Universitat Autònoma de Barcelona

escola d'enginyeria

2148: GEOLOCALITZACIÓ EN TEMPS REAL SOBRE GOOGLE MAPS

Memòria del Projecte Fi de Carrera
d'Enginyeria en Informàtica
realitzat per **Rafael Martín Farré**
i dirigit per **Josep Maria Ganyet Cirera**
Bellaterra,16 de Juny de 2010

El sotasignat, **Josep Maria Ganyet Cirera**

Professor/a de l'Escola Tècnica Superior d'Enginyeria de la UAB,

CERTIFICA:

Que el treball a què correspon aquesta memòria ha estat realitzat sota la seva direcció per en **Rafael Martín Farré**

I per tal que consti firma la present.

Signat: **Josep Maria Ganyet Cirera**

Bellaterra, 16 de Juny de 2010

Agraïments:

Al meu tutor del projecte final de carrera, que sempre m'ha aconsellat i ha estat disponible.

A la meva família, per ajudar-me, en tot moment, en el que he necessitat.

I a la Ceci, la meva companya, que m'ha donat ànims quan més ho necessitava.

Índex de continguts

1 Introducció 1
1.1 Motivació 1
1.2 Objectius 1
1.3 Organització 3
1.3.1 Planificació inicial del temps de treball 3
1.3.2 Planificació real del temps de treball 5
2 L'Estat de l'art 7
2.1 La Geolocalització 7
2.2 Els mapes 8
2.2.1 Superposicions i Tipus de dades. 9
2.3 Les coordenades i el sistema de coordenades 10
2.3.1 Altres sistemes de referenciació espacial: 13
2.4 El sistema GPS: 15
2.4.1 Estructura del sistema: 15
2.4.2 Aplicacions: 16
2.4.3 Conceptes bàsics del GPS: 16
2.4.4 Transmissió del missatge: 21
2.5 Sistema de seguiment de vehicles: 22
2.5.1 Aplicacions: 22
3 Estructura de l'aplicació 24
4 Anàlisi de requeriments. 25
4.1 Requeriments funcionals: 25
4.2 Requeriments no funcionals 30
5 Modelat del comportament 31
5.1 Casos d'ús 31
5.1.1 Pasos per l'obtenció de casos d'ús. 31
5.1.2 Diagrama de casos d'ús. 32
6 Disseny 37
6.1 Disseny de la Base de Dades 37
6.1.1 Versió Preliminar 37
6.1.2 Versió Final 40
6.2 Disseny Algorisme. 43
6.2.1 Diagrama de flux del mòdul de captació: 44
6.2.2 Diagrames de flux del mòdul de processament: 45
7 Implementació 48
7.1 Base de dades. 48
7.2 Mòdul de captació 48
7.3 Mòdul de processament. 51
7.3.1 Inicialització i càrrega: 54
7.3.2 Visualització dels vehicles 59
7.3.3 Gestió de Polígons. 60
7.3.4 Visualitzar Gràfic de Velocitat 63
7.3.5 Visualitzar ruta per dies 63
7.3.6 Gestió d'alertes i notificacions 64
7.3.7 Gestió punts d'interès 67
7.3.8 Seguiment del vehicle 69
7.3.9 Visualitzar ruta amb Street View 70
7.3.10 Visualitzar ruta amb Google Earth 72

7.3.11 Modificar Opcions i Informació del vehicle 73
7.3.12 Crear Ruta 75
7.3.13 Visualitzar tràfic 76
8 Proves 80
8.1 Proves mòdul de captació 80
8.2 Proves mòdul processament 86
8.2.1 Proves de càrrega de vehicles 86
8.2.2 Proves punts d'interès: 87
8.2.3 Proves polígons: 93
8.2.4 Sistema GeoFencing 96
9 Conclusions i treball futur 98
9.1 Conclusions 98
9.2 Treball futur 100
10 Bibliografia 102
11 Annex 104
11.1 Annex 1: Model Entitat-Relació de la base de dades. 104
11.2 Annex 2: Diagrames de classes 105
11.3 Annex 3: Consultes a la Base de Dades. 107
11.4 Annex 4: Resposta XML. 108
11.5 Annex 5. Resposta del servidor de la Direcció General De Tràfic: .. 109

1 Introducció

1.1 Motivació

Les tecnologies, com el temps, avancen i cada cop són més importants en el seu objectiu de facilitar l'adaptació al medi i satisfer les necessitats de les persones, és a dir, de la societat. Aquestes també han millorat la mobilitat de les persones i els béns, gràcies a l'evolució del transport.

En una societat que canvia i es mou tan ràpidament, el transport és una eina molt potent i important, tant des del punt de vista econòmic com social. Aquesta societat, anomenada de la informació, converteix les TIC (Tecnologies de la informació i la comunicació) en els nous motors de desenvolupament i progrés. L'evolució i creixement del transport, per tant, seria impossible sense la informació i la comunicació; vitals per a sistemes de transport avançats, on es realitza un control exhaustiu de la situació geogràfica.

Amb l'evolució de les TIC apareixen nous components informàtics de hardware i software que permeten augmentar la velocitat de transferència de la informació. A més a més, gràcies a la democratització de les tecnologies, eines que fa uns anys solament eren exclusives dels serveis militars, ara, tota la població civil pot fer-ne ús. Com és el cas del sistema GPS, que ens permet obtenir informació de la nostra situació geogràfica en qualsevol punt de la Terra. La necessitat de controlar la mobilitat i la posició geogràfica de persones i béns en temps real, és el motiu que m'ha impulsat a realitzar el projecte de fi de carrera anomenat: **Geolocalització en temps real sobre Google Maps®**.

El projecte es basarà en la creació d'un Sistema de seguiment de vehicles per GPS, que ens permetrà, en qualsevol moment, visualitzar en quina posició geogràfica es troba una certa flota de vehicles.

1.2 Objectius

L'objectiu principal d'aquest projecte és la creació d'un Sistema d'Informació Geogràfica (SIG) sobre vehicles en temps real i sobre la plataforma online cartogràfica Google Maps ® que permeti visualitzar en tot moment on es troben els vehicles controlats, les seves dades com l'hora, la velocitat, les coordenades i les aturades realitzades.

El primer dels objectius que presento és l' organització i visualització de les rutes realitzades per un vehicle. Podent canviar la visualització de rutes anteriors (per minuts, hores, dies, setmanes, mesos i anys), el color, la forma i la visualització interactiva de la ruta sobre la plataforma Google Maps Street View ® i Google Earth ®.

El segon objectiu és la gestió i administració dels punts d'Interès que es representaran sobre el mapa i que representaran un punt de la situació geogràfica al qual se li donarà un significat, un aspecte i una informació addicional.

El tercer objectiu és la gestió i administració dels polígons que es mostraran sobre el mapa i que representaran una zona geogràfica formada per diversos punts. Tindran un nom associat i un control sobre quins vehicles han estat dins la zona. Aquest control produirà una seguit d'alertes que podran ser visualitzades en forma de taula, segons el vehicle que l'ha produït o el Poligon on s'ha produït. Les alertes que es produeixin en temps real també podran ser enviades al correu electrònic dels usuaris que es registrin per un cert polígon i demanin l'enviament. En el correu s'informarà quin vehicle ha fet saltar l'alerta i en quin polígon. També es mostrarà, en una imatge, la ubicació on s'ha produït l'alerta.

El quart objectiu és la gestió i administració dels vehicles que es representaran sobre el mapa i que estaran situats en un punt de la situació geogràfica. Aquests tindran opcions de visualització de vehicle o ruta, gràfics de velocitat i altitud, seguiment del vehicle en temps real, incorporació de nous vehicles, creació i organització de rutes.

El cinquè objectiu és la captació, tractament i emmagatzematge de les dades rebudes des del terminal mòbil amb tecnologia de georeferenciació GPS (Global Positioning System).

El sisè objectiu és la visualització de la Informació del tràfic de l'Estat Espanyol. Es mostraran sobre el mapa una sèrie de marques que ens permetran consultar l'estat i la informació del tràfic.

Per tant, podem resumir els objectius en quatre punts:

1 Control en temps real dels vehicles.
2 Administració i gestió dels Punts d'Interès.
3 Gestió i administració dels Poligons o zones.
4 Gestió i administració dels Vehicles.
5 Captació, tractament i emmagatzematge de le dades rebudes pel GPS.
6 Visualització de la informació del tràfic.

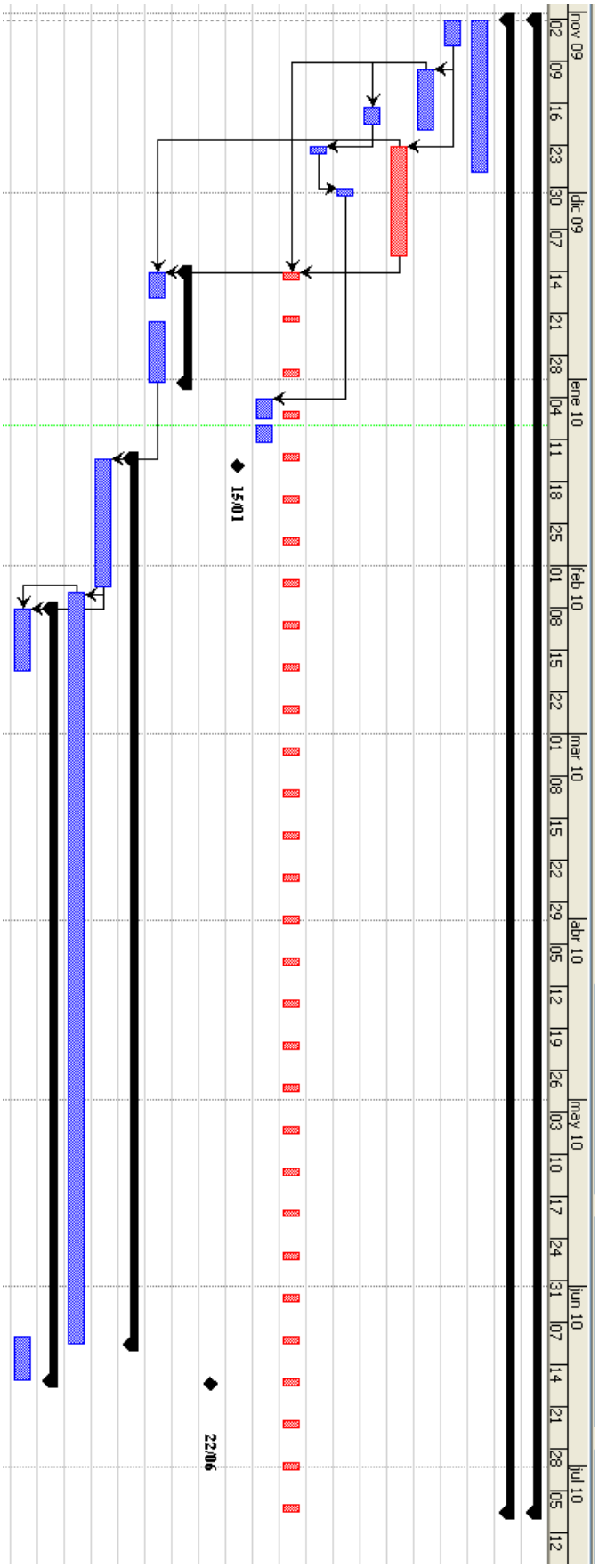
1.3 Organització

1.3.1 Planificació inicial del temps de treball

En un principi es va realitzar una planificació temporal aproximada del treball que comportaria el projecte dividit en tasques i subtasques.

		Nombre	Duración	Inicio	Terminado	Predecesores
1		**Anàlisi de requeriments**	**179 days?**	**2/11/09 8:00**	**8/07/10 17:00**	
2		**Documentació**	**179 days?**	**2/11/09 8:00**	**8/07/10 17:00**	
3		Informació Location Based Services	20 days?	2/11/09 8:00	27/11/09 17:00	
4		Informació Projectes similars	5 days?	2/11/09 8:00	6/11/09 17:00	
5		Informació Aplicació WM GPS	9 days?	10/11/09 8:00	20/11/09 17:00	4
6		Informació API Google Maps	15 days?	23/11/09 8:00	11/12/09 17:00	4
7		Informació trames NMEA data	3 days	16/11/09 16:00	19/11/09 16:00	5SS
8		Informació implementació BBDD	2 days?	30/11/09 8:00	1/12/09 17:00	9
9		Informació transmissió de Dades	2 days?	23/11/09 8:00	24/11/09 17:00	7
10		Informació blogs especialitzats Google Maps	149 days?	14/12/09 8:00	8/07/10 17:00	5SS;6
11		Redactat informe previ	6 days	4/01/10 8:00	11/01/10 17:00	8
12		Lliurament Informe Previ	1 day?	15/01/10 8:00	15/01/10 17:00	
13		Lliurament Memoria	4 days?	17/06/10 7:00	22/06/10 17:00	
14		**Disseny**	**15 days?**	**14/12/09 8:00**	**1/01/10 17:00**	
15		Models BBDD, LLenguatges programació	15 days?	14/12/09 8:00	1/01/10 17:00	6SS;8
16		**Implementació**	**106 days?**	**14/01/10 8:00**	**10/06/10 17:00**	
17		Codificació part servidor (rebre trames GPS)	16 days?	14/01/10 8:00	4/02/10 17:00	15
18		Codifiació part client/servidor	90 days?	5/02/10 8:00	10/06/10 17:00	17
19		**Test**	**93 days?**	**8/02/10 8:00**	**16/06/10 17:00**	
20		Prova de les implementacions	93 days?	8/02/10 8:00	16/06/10 17:00	17;18SS

I vaig obtenir el següent diagrama de Gantt relacionat amb les tasques:

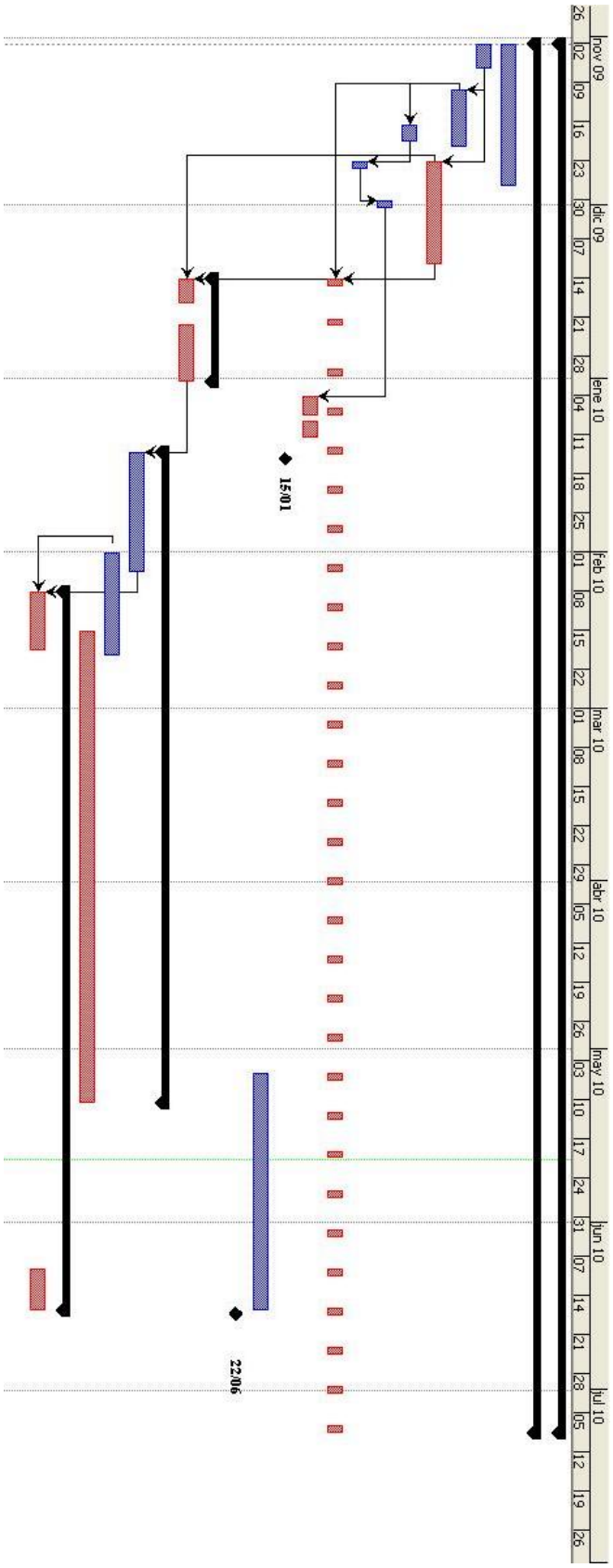


1.3.2 Planificació real del temps de treball

Finalment respectant la majoria de fases i seguint de forma bastant fidel l'organització hi ha hagut alguna variació i ampliació.

		Nombre	Duración	Inicio	Terminado	Predecesores
1		**Anàlisi de requeriments**	**179 days?**	**2/11/09 8:00**	**8/07/10 17:00**	
2		**Documentació**	**179 days?**	**2/11/09 8:00**	**8/07/10 17:00**	
3		Informació Sistemes Informació Geogràfica	20 days?	2/11/09 8:00	27/11/09 17:00	
4		Informació Projectes similars	5 days?	2/11/09 8:00	6/11/09 17:00	
5		Informació Aplicació WM GPS	9 days?	10/11/09 8:00	20/11/09 17:00	4
6		Informació API Google Maps	15 days?	23/11/09 8:00	11/12/09 17:00	4
7		Informació trames NMEA data	3 days	16/11/09 16:00	19/11/09 16:00	5SS
8		Informació implementació BBDD	2 days?	30/11/09 8:00	1/12/09 17:00	9
9		Informació transmissió de Dades	2 days?	23/11/09 8:00	24/11/09 17:00	7
10		Informació blogs especialitzats Google Maps	149 days?	14/12/09 8:00	8/07/10 17:00	5SS;6
11		Redactat informe previ	6 days	4/01/10 8:00	11/01/10 17:00	8
12		Lliurament Informe Previ	1 day?	15/01/10 8:00	15/01/10 17:00	
13		Redacció Memòria	31 days?	5/05/10 8:00	16/06/10 17:00	
14		Lliurament Memòria	4 days?	17/06/10 8:00	22/06/10 17:00	
15		**Disseny**	**15 days?**	**14/12/09 8:00**	**1/01/10 17:00**	
16		Models BBDD, LLenguatges programació	15 days?	14/12/09 8:00	1/01/10 17:00	6SS;8
17		**Implementació**	**83 days?**	**14/01/10 8:00**	**10/05/10 17:00**	
18		Codificació part servidor (rebre trames GPS)	16 days?	14/01/10 8:00	4/02/10 17:00	16
19		Codifiació part servidor	15 days?	30/01/10 8:00	19/02/10 17:00	
20		Codifiació part client	61 days?	15/02/10 8:00	10/05/10 17:00	
21		**Test**	**93 days?**	**8/02/10 8:00**	**16/06/10 17:00**	
22		Prova de les implementacions	93 days?	8/02/10 8:00	16/06/10 17:00	18;19SS

I he obtingut el següent diagrama de Gantt:



2 L’Estat de l’art

2.1 La Geolocalització

La ***Geolocalització***, també anomenada Georreferenciació, és el terme que fa referència al posicionament de la informació en una localització geogràfica.

La Geolocalització s’ha introduït en les nostres vides i en els nostres sistemes d’informació de vàries maneres. Actualment, tots estem familiaritzats amb la informació de diversos objectes que representen un espai geogràfic i les seves característiques sobre la superfície de la terra, com per exemple, mapes i globus terraquis. Utilitzem serveis en línia que ens diuen on està situada una certa adreça, a quina distància està situat un lloc d’un altre i quina ruta hem de prendre per poder arribar-hi. Cada cop més, científics, enginyers, governs, empreses i consultors polítics incorporen *Sistemes d’informació Geogràfica* (GIS) per emmagatzemar i analitzar dades georreferenciades, permetent descobrir patrons de distribució geogràfica que donen suport als qui planifiquen i prenen decisions. En general, estem vivint en una edat on tecnologia i informació estan enriquint la nostra comprensió de la geografica i els seus efectes en les nostres vides i les vides dels altres.

L’àmbit de la geolocalització inclou significats informals referint-se a localitzacions. Habitualment utilitzem topònims i, per representacions formals, les coordenades de longitud i latitud a més d’altres sistemes de referenciació espacial, emprats en la creació de mapes i de navegació. Les representacions formals són empremtes geoespacials – anomenades així perquè mostren sobre el mapa de la superfície de la Terra un punt o àrea concreta on quelcom és localitzat. Aquestes empremtes són les bases pels càlculs matemàtics de la distància i direcció i també per definir relacions espacials. Tenint les empremtes, podem mostrar on estan situats els llocs en un mapa i com aquests s’identifiquen amb districtes administratius, línies de costes, rius, muntanyes o qualsevol punt geogràfic d’interès. Els topònims sense cap tipus de referència espacial no ens ho permeten fer.

Per poder situar i veure els patrons geogràfics i les associacions de la informació necessitem una traducció entre la representació geogràfica formal i informal. Aquest motiu és el que dóna importància als diccionaris geogràfics (gazetteers) que relacionen empremtes geoespacials amb el nom dels llocs, com il·lustra la figura 2.1.

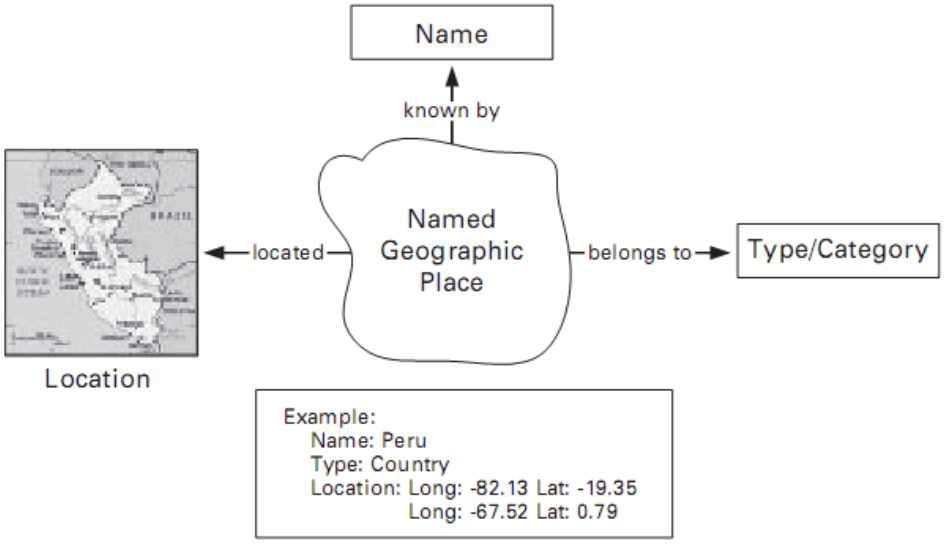


Figura 2.1: Entrada d'un diccionari geogràfic

El procés de traduir les dades identificades per la seva adreça, a dades basades en un sistema de coordenades georeferenciat s'anomena: **geocoding**.

2.2 Els mapes

Els mapes són probablement la forma més familiar de representació d'informació geoespacial d'objectes, especialment com a mapes impresos, però cada cop més com a mapes digitals mostrats sobre un software visualitzador.

Els mapes són informes, representacions o visualitzacions de les dades subjacents en una superfície de dues dimensions (plana). Són productes cartogràfics, que contenen característiques per un propòsit particular i creats segons els principis de la cartografia en termes d'abtracció, simbologia, colors, etiquetes, línies, estils i llegendes. Avui en dia, la visualització dels mapes són incorporats en línia amb una interfície per l'usuari i com a un component integrat en les unitats de sistemes de posicionament global (GPS) portàtils i altres aplicacions.

Els mapes són el resultat final de la reunió (l'exploració, l'agrimensura i la detecció a distància), processament i anàlisi dels punts georreferenciats que documenten la capa de la terra i les seves característiques del paisatge.

El 1854 a Londres, un estudi, fet pel Dr.John Snow, va permetre identificar la causa de l'epidèmia de còlera d'un barri, que no era altra que una bomba d'aigua que abastia una cantonada dels carrers Cambridge i Broad, marcant amb punts els edificis on la gent havia mort. Situar la informació al mapa li va permetre documentar els fets de forma més convincent als líders de la comunitat que taules plenes de dades. Actualment, mitjançant aquesta idea és possible descobrir i combinar dades que pertanyen a àrees geogràfiques concretes i utilitzant

tecnologies GIS, que produeixin mapes que mostrin patrons de distribucions geogràfiques impossibles de documentar anteriorment de forma més clara.

2.2.1 Superposicions i Tipus de dades.

Els mapes estan creats per superposicions d'un conjunt de capes amb característiques pròpies, tal i com capes hidrogràfiques, capes de transport per autopistes i autovies i capes que contenen altres conjunts de dades com patrons de meteorologia. Els visualitzadors de mapes en línia solen donar l'opció d' escollir quina capa es vol mostrar en la vista actual.

Les capes contenen o informació cartogràfica (per exemple: les línies frontereres) representades com un vector de dades o dades amb atributs (com per exemple: la població, altitud, etiquetes, símbols...) representades com un raster de dades.

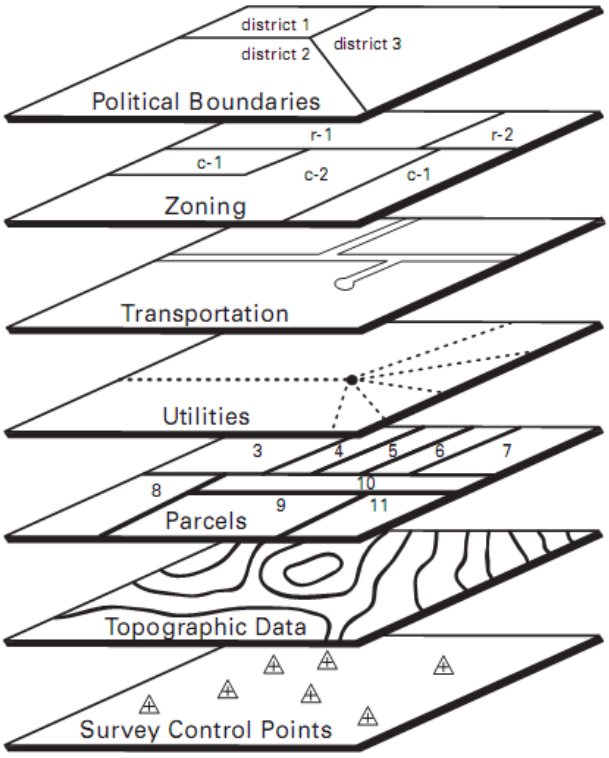


Figura 2.2: Exemple d'un conjunt de capes.

Els **raster** són representats per una matriu o quadrícula de cel·les. Cada cel·la (també coneguda com a pixel) té un o més valors discrets assignats en ella (com una imatge). Aquests valors poden representar categories on la localització és una àrea urbana, suburbana o rural, la densitat de població, la temperatura de l'aigua del mar, etc...Les representacions cartogràfiques dels conjunts de dades del raster solen utilitzar colors, ombres i patrons per mostrar variacions en els seus valors.

Els **vectors** són emmagatzemats com punts desordenats o sèries ordenades de punts. Un punt és una parella de coordenades (longitud i latitud). Una sèrie de punts que comencen i acaben amb el mateix punt i envolten una àrea, són anomenats Polígons.
Exemples de vectors són els arxius com Digital Line Graphs (DLGs) i TIGER/Line.

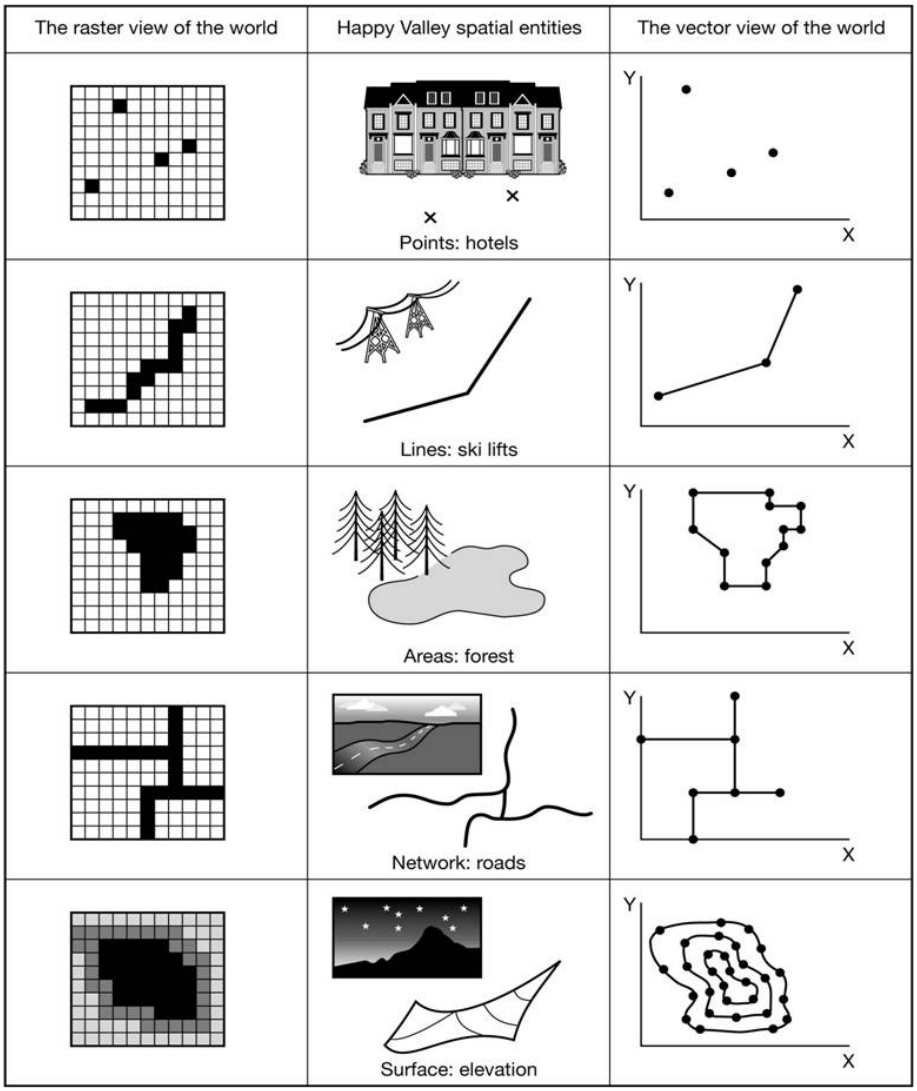


Figura 2.3: Taula comparativa entre rasters i vectors

2.3 Les coordenades i el sistema de coordenades

La forma de la Terra és el·lipsoidal. Si aquesta fos una esfera perfecta, desenvolupar mètodes per realitzar el procés de geocoding seria menys complex. No hi ha un únic model matemàtic de la forma de la Terra i de la seva mida sinó que hi ha diversos models per diferents propòsits. La ciència que determina la mida i la forma de la Terra s'anomea geodesia.

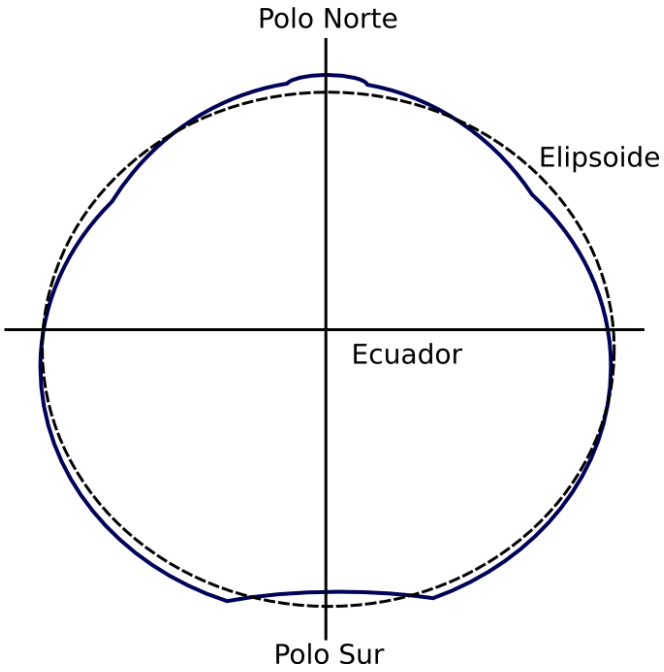


Figura 2.4: Forma el·lipsoidal de la Terra

Hi ha molts sistemes basats en la georreferenciació de la Terra. Tots tenen els components necessaris d'origen: un tipus d'unitat de distància, dos eixos per localitzar una posició en un espai de dues o tres dimensons. El sistema més familiar de georreferenciació de la superfície de la Terra utilitza la longitud i la latitud com a coordenades. L'origen per la latitud és l'equador (0º latitud) i l'origen per la longitud és el meridià principal (0º longitud) que va de pol a pol a través de la localització original de l'observatori Royal Greenwich, també conegut com el meridià de Greenwich. El rang de valors per les latituds és de 90º (des de l'equador fins als pols nord i sud en ambdues direccions).

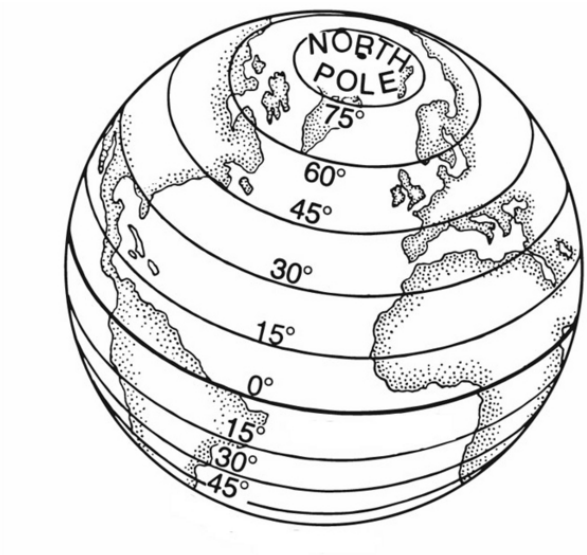


Figura 2.5: Latituds sobre la Terra

El rang de valors per la longitud és de 180º des del principal meridià, conegut com la *International Date Line.*

Quan un sistema de coordenades esfèric que representa la Terra és projectat sobre una superfície plana, el resultat és una representació gràfica dels eixos *x* i *y* que utilitzen les coordenades Cartesianes basades en la geometria desenvolupada per René Descartes. Els graus de longitud van al llarg de l' eix horitzonal de les x i els graus de la latitud van al llarg de l'eix vertical de les y.

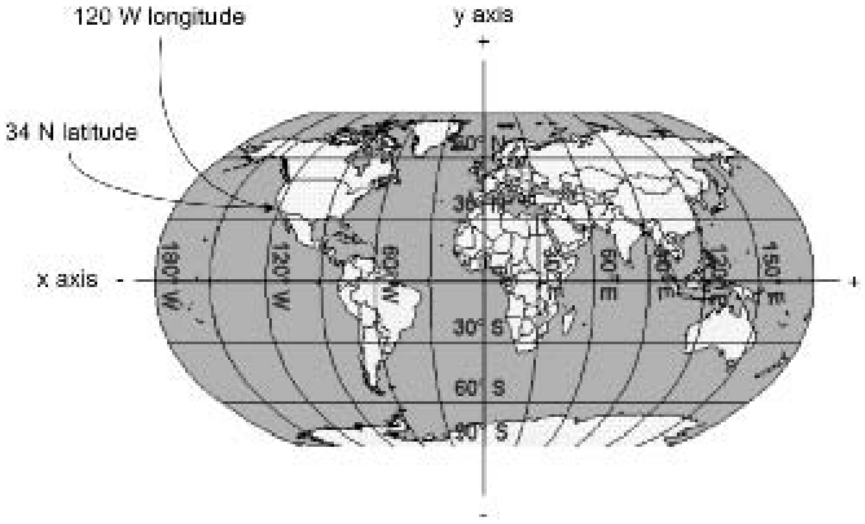


Figura 2.6: Correspondència dels eixos x,y amb la longitud i latitud

Les coordenades de latitud i longitud poden ser representades utilitzan graus, minuts, segons (denotat com ggmmss) o graus decimals (denotat com gg.g). Hi ha dos termes que es poden trobar per aquests dos esquemes. Els graus decimals són anomenats coordenades centesimals, significa que són divisions de cents. L'esquema de grau, minut i segon és anomenat coordenades sexagesimals, i vol dir que són divisions basades en el número 60.

Per la representació de grau, minut i segon, la posició EAST (Est) o WEST (Oest) del principal meridià o NORTH (Nord) o SOUTH (Sud) de l'equador es denotarà amb una lletra abans o després de la cadena de números de la direcció (N,S,E,W).

Per la representació de graus decimals, la convenció és: considerar la posició EST del principal meridià ó NORD de l'equador com a positiva (+) i la posició OEST del principal meridià i SUD de l'equador com a negativa (-). El mateix criteri en els gràfics d'eixos *x* i *y*.

Com a exemple de les dues representacions, mostro el mateix punt que fa referència a la localització de Goleta (Califòrnia) en graus decimals i grau,minut i segon:

- Longitud:
  - Graus decimals : *-119.8266*
  - Grau, minut, segons: *119º49´36´´W*
- Latitud:
  - Graus decimals: *34.4358*
  - Grau, minut, segons: *34º26´8´´N*

2.3.1 Altres sistemes de referenciació espacial:

2.3.1.1 GML (Geography Markup Language):

Aquest sistema permet representar diferents tipus d'empremtes i, a més a més, modelar, transportar i emmagatzemar informació geogràfica.

Alguns tipus d'empremtes són:

- Punt
- Capsa
- LinearString
- LinearRing
- Polígon
- MultiPunt
- MultiLineString
- MultiPolygon
- MultiGeometry

Per exemple, per representar l'empremta més siemple, és a dir, un punt, utilitzem un parell de valors de longitud, latitud.

En graus decimals, el punt és escrit com un valor separat per coma. Per exemple: -120,34 representa un punt 120ª a l'Oest del meridià principal i 34º al Nord de l'equador.

Connectant seqüències de punts, podem anar formant diferents formes geomètriques.

GML especifica que les coordenades geomètricques d'una figura poden ser codificades en XML.

Un exemple d'una capsa:

```
<gml:box>
     <gml:coord>
          <gml:X>-116</gml:X>
          <gml:Y>36</gml:Y>
     </gml:coord>
     <gml:coord>
          <gml:X>-115</gml:X>
          <gml:Y>37</gml:Y>
     </gml:coord>
</gml:box>
```

2.3.1.2 Sistema UTM (Universal Transversal de Mercator):

És un sistema de coordenades basat en la projecció cartogràfica transversa de Mercator, que es construeix com la projecció de Mercator normal però, en comptes de fer-la tangent a l'Equador, es fa la tangent a un meridià. A diferència del sistema de coordenades geogràfic, expresades en longitud i latitud, les magnituts en el sistema UTM s'expressen en metres únicament al nivell del mar que és la base de la projecció de l'el·lipsoide de referència.

La Terra és dividida en zones de 6º que van des de pol a pol i numerats d'Oest a Est, de l'1 al 60, començant per la Internatiola Date line.

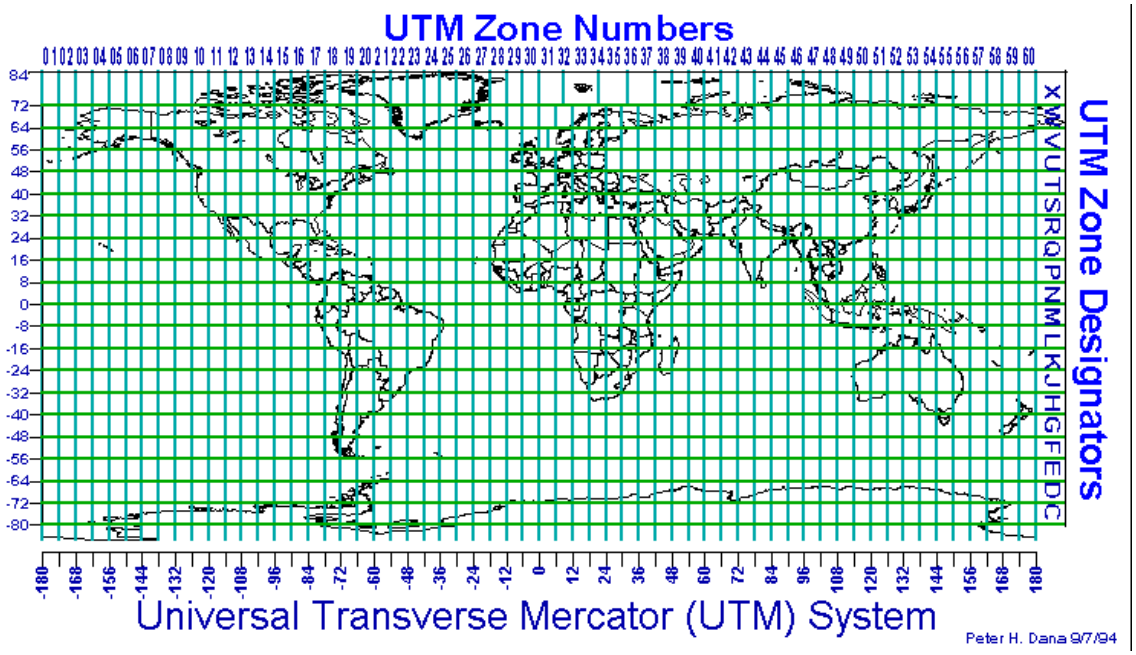


Figura 2.7: Sistema UTM

2.3.1.3 C-squares (Concise Spatial Query and Representation System):

És un sistema de geocodis (actualment un tipus de malla global) que proveeix una base per una simple indexació espacial de dades o de característiques geogràfiques. Va ser desenvolupat per Tony Rees del CSIRO Marine Research en 2001-2002 i escrit en 2003.

El sistema de notació del c-squares incorpora una compacta codificació de les coordenades de latitud i longitud en codi c-square, llegible pels humans, i que pot ser utilitzat per búsqueda espacial o per mostrar en aplicacions de mapes.

En aquest sistema, una sèrie de malles de quadrats són utilitzades per representar àrees de cobertura geogràfica. Els punts localitzats poden ser representats utilitzant un codi pel quadrat on resideix i una notació de tira de text és la representació d'una àrea. La malla del sistema c-squares es basa en la successiva subdivisió de la WMO (World Meteorological Organization) en quadrats de 10x10 graus per la representació de la superfície de la Terra.

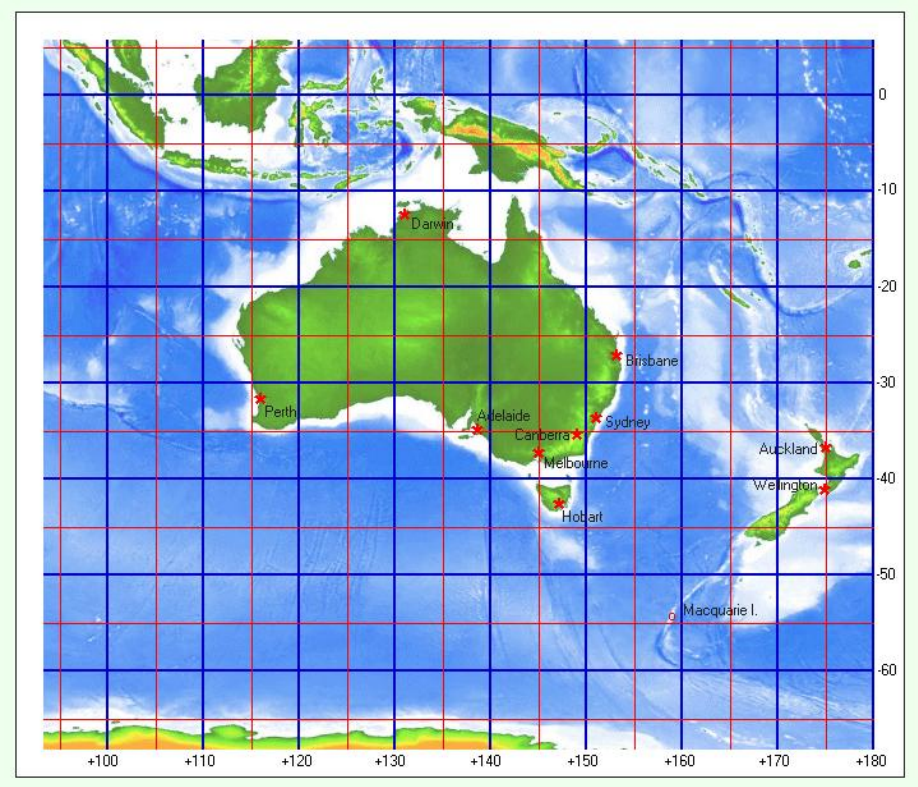


Figura 2.8: C-squares: Els quadrats blaus representen 10 graus i els vermells 5 graus.

2.4 El sistema GPS:

El Sistema de Posicionament Global (GPS) és un sistema de navegació per satèl·lit (GNSS: Global Navigation Satellite System) basat en l'espai que proveeix localitzacions fidedignes i informació horària en totes les condicions meteorològiques, a qualsevol hora i a qualsevol lloc de la Terra on hi hagi una línia sense obstruccions de cobertura de quatre o més satèl·lits GPS. El govern dels EEUU és l'encarregat del manteniment del sistema i és de lliure accés per qualsevol persona amb un receptor GPS.

El sistema va ser creat i desenvolupat pel Departament Americà de Defensa (DOD) i va ser originalment basat i posat en marxa amb 24 satèl·lits.

Va ser establert el 1973 per reduïr el gran nombre de guies de navegació i per superar les limitacions de sistemes de navegació anteriors.

2.4.1 Estructura del sistema:

El GPS està format per tres parts: el segment d'espai, el segment de control i el segment d'usuari. Les Forces Aèrees dels EEUU desenvolupen, s'encarreguen del manteniment i operabilitat dels segments d'espai i control. Els satèl·lits GPS emeten senyals des de l'espai, les quals cada receptor de GPS utilitza per a calcular les seves tres-dimensions (latitud, longitud i altitud) més el temps horari actual.

El segment de l'espai es compon de 24 a 32 satèl·lits en una òrbita mitjana de la Terra i els coets per llançar-los a l'òrbita.
El segment de control es compon d'una estació de control mestra, d'una estació de control alternativa i d'un host d' antenes de terra i estacions monitoritzadores dedicades i compartides.
El segment d'usuari es compon de centenars de milers d'usuaris d' EEUU i aliats militars del servei segur i precís del GPS, de deu milions d' usuaris civils, comercials i científics del servei estàndar de Posicionament.

2.4.2 Aplicacions:

El GPS s'ha convertit en una gran ajuda per la navegació assistida[guiada] i una eina molt important per crear mapes per l'agrimensura, pel comerç, usos científics, seguiment de vehicles, supervisió i hobbies com el geocaching i el waymarking.
La precisió de referència horària que ens proporciona el GPS és utilitzada en moltes aplicacions incloses aplicacions científiques, estudis de terratrèmols i com una font de sincronització per a protocols de xarxes de telefonia mòbil. A més a més, el GPS, segons el website del gps.gov, s'ha convertit en el pilar principal dels sistemes de transport de tot el món, proveïnt navegació per cel, mar i terra. Els serveis d'emergència en catàstrofes depenen del GPS per la localització i les capacitats horàries en les seves misions per salvar vides. La precisió horària que proporciona el GPS és utilitzada en activitats com la banca, telefonia mòbil, i inclús en xarxes elèctriques. Granjers, agrimensors, geòlegs i molts altres porten a cap els seus treballs de forma més eficient, segura, econòmica i precisa utilitzant els senyals lliures i oberts del GPS.

2.4.3 Conceptes bàsics del GPS:

Un receptor de GPS calcula la seva posició amb precisió en el moment en què els senyals són enviats des dels satèl·lits GPS que hi ha sobre la Terra. La precisió del dispositiu GPS oscil·la entre 1-5 metres d'error.
Cada satèl·lit continuament transmet missatges que inclouen:

- L'hora en què el missatge va ser transmés.
- Informació precisa de l'òrbita. (els ephemeris)
- El sistema general de salut i estimatiu de les òrbites de tots els satèl·lits GPS (l'almanac)

El receptor utilitza els missatges que rep per determinar el temps de trànsit de cada missatge i calcula la distància per a cada satèl·lit. Aquestes distàncies amb la localització dels satèl·lits són utilitzades per la possible ajuda del càlcul de la trilateració, depenen de quin algorisme s'utilitza per calcular la posició del receptor. La posició és mostrada en un mapa en moviment o

mostrant solament la latitud,longitud i elevació. Algunes unitats GPS mostren informació derivada com la direcció i la velocitat, calculades del canvi de posicions.
Tres satèl·lits poden semblar suficients per trobar la posició, ja que l'espai té tres dimensions i una posició propera a la superfície de la terra pot ser suposada.
No obstant això, inclús un petit error en el rellotge (oscil·lador de vidre) multiplicat per la velocitat de la llum – a la qual es propaguen els senyals – dóna com a resultat un gran error de posicionament. Per tant, els receptors utilitzen quatre o més satèl·lits per trobar la localització i l'hora. El càlcul precís de l'hora és ocult en moltes aplicacions GPS, les quals solament utilitzen la localització. Poques aplicacions GPS especialitzades utilitzen el temps, les que l'empren inclouen el temps de transferència, la coordinació del tràfic dels senyals i la sincronització de les estacions base de telefonia mòbil.
Encara que quatre satèl·lits siguin requerits per operacions normals, hi ha casos especials que se n'utilitzen menys. Si una variable és coneguda, un receptor pot determinar la seva posició solament utilitzant tres satèl·lits. Per exemple, un vaixell o un avió solen coneixer la seva elevació. Alguns receptors GPS solen utilitzar pistes addicionals o assumpcions (tal i com reutilitzar l'última altitud coneguda, navegació per estimació [dead reckoning], navegació inercial, o incloent informació de l'ordinador de bord del vehicle) per donar una posició estimada quan menys de quatre satèl·lits són visibles.

2.4.3.1 Càlcul de la posició:

Per mostrar una introducció de com funciona el receptor GPS, ignorarem els errors. Utilitzant receptors amb un mínim de quatre satèl·lits visibles, un receptor és capaç de determinar les hores enviades i el satèl·lit corresponent a aquestes hores enviades. Els components x,y,z de posició i l'hora enviada són designats com $[x_i,y_i,z_i,t_i]$, on l'índex *i* és el número de satèl·lit i pot tenir el valor 1,2,3 ó 4. Sabent l'hora indicada al missatge que va ser rebut a t_r, el receptor GPS pot calcular el temps de trànsit del missatge com $(t_r - t_i)$. Assumint que el missatge viatja a la velocitat de la llum *c*, la distància viatjada o pseudorang, *pi* pot ser calculat com $(t_r - t_i)c$.
La posició d'un satèl·lit o pseudorang defineix una esfera, centrada en el satèl·lit amb radi igual al pseudorang. La posició del receptor és qualsevol lloc sobre la superfície de l'esfera. Així amb quatre satèl·lits, la posició del receptor GPS és a la intersecció de les superfícies de quatre esferes o aprop d'aquesta. En un cas ideal sense errors, el receptor GPS seria situat a la intersecció exacta de les quatre superfícies.

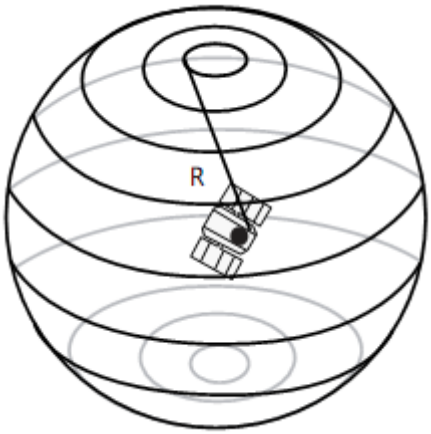


Figura 2.9: Usuari localitzat sobre la superfície de l'esfera

Si la superfície de dos esferes s'interseca a més d'un punt, aquestes s'intersequen en un cercle. La trilateració descriu matemàticament el mètode per determinar la intersecció de tres superfícies d'esferes, donats els centres i els radis de les tres esferes.

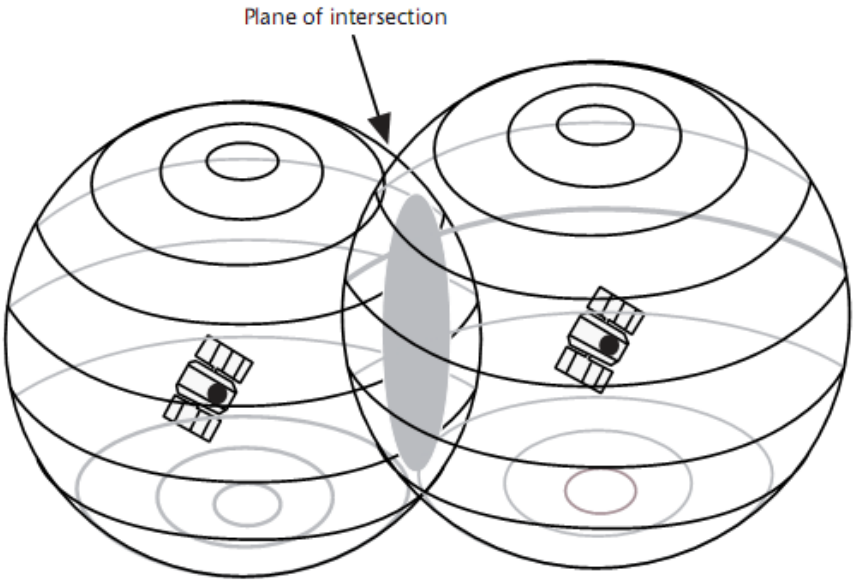
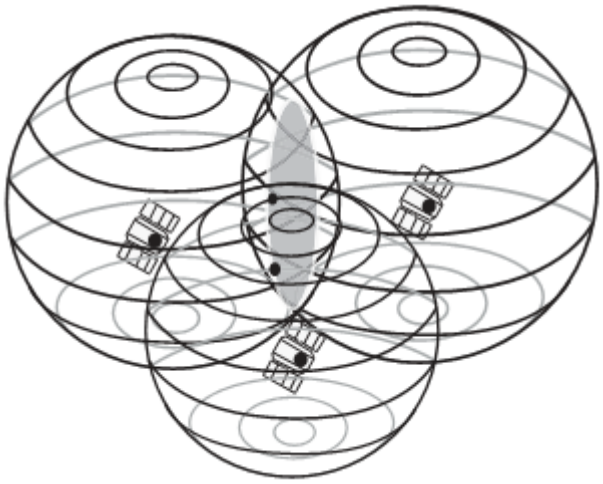
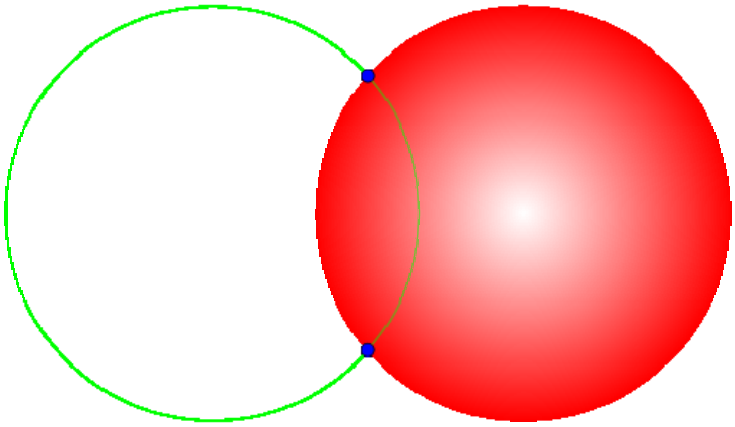


Figura 2.10: Usuari localitzat sobre el perimetre del cercle ombrejat.

La intersecció de la tercera superfície esfèrica amb les dos primeres serà la seva intersecció amb aquest cercle. En la majoria dels casos pràctics això significa que s'intersequen en dos punts.



(a)



(b)

Figura 2.11 : (a) Usuari localitzat en un dels dos punts del cercle ombrejat.
(b) Intersecció d'una superfície esfèrica amb un cercle. La intersecció és marcada amb dos punts.

Per automòbils o altres vehicles que viatgen a prop de la terra, la posició correcta del receptor GPS és la intersecció més propera amb la superfície de la Terra. Per a vehicles espacials, la intersecció més llunyana seria la posició correcta.
La posició correcta pel receptor GPS també és la intersecció més propera a la superfície de l'esfera corresponent al quart satèl·lit.

2.4.3.2 Corregint el rellotge del receptor GPS:

Un dels errors més significants de la font és el rellotge del receptor GPS (oscil·lador de vidre). Degut a la gran llargada del valor de la velocitat de la llum, *c*, la distància estimada des del receptor GPS als satèl·lits, és a dir, el pseudorang, és molt sensible als errors en el rellotge del

receptor GPS. Això suggereix una extremada precisió i un cost elevat en el rellotge del receptor GPS. Per altra banda, molts fabricants de receptors GPS prefereixen fabricar més barat per a mercats massius. La solució a aquest dilema està basada en la forma en què es calcula la intersecció de les superfícies de les esferes.

És probable que la superfície de les tres esferes s'intersequin, ja que el cercle d'intersecció de les dues esferes normalment és bastant gran i, per tant, la tercera esfera, segurament, s'intersecarà amb aquest gran cercle. És poc probable que la superfície de l'esfera corresponent al quart satèl·lit s'intersequi amb algún dels dos punts resultants de la intersecció de les tres primeres esferes, ja que qualsevol error del rellotge podria causar que es perdes un punt d'intersecció. No obstant això, la distància des de la localització estimada i vàlida del receptor GPS a la superfície de la quarta esfera podria ser utilitzada per calcular la correcció del rellotge. Anem a denotar r_4 com la distància des de la localització estimada i vàlida del receptor GPS fins al quart satèl·lit i denotarem com p_4 el pseudorang del quart satèl·lit. Aleshores $da = r_4 - p_4$ és la distància des de la posició calculada del receptor GPS fins a la superfície de l'esfera corresponent al quart satèl·lit. Així el quocient, b = da/c, ens proporciona: *(el temps correcte) – (el temps indicat pel rellotge del receptor)* i el rellotge del receptor pot ser avançat si *b* és positu o endarretir si *b* és negatiu. No obstant això, seria bó tenir en compte que una funció més complexa de *da* seria necessària per estimar l'error de temps en un algorisme iteratiu.

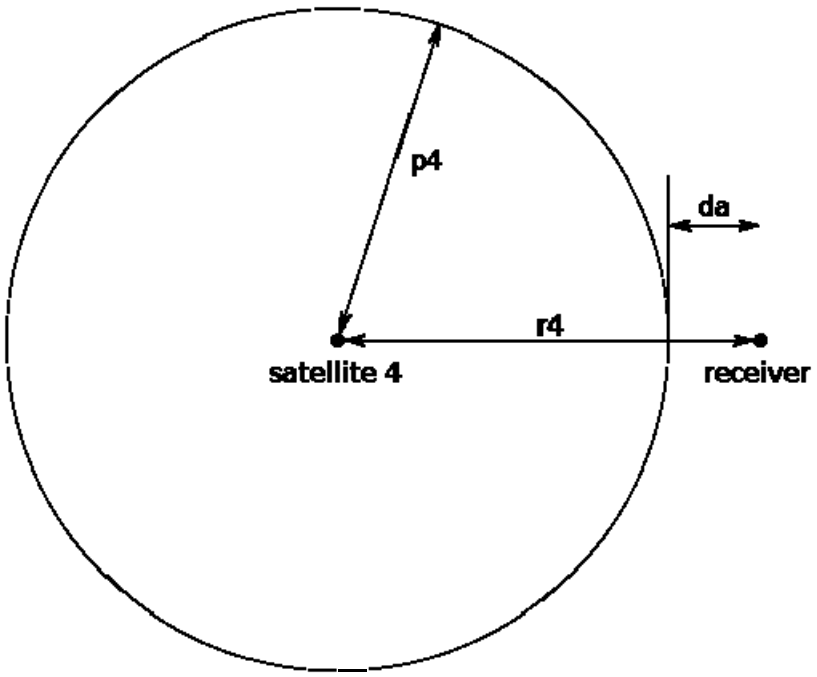


Figura 2.12: Diagrama descrivint el satèl·lit 4, l'esfera, p_4 , r_4 i d_4

En aquest punt, finalitza les explicacions del posicionament del receptor. En aquest projecte no entrarem en càlculs més avançats perque no seran necessaris per l'àmbit del projecte.

2.4.4 Transmissió del missatge:

Molts receptors de GPS solen transmetre les dades de posició a un PC o a un altre dispositiu utilitzant el protocol NMEA.

El protocol NMEA és una especificació per la comunicació entre dispositius marítims electrònics tal i com, sondes nàutiques, sonars, anemòmetres, girocompàs, autopilot, receptors GPS i molts altres tipus d'instruments. Va ser definit i està controlat per National Marine Electronics Association.

L'estàndard NMEA utilitza una simple comunicació sèrie per transmetre una frase a un o més escoltes. Una trama NMEA utilitza tots el seus caràcters en ASCII.

Contingut d'una sentència (frase) enviada a través del protocol NMEA:

- Cada missatge comença pel símbol del dollar ($)
- Els següents cinc caràcters identifiquen: l'emisor utilitzant dos caràcters i el tipus de missatge (tres caràcters).
- Tots els camps són delimitats per una coma.
- Els camps on la dada no està disponible contenen NUL bytes.
- El primer caràcter que inmediatament segueix l'últim camp és un asterisc.
- L'asterisc és seguit per dos dígits de comprovació en format hexadecimal. Els números de comprovació són una OR exclusiva sobre tots els caràcters entre el dollar ($) i l'asterisc (*).

El codi de la sentència més important d'una trama és:

- **GPRMC**: GPS/TRANSIT Recommended Minimum specific data.

El missatge GPRMC conté les dades mínimes per a poder realitzar la localització d'un receptor GPS.

El seu format, camps i contingut de cada un d'ells és:

$GPRMC,hhmmss.ss,A,llll.ll,a,yyyyy.yy,a,x.x,x.x,ddmmyy,x.x,a,m*hh
$GPRMC,1_____,2,3___,4,5_____,6,7__,8__,9_____,10,11,12*13

Número de camp i el seu contingut:

1. UTC hora de la localització
2. Dades d'estat (A=Posició vàlida, V=Avís de la navegació del receptor)
3. Latitud de la localització
4. Nord o Sud de la latitud
5. Longitud de la localització
6. Est or West (Oest) de la longitud
7. Velocitat en nusos.

8. Track made good in degrees True
9. UTC data(dia, mes, any) de la localització
10. Variació magnètica en graus.
11. Est or West (Oest) de la variació magnètica.
12. Indicador de modus, (A=Autònom, D=Diferencial, E=Estimat, N=Dades No vàlides).
13. Números de comprovació (Checksum)

2.5 Sistema de seguiment de vehicles:

El sistema de seguiment de vehicles combina la instal·lació d'un dispositiu electrònic en un vehicle o flota de vehicles, amb un software per a ordinador que permet al propietari, o a un tercer, seguir la localització del vehicle, recollint dades en el procés. Els sistemes de seguiment modern utilitzen tecnologia Global Positioning System (GPS) per a localitzar el vehicle, però hi ha altres tipus de tecnologies que poden ser utilitzades. La informació del vehicle pot ser visualitzada en un mapa electrònic via Internet o en un software especialitzat. Actualment, cada cop més, les autoritats dels transports públics estan adoptant aquest tipus de tecnologia, sobretot en ciutats amb una gran extensió.

Hi ha dos tipus de seguiment de vehicles:

Podem classificar els tipus de seguiment segons el tipus de dispositiu que utilitzen. Aquests són els Passius i els Actius.
Els sistemes de seguiment Passiu solen emmagatzemar en el mateix dispositiu les dades de la localització, velocitat i, a vegades, events com l'encesa o apagada del vehicle. Un cop el vehicle torna a un determinat punt, el dispositiu s'extreu i la informació és descarregada a un ordinador per a la seva posterior evaluació.
Els sistemes de seguiment Passiu recullen les dades de la localització, velocitat i aturades efectuades. Aquesta informació és transmesa en temps-real via telèfon mòbil o xarxes per satèl·lit a un ordinador o a un centre de dades per la seva evaluació.
També es pot combinar els dos tipus de seguiment: quan una xarxa de telefonia mòbil està disponible el dispositiu transmet les dades al servidor, quan la xarxa no està disponible el dispositiu emmagatzema les dades en una memòria interna i transmetrà les dades al servidor quan la xarxa de telefonia mòbil torni a estar disponible.

2.5.1 Aplicacions:

Els sistemes de seguiment de vehicles són comuntment utilitzats per administradors de flotes de vehicles per a funcions com control de rutes, expedicions, informació de bord i seguretat. Junt amb els administradors de flotes, agències del trànsit urbà utilitzen la tecnologia per

diversos propòsits, incloent la monitorització i planificació del servei de bus. Segons la previsió de l'Associació de Transport Public Americà, a finals del 2010, prop de la meitat del trànsit de busos en EEUU ja utilitzarà un sistema de seguiment de vehicles amb tecnologia GPS que alerti de forma automàtica les parades.

Altres aplicacions inclouen la monitorització de la conducció dels empleats d'una empresa o pares amb un conductors joves.

Els sistemes de seguiment també són utilitzats per a prevenció de robatoris i la recuperació d'aquests. La policia pot seguir el senyal que emet el vehicle i seguir i localitzar el vehicle robat. Alguns sistemes de seguiment integren diversos sistemes de seguretat com l'enviament d'alertes al telefon mòbil o el correu si el vehicle s'ha mogut sense cap autorització, o quan entra o abandona una certa zona (geofence).

Podem resumir les aplicacions del sistema de seguiment de vehicles en:

- Recuperació de vehicles robats: seguir la localització d'un vehicle robat per a poder-lo recuperar.
- Administració de Flotes: seguir la localització de les flotes per a poder controlar els temps d'entrega.
- Seguiment de béns: seguiment de vehicles que transporten bens valuosos.
- Administració de serveis: Serveis de reparació o manteniment utilitzen el seguiment dels vehicles dels seus empleats per a poder planificar les seves visites segons la seva localització.
- Seguiment del trànsit: es realitza un seguiment del temps per a poder controlar que els béns no s'aturen durant la ruta i així assegurar la seguretat.
- Vigilància: instal·lar el sistema de seguiment en un vehicle on es vulgui controlar els moviments.

3 Estructura de l'aplicació

En aquest apartat es pretén mostrar una visió general del funcionament de tota l'aplicació, és a dir, la integració de dos mòduls: de captació i de processament. Mostraré un petit esquema (Figura 3.1) per veure quina és la seva funció, com treballen i la relació que tenen entre ells. Per representar aquest esquema, utilitzaré icones fàcils d'interpretar.

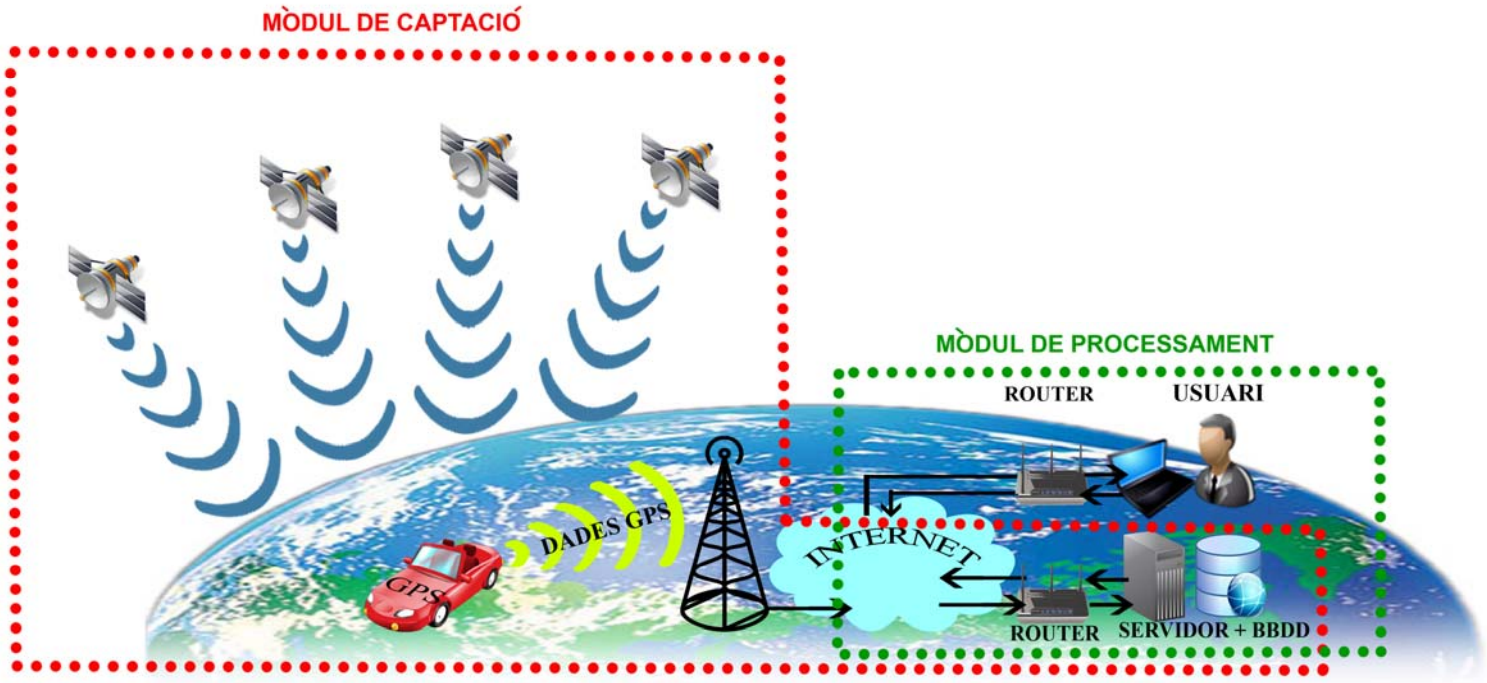


Figura 3.1: Esquema de l'aplicació

L'esquema mostra el funcionament de l'aplicació per al seguiment d'un sol vehicle, però pot ser ampliat a tants vehicles com tinguem configurat el mòdul de captació.

En aquest esquema podem observar el mòdul de captació i el de processament.

El **mòdul de captació**, mitjançant un dispositiu GPS instal·lat en el vehicle a controlar, rep la informació dels satèl·lits GPS i l'envia a través de la xarxa de comunicació sense fils al servidor connectat a Internet. Aquest servidor s'ocupa de trobar les sentències que contenen les dades necessàries pel mòdul de processament, les tracta i les emmagatzema a la Base de Dades.

El **mòdul de processament** a través del navegador web accedeix a l'adreça del servidor, el qual li envia l'aplicació de seguiment del vehicle. L'usuari, utilitzant el navegador, pot visualitzar i controlar tota la informació relacionada amb els vehicles, els polígons i els punts d'interès.

4 Anàlisi de requeriments.

Els requeriments són un conjunt d'idees que el client té sobre què ha de ser el software a desenvolupar. És a dir, són les prestacions del sistema.
Podem classificar els requeriments en dos tipus: requeriments funcionals i requeriments no funcionals.

4.1 Requeriments funcionals:

Els requeriments funcionals ens donen una descripció del comportament del software. Cada requeriment funcional expressa una relació entre les entrades i sortides del sistema, és a dir, especifica les sortides que s'han de produir a partir d'unes determinades entrades i les operacions necessàries per aconseguir això. També s'han d'especificar com s'han de comportar el sistema davant de situacions anormals (entrades invàlides, errors, etc).

Els requeriments funcionals del projecte són:

- **Visualitzar els vehicles sobre el Mapa.**
  - Permetre visualitzar a qualsevol client tots els vehicles i les seves rutes, que estan afegits en la base de dades.
  - Seleccionar un conjunt de punts que pertanyen al vehicle i mostrar-los sobre el mapa, a partir d'una sèrie d'opcions establertes per l'usuari.

- **Visualitzar el tràfic sobre el Mapa.**
  - Permetre a l'usuari mostrar o ocultar la capa amb la informació relacionada del Tràfic, proporcionada per la Direcció General de Tràfic (DGT).
  - A partir de les dades rebudes del servidor de la DGT, mostrem sobre el mapa un seguit d'icones que representen incidències: obres, retencions, càmares de tràfic, estacions meteorològiques, panells informatius. Si cliquem sobre qualsevol d'aquestes icones, obtindrem informació addicional.

- **Visualitzar informació del vehicle.**
  - Permetre a l'usuari mostrar o ocultar el recórrergut d'un vehicle, centrar la seva posició actual en el mapa, mostrar o ocultar la seva icona, visualitzar la ruta per dies i visualitzar un gràfic de velocitat.

- **Crear Ruta.**
    - Permetre a l'usuari crear una ruta a partir d'una adreça o utilitzant els punts d'interès com a fites.
    - A partir d'una adreça introduïda, utilitzem l'API de Google MAPS per realitzar una consulta des del punt on el vehicle està situat actualment fins l'adreça introduïda.
    - A més a més, podem crear una llista ordenable de punts d'interès (fites) i utilitzant l'API de Google Maps realitzem una consulta des del punt on el vehicle està localitzat, passant per tots els punts d'interès de la llista ordenable.
    - Si l'adreça no existeix o bé hi ha més de 23 punts d'interès en la llista ordenable, no és possible realitzar la ruta amb automòbil. Llavors, s'avisa a l'usuari que modifiqui les seves entrades o la ruta.

- **Modificar Opcions i Informació del vehicle.**
    - Permetre a l'usuari modificar les opcions següents del vehicle: nombre de minuts, d'hores, de dies, de setmanes, de mesos o d'anys enrere en què es visualitzarà la ruta, a partir de la data actual o de l'última data rebuda pel vehicle. Veure la ruta completa. Veure la ruta a partir d'un dia en concret. Veure la ruta des d'una data i hora, fins a una altra data i hora especificades. Canviar el sobrenom, el tipus, la marca, el model, la velocitat màxima i el temps per parada del vehicle.
    - A partir de les dades introduïdes per l'usuari, mitjançant calendaris, entrades de text i desplegables, es guarden les opcions introduïdes a la Base de Dades. El programa carregarà les noves dades (punts geogràfics) i mostrarà la nova ruta, segons les opcions introduïdes.

- **Visualitzar la ruta amb Google Maps Street View.**
    - Permetre a l'usuari visualitzar la ruta actual de forma panoràmica com si estiguès a peu de carrer, a través de la finestra creada en Flash per Google Maps Street View. L'usuari tindrà l'opció d'aturar, continuar, tornar a iniciar i canviar la velocitat de reproducció.
    - A partir dels punts que conformen la ruta i, utilitzant l'API de Google Maps, busquem les panoràmiques de cada punt. Per cada panoràmica, calcularem l'orientació utilitzant els punts següents a l'actual i sabent que la inclinació de la càmara és sempre constant.

- **Visualitzar la ruta amb Google Earth.**
    - Permetre a l'usuari visualitzar la ruta actual de forma aèrea a través del plugin Google Earth per Google Maps. L'usuari tindrà l'opció d'aturar, continuar, tornar a iniciar i canviar la velocitat de reproducció.
    - A partir dels punts que conformen la ruta i, utilitzant l'API de Google Earth, situem sobre el mapa aeri cada punt. Unim els punts, dibuixant sobre el mapa una línia.

- **Visualitzar la ruta del vehicle per dies.**
    - Permetre a l'usuari visualitzar la ruta d'un vehicle per dies.
    - Mostrant un calendari, apareixeran els dies del mateix color que la ruta realitzada en aquell dia. Clicant sobre un dia concret, la ruta s'ocultarà o es mostrarà.
    - Rebem tots els punts que formen la ruta del vehicle i es classifiquen per dia. Un cop classificats per dia, la ruta es mostra amb un color determinat. Així obtenim una ruta amb tants colors com en dies s'hagi realitzat.

- **Visualitzar el gràfic de Velocitat.**
    - Permetre a l'usuari veure un gràfic en dos dimensions amb la velocitat, l'hora, minuts i segons en què s'ha realitzat.
    - Rebem tots els punts que formen la ruta del vehicle i es consulta la velocitat per cada punt. Un cop tenim la data, l'hora i la velocitat de cada punt, es mostren sobre el gràfic. A més a més, l'usuari té la possibilitat de clicar sobre qualsevol punt del gràfic i es mostrarà sobre el mapa el punt geogràfic corresponent a aquella hora i velocitat.

- **Visualitzar les alertes.**
    - Permetre a l'usuari visualitzar una taula amb la informació relacionada del vehicle que ha produït l'alerta: el polígon, la longitud, la latitud, el dia i l' hora.
    - En la càrrega inical de tots el vehicles es comprova punt per punt si estan dins d'algún polígon, si és així, s'activa una alerta que es guarda a la Base de Dades amb la seva informació. En el seguiment en temps real també es realitza el mateix procediment.
    - Si l'alerta ja ha estat afegida a la Base de Dades, no es torna a afegir.

- **Gestionar un polígon.**
    - Permetre a l'usuari realitzar un control d'alertes creant un polígon que representi una àrea del mapa. A més a més, un cop creat el polígon aquest, pot ser modificat clicant sobre algún dels seus vértexs, el qual

s'eliminaria. Si cliquem sobre algún vértex i l'arroseguem, aquest adoptarà la nova posició. Un cop creat el polígon, tenim l'opció de guardar-lo a la base de dades, eliminar-lo o centra el mapa per visualitzar-lo.

- Un cop dibuixat el polígon, si volem guardar-lo, recórrerrem vèrtex per vèrtex (és a dir, punt per punt) emmagatzemant-los en la Base de Dades per així més tard poder-los carregar de nou.
- Quan intentem crear un polígon amb un nom que ja ha estat utilitzat, ens alerta que hem de modificar el nom i no ens permet dibuixar/crear el polígon.

- **Gestionar els punts d'interès.**
  - Permetre a l'usuari, fent doble clic en el mapa, crear on vol el lloc d'interès. Després, l'usuari introduïrà la informació següent: nom, tipus de lloc, adreça, comentaris i seleccionarà la icona representativa dins d'un ventall de possibilitats. Un cop creats, es pot clicar sobre els punts d'interès els quals mostraran la informació introduïda anteriorment i que al seu torn es pot modificar. Si cliquem sobre la icona del punt d'interès i l'arroseguem, aquesta es situarà en la nova posició i la seva adreça canviarà automàticament. Podem escollir la icona més adhient pel punt d'interès utilitzant el selector de camps, on es mostren diferents àmbits i situacions.
  - Fent doble clic sobre el mapa, rebem una latitud i longitud que ens permet situar el punt sobre el mapa. Un cop tenim la informació relacionada amb el punt d'interès: nom, tipus de lloc, adreça, comentaris i icona que el representa, els guardem a la Base de Dades. Si al punt d'interès, modifiquem la informació relacionada amb el nom, tipus de lloc, comentaris o icona, aquesta informació s'actualitzarà a la Base de Dades. De la mateixa manera, si arrosseguem el punt d'interès a una nova localització, s'actualitzarà la nova adreça, la nova latitud i longitud de forma automàtica a la Base de Dades.
  - Quan intentem crear un nou punt d'interès amb el mateix nom que un d'existent, el programa ens alerta i no ens deixa continuar amb el procés de creació, impedint l'emmagatzematge a la Base de dades.

- **Afegir, modificar i eliminar un contacte a la llista de direccions.**
  - Permetre a l'usuari afegir informació sobre persones de contacte que volen ser alertades (en temps real) quan es produeixi una entrada d'un vehicle en certs polígons. L'usuari ha d'omplir un formulari amb la informació següent: nom, cognoms, telèfon, correu electrònic i a continuació marcar els polígons dels quals la persona de contacte vulgui rebre una notificació (alerta) i, finalment, ha de marcar la casella que indica si la persona de

contacte vol rebre al correu les notificacions en temps real que es vagin produïnt en els polígons seleccionats

- Una vegada rebuda la informació relacionada amb la persona de contacte s'emmagatzema a la Base de dades. Quan es produeixi una alerta en un polígon, es buscarà la relació dels polígons amb les persones de contacte de la Llista de Direccions i es seleccionaran tots els contactes que estiguin relacionats amb el polígon que ha fet saltar l'alerta. Tot seguit, es crea un correu electrònic que alerta a l'usuari que un vehícle ha estat dins del polígon i, a més, es mostra la imatge d'un mapa amb una marca que indica la localització on s'ha produït l'alerta.
- Quan intentem afegir un nou contacte i el correu electrònic indicat ja està registrat per una altra persona, no ens permet continuar amb el registre d'aquest contacte, fins que no s'introdueixi una direcció de correu diferent.

- **Seguiment**
  - Permetre a l'usuari visualitzar la informació bàsica del vehicle: posició actual (latitud i longitud), velocitat actual, data de l'última dada rebuda i nombre de parades realitzades. A més a més, es permet seleccionar un vehicle i realitzar un seguiment constant (en temps real) de la seva posició i de les seves dades actualitzades.
  - A partir de les dades de l'últim punt, obtenim la latitud i la longitud, la velocitat , la data i les parades realitzades. Aquestes són mostrades en el panell dret i es van actualitzant a mesura que van arribant noves dades del vehicle. La visualització centrada del vehicle en el mapa, es realitza aproximadament cada 3 segons.

- **Visualització Polígons**
  - Permetre a l'usuari visualitzar una llista amb tots els polígons que estan dibuixats sobre el mapa. Clicant sobre cada element de la llista, es desplega la informació addicional: àrea, número de vèrtex i visualitzador del polígon que ens permet centrar-lo sobre el mapa i escollir l'opció mostrar o ocultar.

- **Visualització Punts d'interès**
  - Permetre a l'usuari visualitzar una llista amb tots els punts d'interès que s'han creat. Clicant sobre cada element de la llista, es desplega un visualitzador del punt d'interès que ens permet centrar-lo sobre el mapa i escollir l'opció mostrar o ocultar.

4.2 Requeriments no funcionals

Els requeriments no funcionals són restriccions imposades pel client o pel mateix problema i que afecten al disseny. Normalment, els requeriments no funcionals són quantificables.
Els requeriments no funcionals del projecte són:

- Requeriments de rendiment:
    - Requeriments estàtics (de capacitat);
        - **Usuaris simultanis i Nombre de terminals**:
            - El mòdul de captació que recull, tracta i emmagatzema les dades rebudes per un dispositiu GPS, el nombre de terminals que poden estar controlats en temps real, depèn del nombre de ports del servidor que tinguem disponibles per establir connexions amb cada dispositiu GPS.
            - El mòdul de processament està restringit a un terminal, ja que l'aplicació ha estat pensada per a ser controlada per una sola persona. Malgrat ser una aplicació monousuari, diversos terminals podrien realitzar el control dels vehicles, amb un sol inconvenient: si es produïs una alerta en temps real i, en aquells moments, hi haguès *n* terminals connectats, la notificació, via correu electrònic, s'enviaria *n* cops.
- Restriccions de disseny:
    - Limitacions del hardware:
        - El **Servidor** que recull, tracta i analitza les dades amb connexió a Internet. Pot utilitzar-se sobre qualsevol Sistema Operatiu ja que ha estat implementat en Java.
        - **Telèfon mòbil** amb sistema operatiu Windows Mobile, tecnologia GPS que utilitzi com a protocol de comunicació NMEA i tarifa plana de dades per a la comunicació del dispositiu mòbil i el servidor de dades.
        - El **Sistema de seguiment** pot utilitzar-se sobre qualsevol Sistema Operatiu amb connexió a Internet, amb un navegador que permeti executar JavaScript i consultes AJAX.
- **Objectius de disseny**
    - Aplicació fàcilment ampliable, és a dir, aplicació on es poden incorporar noves funcionalitats de manera senzilla.
    - Aplicació *user friendly*, és a dir, amb un aspecte senzill, on els pasos per a realitzar qualsevol operació són mínims.
    - Fàcil manteniment per part de l'administrador de l'aplicació.

5 Modelat del comportament

5.1 Casos d'ús

Els casos d'ús ofereixen un mitjà sistemàtic i intuïtiu per capturar els requeriments funcionals, centrant-se en el valor afegit per l'usuari. Dirigeixen tot el procés de desenvolupament atès que la majoria d'activitats (planificació, anàlisi, disseny, validació, test...) es realitzen a partir de casos d'ús.

Un cas d'ús especifica el comportament desijtat del sistema. Representen els requeriments funcionals del sistema i descriuen què fa el sistema, no com ho fa.

Els casos d'ús descriuen un conjunt d'interaccions, entre actors externs i el sistema, orientades a satisfer un objectiu d'un actor. Són iniciats per un actor amb un objectiu previst i es completen amb èxit quan el sistema el satisfà. El conjunt complet de casos d'ús especifica totes les possibles formes d'usar el sistema, és a dir, el comportament requerit.

5.1.1 Pasos per l'obtenció de casos d'ús.

1. Identificar els usuaris del sistema:
2. Trobat tots els rols que juguen els usuaris i que són rellevants al sistema.
3. Per cada rol identificar totes les formes (objectius) d'interactuar amb el sistema.
4. Crear un cas d'ús per cada objectiu.
5. Estructurar els casos d'ús.
6. Revisar i validar amb l'usuari.

5.1.2 Diagrama de casos d'ús.

Figura 5.1: Casos d'ús del sistema.

A la figura 5.1 es mostra els casos d'ús del sistema. En aquest solament apareix un usuari, ja que l'aplicació és monousuari, però activitats com: Afegir nou contacte llista de direccions, modificar contacte llista direccions, eliminar contacte llista direccions podria ser realitzar per un usuari administrador.

5.1.2.1 Especificació dels casos d'ús

- **Visualitzar Vehicles**

  Actors: Usuari
  Descripció: Permet a l'usuari visualitzar sobre la plataforma de cartografia digital Google Maps els vehicles, la seva ruta i les parades.

- **Visualitzar Tràfic**

  Actors: Usuari
  Descripció: Permet a l'usuari visualitzar sobre el mapa la informació relacionada amb el Tràfic provinent de la DGT.

- **Crear Polígon**

  Actors: Usuari
  Descripció: A través del ratolí, l'usuari tindrà la possibilitat de dibuixar un polígon marcant sobre el mapa els vèrtexs.

- **Modificar Polígon**

  Actor: Usuari
  Descripció: L'usuari té la possibilitat de modificar la forma del polígon a través d'arrossegar els vèrtexs a una altra posició.

- **Eliminar Polígon**

  Actor: Usuari
  Descripció: Permet a l'usuari eliminar els polígons que ha creat i que també han estat guardats.

- **Visualitzar el gràfic de Velocitat**

  Actor: Usuari
  Descripció: Permet a l'usuari visualitzar un gràfic amb les dades de velocitat i temps. L'usuari també té la possibilitat de visualitzar sobre el mapa la posició geogràfica d'aquella velocitat i temps concret.

- **Visualitzar la ruta del vehicle per dies**

  Actor: Usuari
  Descripció: Utilitzant un calendari, l'usuari serà capaç de mostrar i amagar la ruta clicant sobre el dia amb el mateix color que la ruta.

- **Visualitzar les alertes**

  Actor: Usuari
  Descripció: Permet a l'usuari veure una taula amb totes les alertes dels polígons i dels vehicles.

- **Crear punts d'interès**

  Actor: Usuari
  Descripció: Permet a l'usuari crear un punt en el mapa amb un significat propi.

- **Modificar punt d'interès**

  Actor: Usuari
  Descripció: Permet a l'usuari modificar les dades i l'aspecte del punt d'interès.

- **Eliminar punt d'interès**

  Actor: Usuari
  Descripció: Permet a l'usuari eliminar el punt d'interès i la seva informació.

- **Afegir un nou contacte llista direccions**

  Actor: Usuari
  Descripció: Permet a l'usuari afegir un nou contacte a la llista de direccions.

- **Modificar contacte llista direccions**

  Actor: Usuari
  Descripció: Permet a l'usuari modificar les dades de qualsevol contacte de la llista de direccions.

- **Eliminar contacte llista direccions**

  Actor: Usuari
  Descripció: Permet a l'usuari eliminar qualsevol contacte de la llista de direccions.

- **Visualització Punts interès**

  Actor: Usuari
  Descripció: Permet a l'usuari visualitzar una llista amb tots els punts d'interès creats.

- **Visualització Polígons**

  Actor: Usuari
  Descripció: Permet a l'usuari visualitzar una llista amb tots els polígons creats i informació específica de cadascun: àrea i número de vèrtexs.

- **Seguiment Vehicle**

  Actor: Usuari
  Descripció: Permet a l'usuari realitzar un seguiment visual i constant d'un vehicle.

- **Visualitzar ruta Google Earth**

  Actor: Usuari
  Descripció: Permet a l'usuari visualitzar la ruta realitzada a través de Google Earth amb una perspectiva aèrea.

- **Visualitzar ruta Street View**

  Actor: Usuari
  Descripció: Permet a l'usuari visualitzar la ruta realitzada de manera panoràmica i com si estiguès a peu de carrer, a través de Google Maps Street View.

- **Modificar Opcions i Informació Vehicle**

  Actor: Usuari
  Descripció: Permet a l'usuari canviar les opcions de visualització de ruta i la informació relacionada amb el vehicle: sobrenom, tipus, marca, model, velocitat màxima i temps d'aturada.

- **Crear Ruta**

  Actor: Usuari
  Descripció: Permet a l'usuari crear una ruta per a un vehicle concret, ja pot ser a una adreça concreta o a una llista de fites (punts d'interès).

- **Visualitzar Informació del vehicle**

  Actor: Usuari
  Descripció: Permet a l'usuari visualitzar una llista amb tots el vehicles que s'estan controlant a l'aplicació i els seus controls de visualització: mostrar o ocultar ruta, visualitzar vehicle, mostrar o ocultar vehicle, visualitzar ruta per dies.

6 Disseny

El disseny és el procés d'aplicar diferents tècniques i principis amb el propòsit de definir un sistema amb suficients nivells de detall com per permetre la seva realització física.
El disseny té com a objectiu minimitzar la distància entre el problema i la seva solució. També produir un model o representació del sistema que pugui ser utilitzat, en una fase posterior, a fi d'implementar-lo.
En el nostre projecte, l'aplicació estarà dividida en dos mòduls: un mòdul, anomenat de captació, s'encarregarà de recollir, tractar i emmagatzemar les dades dels dispositius GPS connectats i l'altre mòdul, anomenat de processament, s'encarregarà d'interpretar les dades emmagatzemades pel mòdul de captació i mostrar-les sobre el navegador web.

6.1 Disseny de la Base de Dades

Abans de començar a dissenyar la Base de Dades havíem de conèixer les dades que s'emmagatzemarien en aquesta, per això vaig fer una anàlisi de totes les dades que intervindrien en el projecte. A partir de l'elaboració d'una llista de les dades, vaig començar a estructurar i planificar la Base de Dades.
A mida que s'avançava en el projecte, s'afegien noves funcionalitats i noves relacions entre les taules.
En aquest apartat, mostrarem una **versió preliminar** (o base) del disseny de la Base de Dades i, una **versió final** de la Base de Dades sempre subjecta a la possibilitat d'ampliació i millora.

6.1.1 Versió Preliminar

Per a la realització de la versió preliminar, vaig analitzar quines serien les dades necessàries per a mostrar sobre el mapa els vehicle i els punts d'interès de l'usuari. Documentant-nos sobre el protocol de transmissió de dades del GPS, anomenat NMEA, vaig cercar quines dades serien necessàries per al nostre projecte i aquestes van ser: latitud,longitud,velocitat, data i hora.
El disseny preliminar de la Base de Dades dels vehicles i els punts d'interès em va donar com a resultat tres taules (Figura 6.1). Dues taules per la gestió dels propis vehicles (Informació dels vehicles i dades de posicionament dels vehicles) i una taula per la gestió dels punts d'interès. L'estructura de la Base de Dades va ser molt simple ja que treballava amb poques dades que més tard el mòdul de processament s'ocuparia d'interpretar i processar adequadament per obtenir el seguiment dels vehicles i la gestió dels punts d'interès.

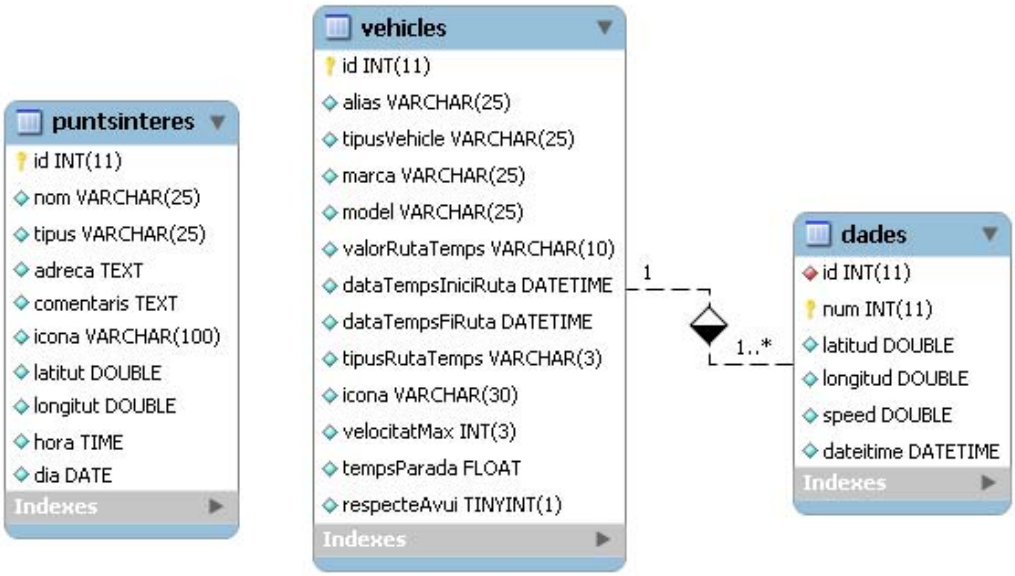


Figura 6.1: Disseny preliminar de la Base de Dades

Explicació de les taules:

- **puntsinteres**:
  - La taula puntsinteres emmagatzema dades relacionades amb el punt d'interès creat per l'usuari, tal i com: informació d'identificació, de posicionament i de creació.

  - Camps:
    - Id : identificador per a cada punt d'interès.
    - Nom: cadena de caràcters que identifica el punt d'interès.
    - Tipus: cadena de caràcters que descriu el tipus de punt d'interès.
    - Adreca : cadena de caràcters que identifica el topònim del punt d'interès.
    - Comentaris: text que l'usuari introdueix com a informació adicional del lloc d'interès.
    - Icona: cadena de caràcters que descriu la ruta i nom de la imatge que representarà el punt d'interès.
    - Latitud: número que representa la latitud del punt d'interès sobre la Terra.
    - Longitud: número que representa la longitud del punt d'interès sobre la Terra.
    - Hora: representa l' hora, minuts i segons en què el punt d'interès va ser creat.

        - Dia: data representada per any-mes-dia en què el punt d'interès va ser creat.

- **Vehicles**:
    - La taula vehicles emmagatzema informació relacionada amb cada vehicle existent a l'aplicació.
    - Camps:
        - Id: Identificador únic del vehicle.
        - Alias: cadena de caràcters que representa el sobrenom del vehicle i que és més entenedora que la id.
        - TipusVehicle: cadena de caràcters que identifica el vehicle amb un tipus en concret.
        - Marca: cadena de caràcters que representa la marca del vehicle.
        - Model: cadena de caràcters que representa el model del vehicle.
        - ValorRutaTemps: cadena de caràcters que identifica el nombre de minuts, hores, dies, setmanes, mesos, anys,interval de dates o dia en concret, enrere pel qual es mostrarà la ruta del vehicle.
        - DataTempsIniciRuta: data i temps que representa l'inici de l'interval a mostrar d'una ruta.
        - DataTempsFiRuta: data i temps que representa el final de l'interval a mostrar d'una ruta.
        - TipusRutaTemps: cadena de caràcters que representa el tipus de Temps escollit per a mostrar la Ruta: Minuts(MIN), Hores(HOR), dies(DIA), Setmanes (SET), Mesos (MES), Anys (ANY), dia en concret (CON) ó Interval de dates (INT).
        - Icona: cadena de caràcters que descriu la ruta i nom de la imatge que representarà el vehicle.
        - VelocitatMax: número que representa la velocitat màxima pel vehicle.
        - TempsParada: número de minuts que representa una parada per aquell vehicle.
        - RespecteAvui: valor booleà que representa si és cert que el nombre de temps enrere que es mostrarà la ruta serà respecte avui (data i hora) ó fals si serà respecte a l'última dada (data i hora) rebuda pel vehicle.

- **Dades**:
    - La taula dades contè tota la informació relacionada amb la localització dels vehicles.
    - Camps:
        - Id: identificador del vehicle.
        - Num: número que representa l'index de cada fila de dades insertades en una data i hora concreta.
        - Latitud: número que representa la latitud del vehicle sobre la Terra en una data i hora concreta.
        - Longitud: número que representa la longitud del vehicle sobre la Terra en una data i hora concreta.
        - Speed: número que representa la velocitat del vehicle en una data i hora concreta.
        - Dateitime: Data i hora en la qual es va insertar la localització concreta del vehicle.

6.1.2 Versió Final

Un cop la versió preliminar de la Base de Dades ja permitia mostrar els vehicle, la seva ruta i la visualització en temps real dels vehicles, vaig començar a dissenyar la base de dades per la resta de funcionalitats especificades pel projecte, com el sistema de Geofencing. El sistema de Geofencing s'ocupa de marcar un perímetre virtual per una àrea geogràfica del món real. Quan un vehicle entra en una àrea amb sistema geofencing, aquesta genera una notificació o alerta. Aquesta notificació, conté informació relacionada amb la localització del vehicle i sol ser enviada com un missatge al telèfon mòbil o al correu electrònic.

Per a implementar el sistema de Geofencing necessitava marcar una zona o àrea del mapa, per això, vaig introduïr el concepte de polígon. A través de marcar una sèrie de punts sobre el mapa podrïem definir una àrea, aquests punts representarien els vèrtexs del polígon i serien emmagatzemats en una taula anomenada *polígons* a la Base de Dades. Un cop construïts els polígons solament ens faltaria el procés de notificació i alertes. El procés d' alertes necessitaria una taula, anomenada *alertes*, que emmagatzemès la informació de les alertes produïdes , informació relacionada dels vehicles i l'hora en què s'haguessin produït. El procés de notificació necessitaria una taula, anomenada *llistadireccions*, que emmagatzemés una llista de contactes amb informació personal del contacte i una taula, anomenada *subscripcio*, que contingués la relació de cada contacte amb els polígons pels quals vol rebre la notificació al correu.
La incorporació d'aquesta nova funcionalitat no canviaria la versió preliminar de la Base de Dades, sinó que afegiria noves relacions i noves taules. Per tant, obtendríem com a resultat una Base de dades amb 7 taules. A la figura 6.2 podem observar el seu disseny físic i al Annex 1 podem observar el seu model Entitat-Relació.

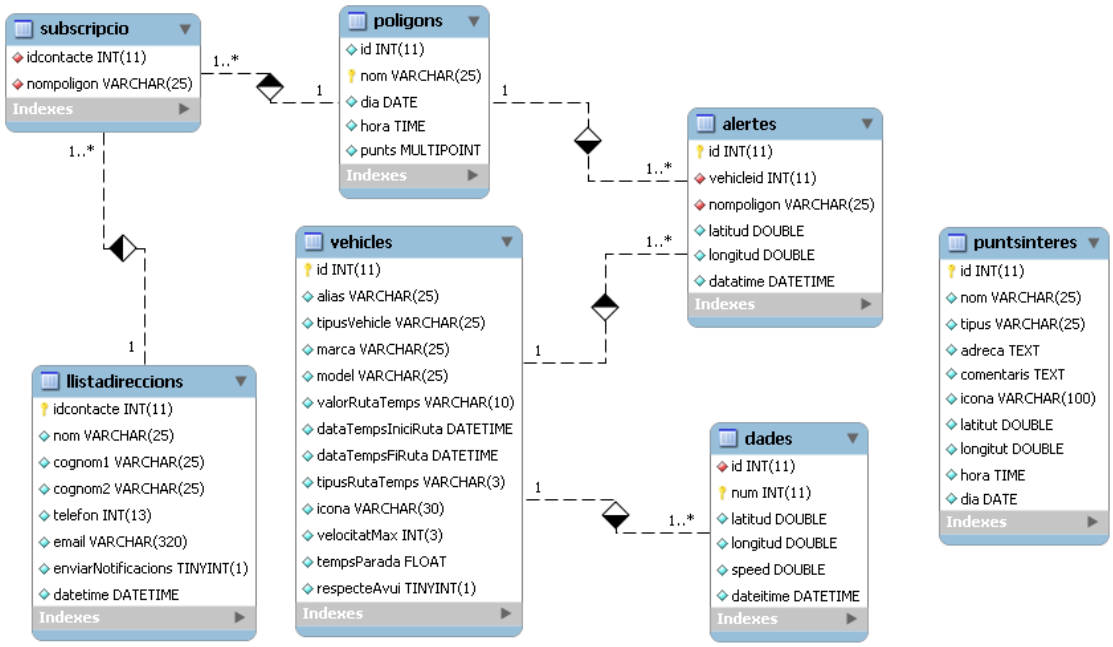


Figura 6.2: Disseny final de la Base de Dades

Com a la versió preliminar de la Base de Dades ja s'han explicat les 3 taules: vehicles, dades i puntsinteres, en aquesta s'explicarà les 4 taules restants que afegeixen funcionalitats a la versió preliminar.

- **poligons**:
  - La taula polígons contè informació relacionada amb la creació del polígon per part de l'usuari, tal i com: informació d'identificació, de posicionament i dates de creació.
  - Camps:
    - id: número identificador del polígon
    - nom: cadena de caràcters que representa el nom únic del polígon.
    - Dia: data que representa el any-mes-hora de quan es va crear el polígon.
    - Hora: hora que representa l'hora:minuts:segons de quan es va crear el polígon.
    - Punts: Classe MultiPoint de Mysql que representa una colecció de geometries composta per elements Point. Per tant, és una colecció de punts que representa els vèrtexs del polígon.
- **Llistadireccions**:
  - La taula direccions contè informació relacionada amb les persones de contacte que volen estar subscrites al sistema de geofencing.

- Camps:
  - Idcontacte: Identificador de la persona de contacte.
  - Nom: cadena de caràcters que representa el nom de la persona de contacte.
  - Cognom1: cadena de caràcters que representa el primer cognom de la persona de contacte.
  - Cognom2: cadena de caràcters que representa el segon cognom de la persona de contacte.
  - Telefon: número que representa el telefon de la persona de contacte.
  - Email: cadena de caràcters que representa el correu electrònic al qual la persona de contacte rebrà les notificacions d'alerta.
  - EnviarNotificacions: booleà que representa si la persona de contacte vol rebre les notificacions al correu o en el cas contrari no vol rebre les notificacions.
  - Datetime: data representada per any-mes-dia i hora:minuts:hora que indica quan es va crear la persona de contacte.

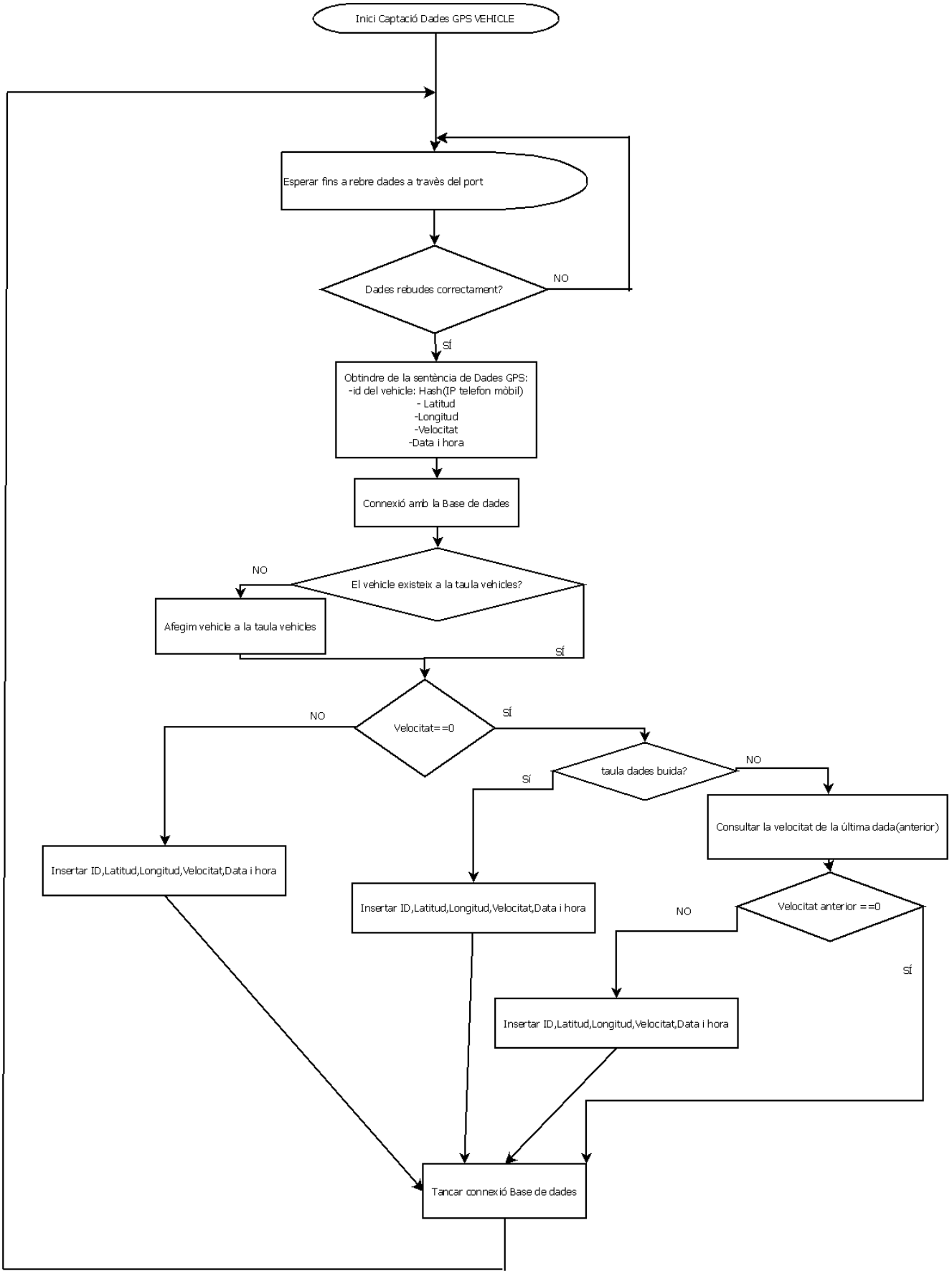
- **Subscripcio**
  - La taula subscripció, és una taula intermitja que contè informació que relaciona les persones de contacte de la llista de direccions amb els polígons emmagatzemats a la base de dades permeten així crear la subscrició dels contactes amb els polígons desitjats.
  - Camps:
    - Idcontacte: identificador de la persona de contacte
    - Nompoligon: cadena de caràcters que representa el nom del polígon.

- **Alertes**
  - La taula alertes contè una llista amb totes les alertes produïdes pels vehicles i en els polígons en els quals s'han produït.
  - Camps:
    - Id: identificador de l'alerta.
    - Vehicleid: identificador del vehicle que ha produït l'alerta.
    - Nompoligon: cadena de caràcters que representa el nom del polígon en el qual s'ha produït l'alerta.
    - Latitud: número que representa la latitud del vehicle en el moment d'haver-se produït l'alerta.
    - Longitud: número que representa la longitud del vehicle en el moment d'haver-se produït l'alerta.
    - Datatime: data representada per any-mes-dia i hora:minuts:segons que indica la data i l'hora de quan es va produir l'alerta.

6.2 Disseny Algorisme.

En aquest apartat es mostra l'estructura general de l'algorisme del mòdul de captació i del de processament, per mostrar l'estructura de l'algorisme utilitzaré els diagrames de flux. Aquests diagrames mostraran el flux per un sol vehicle però poden ser ampliats a tants vehicles com estigui configurat el mòdul de captació.

6.2.1 Diagrama de flux del mòdul de captació:

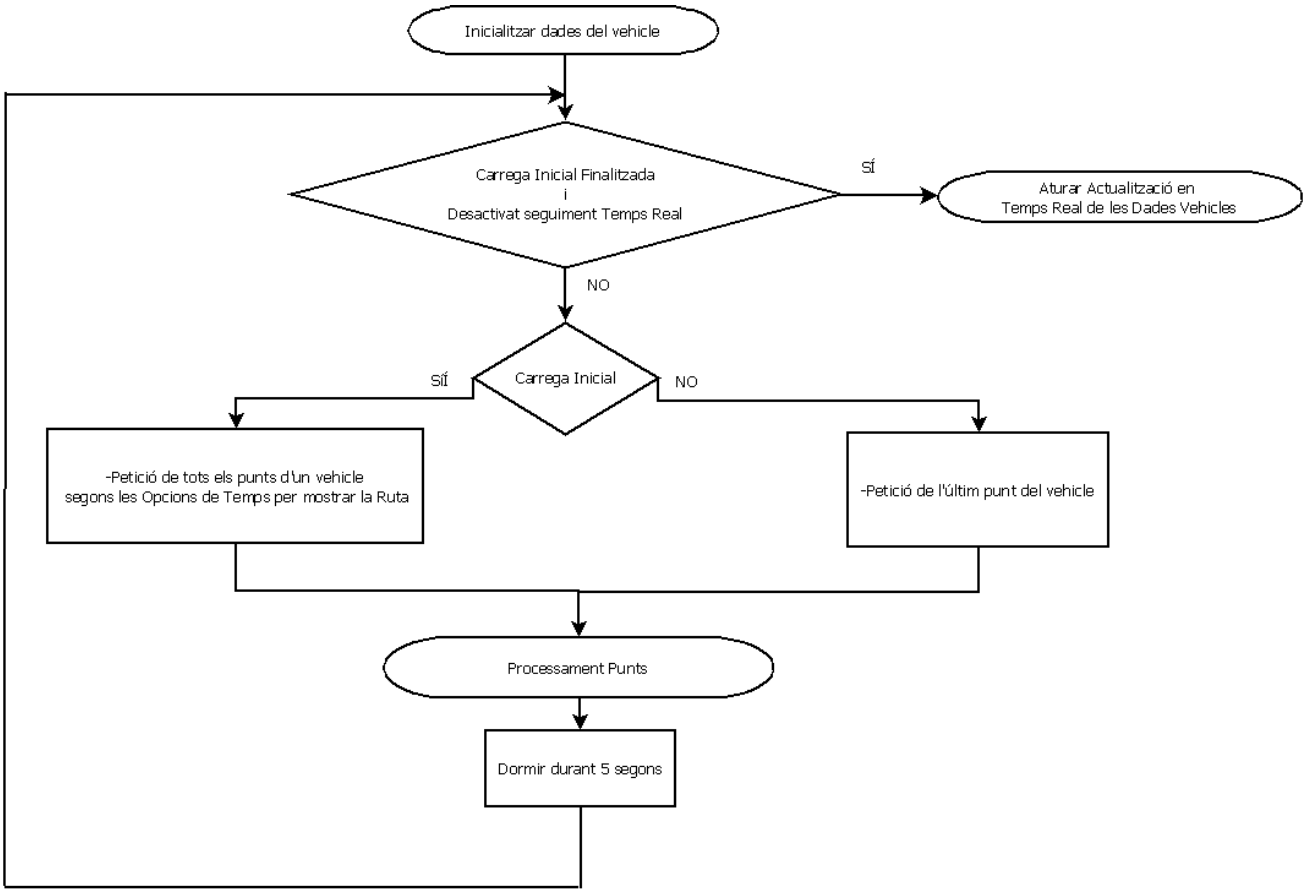


En aquest diagrama de flux podem veure com el mòdul de captació treballa en un bucle on espera la recepció de dades a través d'un port del servidor, és a dir, utilitza l'esquema de client-servidor.

El mòdul s'ocupa de comprovar que les dades rebudes siguin les que volem captar, les tracta i extreu la informació necessària. Abans d'emmagatzemar-les a la Base de Dades es comprova que la velocitat no sigui zero. Si és zero, vol dir que el vehicle està aturat i haurem de comprovar que la última dada emmagatzemada pel vehicle no fos zero. Ja que si anteriorment el vehicle estava aturat i ara continua estant aturat, vol dir que el vehicle no s'ha mogut (no ha canviat de posició) i per tant no cal omplir la base de dades amb les mateixes posicions geogràfiques (punts). Un cop realitzades les comprovacions pertinents, passem a emmagatzemar les dades i a tancar la connexió.

Finalitzades les operacions d' emmagatzematge i tancament de la connexió, el procés torna al punt inicial del bucle per a continuar rebent altres dades i realitzar les mateixes operacions.

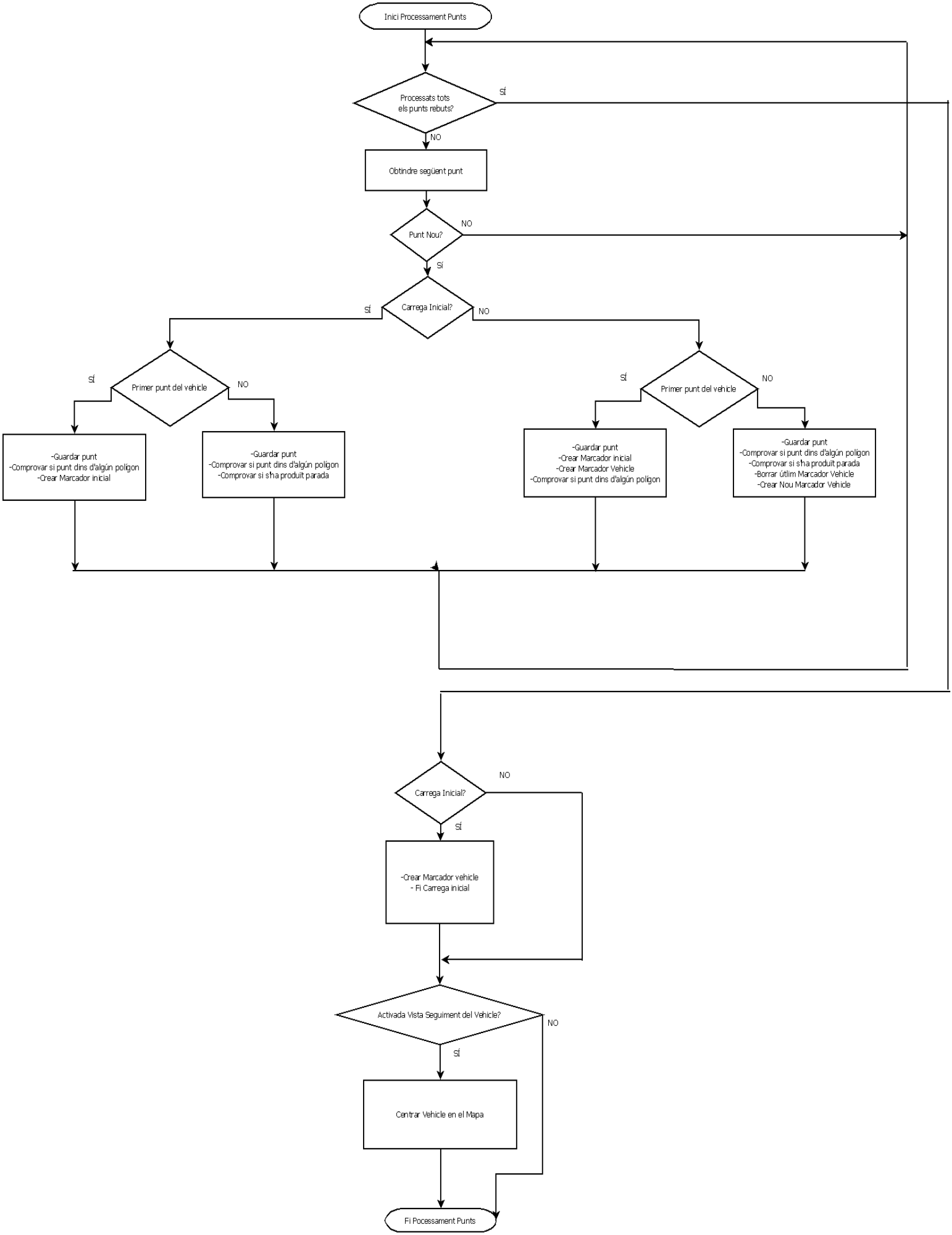
6.2.2 Diagrames de flux del mòdul de processament:



En aquest primer diagrama podem veure com el mòdul de processament treballa en un bucle on realitza peticions de punts a la Base de Dades. Si la petició és al principi, vol dir que la càrrega inicial dels punts que formen la ruta del vehicle s'està realitzant, per tant, es demanen tots els punts que compleixin les opcions establertes per l'usuari. En canvi, si la petició es

realitza després de la càrrega inicial, vol dir que estem comprovant a la Base de Dades si s'ha insertat algún punt nou i, per tant, demanar-lo. Aquesta comprovocació es realitza aproximadament cada 5 segons i desenvolupa l'actualització de les dades en temps real. Un cop fetes o bé la petició de tots els punts per la càrrega inicial o bé la d'un sol punt, el flux del procés es passa al Processament de Punts.

El diagrama del Processament de punts es mostra a continuació:


Inici Processament Punts
Processats tots els punts rebuts?
SÍ
NO
Obtindre següent punt
Punt Nou?
NO
SÍ
Carrega Inicial?
SÍ
NO
Primer punt del vehicle
SÍ
NO
-Guardar punt
-Comprovar si punt dins d'algún polígon
-Crear Marcador inicial
-Guardar punt
-Comprovar si punt dins d'algún polígon
-Comprovar si s'ha produit parada
Primer punt del vehicle
SÍ
NO
-Guardar punt
-Crear Marcador inicial
-Crear Marcador Vehicle
-Comprovar si punt dins d'algún polígon
-Guardar punt
-Comprovar si punt dins d'algún polígon
-Comprovar si s'ha produit parada
-Borrar útlim Marcador Vehicle
-Crear Nou Marcador Vehicle
Carrega Inicial?
NO
SÍ
-Crear Marcador vehicle
- Fi Carrega inicial
Activada Vista Seguiment del Vehicle?
NO
SÍ
Centrar Vehicle en el Mapa
Fi Processament Punts

7 Implementació

En aquesta secció es veurà com s'ha implementat el projecte.

7.1 Base de dades.

Un cop dissenyada la base de dades necessitàvem una eina per a implementar-la. Per la implementació de la base de dades vaig triar MySQL que és un sistema de gestió de base de dades relacional, multi-fil i multi-usuari. A part de ser un dels sistemes més utilitzats també s'ha de tenir en compte que és un software lliure i que dóna als usuaris la llibertat d'executar-lo en qualsevol màquina compatible.
Per a l'administració de la base de dades, és a dir, la creació, estructuració i relació de les taules, utilitzem una eina anomenada "*phpmyadmin*". Phpmyadmin és una eina escrita en PHP que permet l'administració del sistema MySQL a través de pàgines web, utilitzant un navegador. Aquesta eina ve incorporada en alguns softwares, com per exemple xampp, que permeten instal·lar un servidor web en qualsevol ordinador. Phpmyadmin, a més a més, incorpora l'opció de visualitzar la base de dades i crear relacions entre les taules a través del ratolí.

Un cop tenim el Disseny de la Base de dades implementat en MySQL a través de phpmyadmin, accedim i modifiquem la Base de Dades a través de consultes creades en el llenguatge de programació PHP, aquest ens permet crear una connexió i a partir d'aquesta, realitzar les consultes o modificacions que volguem sobre la Base de Dades gestionada per MySQL.
Podem observar la sintàxis d'algunes de les consultes del projecte, a l'Annex 3.

7.2 Mòdul de captació

Un cop tenia la Base de Dades implementada, vaig passar al procés d'implementació del mòdul de captació de les dades GPS, ja que tenia la Base de Dades on emmagatzemar-les.
Per començar la implementació del mòdul de captació necessitava que el dispositiu amb tecnologia GPS transmetés, utilitzant el protocol NMEA, les dades de posicionament, velocitat i temps. Per a transmetre aquestes dades vaig realitzar un procés de búsqueda d'aplicacions per a un dispositiu mòbil amb sistema operatiu Windows Mobile. Com a resultat vaig obtenir una aplicació, anomenada GpsPlex, que s'adaptava millor als meus requeriments.
El GpsPlex és un software gratuït que llegeix la sortida del GPS i reenvia les dades a un port sèrie virtual. Addicionalment GPSPlex permet la comunicació entre diversos programes en una xarxa IP. La pantalla principal del programa presenta l'estat actual del GPS en format gràfic amb una interfície molt simple.

El menú del programa permet a l'usuari configurar l'origen de les dades (GPS Local, Emulador llegint LOG o GPS connectat per TCP/IP) i configurar la sortida d'aquestes (Connexió IP) en format de sentències NMEA.

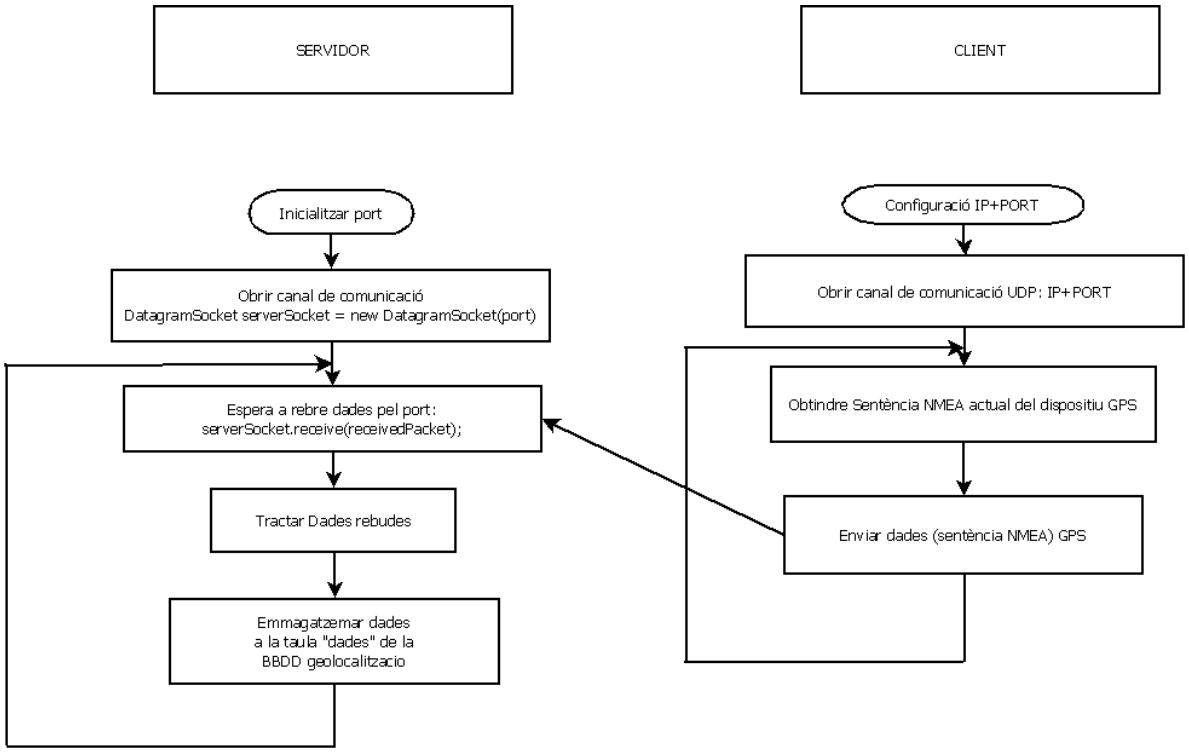
Amb l'aplicació GPSPlex, instal·lada en el dispositiu de telefonia mòbil amb tecnologia GPS, i la configuració per enviar les dades en format NMEA a una adreça IP i un PORT concret, vaig començar a implementar la part del servidor que rep i emmagatzema les dades GPS.

El servidor havia de poder executar-se en qualsevol màquina. Per a poder cumplir aquest requisit, vaig decidir utilitzar el llenguatge de programació Java. Java és un llenguatge de programació orientat a objectes desenvolupat per l'empresa Sun Microsystems. El llenguatge pren molta de la seva sintaxi de C i C++, però té un model d'objectes més simple i elimina eines de baix nivell, que solen induïr a molts errors, com la manipulació directa de punters i memòria. El llenguatge de programació Java ens permet crear aplicacions compilades en *bytecode,* que és un codi intermig més abstracte que el codi màquina. En temps d'execució, el bytecode s'interpreta a codi natiu permetent així que l'aplicació s'executi en qualsevol màquina.

Per a poder desenvolupar l'aplicació en Java vaig utilitzar una eina, anomenada Eclipse, que em permetès crear un projecte, editar-lo i compilar-lo. Eclipse és un entorn de desenvolupament integrat (IDE) de codi obert i multiplataforma. Eclipse disposa d'un Editor de text que resalta la sintaxi i la compilació, és en temps real.

Al començar a implementar el servidor vaig haver de definir l'estructura client-servidor que hauria de tenir l'aplicació.

L'esquema que es mostra a continuació es basa en el seguiment d'un sol vehicle, és a dir, en la connexió per un sol port.



Un cop decidida l'estructura, escollit el llenguatge de programació i l'entorn de desenvolupament, vaig començar a implementar l'aplicació del servidor. L'aplicació, com es pot observar en la secció de Disseny, està formada per 3 classes: GpsRequest, GpsTreatment i GpsSql. A més a més, tenim una quarta classe anomenada GpsServer que treballa com a classe principal i d'inicialització. La classe GpsServer s'ocupa de inicialitzar l'aplicació, és a dir, a partir d'un cert número de port crea escoltes o connexions consecutius al port, per a més informació de classes consulteu l'Annex 2. Per exemple, si el port escollit és el 9996, GpsServer crearà connexions a partir d'aquest de forma consecutiva, és a dir, 9996, 9997, 9998 i 9999, si escollim tenir 4 connexions per 4 vehicles diferents. Aquesta classe principal crea a través de GpsRequest tants fils (*threads*) com vehicles, connexions o ports estiguem seguint, ja que GpsRequest hereda de la classe pare Thread. La classe GpsRequest crea un canal de comunicació (socket) per un port concret. El socket que es crea treballa amb el protocol de transport UDP, ja que l'aplicació GPSPlex instal·lada en el dispositiu mòbil envia les dades del GPS amb aquest protocol. UDP (User Datagram Protocol) és un protocol de la capa de transport encarregada d'efectuar el tranport de les dades de la màquina origen a la de destí, independintzant-la del tipus de xarxa física que s'està utilitzant.

A part que l'aplicació GPSPlex em condicionès a crear una connexió per rebre dades a través d'un port UDP, hi ha altres motius que també m'han fet seguir per aquest camí i no canviar de protocol de transport, com per exemple el TCP. TCP (Transmission Control Protocol) és un protocol de comunicació orientat a connexió i fiable. TCP permet que la comunicació entre dos sistemes s'efectui lliure d'errors, sense pèrdues i amb seguretat. En canvi, UDP no dóna

garanties del lliurament dels seus missatges. En una aplicació on es realitza un enviament cada 1 segon, la garantia de recepció no és el més important, ja que el desplaçament que pot realitzar un vehicle en 1 segon és mínim. El motiu més important pel qual he triat el protocol UDP és perquè permet transmetre amb més velocitat que el protocol TCP, ja que no ha de respondre al transmissor, ni realitzar processos per garantir l'ordre dels paquets, ni garantir la recepció d'aquests, tampoc s'ha d'ocupar d'establir i tancar una connexió.

Un cop creat el canal de comunicació, a través de la classe DatagramSocket de l'API de Java, implementem un bucle que rebi les dades pel canal de comunicació o socket creat. Utilitzem el mètode *receive* de la classe DatagramSocket que guarda el missatge en un objecte de la classe DatagramPacket. Tan aviat com l'objecte DatagramPacket contè el missatge rebut, utilitzem el mètode getData() per a transformar el missatge en una cadena de caràcters (String). Amb aquest String realitzem la comprovació que contingui la cadena de caràcters "GPRMC" per a assegurar-nos que el missatge rebut és una cadena de caràcters en format NMEA. Feta aquesta comprovació, creem un objecte GpsTreatment amb la cadena de caràcters rebuda i l'adreça IP del dispositiu que l'ha enviada.

Amb el constructor de la classe GpsTreatment ens ocupem de buscar la sentència GPRMC per extreure els camps que necessitem i emmagatzemar-los a la Base de dades. Anem recorrent, camp per camp, la sentència "GPRMC" per extreure la latitud, la longitud, l'hora, la data i la velocitat del vehicle. Com en cap sentència rebuda del protocol NMEA s'especifica un identificador del dispositiu, el creem a partir de l'adreça IP que sabem que és única. A aquesta adreça li apliquem un hashCode que ens retorna un valor, el qual utilitzarem com a identificador del vehicle. Un cop tenim tots els camps necessàris, s'executa el mètode transacció de la classe GpsTreatment. Aquest s'ocupa de crear un objecte GpsSql, el constructor del qual crea una connexió amb la classe *Connection* de Java a la Base de dades "geolocalització". Efectuada la connexió a la Base de dades, es realitzen les comprovacions de la velocitat, comentades en la secció del disseny de l'algorisme del mòdul de captació, on es comprova que la velocitat actual no sigui zero per no omplir la Base de Dades amb posicions geogràfiques idèntiques quan el vehicle està aturat. Fetes les comprovacions de la velocitat, es passa a crear una sentència SQL per a insertar les dades rebudes pel dispositiu GPS en la taula *dades* de la Base de Dades *geolocalització*. I és així com captem, tractem i emmagatzemem les dades rebudes pel dispositiu GPS.

7.3 Mòdul de processament.

La implementació del mòdul de processament és la part més complexa i que mès temps ha dut per a la seva realització, ja que a partir d'un conjunt de dades bàsiques rebudes del dispositiu mòbil, s'havien de processar i tractar de tal forma que es puguessin crear diverses funcionalitats.

El mòdul de processament és l'aplicació que permet a l'usuari interactuar amb les dades del vehicle. En la secció de disseny, en l'apartat dels casos d'ús, es pot observar el comportament del mòdul de processament davant de la interacció amb l'usuari.

Abans d'implementar el mòdul de processament, em trobava en la situació que tenia tots els punts de cada vehicle però no tenia forma de mostrar-los. Per això, com la cartografia digital escollida per implementar aquest projecte era Google Maps, em vaig dedicar a llegir i estudiar la seva API (Interfície de programació d' aplicacions) i les seves possibilitats de treball.

Una API és un conjunt de crides a certes biblioteques, que ofereixen accès a certs serveis des dels processos i representen un mètode per a obtenir abstracció en la programació. Les APIs beneficien als programadors, ja que els hi permet utilitzar les seves funcionalitats, evitant el treball de programar des del principi.

A continuació faré una breu explicació de les classes principals de l'API de Google Maps versió 2 i utilitzades en el projecte:

- **GMap2**: crea un nou mapa dins d'un contenidor HTML[1] donat. Sol ser un element div [2]. És la classe central de l'API, Totes les altres són auxiliars d'aquesta.
- **GLatLng**: representa un punt en coordenades geogràfiques de longitud i latitud. S'ha de tindre en compte que alhora de construïr l'ordre en introduïr les coordenades és latitud i longitud.
- **GPoint**: representa un punt en el mapa (GMap2) utilitzant com a coordenades els pixels. No representa un punt sobre la terra perquè no són coordenades geogràfiques, sinó que representa un punt sobre el mapa creat per GMap2.
- **GSize**: representa la mida en pixels[3] d'una àrea rectangular del mapa. L'objecte necessita dos paràmetres amplada (coordenada x) i altura (coordenada y).
- **GEvent**: aquest espai de noms contè funcions que s'utilitzen per a registrar un manipulador d'events, per a events personalitzats o per events del DOM[4], i per llançar events personalitzats.
- **Gmarker**: marca una posició sobre el mapa. Implementa la interficie Goverlay i és afegida al mapa utilitzant el mètode addOverlay de la classe GMap2. Un marcador té una latitud i longitud (GlatLng), que és la posició geogràfica que representa, i una icona.
- **GIcon**: Una icona especifica la imatge utilitzada per a mostrar un marcador (Gmarker) sobre el mapa.
- **Gpolyline**: És una capa superposada al mapa que dibuixa una polilínia. Crea una polilinea a partir d'un Array[5] de vèrtexs. Es pot especificar el color, l'amplada i l'opacitat de la línia.
- **Gpolygon**: És una capa superposada al mapa que dibuixa un polígon. Crea un polígon a partir d'un Array de vèrtexs. Es pot especificar el color, l'amplada i l'opacitat de la línia. Adicionalment també es pot especificar el color que ompli el polígon.

1. HTML: (Hypertext Markup Language) és el llenguatge de marcat predominant per l'elaboració de pàgines web.
2. div: és una marca que defineix una divisió o secció en un document HTML.
3. pixel: és la menor unitat homogènea en color que forma part d'una imatge digital
4. DOM: (Document Object Model) són un conjunt estàndar d'objectes per representar documents HTML i XML.
5. Array: conjunt ordenat d'elements.

- **Gdirections**: s'utilitza per a obtenir resultats que ens indiquin la direcció per a la seva conducció i mostrar-les sobre el mapa o sobre un panell de text.
- **Groute**: Els objectes d'aquesta classe són creats per l'objecte Gdirections per a emmagatzemar informació sobre una ruta simple en el resultat d'una consulta de direccions.
- **Gstep**: Els objectes d'aquesta classe són creats per l'objecte Gdirections per emmagatzemar informació sobre un pas simple d'una ruta.
- **GstreetViewClient**: aquest objecte realitza una búsqueda de dades del sistema Street View basada en els paràmetres passats i els seus mètodes.
- **GXmlHttp**: aquest espai de noms proveeix un factory method per a crear instàncies XmlHttpRequest.

L'API de Google Maps està desenvolupada per treballar amb el llenguatge de programació JavaScript. Javascript és un llenguatge de scripting basat en objectes no tipats i livians, utilitzat per accedir a objectes en aplicacions. Principalment, s'utilitza integrat en un navegador web permetent el desenvolupament d'interficies d'usuari millorades i pàgines web dinàmiques.
Per a crear l'estructuració de la informació de la pàgina web, vaig utilitzar XHTML que és un llenguatge de marques i que té com a objectiu separar la informació i la forma de presentar-la.
La implementació del mòdul de processament la vaig començar desenvolupant la visualització de pocs punts sobre el mapa en XHTML i JavaScript, però aviat em vaig adonar que la implementació de funcions bàsiques i no relacionades amb el projecte era molt costós, per això vaig decidir utilitzar un framework. Un framework és una estructura conceptual i tecnològica de suport. Normalment, pot incluir suport de programes, biblioteques i un llenguatge interpretat per ajudar a desenvolupar i unir els diferents components d'un projecte.
El framework que vaig escollir per a utilitzar amb JavaScript va ser Prototype. Prototype és un framework escrit en Javascript que s'orienta al desenvolupament sençilll i dinàmic d'aplicacions web. És una eina que implementa les tècniques AJAX i que ens simplifica gran part del treball quan es pretén desenvolupar pàgines altament interactives. Prototype té com a funcionalitats addicionals: tecnologia AJAX, ampliar el DOM amb nous mètodes, intercanvi de dades en format JSON i definició de Classes i herència.
Una vegada escollits el llenguatge de programació i el seu Framework per implementar el mòdul de processament, solament em mancava trobar una eina que em permetés crear i editar els arxius de text que continguessin totes les línies de codi. Per aquesta feina vaig escollir l'aplicació Notepad++, que és un editor de text i de codi font lliure amb suport per a diversos llenguatges de programació i amb una sintaxi colorejada.

Per a l'explicació de la implementació de tot el mòdul de processament, el dividirem en diverses seccions, segons les seves funcionalitats. Començarem per la inicialització de l'aplicació, és a dir, la càrrega inicial i periòdica de punts i continuarem per les diverses funcionalitats descrites

en l'anàlisi de requeriments de la secció de disseny. Per obtenir informació de les classes consulteu l'Annex 2.

7.3.1 Inicialització i càrrega:

En aquest apartat explicaré com l'aplicació inicialitza tots els seus components i com aquesta carrega tota la informació dels vehicles.
Feta, des del navegador, una petició al servidor de la pàgina web, es comença a carregar la pàgina web, és aleshores quan a través de l'Event *onLoad* de JavaScript s'activa la inicialització de components i la posterior càrrega.
La inicialització de components s'ocupa de crear nous objectes de les classes definides i donar el control del flux d'execució a cada una d'aquestes classes perquè a través dels seus mètodes s'inicialitzin.

7.3.1.1 Panell de càrrega

Per començar, es crea un Objecte de la classe **ControldeCarrega**. Aquesta classe s'ocupa de controlar tot el que tingui a veure amb la càrrega de l'aplicació. En el constructor de ControldeCarrega es crea un panell de informació que adverteix a l'usuari que esperi fins que l'aplicació s'hagi carregat del tot. Aquest panell es creat a través de l'objecte OverlayMessage[6] que mostra sobre el mapa un element div de HTML amb informació.

7.3.1.2 Càrrega de mapa

Amb el Panell visualitzant-se en el centre de la pantalla principal, passem a la gestió general de vehicles. Aquesta funció la desenvolupa la classe **Vehicles,** el constructor del qual crea el mapa i defineix les opcions d'aquest. També crea un Array que contindrà els identificadors de tots els vehicles i un cluster per a emmagatzemar tots els punts d'interès. Per crear el mapa utilitzem la classe GMap2, de l'API de Google Maps i li passem com a paràmetre al seu constructor el contenidor de HTML on es mostrarà el mapa.

Un cop tenim el mapa creat, hem de retardar l'execució fins que el mapa estigui carregat del tot; per aquest motiu, utilitzem el mètode de la classe ControldeCarrega, anomenat ***mapaCarrega.*** Aquest, a través del mètode *isLoaded* de la classe GMap2, comprova que el mapa estigui carregat. Si el mapa està carregat, l'execució continua; sinó, es retarda l'execució a través del mètode *delay* de Prototype i es torna a fer la comprovació amb el mètode isLoaded fins que el mapa estigui carregat.

6. OverlayMessage: És una classe que permet mostrar missatges sobre el mapa utilitzant un element HTML.

7.3.1.3 Càrrega dels punts d'interès

Amb el mapa carregat, podem donar pas a la càrrega i gestió dels punts d'interès. D'aquesta funció se n'ocupa la classe **Marcadors,** amb el seu constructor, que deshabilita l'opció de fer zoom sobre el mapa a través del doble clic. El doble clic el reservarem per afegir un punt d'interès al mapa. L'objecte creat per la classe Marcadors conté com a atribut més important: un Hash per emmagatzemar els punts d'interès pel seu nom. El mateix constructor de l'objecte Marcadors crida al mètode carregarpuntsinteres que obtè els punts d'interès de la Base de Dades i els carrega al mapa. Per realitzar aquest tasca, emprem tecnologia AJAX combinada amb PHP i XML. AJAX (Asynchronous JavaScript And XML) és una tècnica de desenvolupament web per a crear aplicacions interactives. Aquestes aplicacions s'executen en el client, és a dir, en el navegador de l'usuari mentre es mantè la comunicació asíncrona amb el servidor, en segon pla. D'aquesta forma és possible realitzar canvis sobre les pàgines sense necessitat de recarregar-les, el que significa augmentar la interactivitat, velocitat i usabilitat en les aplicacions. El PHP és un llenguatge de programació interpretat, dissenyat originalment per la creació de pàgines web dinàmiques i s'executa en el servidor (server-side scripting). El XML (eXtensible Markup Language) és un metallenguatge extensible d'etiquetes, desenvolupat pel World Wide Web Consortium (W3C). XML és un estàndar per l'intercanvi d'informació estructurada entre diferents plataformes. La combinació d'aquestes tres tecnologies, ens permet a través del llenguatge JavaScript realitzar una crida AJAX que mantè una comunicació asíncrona i en segon pla amb el servidor, el qual executa el codi PHP que s'ocupa d'obtenir de la Base de dades la informació per la qual hem fet la petició i ens retorna una resposta en format XML. Per tant, a l'hora de carregar els punts d'interès, realitzem una crida AJAX al servidor, especificant com a paràmetres que volem tots els punts d'interès emmagatzemats a la Base de Dades. El codi PHP recull aquests paràmetres i realitza una consulta a la Base de Dades per a obtenir tots el punts d'interès. Un cop ha obtingut tots els punts d'interès, construeix una resposta per a la crida AJAX en format XML. Aquesta resposta conté: latitud i longitud del punt, icona que utilitza, nom, tipus, adreça i comentaris del punt d'interès. AJAX s'ocupa de tractar aquesta resposta obtenint la informació. En el nostre cas obtenim totes les dades del punt d'interès i creem un marcador amb la classe Gmarker de l'API de Google Maps. Per a crear el marcador, primer creem la icona amb la classe Gicon, després les mides amb la classe Gsize i la ruta de la imatge obtinguda de la resposta del servidor, finalment creem un punt geogràfic en el mapa amb la classe GlatLng. Un cop tenim tota aquesta informació, instanciem un nou objecte de la Classe Gmarker, passant-li com a paràmetres al constructor el punt GlatLng, la icona Gicon i l'opció de poder arrossegar-la a una altra localització. I per últim, registrem els events pel marcador, és a dir, associem unes certes accions per quan es produeixi un clic sobre la icona del punt d'interès, o es passi per sobre el ratolí o s'arrossegui el punt d'interès cap a una altra localització. Després d'aquestes passes, afegim el punt al Hash i al cluster de marcadors, creat per la classe Vehicles. El cluster de marcadors s'encarrega de crear i gestionar grans conjunts de marcadors segons el nivell de zoom del mapa. Els

marcadors s'agruparan en conjunts segons la distància als centres dels clusters. Quan un marcador s'afegeix, el cluster de marcadors busca la posició per aquest marcador dins de tots els clusters. Si no la troba, crea un nou cluster amb el nou marcador. A mida que el nivell de zoom va canviant, els grups de cluster canvïen donat que s'agrupen de diferentes formes segons la distància entre els marcadors que es mostra a la pantalla. En l'apartat de la gestió de punts d'interès, s'explicarà de forma més extensa la implementació de la resta de funcionalitats.

7.3.1.4 Càrrega dels polígons

Amb els punts d'interès carregats, donem pas a la càrrega dels polígons que formen part del sistema de geofencing. Carreguem primer els polígons que els vehicles, ja que a la càrrega dels vehicles necessitem els polígons per comprovar si s'ha produït alguna alerta. Per a la càrrega i la gestió dels polígons, creem un objecte de la classe ***GeoFencing***. El constructor d'aquesta classe s'ocupa d'inicialitzar els atributs de l'objecte. Els principals atributs per aquesta classe són: una variable booleana, anomenada *acabatdedibuixar,* que ens indica si el procés de dibuixar el polígon està finalitzat, un Hash anomenat *polilinesdibuixades* que emmagatzema tots els polígons utilitzant com a índex el seu nom. Inicialitzats els atributs de l'objecte creat per la classe GeoFencing, passem a carregar tots el polígons emmagatzemats a la Base de Dades. Per aquesta acció, utilitzem el mètode *carrregarpoligons* de l'objecte creat amb la classe GeoFencing. El mètode *carregarpoligons* utilitza la combinació d'AJAX, PHP i XML. Per a carregar tots els polígons es realitza una crida síncrona al servidor, és a dir, se li especifica a la crida AJAX a través d'un paràmetre que l'execució de la petició no es durà a terme en segon pla, sinó que el flux d'execució s'aturarà fins que AJAX rebi una resposta per part del servidor. Realitzem una petició síncrona ja que volem assegurar-nos que els polígons estiguin carregats abans de començar a carregar els vehicles. El servidor, un cop rep la petició, executa el codi PHP per a obtenir de la Base de dades tots els polígons i la seva informació associada. Extreta la informació de cada polígon construeix una resposta en format XML per la crida AJAX síncrona. Quan rebem la resposta a la crida síncrona d'AJAX realitzem un tractament d'aquesta. La resposta conté el nom del polígon i una llista amb tots els punts o vèrtexs que el formen. Comencem creant punts geogràfics a través de la classe GlatLng, i els anem emmagatzemant en un Array. Després de recórrer tota la llista de punts, utilitzem aquest Array per construïr el polígon amb la classe Gpolygon de l'API de Google Maps. El constructor de Gpolygon necessita un Array de punts geogràfics que representin els vèrtexs i com a opcions ens demana un color per la línia, l'amplada i l'opacitat d'aquesta, el color per omplir el polígon i l'opacitat d'aquest color. Tenint el polígon construït, emmagatzemem aquest objecte en la variable Hash polilineesdibuixades i utilitzant com a índex el nom del polígon. Aquest procés es repeteix per tots els polígons que rebem de la Base de Dades. Per últim, a aquests objectes que representen els polígons els hi registrem una sèrie d'events perquè es disparin quan es faci clic sobre ells o quan es passi el ratolí per sobre. Això permetrà editar el polígon i

que es pugui arrossegar algún dels vèrtexs del mateix a una altra localització. A l'apartat gestió de polígons, s'explicarà de forma més extensa la implementació de la resta de funcionalitats.

7.3.1.5 Càrrega de vehicles

Carregats els polígons, es pot iniciar l'execució de la càrrega dels vehicles. Per a iniciar la càrrega dels vehicles, utilitzem el mètode *creadorVehicles* de la classe Vehicles. Aquest mètode, utilitzant la combinació d' AJAX , PHP i XML s'encarrega d'obtenir tota la informació i opcions dels vehicles. A través d'una petició AJAX al servidor, es demana que s'enviï tota la informació i opcions dels vehicles. La consulta, que s'executa en el servidor i en llenguatge PHP, obté tota la informació dels vehicles que contenen punts en la taula *dades* i informació en la taula *vehicles*. Els vehicles que no tenen informació geogràfica a la taula *dades* no es mostren, encara que estiguin registrats a la taula *vehicles*. Un cop obtinguda la informació i les opcions, el servidor crea una resposta en format XML per a la petició. Aquesta resposta conté la següent infomació: identificador, sobrenom, tipus, marca i model del vehicle. I conté les següents opcions de visualització de la ruta: temps enrere en què s'ha de mostrar la ruta, path de la imatge que ha de representar la icona, velocitat màxima, nombre de minuts per una aturada. Un cop rebuda la resposta, construïm els vehicles amb tota la seva informació i les seves opcions. Totes aquestes dades són passades al constructor de la classe que construïra el vehicle, anomenada ***GeoSeguiment***. L'objecte construït per la classe Geoseguiment, és a dir, el vehicle, és guardat en un atribut Array de la classe Vehicles anomenat geovehicles. També s'emmagatzema a l'atribut Array idsveh de la classe Vehicles, la id del vehicle per a tenir constància de tots els vehicles que s'han creat. Un cop construït el vehicle. es passa a la fase de càrrega del vehicle. segons les seves opcions de ruta. Per realitzar aquesta fase, es crida al mètode *seguiment* de l'objecte construït amb la classe Geoseguiment. El mètode seguiment es pot considerar el cor de l'aplicació, ja que s'encarrega de subministrar constantment dades a l'aplicació. A part de ser l'encarregat de realitzar la càrrega inicial dels punts dels vehicles, també és l'encarregat d'actualitzar en temps real els vehicles, la ruta i la seva informació. En la càrrega inicial, el mètode *seguiment* combina AJAX, PHP i XML per a obtenir tots els punts geogràfics de la taula *dades* que compleixin les opcions. Per fer això, segons el tipus de temps anterior que es vol visualitzar la ruta, es realitzen uns càlculs o uns altres. Aquests càlculs els realitza el mètode obtenirDataSegonsOpcions de la classe Geoseguiment. Primer comprova si el nombre de temps enrere que es vol visualitzar, ha de ser respecte avui o respecte a l'última dada rebuda. Si és respecte avui, es crea un nou objecte Date que contindrà l'hora i la data actuals. Si no és respecte avui, s'accedeix a través d'AJAX, PHP i XML a la data de l'últim punt rebut per aquell vehicle. Una d'aquestes dos dates serà la data base. Tenint ja la data base, obtindrem la data resultant. Per això, només cal transformar el nombre de temps en nombre de minuts, hores, dies, setmanes, mesos o anys enrere a restar a la data base. Si l'opció de tipus de temps és tota la ruta, escollirem l'any 1970 com a data

resultant i si l'opció de tipus de temps és un dia concret, utilitzarem aquesta data com a data resultant.
Amb la data resultant retornada pel mètode obtenirDataSegonsOpcions només cal realitzar una crida AJAX al servidor perquè ens cerqui (a la taula dades) tots els punts posteriors a la data resultant. Aquesta crida retorna una resposta en format XML amb la següent informació dels punts geogràfics: latitud, longitud, velocitat, data i temps en què es va produir. La resposta és tractada pel mètode tracker de la classe Geoseguiment. Aquest mètode recorre punt per punt i els emmagatzema en els atributs Array, anomenats infopunts i punts. Mentre el tracker recorre tots els punts, comprova per cada punt si aquest és el primer, ja que s'haurà de crear amb Gmarker un marcador que indiqui on comença la ruta. També es comprova si el punt està situat en l'àrea d'algun dels polígons i, si és així, s'emmagatzema una alerta a la taula alertes de la Base de dades. Per últim, es comprova si s'ha produït una aturada, és a dir, es comprova que des de l'última dada rebuda fins la dada rebuda recentment no hagi passat un cert nombre de minuts especificats en les opcions. Si és aquest el cas, es crea un nou Array de punts per a separar-lo dels anteriors.
Finalitzada la càrrega dels punts dels vehicles, tindrem un Array on cada posició serà una part de la ruta. Cada aturada fa que es guardi la ruta realitzada fins aleshores en una nova posició de l'Array. La unió de totes aquestes parts de la ruta, ens donaria com a resultat la ruta total sense tenir en compte les aturades. Per tant, cada ruta serà representada per la unió amb una línia dels seus punts entre sí. Aquesta unió la podem representar a través de la classe Gpolyline de l'API de Google Maps. Anirem creant objectes Gpolyline i entre cada parella d'objectes afegirem el marcador d'aturada. Si la ruta no conté atudades, l'Array solament tindrà una sola posició i no caldrà cap marcador per indicar l'aturada. Com a darrera acció, en la càrrega inicial de vehicles, es crearà a l'últim punt, un marcador que representi el vehicle. L'atribut *carregatotal* que indica que s'està executant la càrrega inicial es desactivarà, o sigui, se li assignarà el valor *false*.

7.3.1.6 Càrrega del panell de controls

Carregada tota la informació i dades sobre el mapa, només cal inicialitzar els panells laterals que mostren informació de control i de visualització.

El panell esquerre permet controlar opcions de visualització del vehicle. Aquest panell és creat a partir de la classe ***MenuControl***. El constructor d'aquesta classe compon un panell a través de l'Array de vehicles creats. Per cada vehicle, crea una barra amb la seva id. Quan es clica, apareixen tots els controls de visualització (centrar el vehicle en el mapa, mostrar o ocultar el el seu recorregut, mostrar o ocultar la icona i mostrar la ruta per dies). Aquest panell és creat dinàmicament en JavaScript per a ser mostrat en l'estructura HTML de la pàgina web. Per realitzar el desplegament dels controls de visualització del vehicle (la recollida i desplegament del panell), utilitzem la biblioteca script.aculo.us que permet l'ús de controls i efectes de

visualització. En concret, utilitzem els efectes “slide” que permet fer l’efecte de recollida i desplegament d’un element i l’efecte Morph que indicant-li les propietas d’estil d’un element que es volen modificar se li aplica en un temps indicat.

7.3.1.7 Càrrega del panell de visualització

El panell dret permet la visualització d’una llista dels punts d’interès i polígons, i realitzar un seguiment constant del vehicle que escollim. Aquest panell és creat a partir de la classe ***MenuVisor***, el constructor d’aquesta a partir del l’Array de vehicles creats construeix una llista de vehicles, cada element de la llista té associat un element radio de HTML, al marcar l’element radio permet realitzar un seguiment constant de la posició del vehicle seleccionat. Aquest panell és creat dinàmicament en JavaScript per a ser mostrar en l’estructura HTML de la pàgina web. Per a realitzar el desplegament dels controls de visualització del vehicle, la recollida i desplegament del panell, utilitzem la biblioteca script.aculo.us que permet l’ús de controls i efectes de visualització. En concret, emprem l’efecte “slide” per recollir i desplegar un element i l’efecte Morph per canviar les propietats d’estil del panell en un temps determinat.

La resta de funcionalitats dels panells dret i esquerre seran explicades de forma més extensa en els apartats de visualització de punts d’interès i polígons.

7.3.2 Visualització dels vehicles

En aquest apartat explicarem com l’aplicació mostra els vehicles i va actualitzant les seves posicions i informació periòdicament. Amb la càrrega inicial dels vehicles, punts d’interès, polígons i panells laterals, ens trobem en el punt on l’aplicació es mantè a l’espera de les accions de l’usuari i de la recepció de nous moviments dels vehicles, és a dir, punts geogràfics nous pels vehicles.
Un cop realitzada la càrrega inicial, l’aplicació segueix executant el mètode seguiment de la classe GeoSeguiment. Aquest mètode realitza una crida recursiva a sí mateix, però amb un retard de 5 segons entre crida i crida. A cada execució del mètode, realitza una crida AJAX al servidor, el qual a través del llenguatge PHP consulta a la Base de Dades la última dada insertada per aquest vehicle i crea una resposta en format XML. Aquesta resposta és processada pel mètode *tracker* de la classe GeoSeguiment, o sigui, extreu la informació d’aquesta resposta i comprova si la dada rebuda és nova (diferent a l’última dada processada). Per saber si la dada és nova, utilitzem l’atribut *ultimregistre* de la classe GeoSeguiment que conté el número de fila o registre (atribut *num* a la taula *dades* de la Base de Dades) de la última punt afegit al vehicle. Si l’ *ultimregistre* és diferent al registre rebut en la resposta, significa que s’ha d’afegir el nou punt al vehicle. Si no ho és, es torna el control al mètode

seguiment, que al cap de 5 segons, farà a una crida recursiva a sí mateix. Per afegir el nou punt al vehicle, comprovem que aquest no sigui el primer que rebem.

Si és així, aquest punt s'emprarà per a representar el marcador d'inici a través de les classes GlatLng i Gmarker. Després, s'afegirà el punt a l'Array de punts infopunts i es crearà una nova polilínia, amb Gpolyline, que contindrà un sol punt. També es comprovarà si està situat dins l'àrea d'algún polígon.

Si no és el primer punt, aquest s'afegirà també a l'Array de punts infopunts, es verificarà que no s'hagi produït una aturada i s'inclourà amb el mètode insertVertex a l'últim Gpolyline creat, fent crèixer d'aquesta manera la mida de la ruta del vehicle. Si es produeix una aturada, es crearan un marcador de pausa en la posició d'aquest punt i una nova Gpolyline que representarà la ruta després de l'aturada. Per concloure, canviarem la posició del marcador del vehicle a la nova posició i eliminarem l'anterior Gmarker, creant un nou marcador en la posició actual. Totes aquestes accions, repetides aproximadament cada 5 segons, simularan l'efecte del moviment del vehicle sobre el mapa i l'augment de la seva ruta.

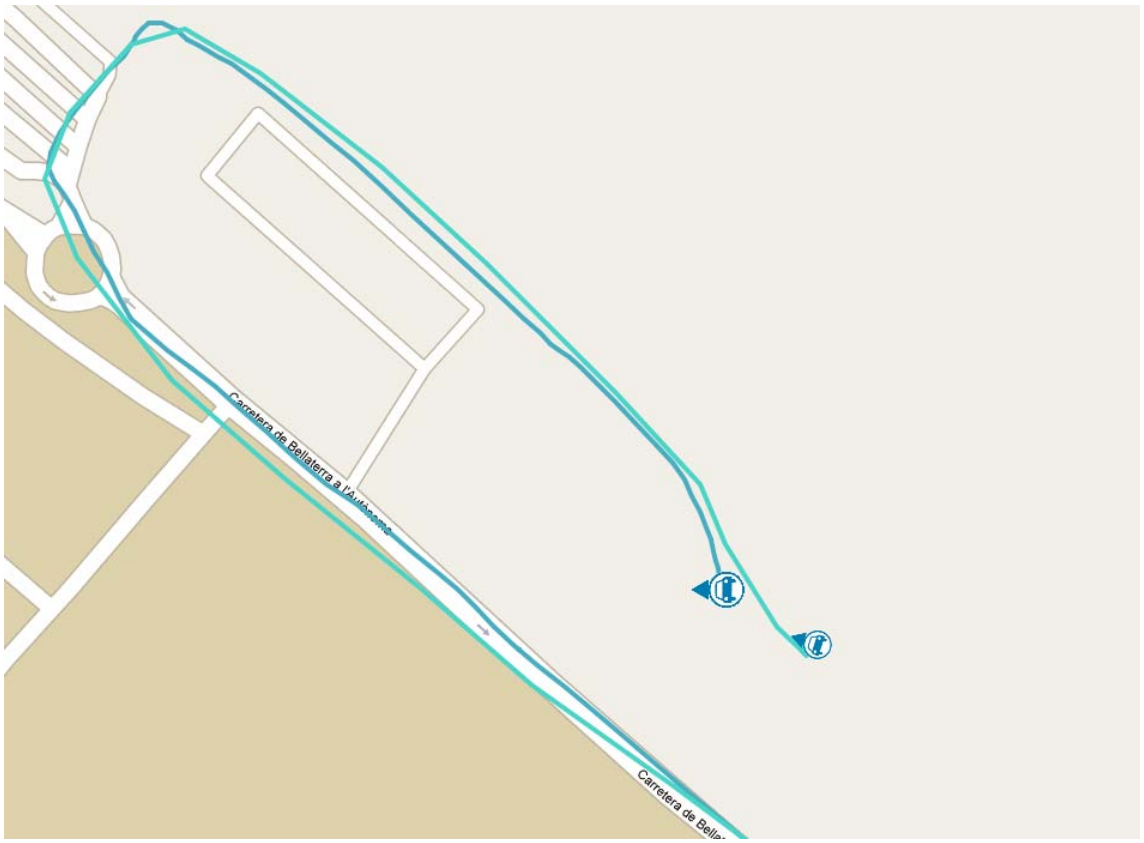


Figura 7.1: Visualització de dos vehicles sobre el mapa

7.3.3 Gestió de Polígons.

La gestió de polígons la realitzem a través d'una estructura en HTML. Aquesta conté un input per introduïr el nom del polígon, el botó per dibuixar i un selector amb tots els polígons existents. A més, de tres botons: un per guardar el polígon dibuixat a la Base de dades, un altre

per eliminar el polígon i un tercer per mostrar el polígon centrat en el mapa. Tota aquesta estructura en HTML és mostrada en una finestra creada per la classe *Window,* basada en el framework Prototype i inspirada pel potencial de la llibreria script.aculo.us. Aquesta finestra és un element *div* que permet introduïr contingut HTML, canviar el tamany de la finestra, minimitzar, maximitzar i mostrar diversos efectes visuals.

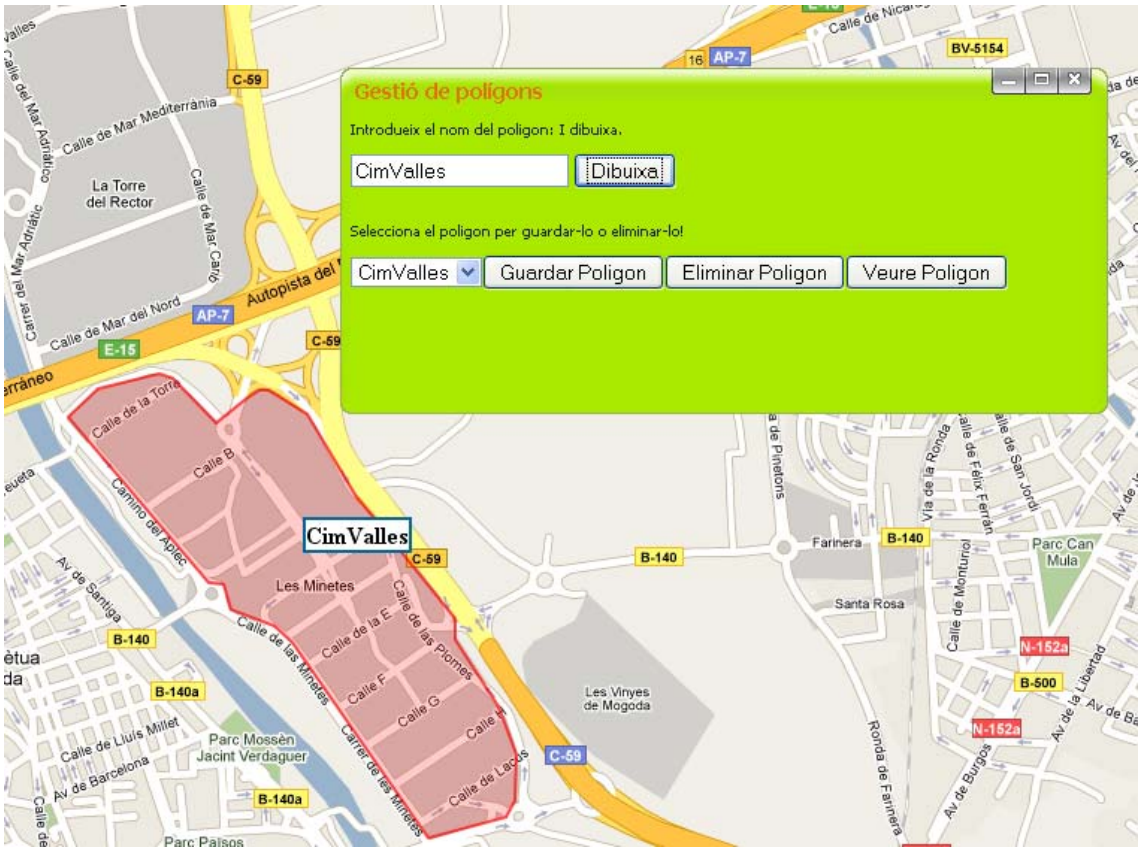


Figura 7.2: Finestra de gestió de polígons

7.3.3.1 Crear Polígon:

Per començar a crear (dibuixar) un polígon, només cal introduïr el nom del polígon i clicar en el botó Dibuixar. Després del clic, s'executa el mètode *verificanom* de la classe GeoFencing que s'encarrega de comprovar el nom, a partir d'una Expressió Regular definida. Aquest nom pot contenir majúscules, minúscuyles i, com a mínim, 3 caràcters. Si el nom és acceptat, cumpleix l' expressió regular. Llavors comprovem si s'ha utilitzat el mateix nom per a crear un altre polígon. Si és afirmatiu, s'adverteix a l'usuari i sinó es passa al mètode *editaPolyline* de la classe *GeoFencin.* Aquest últim crea un objecte de la classe *Gpolygon* amb un Array buit de punts i especifica el color del polígon, l'amplada de la línia, l'opacitat, el color per omplir el polígon i la seva transparència. Abans que l'usuari pugui començar a marcar els vèrtexs del polígon, s'ha de verificar que no hi hagi cap altre polígon dibuixant-se. Aquesta verificació l'executa l'atribut *acabatdedibuixar* de la classe GeoFencing.

Per iniciar la marca dels vèrtexs del polígon, cridem al mètode *adibuixar* de la classe GeoFencing. Aquest emmagatzema en el hash *polilinesdibuixades* el polígon creat i utilitza com a index el seu nom. Habilita el mètode *enableDrawing* i associa els events "mouseover" amb el mètode *enableEditing* i "mouseout" amb el mètode *disableEditing* de *Gpolygon*. També associa l'event "endline" que es dispara quan, a l'hora de dibuixar, unim el primer amb l'últim. Això significa que s'ha finalitzat el dibuix del polígon. És ara quan s'indica que s'ha finalitzat el dibuix del polígon a través de l'atribut acabatdedibuixar. Tot seguit, es mostra el polígon sobre el mapa mitjançant el mètode addOverlay de la class GMap2 i s'associa l'event "click" amb el polígon.
L'event click del polígon s'explica més avall en l'apartat modificar polígon.
Quan tenim el polígon mostrat sobre el mapa i amb els seus events associats, podem emmagatzemar-lo, clicant sobre el botó Guardar Polígon, a la taula *poligons* de la Base de dades. Aquest botó realitza una crida al mètode *controlDibuixar* de GeoFencing que agafa el nom escollit en el selector i crida a *guardarpoligonsbbdd* de la classe Geofencing. El mètode guardarpoligonsbbdd associa un nou event al polígon anomenat "lineupdated" que es dispara quan un vèrtex del polígon és arrossegat a una altra posició. Aquest event s'explicarà de forma més detallada en l'apartat modificar polígon. Amb l'event "lineupdated" associat al polígon, es dóna pas a la crida AJAX al servidor, el qual a través del nom del polígon i els seus punts obtinguts, via mètode getVertex de la classe Gpolygon, insereix un nou registre en la taula *poligons*.

7.3.3.2 Modificar Polígon:

En aquest punt s'explica la implementació de les accions que es produeixen quan es dispara l'event "click" i "lineupdated", modificadors del polígon.

L'event "click" es dispara quan es clica sobre un vèrtex. Al fer clic, es passa l'índex del vèrtex com a paràmetre i s'adverteix a l'usuari amb una finestra de confirmació que el vèrtex clicat s'esborrarà. Si s'accepta, aquest s'elimina amb el mètode *deleteVertex* (Gpolygon) indicant-li l'índex del vèrtex que cal suprimir.
Si cliquem en l'àrea del polígon, creem una etiqueta amb el nom d'aquest. Això es duu a terme amb l' ajuda de la classe Elabel que possibilita visualitzar elements HTML sobre el mapa.

L'event "lineupdated" es dispara quan s'arrossega un vèrtex del polígon. Aquest alerta a l'usuari que ha modificat un polígon i li demana si vol guardar els canvis. Si l'usuari accepta, es crida al mètode *guardarpoligonsbbdd* en mode actualització. Dit d'una altra manera, obtenim la nova localització dels vèrtexs amb el mètode getVertex, el qual retorna un objecte GLatLng. Amb aquesta nova informació es fa una crida AJAX al servidor que realitza una consulta d'actualització per aquell polígon.

7.3.3.3 Eliminar Polígon:

Per a eliminar un polígon, només cal seleccionar-lo amb el selector de polígons de la finestra de gestió i clicar sobre Eliminar. Aquest botó crida al mètode controlBorrar de la classe GeoFencing i adverteix a l'usuari, amb una finestra de confirmació, que s'està a punt de eliminar el polígon. Si l'usuari accepta, s'esborra l'objecte polígon del mapa a través del mètode removeOverlay (GMap2), també s'elimina l'objecte emmagatzemat en el hash polilinesdibuixades i el registre corresponent de la Base de Dades.

7.3.4 Visualitzar Gràfic de Velocitat

En aquest apartat explicarem la implementació del gràfic de velocitat per a cada vehicle. Amb l'ajuda de la llibreria Flotr, basada en el Framework Prototype, que permet dibuixar gràfics en dues dimensions, vaig començar a tractar les dades obtingudes perquè puguessin ser mostrades. El mètode GraficVelocitat de la classe GeoSeguiment és l'encarregat de tractar i mostrar les dades sobre el gràfic. Les dades de velocitat, data i temps estaven emmagatzemades en l'Array d'informació de punts (infopunts). Per això, vaig construïr dos Arrays per a cada eix del gràfic. Un anomentat *velocitats* que contindria totes les velocitats i l'altre anomenat *ticks* que contindria les dates i les hores. Per tant, aquests dos Arrays tindrien la mateixa llargada i fomarien una correspondència entre la velocitat, la data i l'hora. Extreta la informació necessària de l'atribut infopunts, només havia de configurar el gràfic. Per a configurar-lo, vaig utilitzar el mètode *draw* de l'objecte Flot. A aquest li vaig indicar que com a eix "y" utilitzés l'Array velocitats i com a eix "x" utilitzés l'Array ticks. Igualment, amb el mètode draw vaig configurar l'event "mouseover" a fi de què el gràfic mostrés la velocitat i la data d'un punt quan el ratolí estigués al damunt. Com a última configuració, vaig associar l'event "click" amb el gràfic. Aquest em passaria, com a paràmetre, l'índex de l'Array de velocitats i accediria a l'atribut *infopunts,* obtenint l'objecte GlatLng que em permetés crear un marcador (Gmarker) i centrar el mapa amb el mètode setCenter de GMap2.
Tot aquest procés, em mostrava la posició exacta sobre el mapa quan el vehicle assolia una determinada velocitat, data i hora concretes. Per últim, vaig crear una finestra amb la Classe Window que mostrés el gràfic de velocitats.

7.3.5 Visualitzar ruta per dies

En aquest apartat s'explicarà la implementació de la visualització de la ruta per dies. Per mostrar la ruta per dies necessitava un calendari implementat en JavasScript i a poder ser basat en el Framework Prototype. En la cerca de diferents calendaris ja implementats, vaig trobar CalendarView, el qual em permetia incorporar un calendari integrat a la meva web. A la versió original de CalendarView li vaig realitzar un nombre de modificacions perquè s'adaptès

al funcionament requerit. A l'hora de construïr el calendari amb el mètode *setup* de l'objecte *Calendar,* li passava com a paràmetre un objecte anomenat *vehicle* que representava el vehicle del qual voliem visualitzar la ruta per dies. L'objecte servia perquè, abans de construïr el calendari, es cridés al mètode DatesRecórrergudes de l'objecte vehicle (GeoSeguiment). Aquest s'encarregava de recórrer tot l'atribut Array *infopunts* i de transferir la informació a un nou Array *datesrecorregudes,* on cada posició contingués un dia diferent i estigués ordenat de forma cronològica. Amb l'Array datesrecorregudes mostrava sobre el mapa la ruta amb un color diferent per cada dia i intervenia en la construcció del calendari utilitzant el mateix color pels dies on el vehicle hagués circulat. A part de modificar la construcció del calendari, també vaig canviar la gestió de l'event "click". Quan clicava un dia, verificava si aquest era un dels dies seleccionats. Si era així, es deseleccionava amb el mètode *VeureOcultarRecórrergutXTemps* i ocultava la ruta d'aquell dia. En el cas que el dia estigués deseleccionat, el seleccionava de nou, i mostrava la ruta d'aquell dia amb el mètode *VeureOcultarRecórrergutXTemps*.

7.3.6 Gestió d'alertes i notificacions

Aquí tractaré la gestió de les alertes i les notificacions als contactes registrats. La gestió d'alertes i notificacions es realitza al mètode tracker de la classe Geoseguiment. Aquest, com ja he comentat, verificava si el nou punt estava dins de l'àrea delimitada per algun dels polígons creats. El mètode *dinspoligons* s'ocupa de realitzar aquesta funció i de gestionar les alertes i les notificacions. *Dinspoligons* forma part de la classe GeoFencing. Aquest rep els següents paràmetres: el punt (GlatLng), la id del vehicle, la data i hora i un indicador per saber si estem a la càrrega inicial o no. Amb aquesta informació, recorrem tot el hash de polilinesdibuixades i comprovem, amb el mètode Contains[7], si el punt està dins d'aquell polígon.

Si el punt forma part d'aquest polígon, es comença a preparar la gestió d'alertes, la qual n'extreu: la longitud i la latitud (lat i lng de la classe GlatLng).

Per crear l'alerta utilitzarem: el nom del polígon, la id del vehicle, la latitud i longitud i la data i l'hora. És llavors quan es realitza una crida AJAX al servidor, que verificarà si l'alerta ha estat enregistrada. Si no és així, s'emmagatzema a la taula alertes de la Base de Dades.

Les alertes produïdes poden ser consultades gràcies al mètode Alertes (GeoFencing), el qual crea una finestra que permet consultar les alertes segons els polígons on s'ha produït, els vehicles que l'ha produït o totes les alertes. Aquestes consultes són realitzades amb una crida AJAX al servidor indicant el paràmetre que volem consultar: id del vehicle, nom polígon o totes les alertes. Aleshores el servidor amb el llenguatge PHP realitza la consulta i construeix una resposta XML amb les dades resultants. Aquesta resposta es processada pel navegador i mostrada en una taula HTML.

Les notificacions consisteixen en enviar un avís d'alerta als contactes registrats d'un cert polígon. La notificació es realitza amb el mètode dinspoligons (GeoFencing) i solament es produeix quan la càrrega inicial ja ha acabat, és a dir, es notifica al contacte solament quan l'aplicació està processant nous punts en Temps Real. La notificació consisteix en realitzar una

7. Contains: Mètode addicional de la classe Gpolygon que permet comprovar si un punt és dins de l'àrea del polígon.

petició AJAX al servidor amb les dades de l'alerta. El servidor s'encarrega, amb aquestes dades, de cercar els correus electrònics dels contactes subscrits al polígon on s'ha produït. Amb aquestes dades, es construeix un correu electrònic amb la llibreria phpmailer. Al cos d'aquest correu s'indica en quin polígon s'ha produït, el vehicle causant i una imatge de google maps static, que mostra un marcador del lloc.

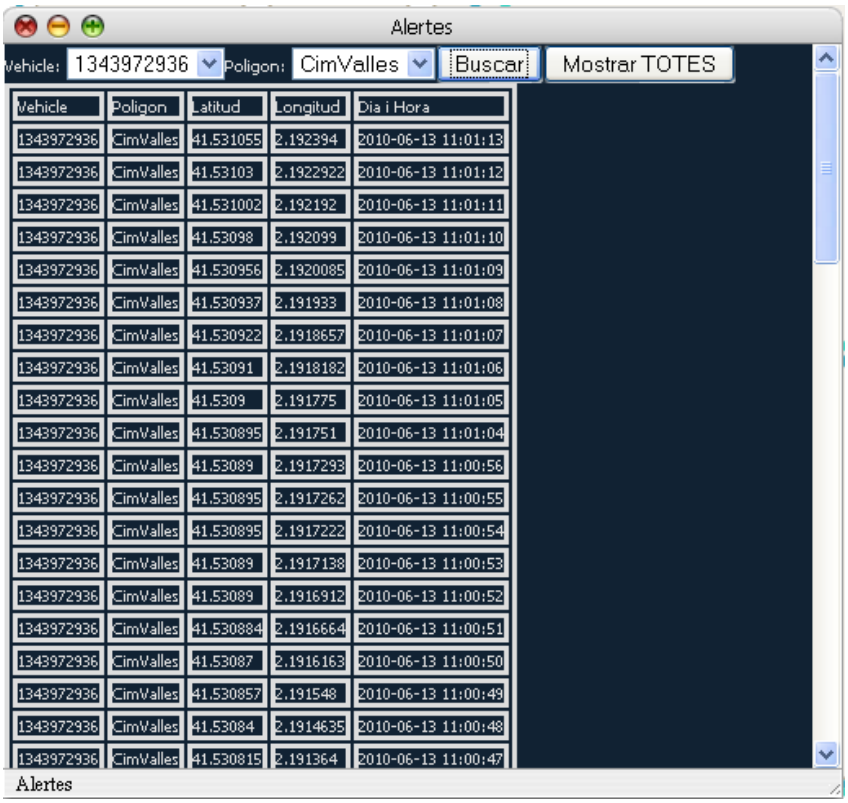


Figura 7.3: Visor d'alertes

Alerta llançada pel vehicle 1343972936 en el poligon CimValles Safata d'entrada | X

de **Servei de Geolocalització** <serveigeolocalitzacio@gmail.com>
per a martinfarre@gmail.com
data 13 de juny de 2010 12:00
assumpte Alerta llançada pel vehicle 1343972936 en el poligon CimValles
enviat per gmail.com
signat per gmail.com
Les imatges d'aquest remitent es mostren sempre. No les mostris d'ara endavant.

Alerta llançada pel vehicle: 1343972936 en el poligon CimValles en el dia i hora :2010-06-13 11:00:56

Figura 7.4: Notificació d'alerta

7.3.7 Gestió punts d'interès

Ara explicaré com s'implementa la gestió dels punts d'interès, és a dir, com es creen, es modifiquen i s' eliminen.
Per entendreu millor, dividiré en el següents punts l'explicació d'aquestes funcionalitats: Crear punt d'interès, modificar punt d'interès i eliminar punt d'interès.

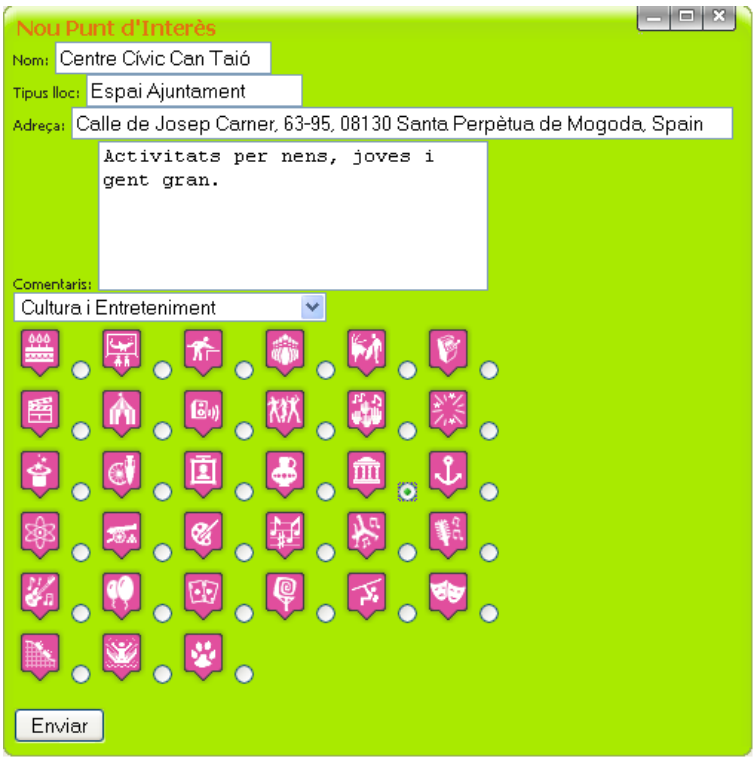


Figura 7.5: Creació d'un punt d'interès

7.3.7.1 Crear punt d'interès.

Durant la càrrega inicial dels punts d'interès, es produeix l'execució del mètode *registraEvent* (Marcadors). Aquest mètode associa l'event "dobleclick" sobre el mapa amb la creació d'un punt d'interès. Si fem doble clic, la funció, creadora del punt d'interès, rep com a paràmetre un objecte (Glatlng) que representa la posició geogràfica on s'ha fet doble clic. Aquest objecte ens permet obtenir l'adreça del punt via mètode *inversegeocoding* (Marcadors). El mètode *inversegeocoding* realitza una crida AJAX al servidor enviant-li com a paràmetres la latitud i la longitud del punt. El servidor utilitza PHP amb la llibreria *Curl*, permetent connectar i comunicar-nos amb diferents tipus de servidors i protocols. La llibreria *Curl* realitza una petició al servidor de Google Maps indicant-li la latitud i la longitud del punt. El servidor de Google Maps ens retorna una resposta en format XML la qual conté la informació sobre aquell punt. D'aquesta informació extreiem l'adreça i creem una resposta en format XML pel navegador que ha realitzat la crida AJAX.

Amb l'adreça rebuda, creem un formulari en HTML amb diversos inputs: nom, tipus, adreça i comentaris del punt d'interès. L'input de l'adreça s'omple amb les dades obtingudes del procés de inversegeocoding. A més a més creem un selector que ens permeti escollir diferents tipus d'icones a través d'un element radio, associant-li l'event "change". Cada cop que canviem de selecció es carregaran uns altres tipus d'icones.
Tota aquesta informació vindrà mostrada en una finestra, creada per la classe Window. Quan l'usuari hagi omplert tots els inputs del formulari i hagi seleccionat una icona, podrà clicar sobre el botó Enviar. Aquest botó cridarà al mètode *enviarformulari,* que recollirà tota la informació per enviar i realitzarà una crida AJAX al servidor. Després, verificarà que no existeixi un altre punt d'interès a la taula puntsdinteres amb el mateix nom. Si no és així, afegirà un nou registre amb tota la informació enviada a la taula puntsdinteres. Afegit el punt d'interès a la Base de dades, aquest es construirà amb el mètode *registrarMarcador*.
El mètode *registrarMarcador* rebrà com a paràmetre tota la informació del marcador, i amb ella crearà un nou marcador en el mapa (amb Gmarker). Al constructor de la classe Gmarker li passarem un Gicon construït amb el path de la icona seleccionada, el punt geogràfic GlatLng on es vol situar el marcador i les opcions per habilitar l'arrossegament del marcador.
Després de la creació del marcador, aquest s'afegirà a l'atribut hash *puntsinteres* de la classe Marcadors, utilitzant el nom com a índex i se li registrarà l'event "click", "mouseover", "mouseout" i "dragend".
Els events "mouseover" i "mouseout" serviran per a que es crei o es destrueixi un Elabel amb el nom del punt d'interès. I l'event "click" i "dragend" els explicaré en l'apartat modificar el punt d'interès.

7.3.7.2 Modificar punt d'interès.

En aquest punt s'explicaré les dues maneres de modificar un punt d'interès: arrossegant el punt d'interès a una altra localització o actualitzant a través d'un formulari les seves dades.

L'event "dragend", associat al marcador del mapa, es dispara quan la icona del mapa s'arrossega d'una localització a una altra. Un cop es deixa caure la icona del marcador en una altra localització, s'executa el mètode *arrossegar* al qual se li passa com a paràmetre la nova posició (GlatLng) i es mostra una finestra de confirmació advertint que el punt d'interès ha canviat de posició. Si l'usuari refusa, s'utilitza el mètode *setLatLng* (Gmarker) per a tornar a la posició anterior. Si accepta, automàticament l'objecte Gmarker canvia la posició del marcador. Es realitza un inversegeocoding amb la latitud i longitud de la nova posició i s'obté la nova adreça. És llavors quan es realitza una petició AJAX al servidor, utilitzant com a paràmetres: nom del punt d'interès, nova adreça, nova latitud i longitud. El servidor actualitza els camps adreca, latitud i longitud de la taula puntsinteres.
L'event "click" es dispara quan es clica sobre qualsevol marcador mostrat sobre el mapa. La funció que gestiona l'event s'ocupa de realitzar una petició AJAX al servidor, utilitzant com a

paràmetre el nom del polígon clicat. El servidor mitjançant el nom, retorna una resposta en format XML amb totes les dades del punt d'interès. Aquestes dades són tractades i extretes de la resposta amb el mètode *recuperadades*, el qual construeix una finestra amb la classe Window i l'estructura en HTML idèntica al formulari per crear un punt d'interès. Cada input del formulari s'omple amb la informació rebuda pel servidor, la qual cosa permet a l'usuari canviar la informació que desitgi. Després d'haver-se modificat les dades, l'usuari pot clicar sobre el botó enviar (enviarDades) per actualitzar les dades al servidor. El mètode enviarDades recull totes les dades del formulari i les envia en una petició AJAX al servidor, el qual utilitza aquestes per actualitzar el registre de la taula puntsinteres del punt d'interès modificat. Realitzada l'actualització de la informació a la Base de dades, el navegador actualitza la informació del marcador mostrat sobre el mapa, com, per exemple, la icona que el representa.

7.3.7.3 Eliminar punt d'interès.

Per a eliminar un punt d'interès, hem de clicar sobre la seva icona del mapa i, així, sen's desplegarà la finestra de modificació de les dades. Només hem de fer clic sobre el botó Eliminar. Aquest botó realitza una crida al mètode *eliminardades* (Marcadors) que processa una petició AJAX al servidor amb el nom del punt d'interès desitjat i el servidor esborra el registre corresponent de la taula puntsinteres. Esborrada la informació del punt d'interès, només cal que el navegador suprimeixi del mapa la icona. Per fer això, s'utilitza el mètode removeOverlay (GMap2) i s'esborra la posició del punt d'interès dins del hash de puntsinteres.

7.3.8 Seguiment del vehicle

En aquest apartat explicaré com es realitza un seguiment constant de la posició d'un vehicle. El panell dret, construït per la classe MenuVisor també crea una pestanya anomenada Seguiment. A partir de l'Array de vehicles creats, elabora una llista de vehicles amb un element radio associat. A cada element li registrem un event "click". Quan es clica sobre la id del vehicle s'executa l'efecte "slide" de la llibreria script.aculo.us la qual ens permet mostrar informació addicional (posició actual (latitud i longitud), velocitat, última dada rebuda (dia i hora) i nombre d'aturades realitzades). Tota aquesta informació l'extraiem de l'última posició de l'atribut Array *infopunts* (informació geogràfica del punt (GlatLng), velocitat del vehicle i l'objecte Date amb la data i hora del punt). Per a extreure la informació del nombre d'aturades realitzades, només cal obtenir la llargada de l'atribut Array *polilinees*.

Quan es clica sobre l'element radio corresponent a un vehicle, aquest executa el mètode *seguimentVehicle* de la classe MenuVisor. El mètode seguimentVehicle recorre tot l'Array de vehicles creats i activa el Seguiment pel vehicle seleccionat i desactiva el Seguiment per a tots els altres vehicles. Per activar el seguiment del vehicle, utilitzem el mètode activarSeguiment de la classe GeoSeguiment. Aquest mètode activa l'atribut *vistaseguir* que permet que a cada

execució del mètode *tracker* es centri la vista del mapa (panTo de GMap2) a l'últim punt rebut pel vehicle (posició actual del vehicle). Centrada la posició del vehicle, s'actualitzen les dades del vehicle a la pestanya seguiment del Menuvisor amb el mètode *actualitzarDadesTR*.

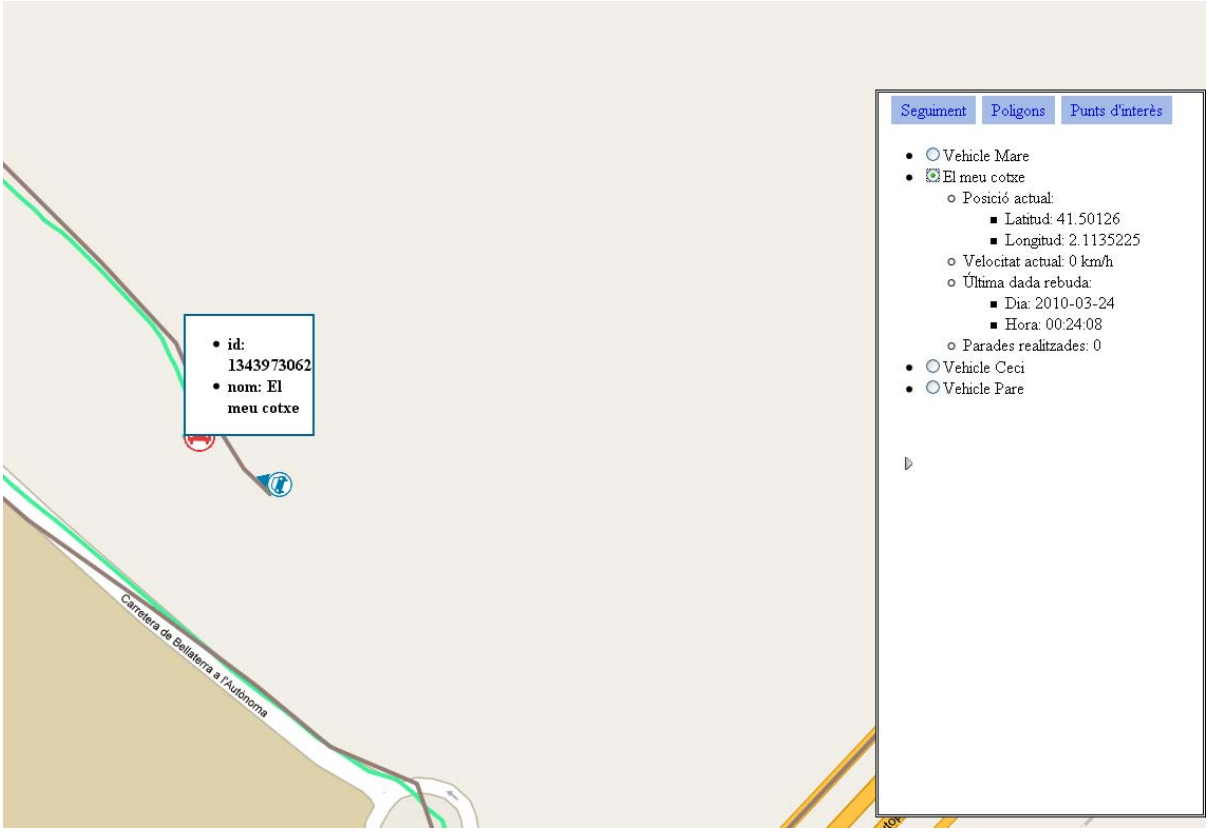


Figura 7.6: Seguiment d'un vehicle

7.3.9 Visualitzar ruta amb Street View

En aquest apartat comentaré com es crea el visor de la ruta amb la tecnologia Street View.
Per començar, instanciem la classe GeoStreet, on el constructor d'aquesta crea una finestra (classe Window) que conté un element div pel visor i un element div pels quatre controls: Iniciar, Aturar, Continuar i selector de velocitat de reproducció. El constructor crea una instància de la classe GstreetViewClient, que ens permetrà buscar informació a la base de dades de Street View. També una altra instància de la classe GstreetviewPanorama la qual construeix un visor flash a partir del contenidor HTML passat com a paràmetre, en el nostre cas un element div.
Un cop construïda la finestra amb els controls i el visor, l'usuari pot clicar en el botó iniciar. Quan clica aquest botó, s'inicialitzen els atributs de la classe GeoStreet i s'associen els events "initialized" i "error" a l'objecte creat amb la classe GstreetViewPanorama. L'event "initialized" es dispara cada vegada que una panoràmica està inicialitzada per a mostrar-se. L'event "error" es dispara quan es produeix algún error en la càrrega de la panoràmica. Amb aquests events registrats, donem pas a la búsqueda de panoràmiques amb el mètode *conduir* de la classe

GeoStreet. El mètode conduir utilitza el client de StreetView (GeoStreetViewClient) i obtè les dades del panorama més proper a una certa posició geogràfica donada per un objecte GlatLng. El primer cop que s'executa li passem el primer objecte GlatLng de l'atribut Array infopunts (GeoSeguiment). Un cop l'objecte GeoStreetViewClient ha obtingut les dades de la panoràmica, aquestes són processades pel mètode *recorregut* de la classe GeoStreet. El mètode *recorregut* s'ocupa d'anar recorrent l'atribut Array *infopunts,* que contindrà tots els punts de la ruta a visualitzar i calcularà els paràmetres pel mètode *setLocationAndPOV* de (GstreetViewPanorama). Aquest mètode ens permetrà establir la localització i el punt de vista (Point Of View) del visor flash. Realitzada aquesta crida, el visor mostrarà la panoràmica amb la localització més propera a l'objecte GlatLng establert. Al mètode setLocationAndPOV li passem, com a localització, la posició per la qual volem veure la panoràmica. Aquesta posició l'extraiem de l'atribut Array infopunts. A més a més, al mètode setLocationAndPOV li passem un objecte de la classe GPOV que indica el punt de vista a utilitzar per la càmara de Street View. Aquest objecte conté com atributs "yaw" i "pitch". "Yaw" és un número que representa l'orientació en graus de la càmara i "pitch" és un número que representa la inclinació en graus de la càmara. Per a calcular el "yaw", el mètode recorregut utilitza el mètode getBearing i li passa com a paràmetre el punt actual i el següent. Però per a que l'orientació sigui més exacta, calculem amb el mètode getBearing l'orientació del següent punt i del posterior al següent. Amb aquestes dos orientacions, fem la mitjana i agafem el valor mig com a orientació, el qual serà assignat a "yaw". El valor de "pitch" sempre ha de romandre constant, així que el mantindrem a 20º.

Tenint la panoràmica amb la localització, l'orientació i la inclinació de la càmara, el mètode recorregut s'ocupa de mostrar al mapa, un marcador (amb la icona d'una càmara) sobre la localització que s'està veient pel visor. Un cop es visualitza la panoràmica, s'incrementa l'índex que recorre l'Array infopunts i es realitza un delay amb el nombre de segons seleccionats pel selector de velocitat de reproducció. Després d'aquest delay, es realitzarà una crida al mètode recorregut però amb l'index de l'Array infopunts incrementat.

A més del botó Iniciar, a la barra de control, podem trobar tres controls: el d'Aturar, que assigna el valor *false* a l'atribut conduint i fa que no s'executi el codi del mètode recorregut; el de Continuar, que assigna el valor *true* a l'atribut conduint i permet que s'executi el codi del mètode recorregut. I finalment, el selector de velocitat, que mitjançant un element *select* i un event "change" associat, permet que cada cop que es selecciona un valor diferent, aquest sigui utilitzat com el retard entre les execucions recursives del mètode recorregut.

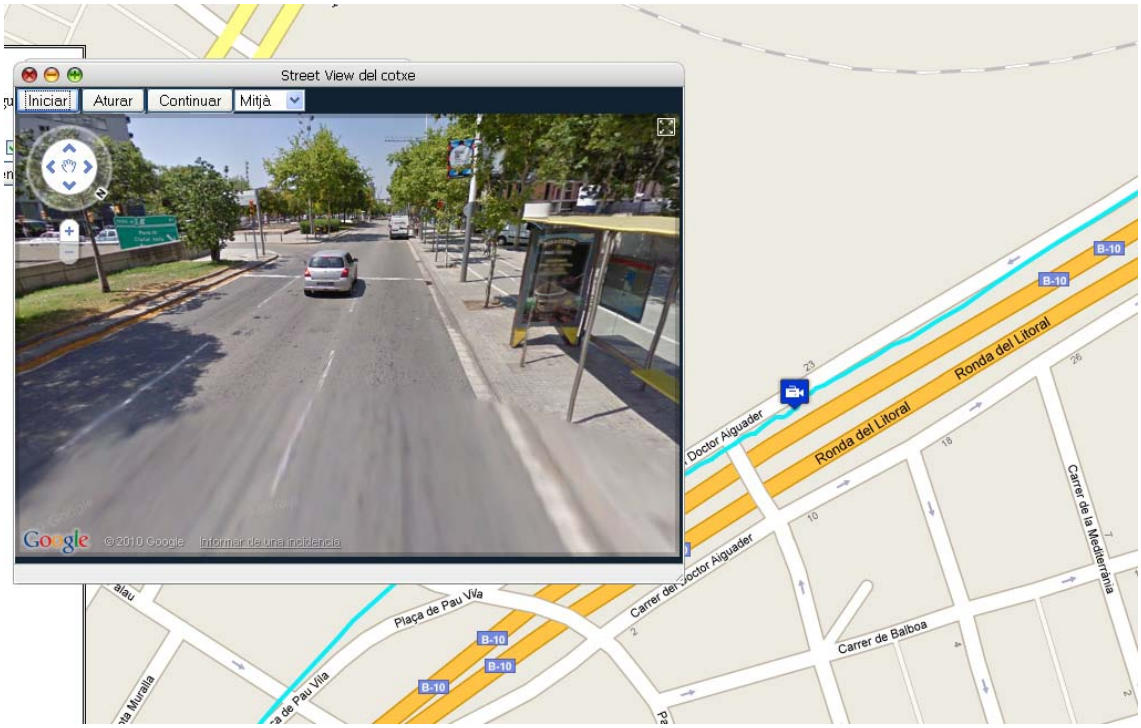


Figura 7.7: Visor Street View

7.3.10 Visualitzar ruta amb Google Earth

En aquest apartat explicaré com s'implementa el visor de la ruta amb tecnologia Google Earth. Per començar, instanciem la classe GeoEarth, on el constructor d'aquesta crea una finestra (classe Window) que conté un element div pel visor i un element div pels quatre controls: Iniciar, Aturar, Continuar i el selector de velocitat de reproducció.

Un cop construïda la finestra amb els controls i el visor, l'usuari pot clicar en el botó iniciar. Quan es clica sobre aquest botó, s'inicialitza l'atribut conduint a true i es crea una instància pel visor flash de Google Earth. La instància es crea a partir de la funció google.earth.createInstance i se li passa com paràmetre: l'identificador de l'element HTML on es mostrarà el visor, una funció que s'executi quan la instància del visor s'ha creat correctament i una funció que s'executi quan la instància del visor no s'ha pogut crear. Com a funció que s'executa quan es crea correctament el visor, li passem el mètode *iniciarInstancia*. Aquest mètode, amb l'objecte *instance* rebut per paràmetre, s'ocupa de configurar el visor per a la seva posterior execució. En el nostre cas, configurem la visibilitat del visor amb el mètode setVisibility i afegim les capes amb el mètode *enableLayerById* que mostren informació sobre carreteres, fronteres de països, edificis en 3D i terrenys en 3D. Realitzada la configuració, passem a mostrar la informació sobre el visor i per això, executem el mètode *recorregut.* Aquest mètode s'ocupa d'anar recorren l'Array de punts infopunts i canviant la ubicació de la vista de càmara, establint una latitud i una longitud amb el mètode setLatitude i setLongitude de la càmara lookAt. Per crear les línies que mostren la ruta sobre el mapa, utilitzem els mètodes

createPlacemark i *createLineString.* La combinació d'aquests amb el mètode *getCoordinates().pushLatLngAl*t, permet especificar una latitud i una longitud per a cada punt de la ruta. Amb la vista de càmara visualitzant la posició actual del punt i la ruta mostrada per una línia que uneix diferents punts, el mètode recorregut s'ocupa de mostrar sobre el mapa un marcador (amb la icona d'una càmara) que representa la localització del punt sobre el mapa Google Earth. Un cop es visualitza la vista de càmara i la ruta fins aquell moment, s'incrementa l'index que recorre l'Array infopunts i executa un delay amb el nombre de segons seleccionats en el control de velocitat de reproducció. Després d'aquest delay, es realitzarà una crida recursiva al mètode recorregut però amb l'index de l'Array infopunts incrementat.

A més del botó Iniciar, a la barra de control, podem trobar tres controls: el d'Aturar, que assigna el valor *false* a l'atribut conduint i fa que no s'executi el codi del mètode recorregut; el de Continuar, que assigna el valor *true* a l'atribut conduint i permet que s'executi el codi del mètode recorregut. I finalment, el selector de velocitat, que mitjançant un element *select* i un event "change" associat, permet que cada cop que es selecciona un valor diferent, aquest sigui utilitzat com el retard entre les execucions recursives del mètode recorregut.

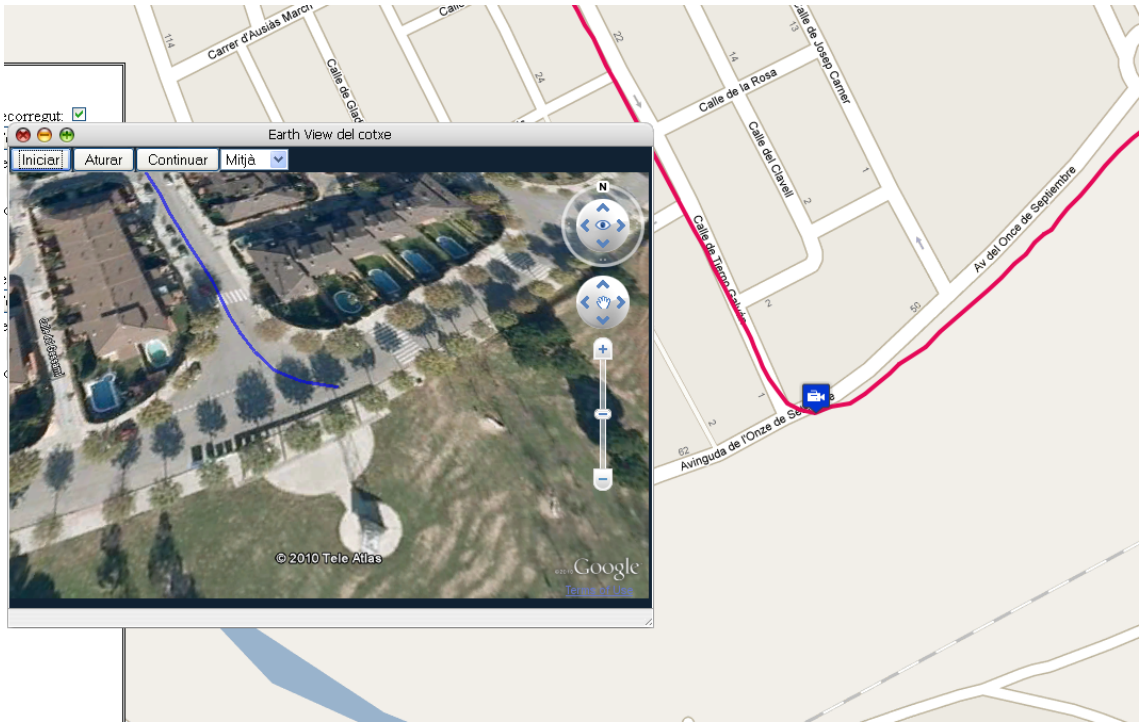


Figura 7.8: Earth View del vehicle

7.3.11 Modificar Opcions i Informació del vehicle

En aquest apartat explicaré la implementació de la visualització, la modificació de la informació i opcions d'un vehicle.

Per a crear la finestra que permet visualitzar i modificar les opcions, utilitzem el mètode crearFinestraOpcions de la classe GeoSeguiment. Aquest mètode amb la classe Window crea

una finestra amb contingut HTML. El contingut HTML que conté aquesta finestra és: data i hora del primer i últim punt rebut del vehicle. Un checkbox per a marcar si volem que el nombre de temps enrere que es visualitzi la ruta sigui respecte avui. Sinó, s'utilitzarà respecte a l'última data rebuda. Un input per a introduïr el número de temps enrere i 7 radios relacionats per a seleccionar el tipus de temps enrere (minuts, hores, dies, setmanes, mesos, anys o tot). Un input que ens permet triar, en un calendari, a partir de quin dia volem veure la ruta. Dos inputs (calendaris) i dos selectors que ens permeten seleccionar l'inici (dia i hora) i la fi (dia i hora) de l'interval de la ruta. I sis inputs per: sobrenom del vehicle, tipus, marca, model, velocitat Màxima i temps per aturada. I finalment el botó de guardar, que executa el mètode *enviarOpcions* de la classe GeoSeguiment. Aquest s'encarrega de recollir les dades segons la selecció de l'usuari i de realitzar una petició AJAX al servidor, que emmagatzema a la taula *vehicles* la informació i les opcions modificades. El servidor respon al navegador si s'ha pogut emmagatzemar correctament la informació i les opcions. Si és així, es carreguen les noves dades als atributs de l'objecte (GeoSeguiment) que representa el vehicle i es tornen a inicialitzar tots els atributs de l'objecte. S' esborren les *polilinees*, marcadors anteriors i es torna a realitzar la càrrega inicial amb les noves opcions especificades.

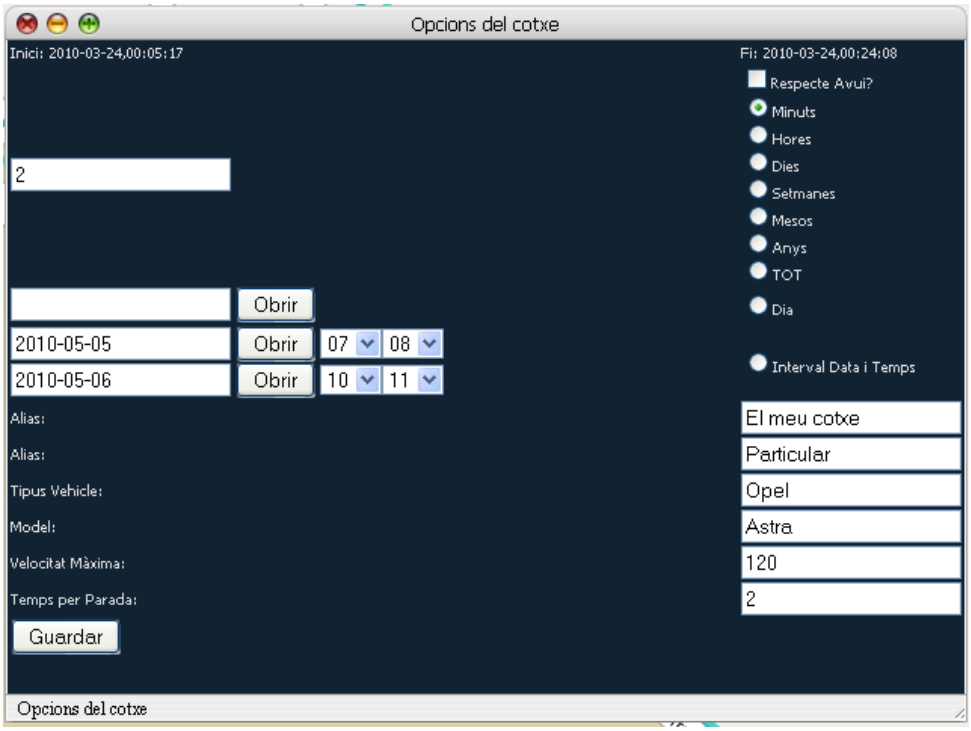


Figura 7.9: Modificar Opcions del vehicle

7.3.12 Crear Ruta

En aquest apartat explicaré com he realitzat la implementació per la creació d'una ruta d'un vehicle.

Per a construir la finestra que permet crear la ruta, vaig utilitzar el mètode *creaFinestraRuta* de la classe Rutes. Aquest mètode instancia un objecte de la classe Gdirections i un objecte de la classe GclientGeocoder. L'objecte Gdirections realitza peticions de rutes i emmagatzema el resultat de les direccions. L'objecte GclientGeocoder es communica directament amb el servidor de Google per a obtenir geocodis d'adreces especificades per l'usuari. A l'objecte GclientGeocoder li associem els events "load" i "error". L'event "load" es dispararà quan s'obtindrà un resultat per la petició de ruta realitzada amb el mètode load de Gdirections. Per contra, l'event "error" es dispararà quan es produirà un error en la petició de ruta. L'elaboració d'una ruta es mostra en una finestra (classe Window) que té com a contingut la creació ràpida i la planificació d'una ruta.

La creació ràpida d'una ruta conté un input per a introduïr l'adreça fins on s'ha de crear la ruta per aquell vehicle. Aquest input té un botó associat anomenat crear Ruta que executa el mètode BuscarQueryRuta de la classe Rutes. El mètode BuscarQueryRuta obté l'adreça introduïda en l'input i realitza una petició a través del mètode load de la classe Gdirections. Amb la resposta rebuda del servidor, s'executa l'event load que obté de l'objecte Gdirections la ruta en forma de polilínia (mètode getPolyline). De l'objecte Gdirections també obtim les instruccions en format text per a arribar al destí. Aquestes instruccions s'obtenen amb el mètode getRoute que li retorna un objecte Route, del qual recorrem cada *step* i extraiem la informació (getDescriptionHtml).

La planificació de la ruta conté una taula amb dos columnes. La primera, mostra una llista amb tots els punts d'interès i la segona és buida. Amb els efectes *draggable*, *droppable* i *sortable* de la llibreria script.aculo.us, permetem a l'usuari arrossegar els punts d'interès a la segona columna i ordenar-los formant així una ruta on cada punt d'interès serà una fita d'aquesta. Els punts d'interès seran *Draggables* mentre que la segona columna serà *Droppable*. Quan deixem arrosseguem i deixem anar un punt d'interès, aquest s'afegirà a la segona columna, i es convertirà en *sortable* per ser ordenat. Quan cliquem sobre el botó crear Ruta, s'executa el mètode *CrearRutaPuntsInteres*, que recorre tota la segona columna recollint els punts d'interès en ordre i creant una ruta amb el mètode load de l'objecte Gdirections. Quan la ruta obté una resposta, es dispara l'event "load" i s'executen les mateixes passes que per la creació d'una ruta ràpida.

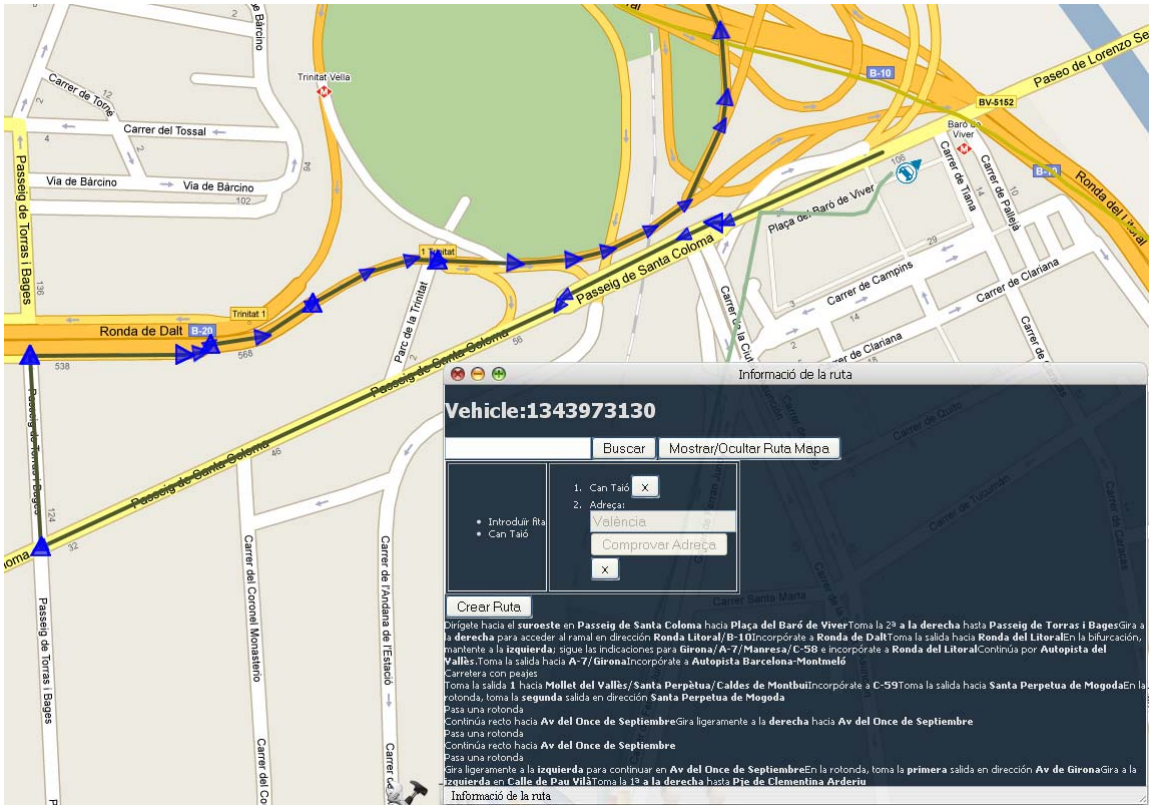


Figura 7.10: Versió preliminar crear rutes

7.3.13 Visualitzar tràfic

En aquest apartat explicaré com s'ha realitzat la implementació de la visualització del tràfic.

En un sistema de seguiment d'una flota de vehicles, el tràfic és un element molt important que pot permetre prendre decisions anticipades. Per aquest motiu, em vaig decidir a utilitzar la capa GtrafficOverlay de l'API de Google Maps, però ràpidament vaig adonar-me que aquesta capa solament disposava d'informació d'alguns països: EEUU, Anglaterra, Finlàndia, Xina i Tailàndia.

Com el servidor de Google Maps no em proveïa de la informació del tràfic, vaig decidir cercar la informació nacional, és a dir, a Espanya. Per això, buscant a la Direcció General de Tràfic algun sistema que em pogués proveïr d'informació del tràfic, vaig topar amb l'aplicació etraffic. Etraffic és un mapa creat en Google Maps, que a partir de peticions AJAX a un servidor de la DGT, obté informació en temps real del tràfic. Aquesta informació es visualitza en forma de marcadors i pot ser de diferents tipus: obres, retencions, incidències meteorològiques, panells informatius, càmares i sensors de tràfic i estacions meteorològiques. Sabent quin tipus d' informació suministrava el servidor de la DGT, vaig analitzar el codi font de l'aplicació. Gràcies a la consola de l'aplicació FireBug[8], vaig poder observar quins paràmetres i quina direcció utilitaven les peticions al servidor de tràfic de la DGT. Quan vaig voler integrar aquesta funcionalitat a la meva aplicació, va resultar que les respostes del servidor eren buides. Cercant informació a la xarxa, em vaig adonar que la tecnologia AJAX era incapaç de llençar peticions

8. Firebug: Eina per editar, debugar i monitoritzar CSS, HTML i JavaScript.

fora del domini en el que s'havia executat, la qual cosa la invalidava com a tecnologia per a utilitzar serveis web de tercers. Però vaig trobar una solució anomenada Cross-Domain, que em permetia, utilitzant un script escrit en PHP, crear un canal de transport per a les peticions AJAX. A aquest script li passo com a paràmetre la URL del servidor i els paràmetres d'aquest. La tecnologia Curl s'encarrega de realitzar una petició al servidor de la DGT i amb els paràmetres rebuts li retorna una resposta. El script retorna aquesta resposta al nostre navegador. Per retornar la resposta del servidor de la DGT, el script configura la sessió de curl amb el mètode curl_setopt i amb les opcions: CURLOPT_URL (que permet especificar la url i els paràmetres del servidor al qual volem accedir) i CURLOPT_RETURNTRANSFER (que permet retornar el resultat de la transferència com un string). Amb el problema de comunicació resolt, vaig començar a integrar el tràfic d'Espanya a la meva aplicació.

El constructor de la classe TraficdEspanya s'encarrega d' inicialitzar les icones de les imatges descarregades una a una del servidor de la DGT. A més a més, el constructor crea un Array pels marcadors que insertarem al mapa i un Array per les finestres que creearem amb la classe EWIndow[9]. Per últim, registra l'event "moveend" al mapa, el qual s'encarrega d'executar el mètode *cercarTraficEspanya* cada cop que es canvia la vista del mapa. Aquest mètode obté la latitud i longitud que representa el Nord-Est del límit de visualització del mapa i la latitud i longitud del Sud-Est del límit de visualització del mapa, és a dir, representen els punts que permeten definir l'àrea de la Terra que s'està visualitzant al mapa. Amb aquests punts, realitzem una petició AJAX al servidor de la DGT a través del script transport. Aquesta petició rebrà com a resposta tota la informació relacionada amb el tràfic però solament de l'àrea actual de visualització del mapa. La resposta rebuda pel servidor, arriba en format JSON i es tractada pel mètode *tractarResposta*. Aquest mètode recorre tots els elements de la resposta i crea per cada un d'ells un marcador sobre el mapa (mètode *crearMarcador)*. El mètode *crearMarcador* construeix un marcador amb la classe Gmarker i els objectes Gicons inicialitzats al principi. Segons si el marcador és un Panell, Camara, Sensor Meterològic o Incidència s'utilitza una icona (Gicon) diferent. A cada marcador se li associen els events "mouseover" i "mouseout" per a mostrar i ocultar etiquetes amb el nom del marcador. També s'associa l'event click, que s'encarrega de realitzar una altra petició al servidor a través del script transport per a obtenir els detalls informatius de cada marcador. A aquesta petició li passem com a paràmetre els codis identificatius de cada marcador. La resposta rebuda per part del servidor de la DGT arriba en format JSON i es tractada pel mètode tractarDetallsElement. Aquest mètode, segons el tipus de marcador que estem tractant, s'encarrega de crear una determinada estructura HTML.

Pels elements Panell:

- Una imatge informativa a l'esquerra
- Informació central
- Una imatge informativa a la dreta

9. Ewindow: Classe addicional a l'API de Google Maps que permet crear els nostres propis info windows.

Figura 7.11: Panell

Pels elements Sensors de tràfic:

- Intensitat del tràfic
- Velocitat mitjana.
- Nivell d'ocupació
- Nombre de vehicles lleugers.

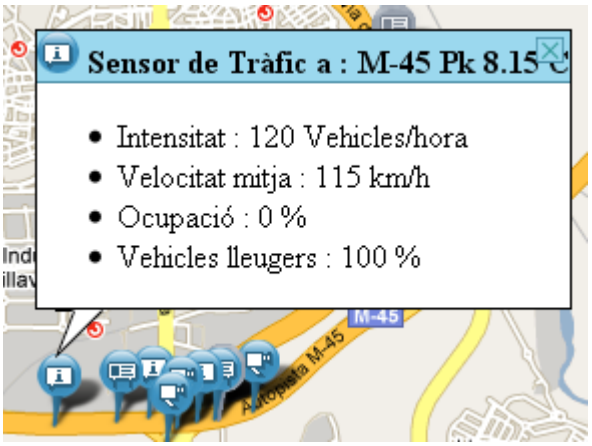


Figura 7.12: Sensor Tràfic

Pels elements Càmara:

- Una captura de la càmara de tràfic.

Figura 7.13: Càmara

Pels elements Sensor meteorològic:

- Quantitat de precipitacions
- Alçada de l'aigua
- Direcció del vent
- Estat de la superfície
- Temperatura de la superfície
- Temperatura de la rosada
- Temperatura de congelació
- Temperatura del subsol
- Nivell de Salinitat
- Radiació Global

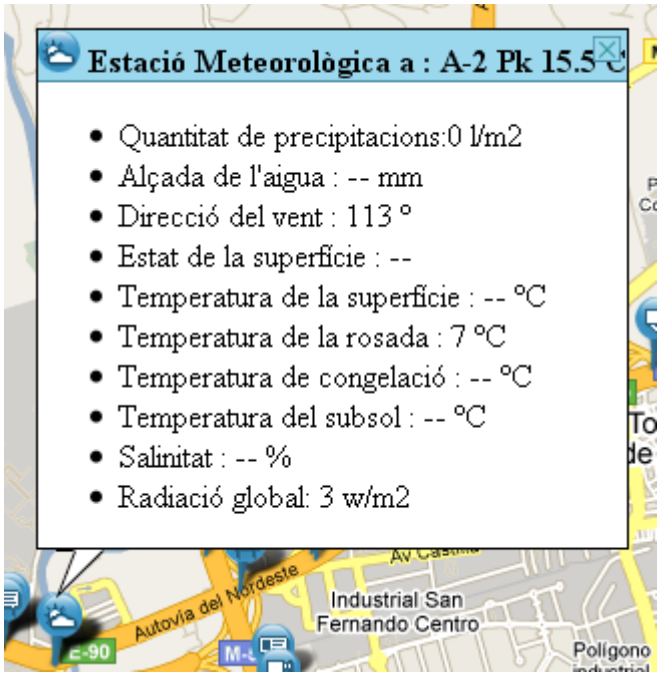


Figura 7.11: Estació Meteorològica

Pels elements Incidència:

- Carretera
- Causa
- Localitat
- Província
- Nivell
- Sentit

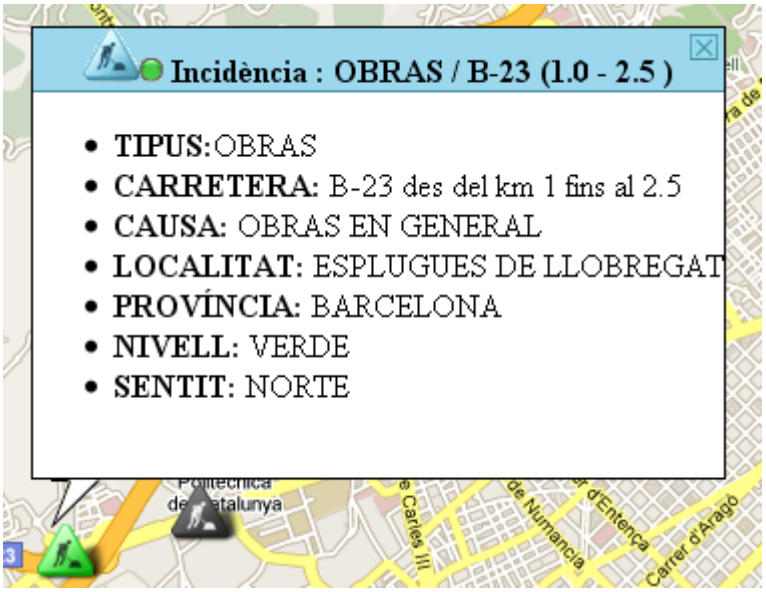


Figura 7.11: Incidència

L'estructura HTML de cada element és incorporada en un nou objecte de la classe Ewindow, el qual ens permet crear petites finestres sobre el mapa i insertar-li codi HTML.
Per a més informació, consulteu l'Annex 5.

8 Proves

La prova és l'avaluació d'un sistema de forma manual o automàtica per tal de verificar que satisfà els requisits especificats o per a identificar diferències entre els resultats esperats i els obtinguts.
Per tant, en aquesta secció realitzarem un conjunt de proves, on s'analitzaran els resultats. Com l'aplicació està dividida en dos mòduls: mòdul de captació i mòdul de processament, les proves es realitzaran sobre els dos mòduls per separat i sobre l'aplicació en conjunt, és a dir, la interacció dels dos mòduls.

8.1 Proves mòdul de captació

En aquest punt ens encarregarem de realitzar proves sobre el mòdul de captació, el qual s'encarrega de captar dades i emmagatzemar-les a la Base de Dades.
La prova realitzada sobre el mòdul de captació ha estat comprovar que aquest realitzava les seves funcionalitats sense que es produïs cap error.
Per començar hem executat l'aplicació GPServer al servidor, aquesta s'ha inicialitzat i s'ha mantingut a l'espera de la recepció de dades per qualsevol dels quatre ports oberts del servidor, en el nostre cas: 9996, 9997, 9998 i 9999.

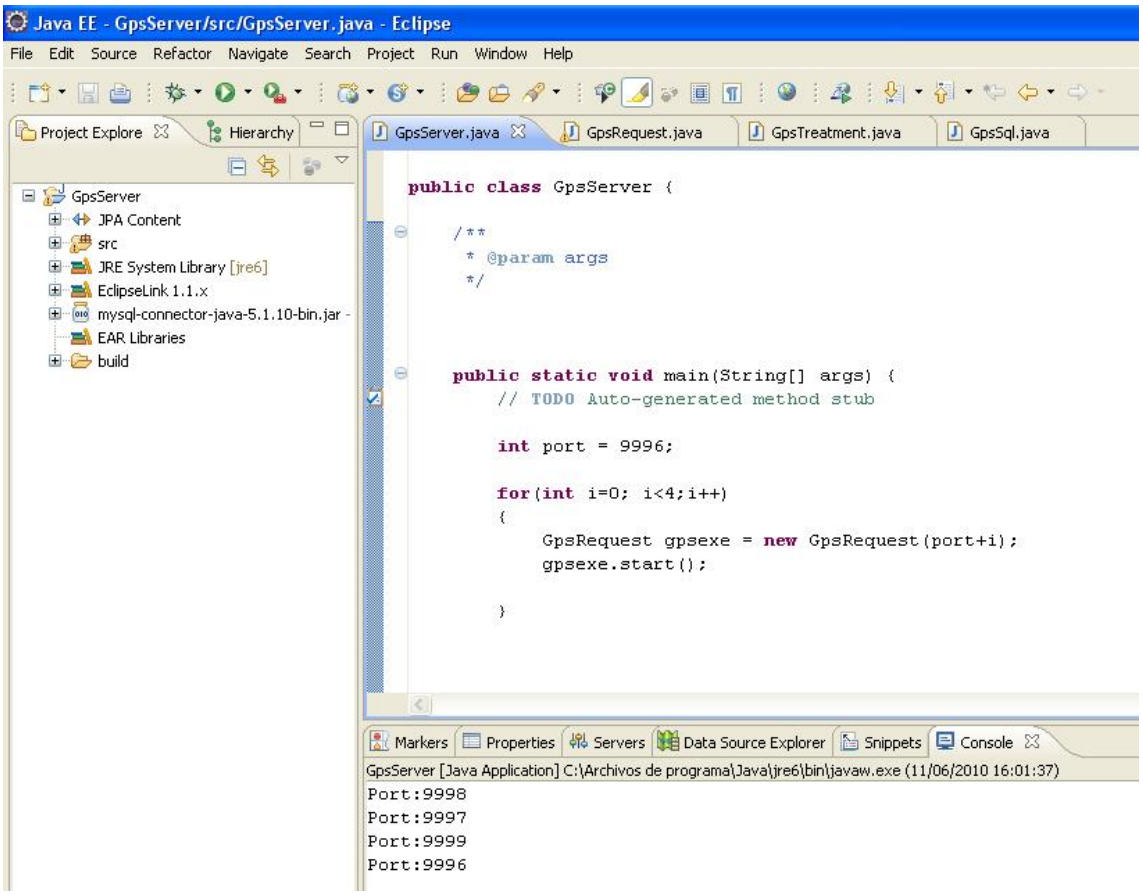


Figura 8.1: GpsServer

Com podem observar a la figura 8.1, des de l'eina Eclipse hem executat l'aplicació GPServer (mòdul de captació) per a que s'inicialitzin els ports i es mantinguin a l'espera de la recepció de dades.

Un cop inicialitzat el mòdul de captació hem donat pas a la configuració i inicialització del dispositiu mòbil amb tecnologia GPS, en el nostre cas un telèfon mòbil amb sistema operatiu Windows Mobile 6.1.

Per començar hem realitzat la connexió del telèfon mòbil a la xarxa sense fils, un cop el telèfon mòbil ha estat connectat hem donat pas a l'execució i configuració de l'aplicació GPSPlex.

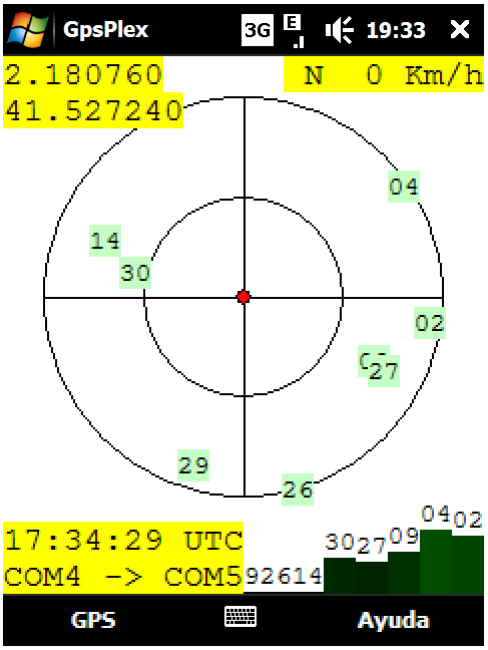


Figura 8.2: GpsPlex

Com podem veure a la figura 8.2, l'aplicació GpsPlex mostra en la part superior esquerra de la pantalla la latitud i longitud actual, a la part superior dreta la velocitat actual, a la part inferior esquerra l'hora actual rebuda pel satèl·lit i a la part inferior dreta el nombre de satèl·lits i la cobertura que tenim per cada un d'ells.

Amb l'aplicació GpsPlex iniciada només ens cal realitzar la configuració d'aquesta, per això cliquem al menú "GPS" i escollim "Nueva Connexión IP". Un cop realitzades aquestes passes podrem observar el menú que es mostra a la figura 8.3.

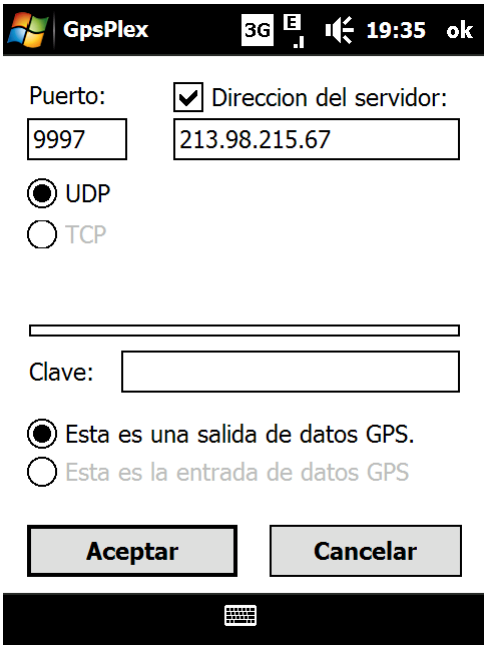


Figura 8.3: GpsPlex connexió

A la figura podem observar com la creació d'una nova connexió IP ens permet introduïr el Port del servidor i la seva direcció IP. En el nostre cas hem introduït com a Port el 9997 i la direcció del servidor 213.98.215.67. Amb la connexió IP creada, l'aplicació GpsPlex ja ha començat a enviar dades a través de la xarxa sense fils al servidor 213.98.215.67 i pel port 9997.

En aquest moment comprovem al mòdul de captació com rebem les dades pel port 9997 i com aquestes són processades.

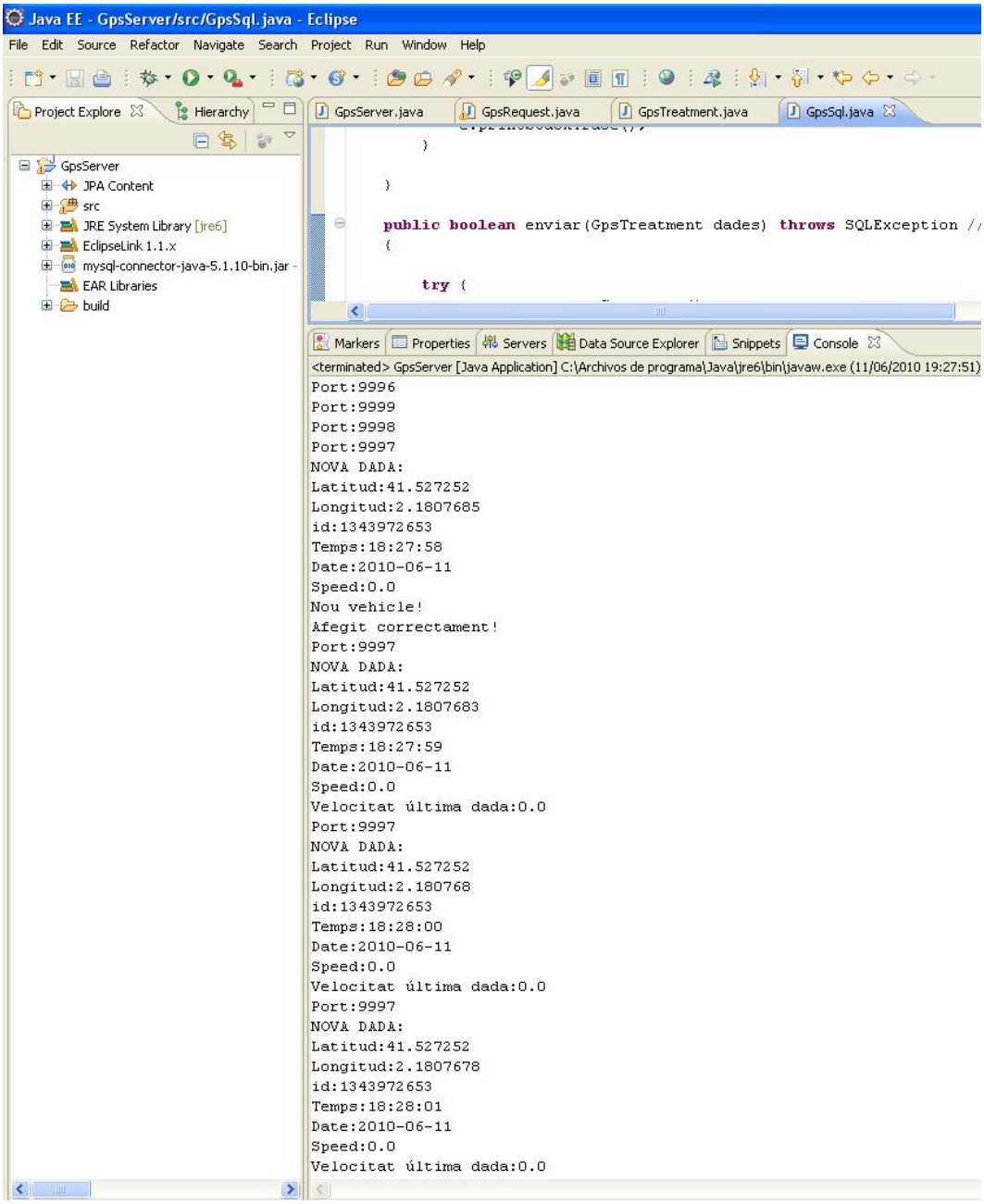


Figura 8.4: GpsServer rebent dades

A la figura 8.4 podem observar com rebem dades pel port 9997 i quin és al tractament d'aquestes, extraient la latitud, la longitud, la id, l'hora i la data. També podem observar com al ser la primera dada que rep d'aquell vehicle, el mòdul de captació ens alerta amb el missatge "Nou Vehicle". El missatge "Afegit correctament" es mostra cada cop que s'afegeix un punt a la Base de dades, en aquest cas solament observem un cop el missatge "Afegit correctament" ja que la velocitat de la primera dada del vehicle és zero i les velocitats rebudes posteriorment també són zero, per tant, no s'afegeixen aquests punts posteriors per a no omplir la Base de Dades amb localitzacions del mateix punt.
Un cop comprovem que el mòdul de captació no produeix cap error, és el moment de comprovar a la Base de Dades que aquestes dades han estat emmagatzemades correctament.

id	alias	tipusVehicle	marca	model
1343972653	Vehicle1343972653	tipus	marca	model

valorRutaTemps	dataTempsIniciRuta	dataTempsFiRuta	tipusRutaTemps
0	1980-05-06 00:00:00	1980-05-07 00:00:00	TOT

icona	velocitatMax	tempsParada	respecteAvui
icona	120	1	0

Figura 8.5: taula vehicles

A la figura 8.5 podem veure com el vehicle amb identificador 1343972653 ha estat afegit correctament a la taula *vehicles* de la Base de dades geolocalització. Les dades insertades són predeterminades i més tard poden ser modificades pel mòdul de processament.
I finalment comprovem que el punt afegit al principi ha estat emmagatzemat a la taula *dades* de la Base de dades geolocalització.

			id	num	latitud	longitud	speed	dateitime
☐	✎	✕	1343972653	2095	41.527252	2.1807685	0	2010-06-11 18:27:58

Figura 8.6: taula dades

Com podem veure a la figura 8.6, el primer punt emmagatzemat pel vehicle amb identificador 1343972653 ha estat emmagatzemat a la taula dades.

Per a comprovar que funciona amb la recepció d'una ruta, és a dir, de diversos punts, mostrarem el conjunt de punts d'una part d'una ruta.

id	num	latitud	longitud	speed	dateitime
1343973130	2096	41.527214	2.1806927	0	2010-06-12 12:36:01
1343973130	2097	41.52716	2.180833	9.26	2010-06-12 12:41:22
1343973130	2098	41.52716	2.180833	9.26	2010-06-12 12:41:22
1343973130	2099	41.527138	2.1808465	10.186	2010-06-12 12:41:23
1343973130	2100	41.527138	2.1808465	10.186	2010-06-12 12:41:23
1343973130	2101	41.5271	2.180875	12.4084	2010-06-12 12:41:24
1343973130	2102	41.5271	2.180875	12.4084	2010-06-12 12:41:24
1343973130	2103	41.526802	2.1810694	14.0752	2010-06-12 12:41:33
1343973130	2104	41.526802	2.1810694	14.0752	2010-06-12 12:41:33
1343973130	2105	41.52677	2.1810894	14.0752	2010-06-12 12:41:34
1343973130	2106	41.52677	2.1810894	14.0752	2010-06-12 12:41:34
1343973130	2107	41.526756	2.1810803	6.6671996	2010-06-12 12:41:38
1343973130	2108	41.526756	2.1810803	6.6671996	2010-06-12 12:41:38
1343973130	2109	41.526733	2.1811302	7.7783995	2010-06-12 12:41:39
1343973130	2110	41.526733	2.1811302	7.7783995	2010-06-12 12:41:39
1343973130	2111	41.526764	2.1811583	6.1116	2010-06-12 12:41:40
1343973130	2112	41.526764	2.1811583	6.1116	2010-06-12 12:41:40
1343973130	2113	41.526733	2.1811857	7.5931997	2010-06-12 12:41:41
1343973130	2114	41.526733	2.1811857	7.5931997	2010-06-12 12:41:41
1343973130	2115	41.526703	2.1812072	8.889601	2010-06-12 12:41:42
1343973130	2116	41.526703	2.1812072	8.889601	2010-06-12 12:41:42
1343973130	2117	41.52668	2.181232	9.6304	2010-06-12 12:41:43
1343973130	2118	41.52668	2.181232	9.6304	2010-06-12 12:41:43
1343973130	2119	41.52666	2.1812496	9.8156	2010-06-12 12:41:44
1343973130	2120	41.52666	2.1812496	9.8156	2010-06-12 12:41:44
1343973130	2121	41.526665	2.1812558	8.7044	2010-06-12 12:41:45
1343973130	2122	41.526665	2.1812558	8.7044	2010-06-12 12:41:45
1343973130	2123	41.526665	2.1812575	6.6671996	2010-06-12 12:41:46
1343973130	2124	41.526665	2.1812575	6.6671996	2010-06-12 12:41:46
1343973130	2125	41.52667	2.1812441	4.2595997	2010-06-12 12:41:47
1343973130	2126	41.52667	2.1812441	4.2595997	2010-06-12 12:41:47
1343973130	2127	41.52667	2.1812286	2.4076	2010-06-12 12:41:48
1343973130	2128	41.52667	2.1812286	2.4076	2010-06-12 12:41:48
1343973130	2129	41.52668	2.1812186	0	2010-06-12 12:41:49
1343973130	2130	41.526676	2.181205	2.4076	2010-06-12 12:41:56

Figura 8.7: Dades d'una ruta

En aquesta figura 8.7 podem visualitzar el conjunt de punts que formen una ruta, podem comprovar com el vehicle inicia la ruta amb velocitat zero i com a partir d'aquest punt, la velocitat, el temps i la posició del vehicle canvien. En aquesta figura també podem comprovar com quan el vehicle es deté, no s'afegeixen més punts en la mateixa posició, sinó que s'espera a que el vehicle comenci a moure's per a insertar nous punts.

8.2 Proves mòdul processament

En aquest punt realitzarem una sèrie de proves per comprovar com el mòdul de processament selecciona i tracta les dades correctament.
Dividirem el conjunt de proves segons la seva funcionalitat:

8.2.1 Proves de càrrega de vehicles

Per a comprovar que la càrrega de vehicles és correcta hem de visualitzar a la Base de Dades quants vehicles hi ha registrats.

id	alias	tipusVehicle	marca	model
1343973062	El meu cotxe	Particular	Opel	Astra
1343973084	Vehicle Ceci	Particular	Opel	Corsa
1343973130	Vehicle Pare	Particular	Renault	Espace

valorRutaTemps	dataTempsIniciRuta	dataTempsFiRuta
2	2010-05-05 07:08:00	2010-05-06 10:11:00
2	2009-11-17 00:01:00	2009-11-17 00:02:00
5	1980-05-06 00:00:00	1980-05-07 00:00:00

tipusRutaTemps	icona	velocitatMax	tempsParada	respecteAvui
MIN	icona	120	2	0
MIN	icona	130	1.5	0
MIN	icona	120	2.5	0

Figura 8.8: vehicles a la taula *vehicles*

Com podem veure a la figura 8.8 hi ha tres vehicles registrats amb identificadors: 1343973062, 1343973084 i 1343973130. En aquest moment hem d'executar el mòdul de processament i comprovar que es carreguen tres vehicles, un cop executada l'aplicació comprovem a través de l'avís inicial que s'està produïnt la càrrega i que s'han carregat tres vehicles, com podem veure a la figura 8.9.

Figura 8.9: Panell inicial de càrrega

Però per a comprovar totalment que aquests vehicles estan carregats del tot a l'aplicació, cerquem al panell esquerra i a la pestanya *seguiment* del panell dret quants vehicles apareixen.

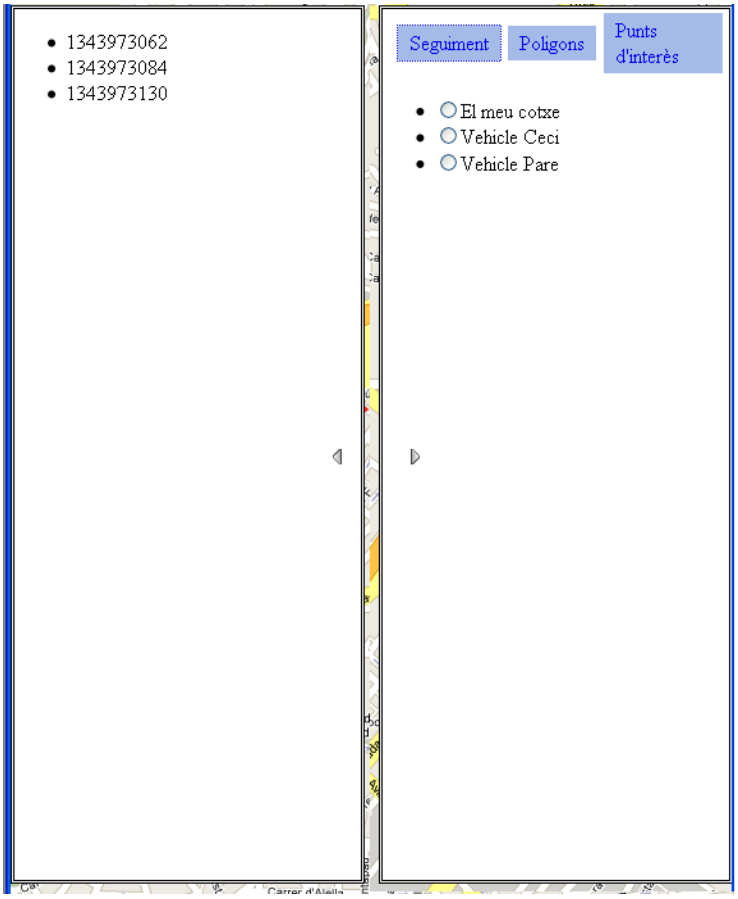


Figura 8.10: Panells laterals

Com podem veure a la figura 8.10, en el panell esquerra apareixen tres vehicles igual que a la pestanya seguiment del panell dret. A més a més a través dels controls del panell esquerra comprovem com aquests tres vehicles es mostren sobre el mapa.

8.2.2 Proves punts d'interès:

Les proves que realitzarem sobre els punts d'interès seran: Creació, Modificar i Eliminar.

8.2.2.1 Crear punt d'interès

Per a crear un punt d'interès farem doble clic sobre el punt geogràfic on volem que s'afegeixi el punt d'interès. Un cop fet el doble clic podrem visualitzar una finestra amb el formulari que apareix a la figura 8.11.

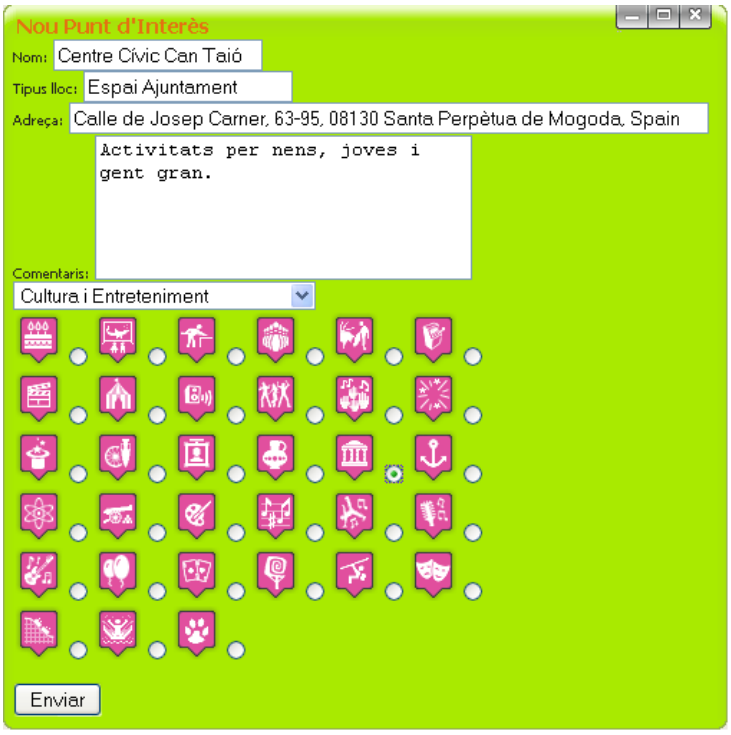


Figura 8.11: Nou Punt d'interès

Un cop introduïdes les dades al formulari, cliquem al botó Enviar, es crea el marcador sobre el mapa que representa el punt d'interès i s'emmagatzema a la taula puntsinteres de la Base de Dades. Com podem comprovar a la figura 8.12 el marcador amb la icona seleccionada s'ha creat.

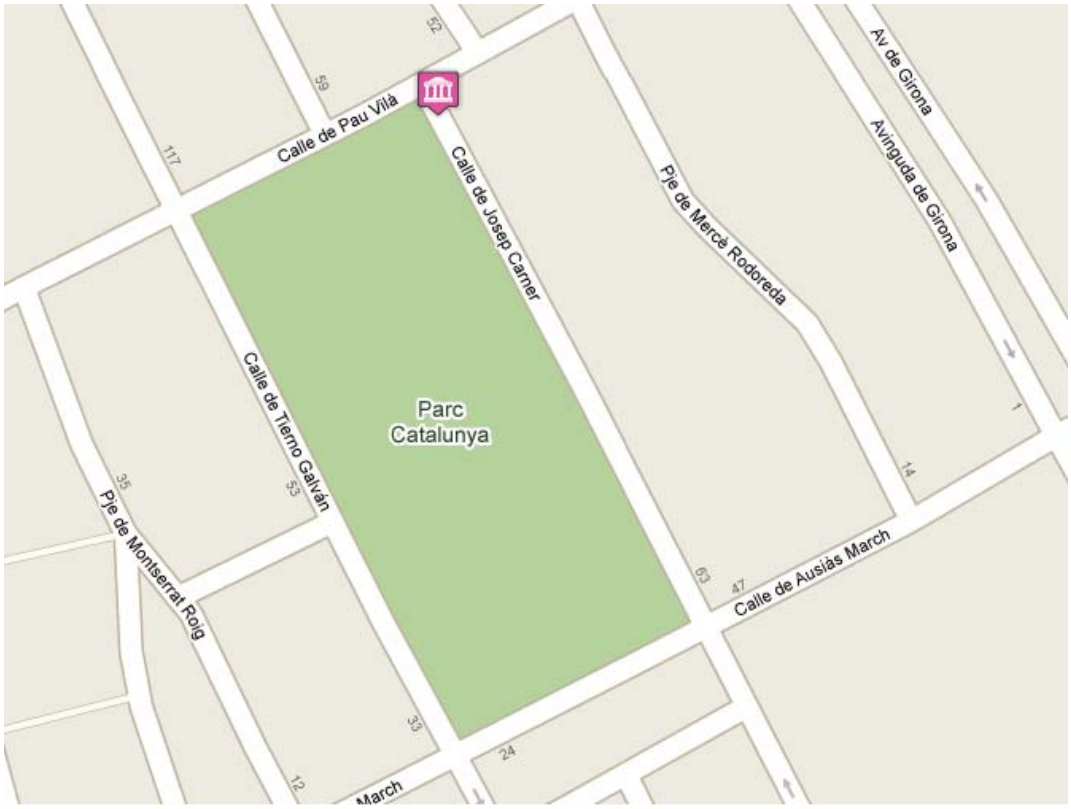


Figura 8.12: Punt d'interès sobre el mapa

Ara comprovem a la taula puntsinteres com s'han emmagatzemat les dades relacionades amb el punt d'interès. Com podem veure a la figura 8.13 a la taula puntsinteres hi ha un registre que correspon al punt d'interès creat.

id	nom	tipus	adreca
124	Centre Cívic Can Taió	Espai Ajuntament	Calle de Josep Carner, 63-95, 08130 Santa Perpètua...

comentaris	icona
Activitats per nens, joves i gent gran.	/icones/CulEnt/museum-historical.png

latitut	longitut	hora	dia
41.5290456069766	2.18014776706696	18:50:37	2010-06-12

Figura 8.13: taula *puntsinteres*

8.2.2.2 Modificar Punt d'interès:

Per a modificar el punt d'interès podem clicar sobre el punt d'interès i modificar la informació o arrossegar el punt d'interès a una altra localització.

- Clic sobre el punt d'interès:

Un cop cliquem sobre el punt d'interès ens apareixen les mateixes dades que hem emmagatzemat a la Base de dades. Ara modificarem la informació del punt d'interès per mostrar com s'actualitza aquesta.

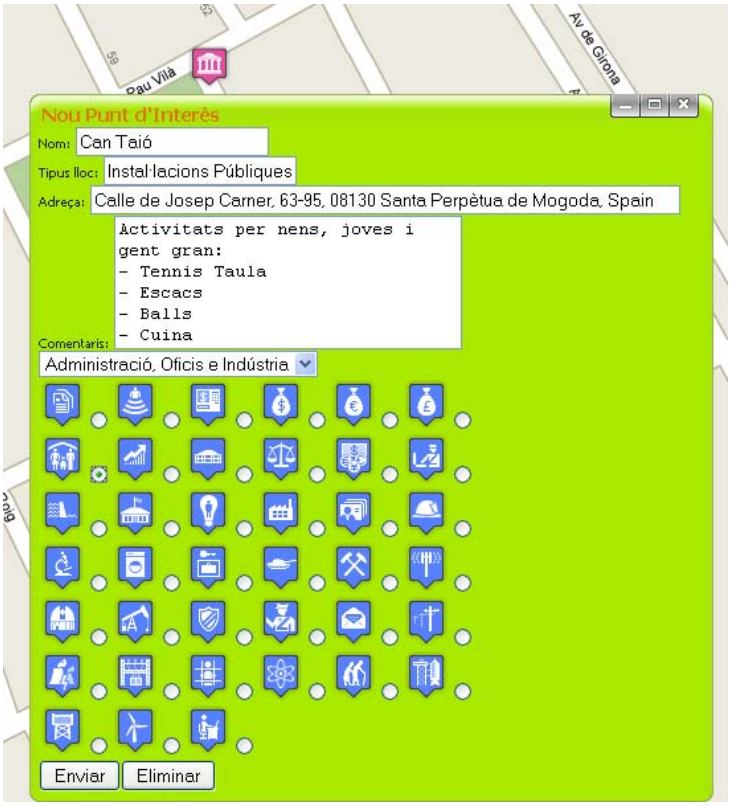


Figura 8.14: Dades punt d'interès

A la figura 8.14 podem observar com hem canviat les dades del punt d'interès i a la figura 8.15 podem veure com l'aplicació ens alerta que el punt d'interès s'actualitzat correctament.

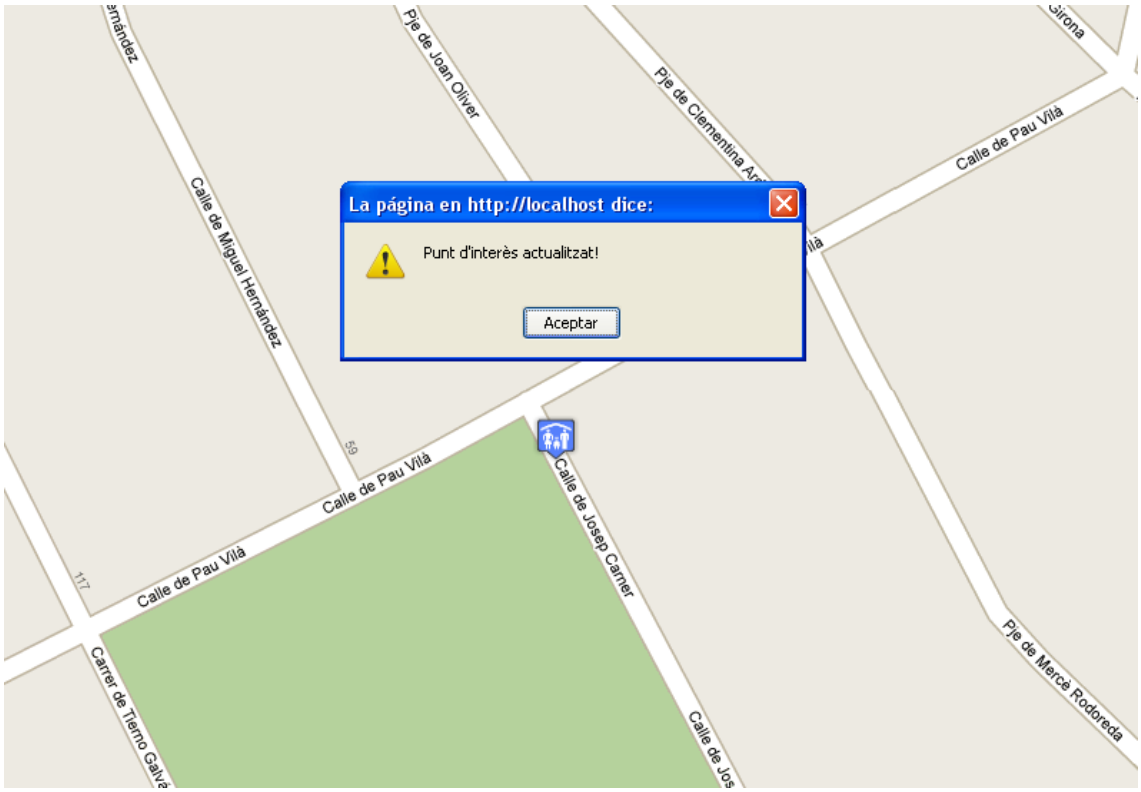


Figura 8.15: Actualització punt d'interès

Ara comprovarem que aquestes dades s'hagin actualitzat correctament a la Base de dades, ho podem comprovar a la figura 8.16.

id	nom	tipus	adreca
124	Can Taió	Instal·lacions Públiques	Calle de Josep Carner, 63-95, 08130 Santa Perpètua...

comentaris	icona
Activitats per nens, joves i gent gran: - Tennis T...	/icones/AdmiOffInd/communitycentre.png

latitut	longitut	hora	dia
41.5290456069766	2.18014776706696	18:50:37	2010-06-12

Figura 8.16: taula *puntsinteres* modificada

- Arrossegar:

Com podem veure a la figura 8.17, un cop fem clic sobre el marcador del punt d'interès i arrosseguem la icona a una altra localització, l'aplicació ens adverteix si volem seguir amb el procés de canvi de posició del marcador. Si acceptem, el marcador canvia de localització i les dades adreça, latitud i longitud.

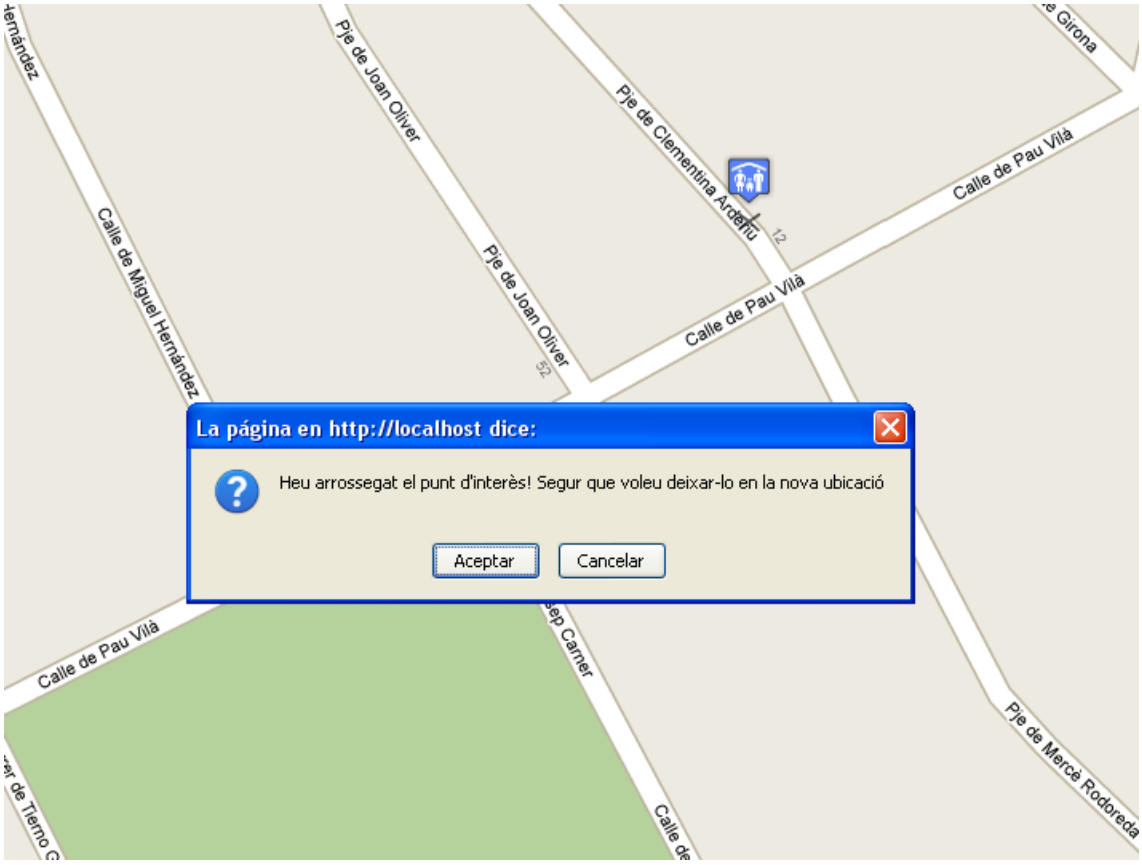


Figura 8.17: Punt d'interès arrossegat

A la figura 8.18 i la figura 8.19 podem observar com les dades han canviat a l'aplicació i a la Base de dades.

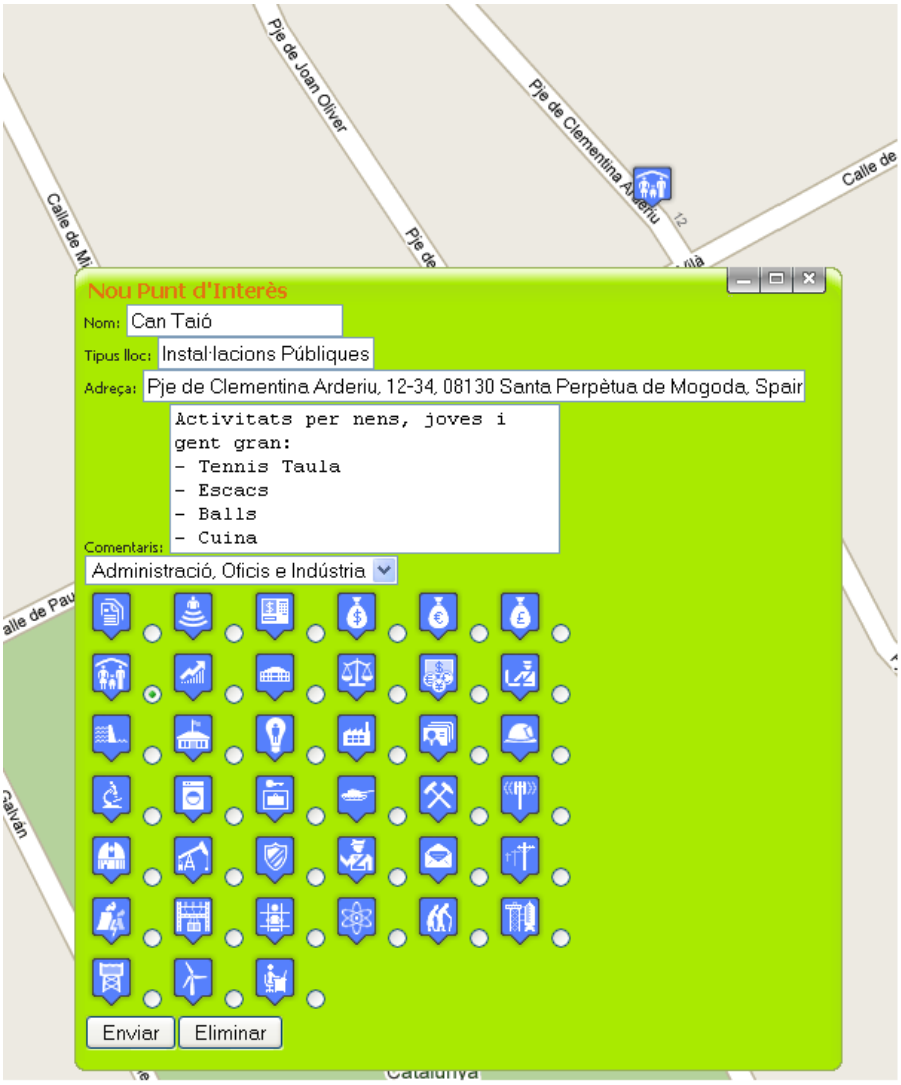


Figura 8.18: Dades modificades del punt d'interès

id	nom	tipus	adreca
124	Can Taió	Instal·lacions Públiques	Pje de Clementina Arderiu, 12-34, 08130 Santa Perp...

comentaris	icona
Activitats per nens, joves i gent gran: - Tennis T...	/icones/AdmiOffInd/communitycentre.png

latitut	longitut	hora	dia
41.5294210933874	2.18055546283722	18:50:37	2010-06-12

Figura 8.19: taula *puntsinteres* modificada

- Eliminar punt d'interès

Quan cliquem sobre el marcador del punt d'interès, la finestra de modificar ens permet eliminar el punt d'interès. Al clicar el punt d'interès s'esborra del mapa i de la Base de Dades.

8.2.3 Proves polígons:

Per comprovar el correcte funcionament dels polígons realitzarem proves sobre les seves funcionalitats:

8.2.3.1 Crear Polígon:

Com es mostra a la figura 8.20, per crear un polígon només cal introduïr un nom i clicar al botó Dibuixa, aleshores ens mostrarà una creu que ens permetrà marcar els vèrtexs del polígon.

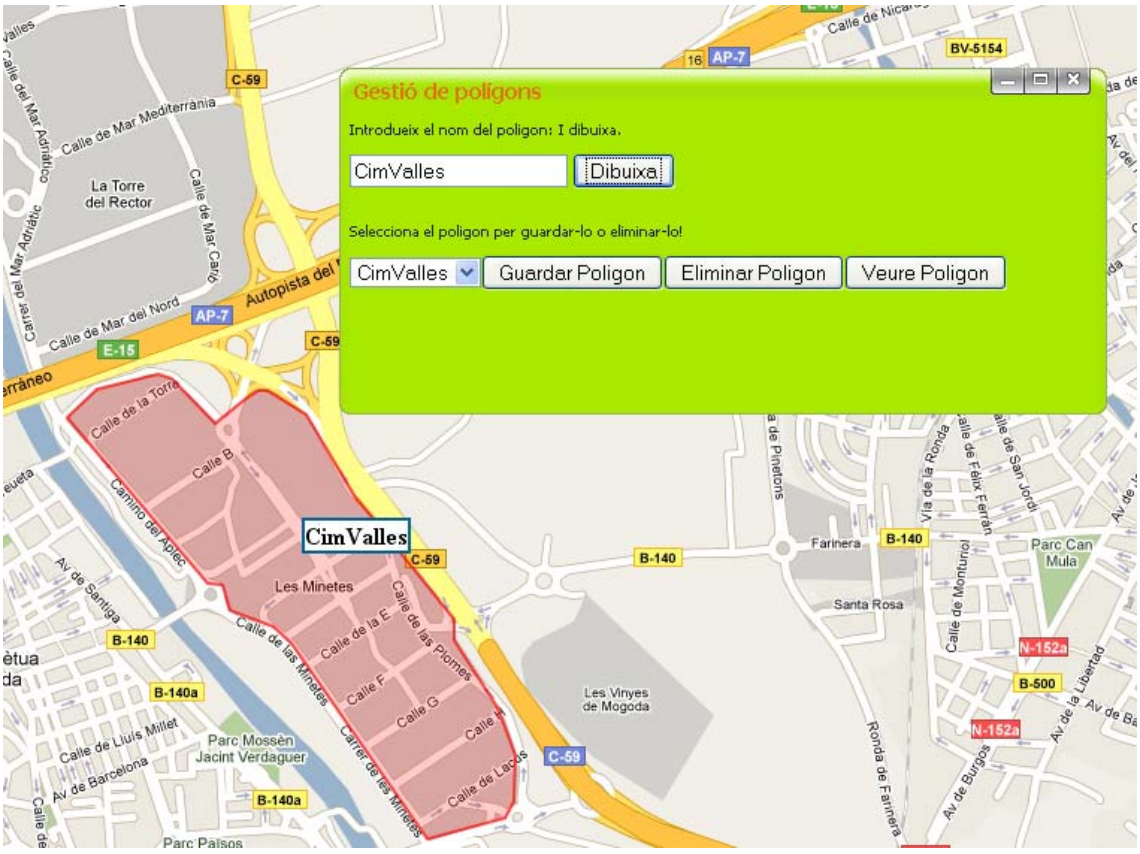


Figura 8.20: Finestra de gestió de polígons

A la figura 8.21 es mostra com un cop dibuixat, si cliquem al botó guardar, les dades del polígon s'emmagatzemen a la taula poligons. La informació relacionada amb els punts no es pot interpretar ja que està emmagatzemada amb el tipus de dades MULTIPOINT, per això, no comprovarem a la Base de dades com els punts del polígon canvien quan es modifica el polígon.

id	nom	dia	hora	punts
87	CimValles	2010-06-12	19:50:24	...

Figura 8.21: taula poligons

8.2.3.2 Modificar Polígon

Per comprovar que la funcionalitat que modifica el polígon funciona tenim dos possibilitats: Clicar sobre un vèrtex ó arrossegar un vèrtex.

- Clic sobre un vèrtex:

Si clicquem sobre un vèrtex, aquest s'eliminarà i automàticament la informació dels punts s'actualitzarà.

Figura 8.22: Avís d'esborrar un vèrtex

A la figura 8.23 podem comprovar com un cop afirmar que volem borrar el vèrtex, el polígon queda en aquesta forma.

Figura 8.23: Polígon amb el vèrtex esborrat

- Arrossegar un vèrtex

A la figura 8.24 podem comprovar que quan s'arrossega un vèrtex, i accceptem l'advertència de l'aplicació, el vèrtex del polígon canvia de localització.

Figura 8.24: Polígon amb el vèrtex arrossegat

8.2.3.3 Eliminar Polígon

Quan ens trobem a la finestra de gestió de polígons, hi ha un selector que ens permet escollir el poligon que volguem, si escollim un polígon i fem clic sobre el botó eliminar, aquell polígon s'esborrarà del mapa i de la taula polígons. Aquesta prova s'ha realitat amb èxit, i el polígon s'ha suprimit del mapa i de la taual polígons.

8.2.4 Sistema GeoFencing

Per comprovar el correcte funcionament del Sistema GeoFencing realitzarem un seguit de proves per al seu test.

8.2.4.1 Afegir contacte

Per comprovar el correcte funcionament d'afegir contacte i subscriure'l a uns determinats polígons, ens dirigim a la finestra d'Afegir a Llista Direccions per a introduïr les dades d'un nou contacte.

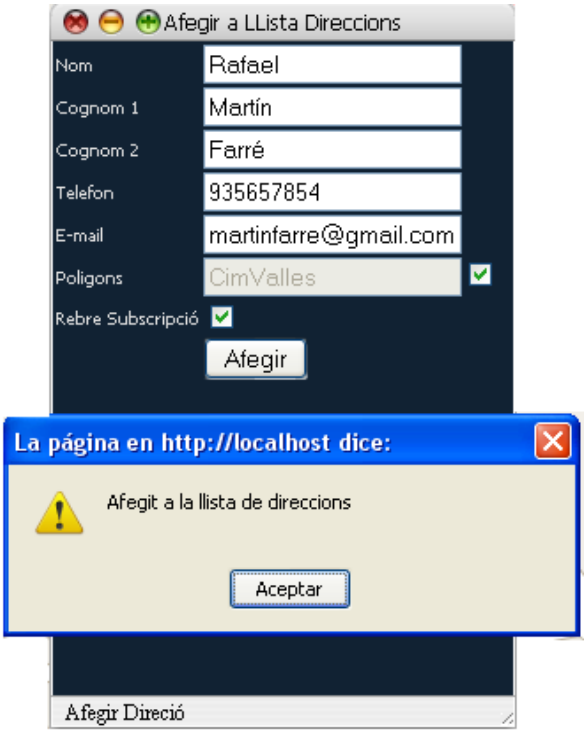


Figura 8.25: Afegir contacte a la llista de direccions

Com podem veure a la figura 8.25, el contacte Rafael ha introduït les seves dades, el seu correu i s'ha registrat pel polígon CimValles. A més a més, ha seleccionat l'opció de rebre subscripció al correu electrònic. Aquesta informació la contrastem amb la informació emmagatzemada a la taula *llistadireccions* i *subscripcio* de la Base de Dades.

idcontacte	nom	cognom1	cognom2	telefon
23	Rafael	Martín	Farré	935657854

email	enviarNotificacions
martinfarre@gmail.com	1

datetime
2010-06-12 10:20:08

Figura 8.26: taula llistadireccions

idcontacte	nompoligon
23	CimValles

Figura 8.27: taula subscripcio

Com podem veure a la figura 8.26, la informació del contacte s'ha afegit correctament a taula *llistadireccions* de la Base de Dades. A més a més , s'ha realitzat correctament la relació entre el polígon CimValles i el contacte Rafael, com es mostra a la taula subscripcio de la figura 8.27.

8.2.4.2 Notificació

En aquest punt, només ens cal posar en marxa el mòdul de captació per anar a realitzar una ruta que passi per dins del polígon CimValles.
Realitzada la ruta, com es mostra a la figura 8.28. Podem observar com el vehicle ha passat per dins del polígon, per tant, s'haurà enviat una notifiació d'alerta a l'usuari Rafael.

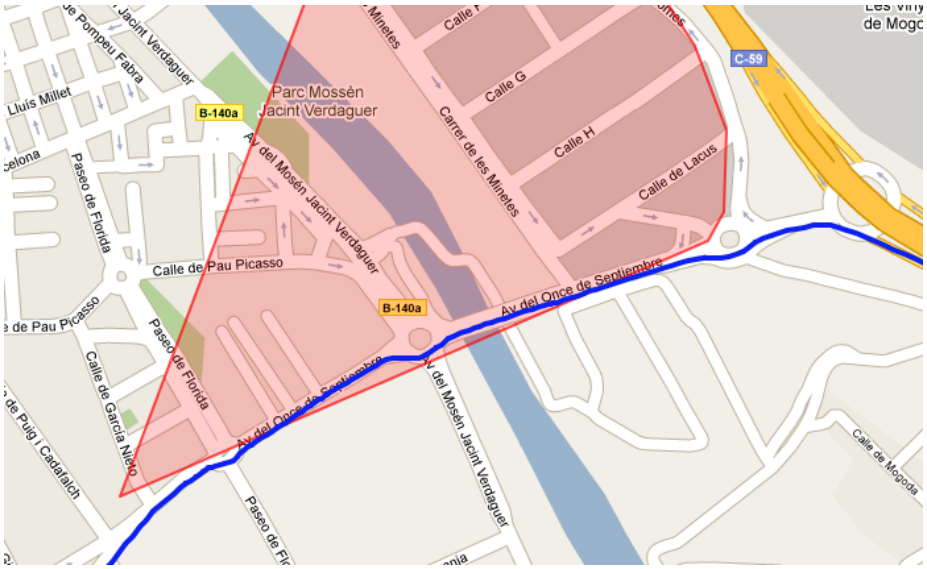


Figura 8.28: Vehicle dins d'un polígon

A la figura 8.29 podem observar el correu del contacte Rafael i veure com el sistema de GeoFencing l'ha avisat d'una alerta.

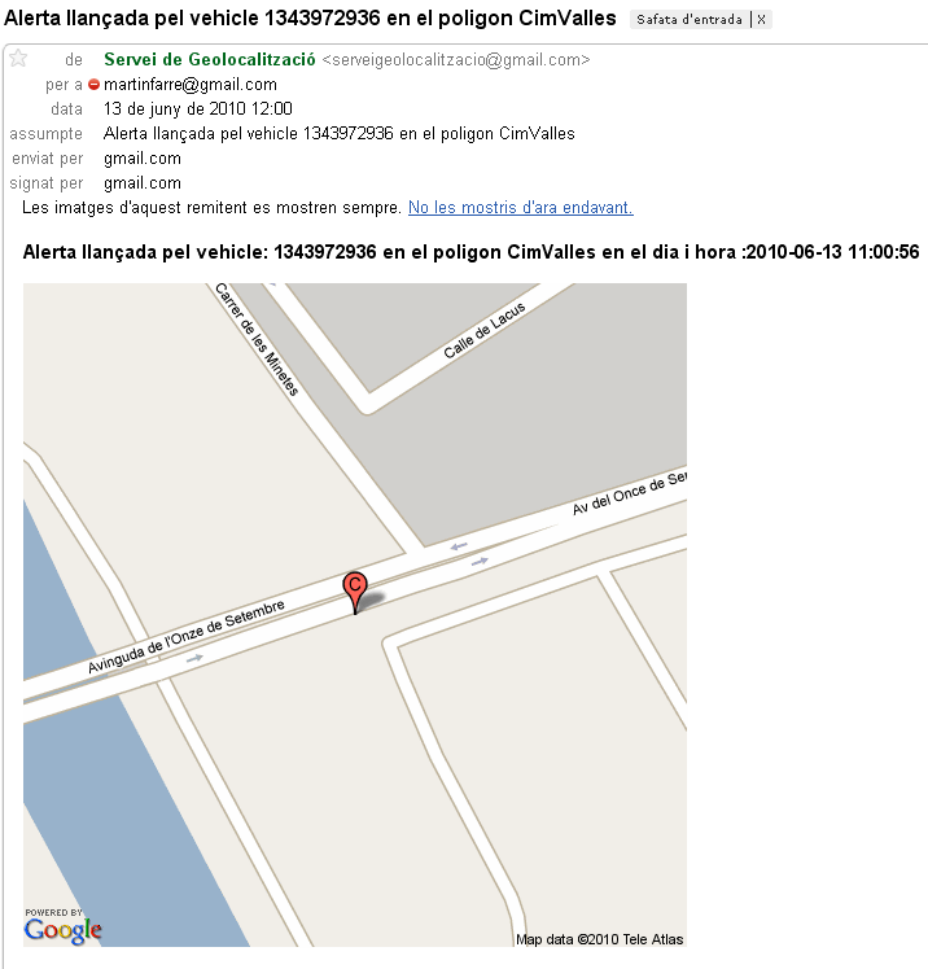


Figura 8.29: Notificació per correu electrònic

9 Conclusions i treball futur.

Tenint en compte els objectius inicials del projecte, els objectius assolits i les proves realitzades anteriorment, aquest capítol té com a finalitat extreure algunes conclusions.

A més a més, en aquest apartat mostraré algunes suggerències per a la millora i ampliació del treball.

9.1 Conclusions

Un cop finalitzat el projecte, em sento satisfet d'haver complert els objectius proposats inicialment, com és el seguiment de vehicles en temps real. A més a més, durant l'elaboració del projecte han sorgit diversos objectius com: l'administració dels punts d'interés, dels polígons i dels vehicles i la visualització de la informació del tràfic sobre el mapa. Aquests objectius també s'han complert i m'han ofert l'oportunitat de descobrir nous camps i funcionalitats.

El projecte ha estat una motivació per indagar en el món de la georreferenciació i les seves aplicacions. M'ha permés, també, introduïr-me en les noves tecnologies de programació web i conèixer el funcionament d'APIS com la de Google Maps. Penso que Google Maps és una eina molt potent i que se li pot treure moltes utilitats i aplicacions. Durant la consulta periòdica de blocs que parlaven sobre aplicacions de Google Maps, m'he adonat d'aquest potencial i de la recent expansió a la xarxa. Les aplicacions, creades arreu del món, s'han anat adaptant als temps i s'han utilitzat per mostrar la informació del dia a dia sobre un mapa. Al llarg del projecte hem pogut veure aplicacions com:

- Localització de la torxa en les olimpiades de Vancouver 2010.
- Capes amb informació climàtica referents a la Cimera del clima de Copenhaguen.
- Mapes amb imatges per satèl·lit del debastament causat pel terratrèmol d'Haití.
- Eines per mostrar informació bàsica a la població després del terratrèmol de Xile (zones on es produïen noves rèpliques, tendes de menjar obertes...)
- Control del tràfic aeri després de l'erupció del volcà islandès (Eyjafjalla).
- Capes amb estimacions sobre les eleccions al Regne Unit.
- Seguiment de la Flotilla de la llibertat (Rachel Corrie) que volia arribar a Gaza.

El meu projecte, orientat a l'àmbit del transport, és avui dia utilitzat per moltes empreses logístiques que necessiten un control precís de la seva flota de vehicles i de les seves mercaderies. Amb la localització del seus vehicles, els administradors de flotes poden prendre decisions i controlar: el comportament dels seus conductors, la seguretat dels seus vehicles i les mercaderies.

A nivell acadèmic, el projecte m'ha ajudat a comprendre el procés bàsic del disseny i implementació d'un software. M'ha permés igualment de descobrir les complicacions d'un programador a l'hora de realitzar un projecte. Penso que al principi va ser una tasca feixuga perquè m'havia de marcar uns objectius, utilitzar una metologia i pautes de treball. Seguint aquest procés de creació de software, he obtingut un resultat que cumpleix els objectius inicials, i, això, personalment, em satisfà.

També haig d'esmentar la feina del meu tutor que, a pesar de la quantitat de projectes que dirigeix, en tot moment ha estat disponible i s'ha mostrat molt atent i encoratjador. Les seves idees i la seva visió avançada de la tecnologia m'han guiat durant tot el projecte.

Per últim, m'agradaria destacar que el projecte realitzat no s'ha basat totalment en la implementació de la visualització de la interfície, sinó que s'ha basat en la implementació de les diverses funcionalitats.

9.2 Treball futur

Les possibles ampliacions i millores poden ser diverses. En aquesta secció parlaré de les que crec que poden ser les més importants:

Ampliacions i millores:

- Crear rutes utilitzant la informació del tràfic:
    - Permetre a l'usuari crear rutes pel vehicle tenint en compte la informació del tràfic. Evitar carreteres amb retencions i incidències. Planificar rutes per les carreteres amb millors temperatures, segons les mercaderies que es transportin.
- Enviar les rutes al vehicle.
    - Permetre enviar i rebre les rutes planificades amb el mòdul de processament. Les rutes podrien ser interpretades pel navegador GPS del vehicle i indicar al conductor les passes a realitzar.
- Migrar el codi a la versió 3 de l'API de Google Maps.
    - El dia 19 de Maig del 2010, la versió 2 de l'API de Google Maps va quedar oficialment obsoleta. Malgrat seguir amb les funcionalitats actives, fora bo migrar a la nova versió. La nova versió de l'API és més ràpida, més integrable per dispositius mòbils i per navegadors d'escriptori.
- Notificacions al mòbil.
    - Permetre als contactes de la llista de contactes rebre les notificacions com un missatge de text al telèfon mòbil.
- Control de velocitat
    - A partir del camp velocitat Màxima de la taula vehicles, controlar que els vehicles no superin aquesta velocitat. Si es supera, crear un registre amb les alertes de velocitat.

- Multiusuari

    - Permetre que hi hagi diferents usuaris registrats i que cada usuari tingui: els seus vehicles, els seus punts d'interès, els seus polígons i les seves opcions de visualització de ruta.

- Millorar interfície

    - El meu projecte, al basar-se en les funcionalitats de l'aplicació, va deixar de banda la interfície i l'aspecte visual d'aquesta. Per això, la millora d'aquesta permetria que l'usuari es sentís més comode.

- Aplicació dispositiu mòbil

    - El GpsPlex és una aplicació que ens ha ajudat molt en l'enviament de les dades del GPS a la Base de Dades. Però, per la nostra aplicació caldria crear-ne una de nova pel dispositiu mòbil, que ens permetés enviar un identificador, especificat per l'usuari i no haver-lo de calcular a través de l'adreça IP com fem ara.

10 Bibliografia

[1] Linda L.Hill. Georeferencing. The geographic associations of information. MIT Press. 2006.

[2] Elliott D.Kaplan, Christopher J. Hegarty. Understanding GPS. Principles and Applications. Artech House. 2006

[3] Michael Purvis, Jeffrey Sambells, Cameron Turner. Beginning Google Maps applications with PHP and AJAX. Apress. 2006

[4] Global Positionig System. http://en.wikipedia.org/wiki/Global_Positioning_System

[5] Roca, Xavier. Apunts Enginyeria del software I (Anàlisi de requeriments)

[6] Roca, Xavier. Apunts Enginyeria del software I (Casos d'ús)

[7] Vehicle tracking System. http://en.wikipedia.org/wiki/Vehicle_tracking_system

[8] Glenn Baddeley. GPS- NMEA sentence information. http://home.pacific.net.au/~gnb/gps/nmea.html#gprmc

[9] Google Maps JavaScript API V2 Reference. http://code.google.com/intl/esES/apis/maps/documentation/javascript/v2/reference.html

[10] Google Maps API Tutorial. http://econym.org.uk/gmap/

[11] Google Geo Developers Blog. http://googlegeodevelopers.blogspot.com/

[12] Google Maps Mania. http://googlemapsmania.blogspot.com/

[13] Google LatLong. http://google-latlong.blogspot.com/

[14] Some experiments with the Google Maps API. http://maps.forum.nu/

[15] MySQL 5.0 Reference Manual. http://dev.mysql.com/doc/refman/5.0/es/index.html

[16] Prototype API Documentation. http://api.prototypejs.org/

[17] Script.aculo.us API Reference. http://wiki.github.com/madrobby/scriptaculous/

[18] Flotr Documentation. http://solutoire.com/flotr/docs/

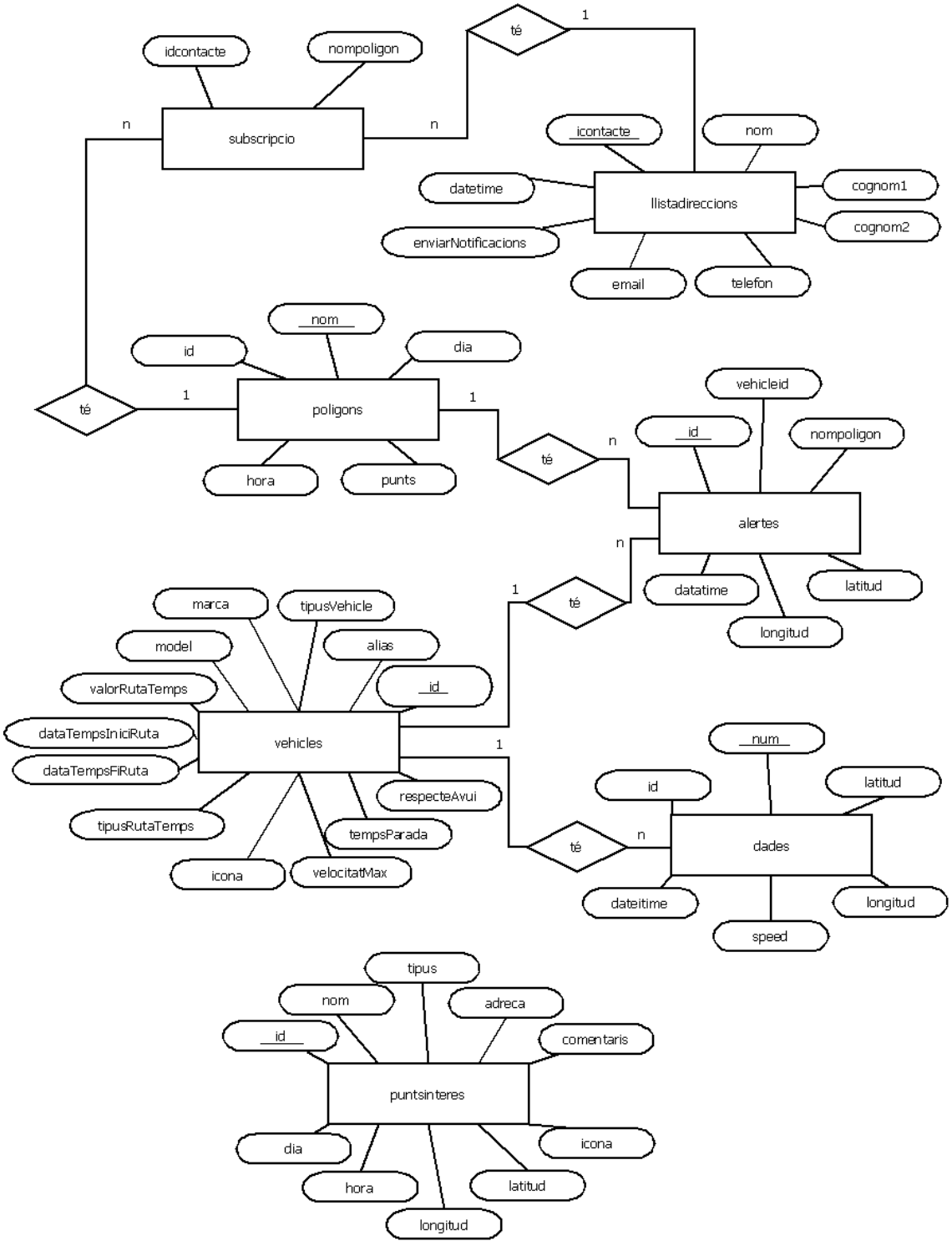
[19] Prototype Window Class: Documentation.

http://prototype-window.xilinus.com/documentation.html

[20] PHP: Function Reference. http://www.php.net/manual/en/funcref.php

11 Annex

11.1 Annex 1: Model Entitat-Relació de la base de dades.

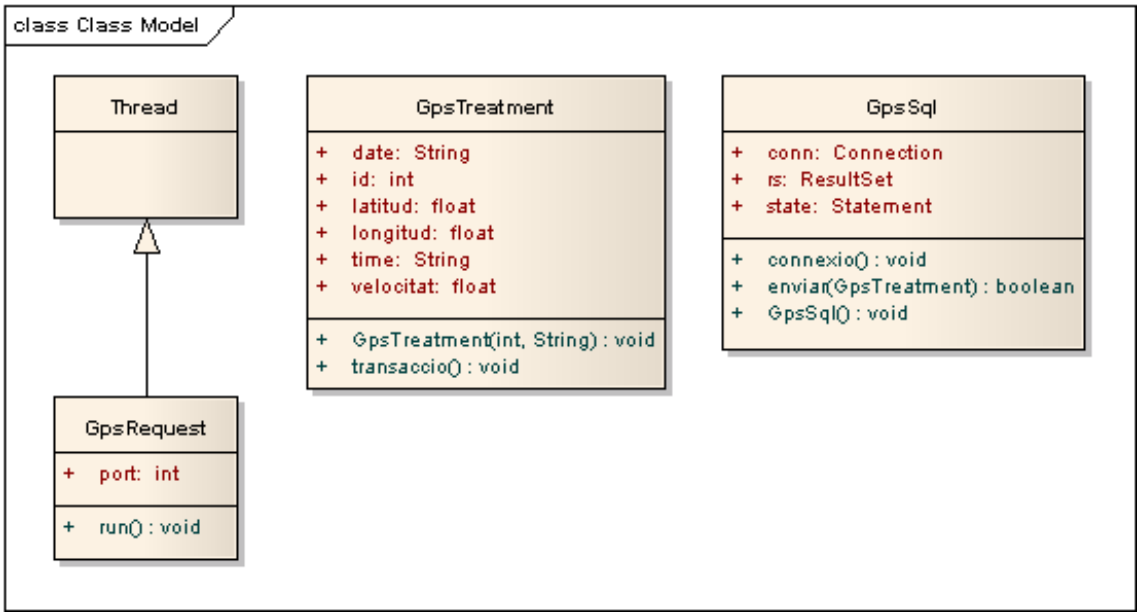


11.2 Annex 2: Diagrames de classes

En aquest annex mostrarem el diagrama de classes per al mòdul de processament i pel mòdul de captació.

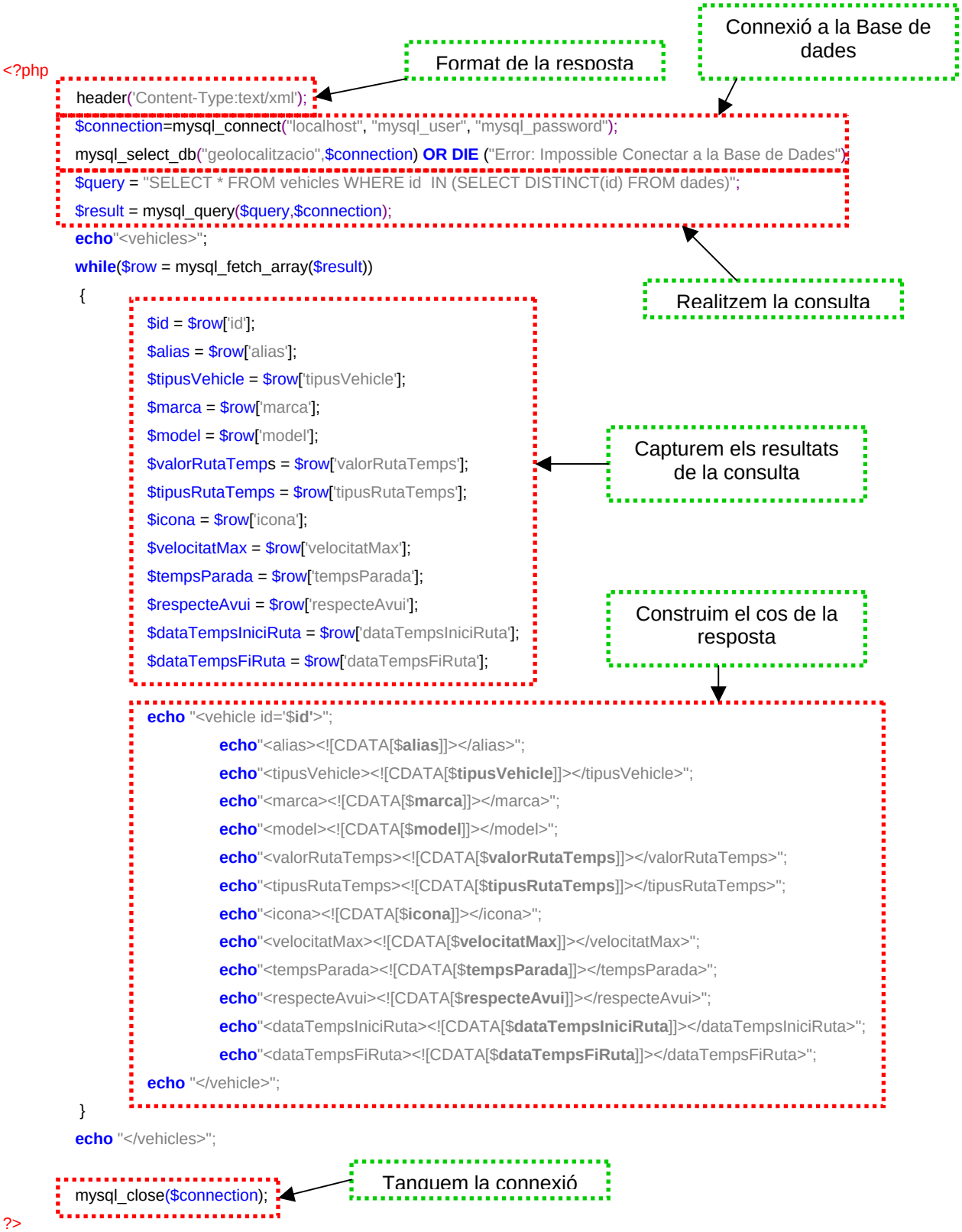
El diagrama de les classes principals del mòdul de processament, es mostra en la següent figura:

El diagrama de classes del mòdul de captació és:



11.3 Annex 3: Consultes a la Base de Dades.

En aquest annex es mostra el codi d'una consulta a la Base de Dades del nostre projecte. El codi està escrit en llenguatge PHP i realitza la consulta per rebre tota la informació dels vehicles disponibles. Després de realitzar la consulta, crea una resposta en format XML.




```php
<?php
    header('Content-Type:text/xml');
    $connection=mysql_connect("localhost", "mysql_user", "mysql_password");
    mysql_select_db("geolocalitzacio",$connection) OR DIE ("Error: Impossible Conectar a la Base de Dades");
    $query = "SELECT * FROM vehicles WHERE id  IN (SELECT DISTINCT(id) FROM dades)";
    $result = mysql_query($query,$connection);
    echo"<vehicles>";
    while($row = mysql_fetch_array($result))
    {
        $id = $row['id'];
        $alias = $row['alias'];
        $tipusVehicle = $row['tipusVehicle'];
        $marca = $row['marca'];
        $model = $row['model'];
        $valorRutaTemps = $row['valorRutaTemps'];
        $tipusRutaTemps = $row['tipusRutaTemps'];
        $icona = $row['icona'];
        $velocitatMax = $row['velocitatMax'];
        $tempsParada = $row['tempsParada'];
        $respecteAvui = $row['respecteAvui'];
        $dataTempsIniciRuta = $row['dataTempsIniciRuta'];
        $dataTempsFiRuta = $row['dataTempsFiRuta'];
        echo "<vehicle id='$id'>";
                echo"<alias><![CDATA[$alias]]></alias>";
                echo"<tipusVehicle><![CDATA[$tipusVehicle]]></tipusVehicle>";
                echo"<marca><![CDATA[$marca]]></marca>";
                echo"<model><![CDATA[$model]]></model>";
                echo"<valorRutaTemps><![CDATA[$valorRutaTemps]]></valorRutaTemps>";
                echo"<tipusRutaTemps><![CDATA[$tipusRutaTemps]]></tipusRutaTemps>";
                echo"<icona><![CDATA[$icona]]></icona>";
                echo"<velocitatMax><![CDATA[$velocitatMax]]></velocitatMax>";
                echo"<tempsParada><![CDATA[$tempsParada]]></tempsParada>";
                echo"<respecteAvui><![CDATA[$respecteAvui]]></respecteAvui>";
                echo"<dataTempsIniciRuta><![CDATA[$dataTempsIniciRuta]]></dataTempsIniciRuta>";
                echo"<dataTempsFiRuta><![CDATA[$dataTempsFiRuta]]></dataTempsFiRuta>";
        echo "</vehicle>";
    }
    echo "</vehicles>";
    mysql_close($connection);
?>
```

Tipus de consultes utilitzades en el projecte, escrites en MySQL:

- Consulta de tots els vehicles disponibles:
  - SELECT * FROM vehicles WHERE id IN (SELECT DISTINCT(id) FROM dades)

- Insertar un nou contace:
  - INSERT INTO llistadireccions (nom, cognom1, cognom2, telefon, email, enviarNotificacions, datetime) VALUES('$nom', '$cognom1', '$cognom2', '$telefon','$email','$subscripcio','$datetime');"

- Actualització d'un punt d'interès:
  - UPDATE puntsinteres SET nom='$nom', tipus='$tipus', adreca='$adreca', comentaris='$comentaris', icona='$icona' WHERE nom='$antnom';";

- Eliminar polígon:
  - DELETE FROM poligons WHERE nom='$nom';

11.4 Annex 4: Resposta XML.

En aquest annex mostrem una resposta del servidor, amb la informació d'un punt d'interès, en format XML .


```
− <puntsinteres>
  − <punt lat="41.5294210933874" lng="2.18055546283722" icona="/icones/AdmiOffInd/communitycentre.png">
      <nom>Can Taió</nom>
      <tipus>Instal·lacions Públiques</tipus>
    − <adreca>
        Pje de Clementina Arderiu, 12-34, 08130 Santa Perpètua de Mogoda, Spain
      </adreca>
    − <comentaris>
        Activitats per nens, joves i gent gran: - Tennis Taula - Escacs - Balls - Cuina
      </comentaris>
    </punt>
  </puntsinteres>
```

11.5 Annex 5. Resposta del servidor de la Direcció General De Tràfic:

En aquest annex mostrem la resposta, en format JSON, del servidor de la DGT.
El **JSON** (JavaScript Object Notation) és un format lleuger per l'intercanvi de dades. JSON és un subconjunt de la notació literal d'objectes de JavaScript que no requereix de l'ús del XML. S'utilitza com alternativa al XML en les respostes AJAX.
Realitzem una consulta al servidor de la DGT per la informació del tràfic. Rebuda la resposta, n'extraiem la informació i la mostrem sobre el mapa. Aquesta és la resposta que obtenim:

[{"causa":"OBRAS EN GENERAL"
,"lat":41.35027,"nivel":"VERDE","icono":"INCIDmapOBRverd.png","pkIni":607.8,"descripcion":"",
"tipoInci":"OBRAS","provincia":"BARCELONA","pkFinal":607.2,"tipo":"Incidencia","sentido":"NO
RTE","alias":"OBRAS / A-2 (607.8 - 607.2)","fecha":"14/06/2010","carretera":"A-
2","codEle":"OB006140061","estado":1,"lng":2.0608208,"poblacion":"SANT JOAN
DESPI","hora":"21:48"},
{"causa":"OBRAS EN GENERAL"
,"lat":41.59072,"nivel":"VERDE","icono":"INCIDmapOBRverd.png","pkIni":560,"descripcion":"","ti
poInci":"OBRAS","provincia":"BARCELONA","pkFinal":564,"tipo":"Incidencia","sentido":"NORTE
","alias":"OBRAS / A-2 (560.0 - 564.0)","fecha":"15/06/2010","carretera":"A-
2","codEle":"OB006150050","estado":1,"lng":1.6793107,"poblacion":"CASTELLOLI","hora":"11:1
2"},...]

La resposta conté la informació del tràfic de l'àrea geogràfica que estem visualitzant en el mapa. La resposta és un array d'elements de tràfic però solament mostro la resposta de tres incidències perquè la resposta és molt extensa i apareixen molts elements: incidències, sensors, càmares...
Els camps de la resposta d'una incidència són:

- Causa de la incidència
- Latitud de la incidència
- Nivell de la incidència
- Icona que representa la incidència
- Kilòmetre inicial de la incidència
- Descripció de la indicèndia
- Tipus d'incidència
- Província de la incidència
- Kilòmetre final de la incidència
- Tipus d'element de la resposta: en aquest cas "Incidència"
- Sentit de la incidència
- Alias de la incidència

- Data de la incidència
- Carretera on s'ha produït la incidència
- Codi de l'element
- Estat de l'element
- Longitud de la incidència
- Població on s'ha produït la incidència
- Hora de la incidència

Amb els elements mostrats sobre el mapa en forma de marcador, si cliquem sobre algun d'aquests, rebrem una determinada resposta segons el tipus d'element que sigui.
Per exemple, en fer clic sobre un panell d'informació, la resposta rebuda pel servidor de la DGT és:

{"NumAlternancias":"1","drawIz2":"apagado.gif","mensaje2":"","imgTxtDer2":"senal.255","drawDer2":"apagado.gif","drawDer1":"isalida.gif","imgTxtIzq1":"senal.138","imgTxtDer1":"senal.233","drawIz1":"plconges.gif","imgTxtIzq2":"senal.255","mensaje1":"EN SALIDA 9 STA EUGENIA","tipo":"Panel_CMS"}

D'aquesta resposta que hem rebut extraiem:
La imatge esquerra del panell indicada pel camp *drawIz1*, la informació que es mostra en el centre del panell, indicada en el camp *mensaje1* i la imatge dreta del panell indicada pel camp *drawDer1*. El resultat de processar aquesta informació és una finestra que mostra els detalls de l'element, com es pot veure en la següent figura:

Per cada element que cliquem extreurem la informació necessària i formarem la seva corresponent finestra de detalls.

Signat: **Rafael Martín Farré**

Bellaterra, 16 de Juny de 2010

ABSTRACT

L'objectiu d'aquest projecte és crear un sistema de seguiment d'una flota de vehicles amb GPS en temps real. A partir d'un mòdul de captació, el servidor recull la informació geogràfica dels vehicles i l'emmagatzema. I amb un mòdul de processament, es mostra i controla els vehicles, els punts d'interès i els polígons del sistema de Geofencing. En primer lloc, faig una introducció a l'estat de l'art dels sistemes de seguiment de vehicles. A continuació, analitzo els requeriments, especifico el comportament desitjat del sistema, explico el disseny i la implementació. Per últim, faig un seguit de proves per extreure'n les conclusions.

El objetivo de este proyecto es crear un sistema de seguimiento de una flota de vehículos con GPS en tiempo real. A partir de un módulo de captación, el servidor recoge la información geográfica de los vehículos y lo almacena. Y con un módulo de procesamiento, se muestra y controla los vehículos, los puntos de interés y los polígonos del sistema de Geofencing. En primer lugar, hago una introducción al estado del arte de los sistemas de seguimiento de vehículos. A continuación, analizo los requerimientos, especifico el comportamiento deseado del sistema, explico el diseño y la implementación. Por último, hago una serie de pruebas para extraer las conclusiones.

The aim of this project is to create a tracking system of a fleet of vehicles with GPS in real time. From a capture module, the server collects and stores the geographical information of the vehicles. And, with a processing module, are shown and controlled vehicles, points of interest and polygons of Geofencing system too. First of all, I make an introduction to the state of the art of the vehicle tracking systems. Next, I analyze the requirements, I specify the wished behaviour of the system, I explain the design and the implementation. Finally, I make a series of tests to extract the conclusions.