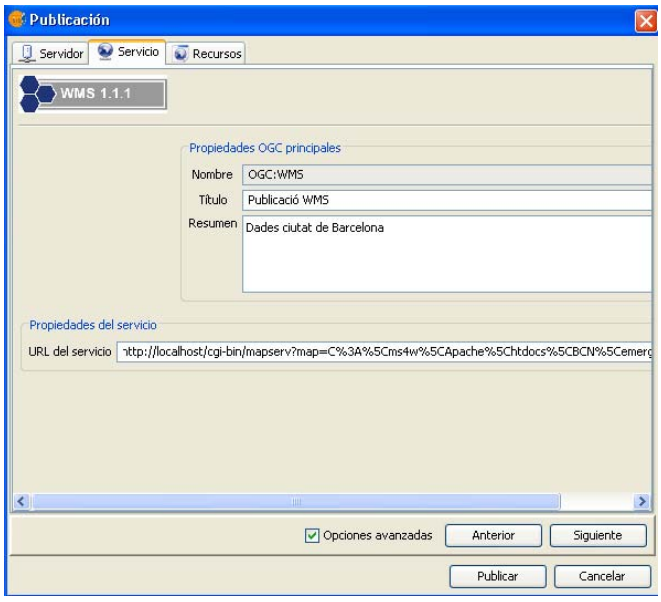

Figura 6.6 – paràmetres servei WMS

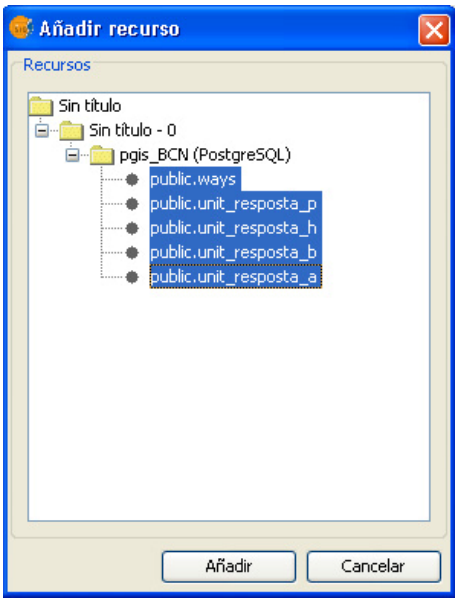

Figura 6.7 – Tria de recursos

En aquesta darrera etapa, les opcions avançades permetran habilitar les capes per tal de que aquestes siguin consultables, permetent l'operació *GetfeatureInfo*. Aquesta opció no afectarà ja que sols serà activable amb capes provinents de shapefiles.

En el punt final, en cas de no produir-se cap error, es mostrarà un missatge informatiu informant de la correcta publicació.

6.1.2 Modificacions dels fitxers de publicació

Durant el procés de publicació, apart del fitxer MAP, es generaran tres fitxers addicionals: el fitxer de símbols (.sym), el fitxer de fonts (fonts.txt) i un fitxer de font (Vera.ttf). En aquests fitxers s'hauran de modificar lleugerament:

- Fitxer de Símbols (.sym): Es tindrà que afegir la instrucció SYMBOLSET al inici del fitxer i la instrucció END al final.
- Fitxer MAP (.MAP): És un fitxer que està composat de varis objectes, dins dels quals es defineixen varis paràmetres. S'hauran de fer modificacions en cadascun d'aquests objectes. Seguidament es veuran per separat:
  - *Objecte WEB*: Per tal de poder-se realitzar l'operació GetFeatureInfo caldrà definir els paràmetres:

```
HEADER templates/header.html
TEMPLATE "MapName.html"
FOOTER templates/footer.html
```

A dins de l'objecte WEB hi ha l'objecte METADATA que caldrà modificar i afegir nous paràmetres:

- Primerament, caldrà modificar el paràmetre *wms_onlineresource*, canviant els símbols que s'han generat per defecte al crear el fitxer MAP i afegir-hi els paràmetres *service* i *version*. El seu aspecte final serà:

```
"wms_onlineresource" "http://localhost:9000/cgi-
bin/mapserv?service=WMS&version=1.1.0&map=/ms4w/Apache/htdocs/BCN/emergencies.map"
```

- També caldrà afegir els nous paràmetres:

```
"wms_service" "WMS"
"wms_formats" "png gif jpeg"
"wms_feature_info_mime_type" "text/html"
```

  - *Objecte LAYER*: De forma anàloga a l'objecte WEB, per a l'objecte LAYER caldrà definir els paràmetres:

```
HEADER "templates/LayerName_header.html"
TEMPLATE "LayerName.html"
FOOTER "templates/LayerName_footer.html"
```

A dins de l'objecte LAYER, caldrà fer les modificacions:

- En la propietat METADATA de l'objecte LAYER caldrà afegir-hi:

```
"wms_include_items"  "all"
```

- En cas de voler fer una reprojecció posterior de les dades[38] caldrà modificar la referència al camp de la geometria tot canviant-li el seu nom ja que pot donar error. Deixant-la com:

```
DATA "geom from public.unit_resposta_a using unique id using srid=4326"
```

Aquestes seran totes les modificacions que caldrà fer en l'objecte MAP.

6.1.3 Proves de funcionament

Un cop fetes les modificacions en els arxius que s'han generat en la publicació, es faran dues proves de funcionament per tal de comprovar que tot funciona de forma correcta. Aquestes proves són crides a dues funcions WMS. Aquestes crides permetran comprovar el correcte funcionament del servei WMS publicat. Aquestes comprovacions es faran a través d'un client lleuger (un navegador web).

1) getCapabilities

És una operació obligatòria per comprovar la correctesa del procés es fa una crida a aquesta funció del servei.

Per a la crida serà necessari especificar una sèrie de paràmetres obligatoris (VERSION, REQUEST, SERVICE). En el nostre cas, la crida serà tal i com segueix:

```
http://localhost:9000/cgi-bin/mapserv.exe?service=WMS&request=getCapabilities&map=/ms4w/apache/htdocs/BCN/emergencies.map&version=1.0.0
```

La resposta serà un document en format XML amb informació sobre les capes i les seves característiques i altres metadates com els sistemes de referència que suporta i l'àmbit geogràfic. Les metadates que descriuen el contingut tenen que descriure tant el joc de dades com cadascuna de les capes per separat.

2) getMap

És una operació obligatòria, es tracta d'un servei que genera una imatge estàtica d'un mapa.

Aquesta operació serà necessari definir una URL on es demanarà la sortida d'un mapa definit com una imatge en format digital. L'operació està composada de paràmetres obligatoris (VERSION, REQUEST, LAYERS, STYLES, SRS, BBOX, WIDTH, HEIGHT, FORMAT) i altres d'opcionals (BGCOLOR, TRANSPARENT, SLD, EXCEPTIONS, etc).

En el nostre cas, la crida serà tal i com segueix:

```
http://localhost:9000/cgi-bin/mapserv.exe?map=C%3A%5Cms4w%5CApache%5Chtdocs%5CBCN%5Cemergencies.map&Service=WMS&VERSION=1.1.0&request=getMap&Layers=public.ways&SRS=EPSG:4326&BBOX=33.5508,28.8462,39.4306,35.1647&WIDTH=2048&HEIGHT=2048&FORMAT=image/png
```

3) getFeaturesInfo

Es tracta d'una operació opcional que retorna informació sobre entitats i objectes mostrats en el mapa. Respon a consultes bàsiques sobre el contingut del mapa com podrien ser determinats atributs sobre una determinada capa. Al ser una operació opcional, sols es comentaran les principals característiques.

Al ser opcional, sols es comentaran les principals característiques. Cal dir que en aquesta operació, en la que també serà necessari definir una URL, hi ha un conjunt de paràmetres obligatoris com VERSION, REQUEST i QUERY_LAYERS.

En la seva execució, si no ha ocorregut cap error, la resposta que s'obtindrà anirà en funció amb la consulta realitzada.

6.1.4 Contingut del fitxer MAP

El fitxer MAP que s'ha utilitzat perquè MapServer faci de servidor WMS i subministri dades de la base de dades PostGIS és el següent:

```
MAP
  NAME map_generated_by_gvsig
  EXTENT 2.0742169 41.3576806 2.2446549 41.4705901
  SHAPEPATH "C:\Archivos de programa\PostgreSQL\8.4\data\base\26872"
  DEBUG ON
  SYMBOLSET "emergencies.sym"
  FONTSET "fonts.txt"
  LEGEND
    IMAGECOLOR -1 -1 -1
    LABEL
      FONT "vera"
      ANGLE FOLLOW
      COLOR 0 0 0
      ENCODING "UTF-8"
      TYPE truetype
      SIZE 8
    END
    STATUS ON
    TRANSPARENT ON
  END
  WEB
    HEADER templates/header.html
    TEMPLATE "MapName.html"
    FOOTER templates/footer.html
    IMAGEPATH "c:\temp"
    METADATA
      # "wms_encoding" "UTF-8"
        "wms_version" "1.1.0"
        "wms_formats" "png gif jpeg"
        "wms_title" "Mapserver WMS"
        "wms_abstract" ""
        "wms_srs" " EPSG:4326"
        "wms_service" "WMS"
        "wms_feature_info_mime_type" "text/html"
       "wms_onlineresource"
       "http://localhost:9000/cgi-bin/mapserv?service=WMS&version=1.1.0&map=/ms4w/Apache/htdocs/BCN/emergencies.map"
    END
  END

  PROJECTION
    "init=epsg:4326"
  END
  LAYER
    HEADER "templates/LayerName_header.html"
    TEMPLATE "LayerName.html"
    FOOTER "templates/LayerName_footer.html"
    NAME "public.unit_resposta_p"
    STATUS ON
    DEBUG ON
    TYPE POINT
    DATA "geom from public.unit_resposta_p using unique id using srid=4326"
    CONNECTIONTYPE POSTGIS
    CONNECTION "user=postgres password=1234 host=localhost port=5432 dbname=pgis_BCN"
    MAXSCALE -1.0
    MINSCALE -1.0
    TRANSPARENCY 100
    DUMP TRUE
    TEMPLATE "."
    SIZEUNITS pixels
    PROJECTION
      "init=epsg:4326"
    END
    CLASS
      STYLE
        COLOR 14 152 13
        SIZE 5
      END
      SYMBOL "square"
      NAME "default"
    END
```

Figura 6.8. Fitxer MAP

```
    METADATA
      "wms_title" "public.unit_resposta_p"
      "wms_include_items" "all"
      "wms_abstract" "generated by gvSIG"
      "wms_extent" "2.12032090000127 41.3715777018209 2.19080391572109 41.4485544000553"
      "wms_layer_group" "/pgis_BCN (PostgreSQL)"
      "gml_include_items" "all"
    END
  END # Layer
  LAYER
    HEADER "templates/LayerName_header.html"
    TEMPLATE "LayerName.html"
    FOOTER "templates/LayerName_footer.html"
    NAME "public.unit_resposta_a"
    STATUS ON
    DEBUG ON
    TYPE POINT
    DATA "geom from public.unit_resposta_a using unique id using srid=4326"
    CONNECTIONTYPE POSTGIS
    CONNECTION "user=postgres password=1234 host=localhost port=5432 dbname=pgis_BCN"
    MAXSCALE -1.0
    MINSCALE -1.0
    TRANSPARENCY 100
    DUMP TRUE
    TEMPLATE "."
    SIZEUNITS pixels
    PROJECTION
      "init=epsg:4326"
    END
    CLASS
      STYLE
        COLOR 31 169 81
        SIZE 5
      END
      SYMBOL "square"
      NAME "default"
    END
    METADATA
      "wms_title" "public.unit_resposta_a"
      "wms_include_items" "all"
      "wms_abstract" "generated by gvSIG"
      "wms_extent" "2.12722689999626 41.3755150000003 2.20817819997248 41.4162682000827"
      "wms_layer_group" "/pgis_BCN (PostgreSQL)"
      "gml_include_items" "all"
    END
  END # Layer
  LAYER
    HEADER "templates/LayerName_header.html"
    TEMPLATE "LayerName.html"
    FOOTER "templates/LayerName_footer.html"
    NAME "public.unit_resposta_b"
    STATUS ON
    DEBUG ON
    TYPE POINT
    DATA "geom from public.unit_resposta_b using unique id using srid=4326"
    CONNECTIONTYPE POSTGIS
    CONNECTION "user=postgres password=1234 host=localhost port=5432 dbname=pgis_BCN"
    MAXSCALE -1.0
    MINSCALE -1.0
    TRANSPARENCY 100
    DUMP TRUE
    TEMPLATE "."
    SIZEUNITS pixels
    PROJECTION
      "init=epsg:4326"
    END
    CLASS
      STYLE
        COLOR 16 181 84
        SIZE 5
      END
      SYMBOL "square"
      NAME "default"
    END
    METADATA
      "wms_title" "public.unit_resposta_b"
                  "wms_include_items" "all"
      "wms_abstract" "generated by gvSIG"
      "wms_extent" "2.14355519999993 41.37688706544 2.19911960000384 41.4429312998981"
      "wms_layer_group" "/pgis_BCN (PostgreSQL)"
      "gml_include_items" "all"
    END
  END # Layer
```

Figura 6.8. Fitxer MAP

```
  LAYER
    HEADER "templates/LayerName_header.html"
    TEMPLATE "LayerName.html"
    FOOTER "templates/LayerName_footer.html"
    NAME "public.unit_resposta_h"
    STATUS ON
    DEBUG ON
    TYPE POINT
    DATA "geom from public.unit_resposta_h using unique id using srid=4326"
    CONNECTIONTYPE POSTGIS
    CONNECTION "user=postgres password=1234 host=localhost port=5432 dbname=pgis_BCN"
    MAXSCALE -1.0
    MINSCALE -1.0
    TRANSPARENCY 100
    DUMP TRUE
    TEMPLATE "."
    SIZEUNITS pixels
    PROJECTION
      "init=epsg:4326"
    END
    CLASS
      STYLE
        COLOR 215 200 25
        SIZE 5
      END
      SYMBOL "square"
      NAME "default"
    END
    METADATA
      "wms_title" "public.unit_resposta_h"
      "wms_include_items" "all"
      "wms_abstract" "generated by gvSIG"
      "wms_extent" "2.13846155853495 41.3889943449329 2.1771948999261 41.4274814985103"
      "wms_layer_group" "/pgis_BCN (PostgreSQL)"
      "gml_include_items" "all"
    END
  END # Layer
  LAYER
    HEADER "templates/LayerName_header.html"
    TEMPLATE "LayerName.html"
    FOOTER "templates/LayerName_footer.html"
    NAME "public.ways"
    STATUS ON
    DEBUG ON
    TYPE LINE
    DATA "geom from public.ways using unique gid using srid=4326"
    CONNECTIONTYPE POSTGIS
    CONNECTION "user=postgres password=1234 host=localhost port=5432 dbname=pgis_BCN"
    MAXSCALE -1.0
    MINSCALE -1.0
    TRANSPARENCY 100
    DUMP TRUE
    TEMPLATE "."
    SIZEUNITS pixels
    PROJECTION
      "init=epsg:4326"
    END
    CLASS
      STYLE
        COLOR 136 124 6
        WIDTH 1
      END
      NAME "default"
    END
    METADATA
      "wms_title" "public.ways"
      "wms_include_items" "all"
      "wms_abstract" "generated by gvSIG"
      "wms_extent" "2.0742169 41.3576806 2.2446549 41.4705901"
      "wms_layer_group" "/pgis_BCN (PostgreSQL)"
      "gml_include_items" "all"
    END
  END # Layer
```

Figura 6.8. Fitxer MAP

```
LAYER
    HEADER "templates/LayerName_header.html"
    TEMPLATE "LayerName.html"
    FOOTER "templates/LayerName_footer.html"
    NAME "public.result_amb"
    STATUS ON
    DEBUG ON
    TYPE LINE
    DATA "geom from public.result_amb using unique gid using srid=4326"
    CONNECTIONTYPE POSTGIS
    CONNECTION "user=postgres password=1234 host=localhost port=5432 dbname=pgis_BCN"
    MAXSCALE -1.0
    MINSCALE -1.0
    TRANSPARENCY 100
    DUMP TRUE
    TEMPLATE "."
    SIZEUNITS pixels
    PROJECTION
      "init=epsg:4326"
    END
    CLASS
      STYLE
        COLOR 36 224 6
        WIDTH 1
      END
      NAME "default"
    END
    METADATA
      "wms_title" "public.result_amb"
      "wms_include_items" "all"
      "wms_abstract" "generated by gvSIG"
      "wms_extent" "2.0742169 41.3576806 2.2446549 41.4705901"
      "wms_layer_group" "/pgis_BCN (PostgreSQL)"
      "gml_include_items" "all"
    END
  END # Layer
  LAYER
    HEADER "templates/LayerName_header.html"
    TEMPLATE "LayerName.html"
    FOOTER "templates/LayerName_footer.html"
    NAME "public.result_bom"
    STATUS ON
    DEBUG ON
    TYPE LINE
    DATA "geom from public.result_bom using unique gid using srid=4326"
    CONNECTIONTYPE POSTGIS
    CONNECTION "user=postgres password=1234 host=localhost port=5432 dbname=pgis_BCN"
    MAXSCALE -1.0
    MINSCALE -1.0
    TRANSPARENCY 100
    DUMP TRUE
    TEMPLATE "."
    SIZEUNITS pixels
    PROJECTION
      "init=epsg:4326"
    END
    CLASS
      STYLE
        COLOR 136 204 63
        WIDTH 1
      END
      NAME "default"
    END
    METADATA
      "wms_title" "public.result_bom"
      "wms_include_items" "all"
      "wms_abstract" "generated by gvSIG"
      "wms_extent" "2.0742169 41.3576806 2.2446549 41.4705901"
      "wms_layer_group" "/pgis_BCN (PostgreSQL)"
      "gml_include_items" "all"
    END
  END # Layer
```

Figura 6.8. Fitxer MAP

```
LAYER
  HEADER "templates/LayerName_header.html"
  TEMPLATE "LayerName.html"
  FOOTER "templates/LayerName_footer.html"
  NAME "public.result_hos"
  STATUS ON
  DEBUG ON
  TYPE LINE
  DATA "geom from public.result_hos using unique gid using srid=4326"
  CONNECTIONTYPE POSTGIS
  CONNECTION "user=postgres password=1234 host=localhost port=5432 dbname=pgis_BCN"
  MAXSCALE -1.0
  MINSCALE -1.0
  TRANSPARENCY 100
  DUMP TRUE
  TEMPLATE "."
  SIZEUNITS pixels
  PROJECTION
    "init=epsg:4326"
  END
  CLASS
    STYLE
      COLOR 236 24 126
      WIDTH 1
    END
    NAME "default"
  END
  METADATA
    "wms_title" "public.result_hos"
   "wms_include_items" "all"
    "wms_abstract" "generated by gvSIG"
    "wms_extent" "2.0742169 41.3576806 2.2446549 41.4705901"
    "wms_layer_group" "/pgis_BCN (PostgreSQL)"
    "gml_include_items" "all"
  END
END # Layer
LAYER
  HEADER "templates/LayerName_header.html"
  TEMPLATE "LayerName.html"
  FOOTER "templates/LayerName_footer.html"
  NAME "public.result_pol"
  STATUS ON
  DEBUG ON
  TYPE LINE
  DATA "geom from public.result_pol using unique gid using srid=4326"
  CONNECTIONTYPE POSTGIS
  CONNECTION "user=postgres password=1234 host=localhost port=5432 dbname=pgis_BCN"
  MAXSCALE -1.0
  MINSCALE -1.0
  TRANSPARENCY 100
  DUMP TRUE
  TEMPLATE "."
  SIZEUNITS pixels
  PROJECTION
    "init=epsg:4326"
  END
  CLASS
    STYLE
      COLOR 236 124 236
      WIDTH 1
    END
    NAME "default"
  END
  METADATA
    "wms_title" "public.result_pol"
   "wms_include_items" "all"
    "wms_abstract" "generated by gvSIG"
    "wms_extent" "2.0742169 41.3576806 2.2446549 41.4705901"
    "wms_layer_group" "/pgis_BCN (PostgreSQL)"
    "gml_include_items" "all"
  END
END # Layer
```

Figura 6.8. Fitxer MAP

```
  LAYER
    HEADER "templates/LayerName_header.html"
    TEMPLATE "LayerName.html"
    FOOTER "templates/LayerName_footer.html"
    NAME "public.lloc_incident"
    STATUS ON
    DEBUG ON
    TYPE LINE
    DATA "geom from public.lloc_incident using unique gid using srid=4326"
    CONNECTIONTYPE POSTGIS
    CONNECTION "user=postgres password=1234 host=localhost port=5432 dbname=pgis_BCN"
    MAXSCALE -1.0
    MINSCALE -1.0
    TRANSPARENCY 100
    DUMP TRUE
    TEMPLATE "."
    SIZEUNITS pixels
    PROJECTION
      "init=epsg:4326"
    END
    CLASS
      STYLE
        COLOR 48 24 126
        WIDTH 1
      END
      NAME "default"
    END
    METADATA
      "wms_title" "public.lloc_incident"
      "wms_include_items" "all"
      "wms_abstract" "generated by gvSIG"
      "wms_extent" "2.0742169 41.3576806 2.2446549 41.4705901"
      "wms_layer_group" "/pgis_BCN (PostgreSQL)"
      "gml_include_items" "all"
    END
  END # Layer
END # Map File
```

Figura 6.8. Fitxer MAP

En aquest fitxer MAP, cal assenyalar com aspectes més rellevants: l'especificació de la projecció (EPSG:4326) i les capes definides (ways, unit_resposta_a, unit_resposta_b, unit_resposta_p, unit_resposta_h, lloc_incident, result_amb, result_bom, result_pol, result_hos).

Les capes definides, contindran:

- ways: la taula on hi ha definides les vies del municipi.
- *unit_resposta_a, unit_resposta_b, unit_resposta_p, unit_resposta_h*: la distribució en diferents capes de les diferents unitats de resposta de la ciutat.
- lloc_incident: Conté la ubicació d'on s'ha produït el incident. L'entrada a aquesta taula es realitza una vegada l'usuari ha fet click a sobre del mapa.
- *result_amb, result_bom, result_pol, result_hos*: Aquestes taules contenen la ruta desde el lloc del incident cap a les diferents unitats de resposta que siguin més properes al lloc del incident. Les dades s'emplenen mitjançant consultes SQL definides més endavant.

La capes ways, *result_amb, result_bom, result_pol, result_hos* seran de tipus *line*. Les demés capes es mostraran mitjançant el símbol *circle*. Els símbols es troben definits en el fitxer *symbol.sym* comentat anteriorment.

Com es veurà més endavant, per a cada una d'aquestes capes s'establirà una connexió a la BD pgis_BCN i es sel.leccionaran les dades corresponents mitjançant la columna de geometria (geom) de cadascuna d'elles.

6.2. Implementació de la Base de Dades

El disseny de la base de dades és un reflex de la concepció inicial de programari. Donat que s'ha concebut de tal manera que sigui el més intuïtiu possible per a l'usuari, s'ha considerat el posar tota la càrrega de treball i càlcul en la base de dades. Aquesta és una bona forma d'alliberar a l'usuari d'operativa innecessària i fer, alhora, el disseny més minimalista. La forma de concentrar tota la càrrega i el càlcul ha estat mitjançant el disseny de taules i consultes específiques per a tal fi.

Per al càlcul de les rutes òptimes, s'han considerat les següents taules i consultes:

Taules		Consultes	
lloc_incident hosp_assignat cost_incid_unitat	result_amb result_bom result_hos result_pol	calc_amb1 calc_amb2 calc_bom1 calc_bom2	calc_pol1 calc_pol2 calc_hos2

Es veurà primer, un esquema gràfic que ajudarà a entendre l'operativa del programari:

1) Inicialment l'usuari marcarà en el mapa el lloc on s'ha produït el incident.
2) L'event de marcar farà que les dades de la seva localització geogràfica quedi emmagatzemades en la taula *lloc_incident*.
3) De forma anàloga, depenent de la tipologia de l'accident, l'usuari triarà un hospital que pugui atendre al accidentat. Queda fora de l'abast d'aquest projecte la funcionalitat d'assignar l'hospital més idoni de forma automàtica. En aquest projecte serà l'usuari qui coneixerà l'hospital més idoni.
4) De forma anàloga al segon punt, també s'emmagatzema aquesta dada. En aquest cas, en la taula *hosp_assignat*.

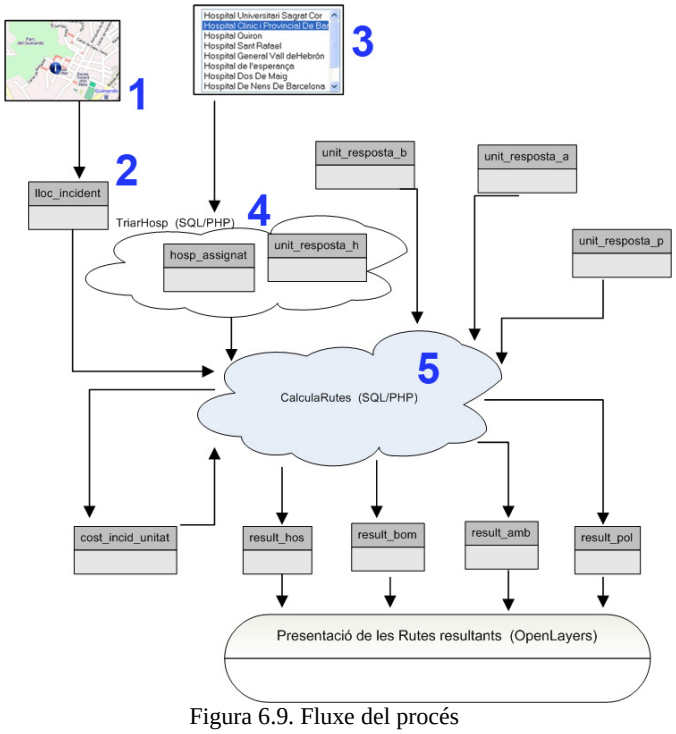

Figura 6.9. Fluxe del procés

5) Finalment, l'usuari interaccionarà amb el botó '*càlcul de rutes*'. Aquest iniciarà el procés de càlcul de rutes intern que es detalla tot seguit:

El procés del càlcul de les rutes òptimes es redueix en l'execució, de forma seqüencial, de les consultes descrites anteriorment. S'haguessin pogut unir totes les consultes en una única consulta. Però a nivell de claredat s'ha decidit implementar-ho d'aquesta manera. També, fent-ho d'aquesta manera, tampoc se n'ha ressentit la velocitat en el càlcul pròpiament dit.

La nomenclatura de les consultes descriu la seva funcionalitat. Així, hi hauran 2 consultes per a cada una de les unitats de emergència (policia, bombers i ambulàncies): *calc_pol, calc_bom, calc_amb*. En els hospitals (*calc_hos*) sols es disposa de una consulta. Aquest fet ve determinat perquè és el propi usuari qui triarà l'hospital destinatari del accidentat. A nivell explicatiu es farà referència sols a les consultes *calc_amb1 i calc_amb2*. De totes formes, l'operativa serà anàloga per la resta de les unitats.

La **primera consulta** (calc_amb1) es pot dir que està composada de 2 instruccions bàsiques:

La *primera instrucció***:** interactuarà amb la taula de treball *cost_incid_unitat*, eliminant els registres d'anteriors càlculs. Cal observar que cada consulta, tot i que comparteixen una taula comuna, sols interactuarà amb les dades que li afecte (ambulàncies (unitat='AMB'), policia (unitat='POL'), bombers (unitat='BOM')).

La *segona instrucció***:** és la que s'encarrega de tot el procés de càlcul a base de consultes imbricades. Seguidament, es farà una descripció detallada:

En la figura 6.10 es pot observar les dues instruccions bàsiques assenyalades amb blau.

Com s'ha comentat, aquesta primera consulta, inserirà en la taula de treball *cost_incid_unitat* el cost que suposa anar desde el punt on s'ha produït l'accident (emmagatzemat en la taula lloc_incident) fins a cadascuna de les unitats d'emergència (en aquest cas, les ambulàncies).

Es a dir, per a cada unitat d'ambulancies es disposarà d'una sentència:

```
insert into cost_incid_unitat (origen, desti, cost, unitat)
```

Tot seguit es descriurà cadascun dels paràmetres de forma detallada:

- *origen*: serà el lloc on s'ha produït el incident.

per a l'obtenció d'aquest valor, n'hi haurà prou amb l'execució d'aquesta subconsulta:

```
select vertices_tmp.id
from vertices_tmp, lloc_incident
where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1
```

Aquesta consulta obtindrà la dada emmagatzemada prèviament pel programari en el moment en que l'usuari ha marcat un punt sobre el mapa.

- *desti*: serà cadascuna de les unitats de resposta (en aquest cas ambulàncies).

S'Obtindrà aquest valor mitjançant amb l'execució de la subconsulta:

```
select vertices_tmp.id
from vertices_tmp, unit_resposta_a
where vertices_tmp.the_geom=unit_resposta_a.the_geom
and unit_resposta_a.id=1
```

- *cost*: serà el cost que suposa anar desde el punt origen fins al destí entrats anteriorment.

Per al càlcul del cost es farà servir l'algorisme A* al suposar, com s'ha descrit en l'apartat anterior, una clara millora respecte el de Dijkstra. L'algorisme de dijkstra té associada la funció wrapper *shortest_path_astar*, descrita en l'apartat anterior.

Aquesta funció se li entrarà l'origen i el destí, els mateixos dos paràmetres descrits anteriorment:

```
select sum(cost)
from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id from
vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select
vertices_tmp.id from vertices_tmp, unit_resposta_a where vertices_tmp.the_geom=unit_resposta_a.the_geom and
unit_resposta_a.id=1
), false, false))
```

- *unitat*: serà un literal amb la descripció de la unitat de resposta (en cas d'hospitals, serà 'AMB'). D'aquestà forma permetrà fer un seguiment més acurat dels càlculs del programa davant de possibles errors en el càlcul.

Com es pot observar, la consulta conté la totalitat de les unitats de resposta. En cas de les ambulàncies serien 7. El motiu de fer-se d'aquesta manera ha estat la simplificació i la poca volatilitat de les dades. Cal recordar que, en el nostre problema, tenim unes unitats de resposta (hospitals, policia, bombers) que variaran molt poc en el temps i pràcticament sempre seran les mateixes. D'aquí que s'hagi optat per aquesta implementació. De la mateixa manera, això suposa tenir tota la càrrega de treball en el servidor i alliberar de codi la part del client que hauria de fer aquesta sel.lecció

A tall d'exemple, es mostrarà el contingut parcial de la taula cost_incid_unitat un cop executada la corresponent consulta. Les dades que es mostren són de la unitat d'ambulàncies (AMB) i es troben ordenades per cost de forma ascendent.

origen	desti	cost	tipus
1191	5692	1.25115258379097	AMB
1191	10116	2.18273899245912	AMB
1191	1447	2.4969749160743	AMB
1191	1253	3.98549723789763	AMB
1191	1150	4.00236620718513	AMB
1191	6883	4.08635214785859	AMB
1191	2742	6.75061503959111	AMB

```
delete  from cost_incid_unitat where unitat='AMB';
insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a
where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=1
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select
vertices_tmp.id from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and
lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a where
vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=1
), false, false)), 'AMB');
insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a
where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=2
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select
vertices_tmp.id from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and
lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a where
vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=2
), false, false)), 'AMB');
insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a
where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=3
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select
vertices_tmp.id from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and
lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a where
vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=3
), false, false)), 'AMB');
insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a
where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=4
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select
vertices_tmp.id from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and
lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a where
vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=4
), false, false)), 'AMB');
insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a
where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=5
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select
vertices_tmp.id from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and
lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a where
vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=5
), false, false)), 'AMB');
insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a
where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=6
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select
vertices_tmp.id from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and
lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a where
vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=6
), false, false)), 'AMB');
insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a
where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=7
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select
vertices_tmp.id from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and
lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a where
vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=7
), false, false)), 'AMB');
```

Figura 6.10. Funció SQL 'calc-amb1'

Un cop completada aquesta fase, amb la càrrega de dades, es procedirà a l'execució de la segona consulta (Figura 6.3).

La **segona consulta (**calc_amb2), es nodrirà de les dades de la taula que vista abans i triarà el registre que li suposi un cost menor.

```
DROP TABLE IF EXISTS result_amb;
CREATE TABLE result_amb(gid int4) with oids;
SELECT AddGeometryColumn('result_amb', 'the_geom', '4326', 'MULTILINESTRING', 2);
INSERT INTO result_amb(the_geom)
SELECT the_geom FROM astar_sp_delta('ways', (select desti from cost_incid_unitat where cost=(SELECT min(cost) from cost_incid_unitat where unitat='AMB'))
, (select origen from cost_incid_unitat where cost=(SELECT min(cost) from cost_incid_unitat
where unitat='AMB')) , 0.1);
```

Figura 6.11. Funció SQL 'calc-amb2'

Per a l'execució d'aquesta segona consulta es farà servir una taula (*result_amb*) que contindrà la composició de la ruta òptima entre els dos punts.

Cal destacar, en aquesta consulta, la part en que agafa el menor valor dels valors vistos abans:

```
select desti from cost_incid_unitat where cost=(SELECT min(cost) from cost_incid_unitat where unitat='AMB'));
```

Com s'ha pogut observar, s'ha minimitzat al màxim la interacció de l'usuari amb el programari. Alhora els passos que aquest haurà de seguir són molt fàcils i intuïtius. Cal observar que s'ha descartat incloure el càlcul de les rutes en un dels events de sel.lecció donat que, fent-ho d'aquesta manera, no permetia toleràncies a errades per part de l'usuari en la sel.lecció. Es a dir, a vegades a l'usuari li pot convenir tornar a marcar el punt assenyalat amb anterioritat en un altre lloc sense que això impliqui l'execució cada vegada de tot el procés de càlcul. Per tant, per permetre aquesta tolerància, s'ha considerat més òptim situar el procés a dins del botó i que l'usuari haurà de premer de forma expressa.

De forma anàloga, en el disseny també s'ha contemplat la possibilitat de que aquest programari pugui ser multiusuari i amb accés concurrent per part d'ells. Els canvis que s'haurien de fer per tal efecte estarien a nivell de disseny de les taules anteriors. Aquestes taules haurien de tenir la possibilitat de poder emmagatzemar el nom del usuari. Tota la operativa i la lògica seguiria essent la mateixa.

6.3. Connexió entre PostGIS, MapServer i OpenLayers

Fins ara, s'ha vist com crear el fitxer MAP i els canvis que calia fer-hi. A continuació, s'ha parlat del disseny de la base de dades amb la creació de les taules i de les consultes SQL necessàries per tal de calcular les rutes òptimes. Ara es veurà com s'integraran tots aquests elements per tal de dotar de funcionalitat l'aplicatiu.

L'element central serà una pàgina HTML (Figura 6.12) que serà la que integrarà tots els elements. Com es pot apreciar, el disseny de la interfície d'usuari s'ha fet seguint les especificacions del World Wide Web Consortium (W3C).

El disseny de la interfície, s'ha fet de tal manera que fos el més intuïtiva possible per a l'usuari. El fet de posar en el cantó del servidor tot el procés de càlcul ha ajudat en gran mesura a fer un disseny molt més intuïtiu.

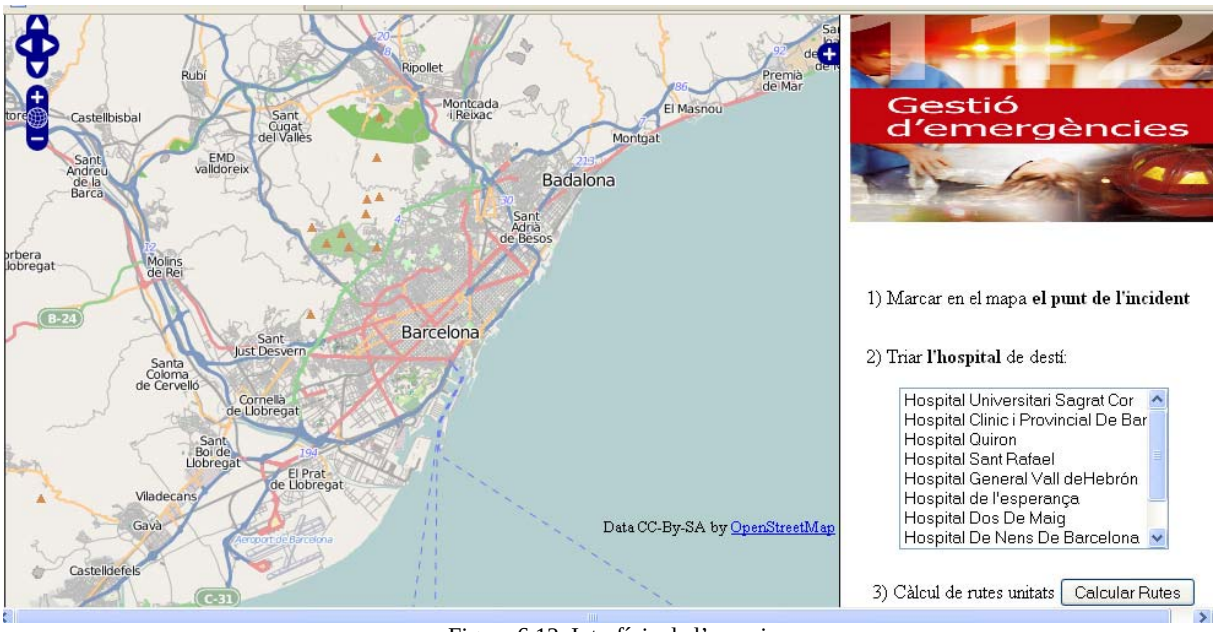

Figura 6.12. Interfície de l'usuari

Com es pot apreciar, el càlcul de les rutes consta de 3 passos clarament definits: triar en el mapa el lloc del incident, triar l'hospital més idoni per traslladar a l'accidentat i el propi càlcul de les rutes.

Com es pot observar, gràcies a OpenLayers, en el mapa s'hi pot afegir tota la funcionalitat que es cregui necessària: zooms, desplaçaments, visualització de coordenades del punter, etc. En aquest projecte s'ha optat per oferir sols les eines que s'han cregut necessàries.

Un altre aspecte a destacar és la distribució que s'ha fet a nivell de capes. Aquesta descripció queda reflexada amb l'etiqueta descriptiva que porta integrada la imatge (Figura 6.13).

Fig. 6.13. Etiqueta descriptiva

La distribució, s'hagués pogut fer de vàries formes depenent de molts factors. En aquesta primera versió del programari, OpenLayers mostra les següents capes:

- Una capa tipus OSM, provinent de OpenStreetMaps que mostra els carrers de la ciutat i dóna un aspecte visual idoni per la posterior inserció dels punts on s'ha produït una emergència.

- Capes WMS. Una capa per a cada unitat de resposta diferent (bombers, policia ,hospital i ambulància) i una cinquena per mostrar el resultat del càlcul de les rutes òptimes.

- Una capa de tipus vector que servirà per mostrar els punts sel.leccionats en el mapa.

Cal recordar que el càlcul de les rutes es realitzarà mitjançant consultes SQL que ja s'han explicat detalladament en l'apartat anterior. El motiu de fer-ho d'aquesta manera ha estat que SQL proporciona un estàndard ideal per a futures migracions de BBDD. D'aquesta manera, es podría aprofitar tota la seva funcionalitat per a la nova base de dades.

De totes formes, en aquest projecte, aquestes consultes que proporcionen tota la funcionalitat, s'han extret per tal que poguessin posar-se com a instruccions dins del llenguatge PHP i es podessin, d'aquesta manera, cridar desde la Interfície de l'usuari.

De forma anàloga, es farà servir el llenguatge PHP, un cop s'hagi fet clic sobre el mapa o s'hagi triat l'hospital de destinació, el programari es connectarà a la base de dades i s'emmagatzemaran els corresponents punts previs al càlcul.

Un cop efectuat el càlcul, la geometria de la ruta resultant és emmagatzemada en format WKT (Well Known Text) i es retorna com un fitxer XML a OpenLayers que la interpretarà i dibuixarà la ruta en una capa vectorial. L'usuari podrà visualitzar el resultat tot activant en l'etiqueta l'opció corresponent vista anteriorment.

Degut a la flexibilitat que ofereix OpenLayers, aquesta no és la única alternativa possible a l'hora de realitzar consultes a la base de dades. El fet d'optar per aquesta sol.lució, com ja s'ha comentat en repetides ocasions, ha estat la òptima interacció amb els usuaris.

Seguidament, es comentarà amb detall el codi HTML de la pàgina i els corresponents scripts d'OpenLayers.

6.3.1 Codi HTML i Script OpenLayers

La pàgina amb que l'usuari interaccionarà estarà composada de diferents parts, cadascuna amb la seva funcionalitat.

Al inici de la capçalera és a on s'especifica la localització de la llibreria OpenLayers necessaria per a poder treballar. Hi ha dues formes d'especificar-la: de forma local i remota.

```
<title> Gesti&oacute; d&acute;emerg&egrave;ncies </title>
<script src="http://www.openlayers.org/api/OpenLayers.js"></script>
```

Seguidament, és on comença l'script de OpenLayers. S'especificaran els estils de les etiquetes (arxius d'imatge *png* externs) dels punts d'emergència que establim i un tipus de variable (de la classe OpenLayers) que servirà per emmagatzemar els punts als quals s'hi faci clic a sobre del mapa.

```
var SinglePoint = OpenLayers.Class.create();
                        SinglePoint.prototype = OpenLayers.Class.inherit(OpenLayers.Handler.Point, {
                        createFeature: function(evt) {
                                this.control.layer.removeFeatures(this.control.layer.features);
                                OpenLayers.Handler.Point.prototype.createFeature.apply(this, arguments);
                        }
                        });

 var start_style = OpenLayers.Util.applyDefaults({
    externalGraphic: "inici.png",
    graphicWidth: 24,
    graphicHeight: 27,
    graphicYOffset: -27,
    graphicOpacity: 1
 }, OpenLayers.Feature.Vector.style['default']);
```

Seguidament, es declararan les variables que es faran servir a nivell global en les diferents funcions de les que està composat el script.

```
var map, start;
```

Seguidament, comença la funció init(). Aquesta funció s'executarà al carregar-se la pàgina. Es definirà una variable, options, on s'especificaran les característiques del mapa que es mostrarà inicialment: projection (la projecció), units (les unitats), numZoomLevels (els nivells de Zoom), la ressolució màxima i l'extensió màxima que se li voldrà donar.

El proper pas serà crear l'objecte map (mapa) al qual se li afegiran dos controls opcionals, un per a poder canviar les capes mostrades i l'altre per poder mostrar sobre el mapa les coordenades de la posició del cursor.

```
var options = {
        projection: new OpenLayers.Projection("EPSG:4326"),
        units: "m",
        numZoomLevels: 18,
        maxResolution: 156543.0339,
        maxExtent: new OpenLayers.Bounds( 226247, 5060351, 254733, 5081973)
        }

        map = new OpenLayers.Map('map', options);
        map.addControl(new OpenLayers.Control.LayerSwitcher());
        map.addControl(new OpenLayers.Control.MousePosition());
```

A continuació es declararan les capes que s'afegiran al mapa. La primera capa (mapnik) serà de tipus OSM i aportarà la capa base d'OpenStreetMap.

Les demés capes aportaran informació de la base de dades PostGIS. Així, layer1, layer2, layer3 i layer4 aportaran informació sobre la ubicació de les diferents unitats de resposta, respectivament, ambulàncies, bombers, hospitals i policia.

La darrera capa que s'afegirà contindrà el resultat de les rutes entre el punt del incident cap a les unitats de resposta que siguin més properes.

```
var mapnik = new OpenLayers.Layer.OSM("OSM base layer");

 var layer1 = new OpenLayers.Layer.WMS( "Unit. Ambulancies",
        "http://localhost:9000/cgi-bin/mapserv?map=/ms4w/apache/htdocs/BCN/emergencies.map", {layers: "public.unit_resposta_a"} );

 var layer2 = new OpenLayers.Layer.WMS( "Unit. Bombers",
        "http://localhost:9000/cgi-bin/mapserv?map=/ms4w/apache/htdocs/BCN/emergencies.map", {layers: "public.unit_resposta_b"} );

 var layer3 = new OpenLayers.Layer.WMS( "Unit. Hospitals",
        "http://localhost:9000/cgi-bin/mapserv?map=/ms4w/apache/htdocs/BCN/emergencies.map", {layers: "public.unit_resposta_h"} );

 var layer4 = new OpenLayers.Layer.WMS( "Unit. Policia",
        "http://localhost:9000/cgi-bin/mapserv?map=/ms4w/apache/htdocs/BCN/emergencies.map", {layers: "public.unit_resposta_p"} );

  var layer5 = new OpenLayers.Layer.WMS( "Rutes Calculades",
        "http://localhost:9000/cgi-bin/mapserv?map=/ms4w/apache/htdocs/BCN/emergencies.map",  {layers:  "public.result_amb,
public.result_bom, public.result_hos, public.result_pol"} );


  start = new OpenLayers.Layer.Vector("Punt d'origen", {style: start_style});
```

Finalment, s'afegeixen explícitament les capes al mapa, es transforma amb unes coordenades que asseguren que s'estigui en la projecció correcte i, finalment, s'enquadra la vista a la ciutat de Barcelona.

```
map.addLayers([mapnik, layer1, layer2, layer3, layer4, layer5, start]);
var proj4326 = new OpenLayers.Projection("EPSG:4326");
var limits = new OpenLayers.Bounds(2.064378, 41.324683, 2.256328, 41.470384);
limits.transform(proj4326, map.getProjectionObject());
map.zoomToExtent(limits);
```

Per acabar, s'afegeix a la funció *init()* el control dels clics que es facin sobre el mapa. Al fer un clic es emmagatzemarà el punt de la incidència en la variable start.

```
controls = {
  start: new OpenLayers.Control.DrawFeature(start, SinglePoint),
            }
              for (var key in controls) {
               map.addControl(controls[key]);
              }
```

Seguidament, es descriu la funció que s'executarà al prémer el botó de la pàgina, la funció *calcular().*

Aquesta funció es que farà serà guardar les dades de l'hospital triat en la taula corresponent i, seguidament, executar les consultes SQL definides amb anterioritat per al càlcul de les rutes. Aquestes funcions emplenaran les taules corresponents i seran visualitzades activant la capa corresponent en el navegador.

A nivell intern, es fa una crida a la pàgina *calcularutes.php* que és l'encarregada de fer tot el procés en un segon pla.

```
            // Al fer 'clic' a sobre del botó s'executarà la funció de calcular Aquesta funció ens farà tot el procés,
            // desde la inserció del punt de l'emergència  i l'hospital destinació fins el càlcul de les rutes.

            function calcular(){
                        var startPoint = start.features[0];
                        // En cas de que hi hagi un punt entrat, fem el procés
                        if (startPoint) {
                              var result = {
                                          startpoint: startPoint.geometry.x + ' ' + startPoint.geometry.y,
                                                };

                              // Guardem els valors triat als mapa i els triats en la llista d'hospitals en les corresponents taules
                              OpenLayers.loadURL("PuntIncident.php?startpoint=" + ' ' + startPoint.geometry.x+startPoint.geometry.y, null, null,
mostraRuta, null);
                              OpenLayers.loadURL("triarHosp.php", null, null, mostraRuta, null);

                              // i calculem les rutes
                              OpenLayers.loadURL("calcularutes.php", null, null, mostraRuta, null);

                              }

                        }
```

Finalment, es mostrarà el codi HTML de la pàgina en sí, on es podrà veure les diferents opcions dels hospitals, la càrrega del logo de l'aplicatiu, les indicacions dels passos a seguir, el botó que permetrà el càlcul de les rutes, etc.

```
<body onload="init()">

    <br><br>
    <table cellpadding="0" cellspacing="0" border="0" align="center" valign="middle" width="1000" height="100">
    <tr align="center" valign="middle">
    <td width="680" align="center"><div style="width:100%; height:100%" id="map"></div></td>
    <td width="320" align="center">
    <p><img src="emergencies.png" width="300" height="180" alt="emergencies" longdesc="emergencies"></p>
    <br>
    <table width="280" border="0">
            <tr>
    <td height="46" colspan="2">1) Marcar en el mapa <strong>el punt del incident</strong></td>
    </tr>
            <tr>
    <td height="46" colspan="2">2) Triar <strong>l'hospital</strong> de dest&iacute;: </td>
    </tr>
    <td height="114" colspan="2" align="center"><p>
    <select name="llistaHosp" size="8" id="llistaHosp"  onChange="llista(this.value)" style="width:220px;">
      <option value="1">Hospital Universitari Sagrat Cor</option>
      <option value="2">Hospital Clinic i Provincial De Barcelona</option>
      <option value="3">Hospital Quiron</option>
      <option value="4">Hospital Sant Rafael</option>
      <option value="5">Hospital General Vall deHebr&oacute;n</option>
      <option value="6">Hospital de l'esperan&ccedil;a</option>
      <option value="7">Hospital Dos De Maig</option>
      <option value="8">Hospital De Nens De Barcelona</option>
      <option value="9">Hospital Pere Virgili</option>
      <option value="10">Hospital Evangelico</option>
    </select>
    </td>
    </tr>

</table>
<br>
 3) Càlcul de rutes unitats
<button onClick="Calcular()">Calcular Rutes</button>
</td>
</tr>
</table>
   </body>
```

6.3.2 Fitxer Imputació dades (TriarHosp.php)

La funcionalitat d'aquest fitxer (figura 6.14) és carregar la taula que es necessita per al càlcul de rutes amb les dades de l'hospital triat. Com es pot veure en el codi, una vegada l'usuari ha triat un hospital de la llista i ha premut el botó de calcular les rutes, es guarden les dades d'aquest hospital en la taula 'hosp_assignat'. Aquesta taula la farà servir el procés descrit posteriorment (calcularutes) per calcular-ne la ruta més òptima cap a l'hospital triat.

El procés per guardar les dades de l'hospital triat ha estat molt simple. L'índex de l'opció triada per l'usuari en triar l'hospital desde l'entorn web coincideix amb la clau primària de les unitats de resposta dels hospitals. Per tant, el que s'ha fet és triar la unitat de resposta (hospital) que incidiís amb l'índex triart per l'usuari al fer 'cilc' en el desplegable i el resultat guardar-lo en la taula 'hosp_assignat'. D'aquesta manera el procés es fa d'una forma molt neta i entenedora.

```
<?php
// Definim els paràmetres per la connexió amb la BD
define("PG_DB" , "pgis_BCN");
define("PG_HOST", "127.0.0.1");
define("PG_USER", "postgres");
define("PG_PASS", "1234");
define("PG_PORT", "5432");



// Mirem qui és l'hospital que s'ha triat i carreguem les dades a la taula

 var id_hosp = document.getElementById("llistaHosp");

 $IDX_TRIAT = id_hosp.selectedIndex;



 // controlem que n'hagi triat un abans de grabar
 if id_hosp.selectedIndex != -1){

     // Ens connectem a la BD
     $con = pg_connect("dbname=".PG_DB." host=".PG_HOST." password=".PG_PASS." user=".PG_USER);

                 // Borrem la sel.lecció anterior
                 $sql0 = "delete from hosp_assignat;";
                 $query = pg_query($con,$sql0);

                 // Donat que el identificador del hospital que ha triat l'usuari a través de la plana web coincideix
                 // amb el camp clau de l'hospital, el que fem és accedir a les dades del hospital desitjat i guardem el
                 // seu resultat en la taula de l'hospital triat (hosp_assignat) per a poder els càlculs.

     $sql1= "insert into hosp_assignat (id, name, lat, lon, descr, the_geom) values
     (1, (select  name from unit_resposta_h where id=" .IDX_TRIAT. "), (select  lat from unit_resposta_h where id=" .IDX_TRIAT. "),
     (select  lon from unit_resposta_h where id=" .IDX_TRIAT. "), (select  descr from unit_resposta_h where id=" .IDX_TRIAT. "),
     (select  the_geom from unit_resposta_h where id=" .IDX_TRIAT. "));";

                 $query = pg_query($con,$sql1);


     // Tanquem la connexió a la BD
     pg_close($con);

    }


?>
```

Figura 6.14. Fitxer TriarHosp.php

Associat al procés anterior, hi ha el procés que guarda les dades del punt d'incidència sobre el mapa que ha marcat l'usuari.

Bàsicament el que fa el procés, descrit a continuació, és recuperar els paràmetres que es passen en la crida de la funció en forma de llistes i gravar-les en la taula '*lloc_incident*'. Aquestes dades, juntament amb les entrades en altres taules, les farà el servir el procés de càlcul de les rutes.

```
<?php
// Definim els paràmetres per la connexió amb la BD
define("PG_DB" , "pgis_BCN");
define("PG_HOST", "127.0.0.1");
define("PG_USER", "postgres");
define("PG_PASS", "1234");
define("PG_PORT", "5432");



// Seguidament, guardem les coordenades dels punts d'origen (del incident) que s'han passat com a
// paràmetres en la variable startPoint com a llista (x, y).

   $start = split(' ',$_REQUEST['startpoint']);
   $startPoint = array($start[0], $start[1]);

   $INICI_X = startPoint.geometry.x;
   $INICI_Y = startPoint.geometry.y;


   // Ens connectem a la BD
   $con = pg_connect("dbname=".PG_DB." host=".PG_HOST." password=".PG_PASS." user=".PG_USER);

   // Borrem les dades que hi hagi d'abans (de càlculs anteriors)

   $sql = "delete from lloc_incident;";
   $query = pg_query($con,$sql);


   // Inserim les dades relatives al nou punt entrat en el mapa

    $sql1= "insert into lloc_incident (id, name, lat, lon) values
    (1, 'punt del incident', " .INICI_X. "," .INICI_Y. ");";

   $query = pg_query($con,$sql1);


    // Tanquem la connexió a la BD
    pg_close($con);


?>
```

6.3.3 Fitxer del càlcul de rutes (calcularutes.php)

Aquest fitxer (figura 6.15), executat al prémer el botó 'Calcular Rutes', serà l'encarregat de comunicar-se amb la base de dades PostGIS i realitzar les consultes.

Com s'ha comentat amb anterioritat, s'ha fet una adaptació de les consultes dissenyades per oferir tota la potència de càlcul amb la mínima intervenció per part de l'usuari.

Com es pot observar, inicialment, es defineix, en varies variables, els paràmetres necessaris per a la connexió: el nom de la base de dades, el servidor, el port de connexió i les dades de l'usuari

Seguidament, per a les diferents unitats de resposta (Ambulancies, bombers, policia i hospitals), es composen les consultes SQL i s'executen. La seva execució es guarda en diverses taules de treball en format WKT, apunt per ser visualitzades.

```
<?php
// Definim els paràmetres per la connexió amb la BD
define("PG_DB" , "pgis_BCN");
define("PG_HOST", "127.0.0.1");
define("PG_USER", "postgres");
define("PG_PASS", "1234");
define("PG_PORT", "5432");

// Ens connectem a la BD
$con = pg_connect("dbname=".PG_DB." host=".PG_HOST." password=".PG_PASS." user=".PG_USER);

//#############################################################################################
// Composem les Consultes a executar corresponents a les AMBULANCIES
//#############################################################################################

$sql_01 = "delete  from cost_incid_unitat where unitat='AMB';";

$sql_02 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a
where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=1
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_a where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=1
), false, false)), 'AMB');";

$sql_03 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a
where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=2
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_a where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=2
), false, false)), 'AMB');";

$sql_04 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a
where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=3
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_a where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=3
), false, false)), 'AMB');";

$sql_05 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a
where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=4
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_a where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=4
), false, false)), 'AMB');";

$sql_06 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a
where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=5
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_a where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=5
), false, false)), 'AMB');";

$sql_07 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a
where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=6
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_a where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=6
), false, false)), 'AMB');";

$sql_08 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_a
where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=7
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_a where vertices_tmp.the_geom=unit_resposta_a.the_geom and unit_resposta_a.id=7
), false, false)), 'AMB');";
```

Figura 6.15. Fitxer calcularutes.php

```
$sql_09 = "DROP TABLE IF EXISTS result_amb;";

$sql_10 = "CREATE TABLE result_amb(gid int4) with oids;";

$sql_11 = "SELECT AddGeometryColumn('result_amb', 'the_geom', '4326', 'MULTILINESTRING', 2);";

$sql_12 = "INSERT INTO result_amb(the_geom)
SELECT the_geom FROM astar_sp_delta('ways', (select desti from cost_incid_unitat where cost=(SELECT min(cost) from cost_incid_unitat
where unitat='AMB'))
, (select origen from cost_incid_unitat where cost=(SELECT min(cost) from cost_incid_unitat
where unitat='AMB'))
, 0.1);";

// Executem les sentències SQL corresponents a les AMBULANCIES
$query_01 = pg_query($con,$sql_01);
$query_02 = pg_query($con,$sql_02);
$query_03 = pg_query($con,$sql_03);
$query_04 = pg_query($con,$sql_04);
$query_05 = pg_query($con,$sql_05);
$query_06 = pg_query($con,$sql_06);
$query_07 = pg_query($con,$sql_07);
$query_08 = pg_query($con,$sql_08);
$query_09 = pg_query($con,$sql_09);
$query_10 = pg_query($con,$sql_10);
$query_11 = pg_query($con,$sql_11);
$query_12 = pg_query($con,$sql_12);

//#############################################################################################
// Composem les Consultes a executar corresponents als BOMBERS
//#############################################################################################

$sql_01 = "delete  from cost_incid_unitat where unitat='BOM';";


$sql_02 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_b
where vertices_tmp.the_geom=unit_resposta_b.the_geom and unit_resposta_b.id=1
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_b where vertices_tmp.the_geom=unit_resposta_b.the_geom and unit_resposta_b.id=1
), false, false)), 'BOM');";

$sql_03 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_b
where vertices_tmp.the_geom=unit_resposta_b.the_geom and unit_resposta_b.id=2
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_b where vertices_tmp.the_geom=unit_resposta_b.the_geom and unit_resposta_b.id=2
), false, false)), 'BOM'); ";

$sql_04 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_b
where vertices_tmp.the_geom=unit_resposta_b.the_geom and unit_resposta_b.id=3
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_b where vertices_tmp.the_geom=unit_resposta_b.the_geom and unit_resposta_b.id=3
), false, false)), 'BOM'); ";

$sql_05 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_b
where vertices_tmp.the_geom=unit_resposta_b.the_geom and unit_resposta_b.id=4
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_b where vertices_tmp.the_geom=unit_resposta_b.the_geom and unit_resposta_b.id=4
), false, false)), 'BOM'); ";


$sql_06 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_b
where vertices_tmp.the_geom=unit_resposta_b.the_geom and unit_resposta_b.id=5
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_b where vertices_tmp.the_geom=unit_resposta_b.the_geom and unit_resposta_b.id=5
), false, false)), 'BOM'); ";
```

Figura 6.15. Fitxer calcularutes.php

```
$sql_07 = "DROP TABLE IF EXISTS result_bom;";

$sql_08 = "CREATE TABLE result_bom(gid int4) with oids;";

$sql_09 = "SELECT AddGeometryColumn('result_bom', 'the_geom', '4326', 'MULTILINESTRING', 2);";

$sql_10 = "INSERT INTO result_bom(the_geom)
SELECT the_geom FROM astar_sp_delta('ways', (select desti from cost_incid_unitat where cost=(SELECT min(cost) from cost_incid_unitat
where unitat='BOM'))
, (select origen from cost_incid_unitat where cost=(SELECT min(cost) from cost_incid_unitat
where unitat='BOM'))
, 0.1);";

// Executem les sentències SQL corresponents als BOMBERS

$query_01 = pg_query($con,$sql_01);
$query_02 = pg_query($con,$sql_02);
$query_03 = pg_query($con,$sql_03);
$query_04 = pg_query($con,$sql_04);
$query_05 = pg_query($con,$sql_05);
$query_06 = pg_query($con,$sql_06);
$query_07 = pg_query($con,$sql_07);
$query_08 = pg_query($con,$sql_08);
$query_09 = pg_query($con,$sql_09);
$query_10 = pg_query($con,$sql_10);

//#############################################################################################
// Composem les Consultes a executar corresponents a la POLICIA
//#############################################################################################

$sql_01 = "delete  from cost_incid_unitat where unitat='POL';";

$sql_02 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_p
where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=1
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_p where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=1
), false, false)), 'POL');";

$sql_03 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_p
where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=2
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_p where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=2
), false, false)), 'POL');";

$sql_04 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_p
where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=3
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_p where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=3
), false, false)), 'POL');";

$sql_05 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_p
where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=4
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_p where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=4
), false, false)), 'POL');";


$sql_06 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_p
where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=5
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_p where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=5
), false, false)), 'POL');";
```

Figura 6.15. Fitxer calcularutes.php

```
$sql_07 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_p
where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=6
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_p where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=6
), false, false)), 'POL');";

$sql_08 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_p
where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=7
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_p where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=7
), false, false)), 'POL');";

$sql_09 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_p
where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=8
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_p where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=8
), false, false)), 'POL');";

$sql_10 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_p
where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=9
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_p where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=9
), false, false)), 'POL');";

$sql_11 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_p
where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=10
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_p where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=10
), false, false)), 'POL');";

$sql_12 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_p
where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=11
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_p where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=11
), false, false)), 'POL');";

$sql_13 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_p
where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=12
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_p where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=12
), false, false)), 'POL');";

$sql_14 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_p
where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=13
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_p where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=13
), false, false)), 'POL');";

$sql_15 = "insert into cost_incid_unitat (origen, desti, cost, unitat) values (( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1 ),(select vertices_tmp.id from vertices_tmp, unit_resposta_p
where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=14
), (select sum(cost) from shortest_path_astar('select gid as id, source, target, length as cost, x1, y1, x2, y2 from ways', ( select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom and lloc_incident.id=1 ),(select vertices_tmp.id from
vertices_tmp, unit_resposta_p where vertices_tmp.the_geom=unit_resposta_p.the_geom and unit_resposta_p.id=14
), false, false)), 'POL');";
```

Figura 6.15. Fitxer calcularutes.php

```
$sql_16 = "DROP TABLE IF EXISTS result_pol;";

$sql_17 = "CREATE TABLE result_pol(gid int4) with oids;";

$sql_18 = "SELECT AddGeometryColumn('result_pol', 'the_geom', '4326', 'MULTILINESTRING', 2);";

$sql_19 = "INSERT INTO result_pol(the_geom)
SELECT the_geom FROM astar_sp_delta('ways', (select desti from cost_incid_unitat where cost=(SELECT min(cost) from cost_incid_unitat
where unitat='POL'))
, (select origen from cost_incid_unitat where cost=(SELECT min(cost) from cost_incid_unitat
where unitat='POL'))
, 0.1);";


// Executem les sentències SQL corresponents a la POLICIA

$query_01 = pg_query($con,$sql_01);
$query_02 = pg_query($con,$sql_02);
$query_03 = pg_query($con,$sql_03);
$query_04 = pg_query($con,$sql_04);
$query_05 = pg_query($con,$sql_05);
$query_06 = pg_query($con,$sql_06);
$query_07 = pg_query($con,$sql_07);
$query_08 = pg_query($con,$sql_08);
$query_09 = pg_query($con,$sql_09);
$query_10 = pg_query($con,$sql_10);
$query_11 = pg_query($con,$sql_11);
$query_12 = pg_query($con,$sql_12);
$query_13 = pg_query($con,$sql_13);
$query_14 = pg_query($con,$sql_14);
$query_15 = pg_query($con,$sql_15);
$query_16 = pg_query($con,$sql_16);
$query_17 = pg_query($con,$sql_17);
$query_18 = pg_query($con,$sql_18);
$query_19 = pg_query($con,$sql_19);


//#############################################################################################
// Composem les Consultes a executar corresponents als HOSPITALS
//#############################################################################################

$sql_01 = "DROP TABLE IF EXISTS result_hos;";

$sql_02 = "CREATE TABLE result_hos(gid int4) with oids;";

$sql_03 = "SELECT AddGeometryColumn('result_hos', 'the_geom', '4326', 'MULTILINESTRING', 2);";

$sql_04 = "INSERT INTO result_hos(the_geom)
SELECT the_geom FROM astar_sp_delta('ways', (select vertices_tmp.id
from vertices_tmp, lloc_incident where vertices_tmp.the_geom=lloc_incident.the_geom
and lloc_incident.id=1)
, (select vertices_tmp.id
from vertices_tmp, hosp_assignat where vertices_tmp.the_geom=hosp_assignat.the_geom
and hosp_assignat.id=1)
, 0.1);";


// Executem les sentències SQL corresponents als HOSPITALS

$query_01 = pg_query($con,$sql_01);
$query_02 = pg_query($con,$sql_02);
$query_03 = pg_query($con,$sql_03);
$query_04 = pg_query($con,$sql_04);



// Tanquem la connexió a la BD
pg_close($con);


?>
```

6.4. Exemple de funcionament

En aquest apartat es mostrarà, de forma visual, el funcionament de l'aplicatiu a través de captures de pantalla.

S'ha fet una tria sobre el mapa d'un punt en el que ha ocorregut un accident. S'ha triat un punt que estigui al mig de varies unitats de resposta per tal de poder verificar fàcilment, de forma visual, la correctesa del resultat (Figura 6.16).

Figura 6.16. Es tria el lloc del incident

Figura 6.17. Es tria l'hospital

Un cop triat el lloc del accident, es determinarà l'hospital al que es vol traslladar a l'accidentat (figura 6.17). Per acabar s'executarà l'opció de *calcular les rutes.* En executar el procés, s'executaran les consultes SQL descrites amb anterioritat i es realitzaran els càlculs. El resultats aniran a parar a les corresponents taules.

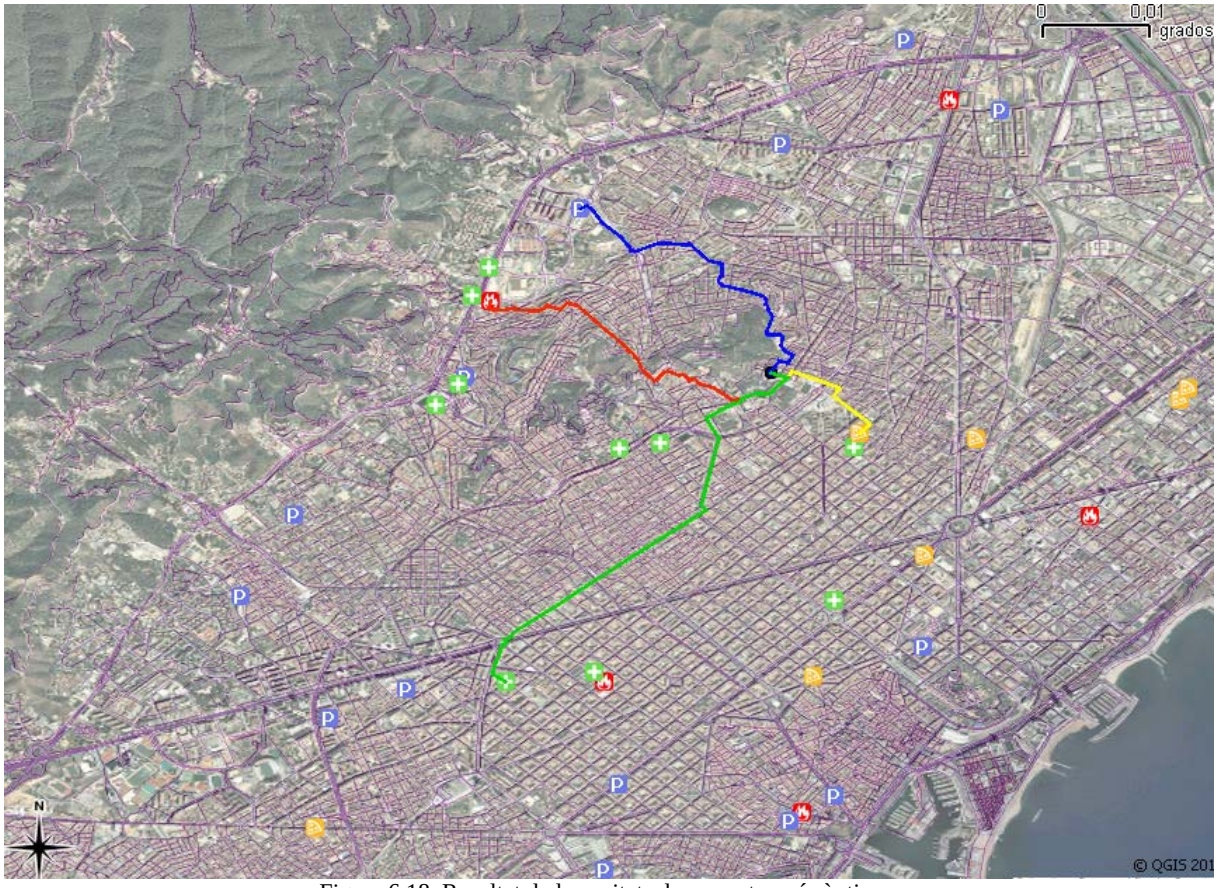

Figura 6.18. Resultat de les unitats de resposta més òptimes

En la figura anterior (Figura 6.18) es pot apreciar el resultat del càlcul final de les rutes. El resultat global s'ha mostrat a través d'un navegador pesat (qGIS). El motiu ha estat que el nostre disseny amb OpenLayers sols permetia visualitzar les diferents rutes de forma separada al estar assignades a diferents capes. S'ha considerat que, com a presentació, seria molt millor la mostra conjunta dels resultats.

Si s'observa la imatge anterior es poden observar les diferents unitats de resposta assignades amb un color diferenciador.

Així, per exemple, de les unitats de resposta policials (color blau) disponibles, s'ha presentat com a unitat de resposta que atendrà el incident la que es troba més proper al lloc del inicident. Es pot apreciar visualment com les altres unitats de resposta policials (també en color blau) queden molt més allunyades que la unitat de resposta que ha sortit escollida.

El mateix raonament per les unitats de resposta de les ambulancies (groc) i la dels bombers (vermell). Associades a la unitat de resposta i al lloc del incident es mostra la ruta més òptima cap a aquestes. A la ruta se li ha assignat un color amb cooncordància amb les diferents unitats de resposta.

En quan als hospitals (color verd), pel sol fet de tenir-lo que triar nosaltres, el programa es limitarà a calcular-ne la ruta més òptima desde el lloc del incident fins a la seva ubicació.

7. Conclusions i linies futures

En aquest apartat s'exposen les conclusions del treball realitzat i les possibles ampliacions i línies de treball futures.

Conclusions:

- Com s'ha pogut demostrar en aquest projecte, sols amb eines de programari lliure s'ha pogut desenvolupar un aplicatiu apte per a l'ús professional.

- Tal com es comentava en la introducció, la gran majoria de les eines de software lliure que s'ha fet servir disposen d'una abundant documentació i d'una comunitat de desenvolupadors a qui remetre's en cas de dificultats.

- El projecte d'OpenStreetMaps és un projecte molt consolidat i és una molt bona font de dades geogràfiques. De totes formes, cal assenyalar que, depenent del municipi que es desitgi les dades, poden no ser-hi o bé estar incomplertes.

- Mitjançant OpenLayers i JavaScript permeten fer una presentació de qualitat de la informació extreta del SIG. Tot i això, també s'ha de comentar que s'hi han trobat a faltar eines que permetessin solventar alguns del problemes ocorreguts (donat que l'eina Firebug no era suficient). Concretament al intentar visualitzar capes WMS no hi havia informació del perquè no era possible fer-ho.

- De forma paral·lela al punt anterior cal assenyalar l'existència de diverses extensions (com la de publicació del client pesat gvSIG) que permetien crear fitxers, com el MAP, necessaris per la seva visualització amb OpenLayers.

- Degut a la necessitat d'utilitzar PHP per tal de confeccionar planes dinàmiques i degut a la poca utilitat que suposava en aquest apartat l'ús de Firebug, s'ha hagut d'adaptar provar, primerament el codi utilitzat a un altre llenguatge, concretament amb l'entorn .NET, per tal de disposar d'una eina que permetés fer un "pas a pas" i comprovar la correctesa de la instruccions.

- Cal recordar que en aquest projecte es calculen les rutes prenent com a origen i destí els nodes més propers. Això fa que s'obviïn les arestes tant a nivell informatiu com a l'hora de fer els càlculs. Això provocarà petits errors de càlcul a l'hora de determinar la unitat de resposta més adient.

Tot i que s'han complert els requeriments que es demanaven en l'enunciat, encara són moltes les **millores** que es podrien realitzar:

- El primer aspecte a comentar és la possibilitat que el programari fós multiusuari. S'ha comentat amb anterioritat que el disseny de la base de dades estava preparat a tal efecte i sols s'haurien d'introduir en diversos punts del procés les dades de l'usuari en curs.

- Una altra millora vindria relacionada amb la tria de l'hospital de destinació en funció d'un conjunt de criteris, encara per definir, que no tingués que ser l'usuari que l'escollís de forma manual.

- Relacionat amb el punt anterior, es podrien oferir alguna alternativa a les unitats de resposta més propera en cas d'incidències, com talls sobtats de circulació o gran densitat de trànsit en determinats moments. La sol.lució passaria per oferir dues alternatives, com a mínim, per a cada unitat de resposta.

- Una altre de les millores seria poder oferir a l'usuari una estimació de la distància i del temps que trigarien les diferents unitats de resposta fins al punt on ha ocorregut el incident. Per dur a terme aquesta millora seria suficient en modificar, per a cada unitat de resposta, la consulta SQL del càlcul del cost introduint-hi la funció wrapper *ST_DISTANCE*. Aquesta funció ens retornarà la distància en metres que separa a dos punts (per al càlcul utilitza els camps the_geom origen i destí).

- Una millora associada al punt anterior seria el càlcul del temps que trigaria en arribar al punt de l'emergència. Aquesta dades seria difícil de determinar donat que estaria condicionada, a més de la distància, a altres factors com el trànsit del moment en aquest punt del municipi.

Per acabar, comentar que aquesta ha estat la meva primera experiència amb el món dels SIG. El treball ha estat un procés on, apart d'assimilar els diferents conceptes propis dels SIG, s'han hagut de vèncer moltes dificultats i consultar molta documentació.

Les principals dificultats han estat en l'apartat de OpenLayers, dificultats que han alentit el procés molt més de lo esperat. El fet de ser un projecte amb la intervenció d'una gran varietat de programari i sistemes (Linux, BD, pàgines web, JavaScript, etc) han fet que la seva realització hagi estat particularment difícil.

8. Referències

1 **Osm2pgrouting tool,** [Consulta 6 d'octubre de 2010], <http://pgrouting.postlbs.org/wiki/tools/osm2pgrouting>

2 **Ubuntu Desktop edition**, [Consulta 5 d'octubre de 2010], <http://www.ubuntu.com/>

3 **Bayarri, Salvador (2009)** "Infraestructuras de Datos Espaciales (y más allá) con Tecnologías Libres" , [Consulta 12 d'octubre de 2010], <http://www.idee.es/resources/presentaciones/GTIDEE_Malaga_2009/IDEsSoftwareLibreIVER.pdf>

4 **MapServer, Open Source Web Mapping**, [Consulta 28 setembre de 2010], <http://mapserver.org>

5 **PostgreSQL, [**Consulta 28 setembre de 2010], < http://www.postgresql.org/>

6 **PostGIS spatial database extension for PostgreSQL**, [Consulta 28 setembre de 2010], <http://postgis.refractions.net>

7 **PGCon - PostgreSQL Conference for Users and Developers**, [Consulta 15 d'Octubre de 2010], <http://www.pgcon.org/2008/schedule/attachments/52_postgis.pdf>

8 **University of Zurich – Department of Geography**, [Consulta 15 d'Octubre de 2010], < http://www.geo.unizh.ch/oai/spatialdb/folien/ogcsf.pdf>

9 **PostGIS Reference** [Consulta 17 d'octubre de 2010], <http://postgis.refractions.net/docs/reference.html>

10 **Geometry Engine Open Source,** [Consulta 12 d'Octubre de 2010], <http://trac.osgeo.org/geos>

11 **Cartografic Projections Library,** [Consulta 12 d'Octubre de 2010], <http://trac.osgeo.org/proj>

12 **pgRouting project**, [Consulta 28 de setembre de 2010], <http://pgrouting.postlbs.org/>

13 **Genetic Algorithm Utility Library (GAUL)**, [Consulta 9 d'octubre de 2010], <http://sourceforge.net/projects/gaul/files/gaul-devel/0.1850-0/gaul-devel-0.1850-0.tar.gz/download/>

14 **Pgrouting – postlbs - Ticket #160** , [consulta 9 d'octubre de 2010], <http://pgrouting.postlbs.org/ticket/160>

15 **Pgrouting – postlbs - Message #1426** , [consulta 10 d'octubre de 2010], <http://pgrouting.postlbs.org/discussion/message/1426>

16 **OpenLayers: Free Maps for the Web**, [Consulta 6 d'octubre de 2010], <http://www.openlayers.org/>

17 **gvSIG, Conselleria d'infrastructures i transport**, [Consulta 8 d'octubre de 2010], <http://www.gvsig.gva.es/>

18 **uDIG, Userfriendly Desktop Internet GIS**, [Consulta 8 d'octubre de 2010], <http://udig.refractions.net/>

19 **Quantum GIS project**, [Consulta 8 d'octubre de 2010], <http://www.qgis.org/>

20 **Firebug**, [Consulta 7 d'octubre de 2010], <http://getfirebug.com/>

21 **NotePAD++**, [Consulta 8 d'octubre de 2010], < http://notepad-plus-plus.org/>

22 **Cartociudad,** [Consulta 7 d'octubre de 2010], <http://www.cartociudad.es/visor/>

23 **OpenStreetMaps,** [Consulta 8 d'octubre de 2010], <http://www.openstreetmap.org>

24 **Osm2pgsql**, [Consulta 10 d'octubre de 2010], <http://wiki.openstreetmap.org/wiki/osm2pgsql>

25 **Osm2pgr**, [Consulta 10 d'octubre de 2010], <http://rapidshare.com/files/353030545/osm2pgr-0.8.zip.html>

26 **OpenGeo suite** [Consulta 11 d'octubre de 2010], <http://opengeo.org/>

27 **OSMTranslator**, [Consulta 11 d'octubre de 2010], < http://www.polygongis.com/OSMTranslator.aspx>

28 **Osm2pgrouting**, [Consulta 11 d'octubre de 2010], <http://pgrouting.postlbs.org/wiki/tools/osm2pgrouting>

29 **Institut Cartogràfic de Catalunya**, [Consulta 17 d'octubre de 2010], <http://www.icc.es/>

30 **How to useHow to Use pg_dump and pg_restore with Postgres Plus(R) in Windows(R)** [Consulta 12 d'octubre de 2010] <http://www.enterprisedb.com/learning/tutorial/postgresql_dump_restore_windows.do>

31 **Servei de mapes del Institut Cartogràfic de Catalunya**, [Consulta 19 d'octubre de 2010], <http://shagrat.icc.es/lizardtech/iserv/ows?>

32 **Google Maps** [Consulta 19 d'octubre de 2010], <http://maps.google.es/>

33 **Management Functions** [Consulta 24 d'octubre de 2010] <http://postgis.refractions.net/documentation/manual-1.4/AddGeometryColumn.html>

34 **PostGIS: Data Management and Queries** [Consulta 24 d'octubre de 2010] <http://postgis.refractions.net/documentation/manual-1.4/ch04.html#Manual_Register_Spatial_Column>

35 **FOSS4G 2009 TOKYO,** [Consulta 24 d'octubre de 2010] <http://www.osgeo.jp/wordpress/wp-content/uploads/2009/11/workshop_manual.pdf>

36 **PgRouting 1.02 on Win32,** [Consulta 24 d'octubre de 2010] <http://www.davidgis.fr/documentation/pgrouting-1.02/>

37 **Extensió de Publicació pgRouting,** [Consulta 15 de Novembre de 2010] <http://www.gvsig.org/web/projects/gvsig-desktop/official/gvsig-1.1/extensiones-gvsig-1.1/extension-de-publicacion/>

38 **Web-based Routing: An Introduction to pgRouting with OpenLayers,** [Consulta 22 de Novembre de 2010] <http://files.postlbs.org/foss4g2007/W-12/foss4g_w12/docs/foss4g_w12_manual.pdf/>