

(Creative Commons)

Aquest treball està subjecte –excepte que s’indiqui el contrari– en una llicència de Reconeixement-NoComercial-SenseObraDerivada 2.5 Espanya de Creative Commons. Podeu copiar-lo, distribuir-lo i transmetre’l públicament sempre que citeu l’autor i l’obra, no es faci un ús comercial i no es faci còpia derivada. La llicència completa es pot consultar a <http://creativecommons.org/licenses/by-nc-nd/2.5/es/deed.es>.

# SIG web per a la representació dels jaciments de Molins de Rei amb Google Maps

MEMÒRIA

Alumne: Marcelino Serio López  
Dirigit per Pere Juanola Juanola

Treball final de carrera  
Enginyeria tècnica en telecomunicació - Telemàtica

Curs 2009-10, 2n semestre, juny 2010



Universitat Oberta  
de Catalunya

[www.uoc.edu](http://www.uoc.edu)

*A Antònia, Marisa, Pablo i Carlos, els meus companys de viatge per la UOC*

***Agraïments***

*Al meu consultor del TFC, en Pere Juanola,  
pel seu guiatge i suport*

## RESUM

L'evolució del programari de sistemes d'informació geogràfica (*SIG*, d'ara endavant), lligada a la disponibilitat tecnològica i econòmica dels sistemes informàtics i de comunicació de cada moment, ha estat espectacular en els darrers gairebé 50 anys.

Abans de la dècada dels 80 el desenvolupament d'aquests sistemes era gairebé exclusiu de departaments governamentals, militars i universitaris. Posteriorment s'obririen pas empreses que els desenvoluparien i comercialitzarien a preus força elevats degut al seu alt cost de realització i a l'encara reduïda demanda. Ja a la dècada dels 90, van aparèixer els SIG d'escriptori amb la millora de prestacions dels ordinadors personals però encara amb alts costos de llicències.

En els darrers anys, però, estan apareixent alternatives com sistemes de codi obert o programari lliure, a més de l'alternativa tecnològica dels SIG web on tant les dades geogràfiques com els processos que sobre aquestes s'han d'executar estan localitzats en servidors d'Internet de manera que els usuaris hi accedeixen des dels seus navegadors.

En aquest sentit cal fer especial esment de Google Maps que, dins el món *Web 2.0*, permet la creació de *mashups*, és a dir, aplicacions web que integren les funcionalitats geogràfiques de Google Maps amb dades d'altres fonts.

En aquest treball final de carrera (TFC, d'ara endavant) s'empra Google Maps per crear un mashup per la representació dels jaciments de Molins de Rei. En concret es vol que els usuaris puguin consultar el fons documental del Museu de Molins de Rei de manera que puguin accedir a la informació de documents i imatges lligats a la localització del jaciment corresponent.

## ÍNDEX

RESUM .....	1
ÍNDEX.....	2
ÍNDEX DE FIGURES.....	4
1. Introducció.....	5
1.1. Justificació: context, punt de partida i aportació del TFC.....	5
1.2. Objectius.....	5
1.2.1. Objectius generals i específics del TFC .....	6
1.2.2. Objectius del SIG web a desenvolupar.....	6
1.2.3. Abast i possibles ampliacions.....	7
1.3. Enfocament i mètode seguit .....	7
1.4. Planificació .....	8
1.4.1. Fase d'aprenentatge i preparació de l'entorn del treball.....	8
1.4.2. Fase d'adquisició de dades per la representació.....	8
1.4.3. Fase d'implementació de l'aplicació .....	9
1.4.4. Fase de documentació de l'aplicació .....	9
1.4.5. Calendari.....	10
1.4.6. Anàlisi de riscos i camí crític .....	10
1.4.7. Fites .....	12
1.4.8. Diagrama de Gantt.....	12
1.4.9. Productes obtinguts .....	13
1.4.10. Descripció dels capítols de la memòria .....	13
2. Preparació de l'entorn de treball.....	14
2.1. Instal·lació del servidor web Apache.....	14
2.2. Creació de la base de dades amb MySQL .....	15
2.3. Instal·lació de l'IDE Aptana de Javascript .....	15
2.4. Obtenció d'una clau de Google Maps .....	15
2.5. Creació d'un mashup senzill.....	16
2.5.1. Visualitzar el mapa .....	16
2.5.2. Cridar la funció per obtenir les dades.....	17
2.5.3. Emprar la tecnologia AJAX .....	18
2.5.4. La consulta MySQL .....	19
2.6. Resum de la tecnologia a emprar.....	20
3. Adquisició de les dades .....	21
3.1. Preparació de les dades geogràfiques .....	21
3.2. Les dades del fons documental del museu .....	23

4. Anàlisi i disseny de l'aplicació .....	24
4.1. Anàlisi dels requeriments .....	24
4.2. Disseny de la base de dades.....	25
4.3. Disseny de la interfície .....	26
5. Implementació de l'aplicació .....	27
5.1. Implementació de la base de dades.....	27
5.2. Implementació del sistema d'arxius .....	29
5.3. Implementació de les funcionalitats .....	29
5.3.1. Mostrar els límits municipals .....	29
5.3.2. Mostrar el museu i els jaciments .....	34
5.3.3. Mostrar els jaciments per període.....	37
5.3.4. No permetre la sortida dels límits municipals .....	43
5.3.5. Implementar el menú contextual dels jaciments.....	44
5.3.6. Mostrar el plànol del museu amb la distribució per jaciments .....	44
5.3.7. Calcular rutes cap al museu i jaciments.....	45
6. Manual d'usuari per l'execució de l'aplicació .....	54
7. Valoració econòmica .....	56
7.1. Recursos necessaris .....	56
7.2. Valoració del cost econòmic.....	56
8. Conclusions .....	57
9. Glossari.....	58
10. Bibliografia.....	60
10.1. Documents .....	60
10.2. Enllaços a Internet .....	60

## ÍNDIX DE FIGURES

Figura 1.1. – Diagrama de Gantt amb la planificació del TFC .....	12
Figura 2.1. – Instal·lació d'Apache .....	14
Figura 2.2. – Comprovació d'instal·lació d'Apache.....	14
Figura 2.3. – Creació de la base de dades amb MySQL.....	15
Figura 2.4. – Mashup senzill .....	16
Figura 2.5. – Comparació entre el models clàssic i AJAX d'aplicacions web.....	20
Figura 3.1. – Comprovació de coordenades UTM 31N ED50 amb la web de l'ICC.....	21
Figura 3.2. – Conversió de coordenades geodèsiques UTM 31N ED50 a geogràfiques .....	21
Figura 3.3. – Conversió de coordenades geogràfiques d'ED50 a ETRS89.....	22
Figura 3.4. – Conversió de coordenades geogràfiques ETRS89 a notació decimal.....	22
Figura 3.5. – Comprovació de coordenades geogràfiques ETRS89 amb Google Maps .....	22
Figura 3.6. – Fons documental d'un jaciment .....	23
Figura 4.1. – Model relacional de la base de dades .....	25
Figura 4.2. – Disseny de la interfície.....	26
Figura 5.1. – Creació de taula jaciments de la base de dades.....	27
Figura 5.2. – Definició dels camps de la taula jaciments .....	27
Figura 5.3. – Inserció de registres a la taula jaciments.....	28
Figura 5.4. – Sistema d'arxius .....	29
Figura 5.5. – Mapa centrat a nivell de zoom 13 .....	30
Figura 5.6. – Nivells de zoom 11, 12 i 13 .....	31
Figura 5.7. – Nivells de zoom 15 i 16 sobre mapes de tipus híbrid .....	31
Figura 5.8. – Alertes a l'intentar excedir els nivells de zoom o els límits.....	31
Figura 5.9. – Formulari museu .....	34
Figura 5.10. – Mostrar el museu i les seves dades .....	34
Figura 5.11. – Formulari jacimentsxperiode .....	37
Figura 5.12. – Mostrar jaciments per període i les seves dades .....	38
Figura 5.13. – Menú contextual dels jaciments: accés a dades .....	44
Figura 5.14. – Menú contextual dels jaciments: accés a imatges i documents.....	44
Figura 5.15. – Plànol del museu amb la distribució per jaciments.....	45
Figura 5.16. – Formulari ruta.....	45
Figura 5.17. – Exemple de ruta .....	51
Figura 5.18. – Alerta d'error de ruta per origen incorrecte .....	52
Figura 5.19. – Alerta del punt d'origen .....	53
Figura 6.1. – Parts de l'aplicació .....	54
Figura 6.2. – Controls zoom i tipus de mapa .....	55
Figura 6.3. – Museu i accés al plànol .....	55
Figura 6.4. – Jaciments i accés a imatges i documents .....	55
Figura 6.5. – Ruta amb la seva informació .....	55

## **1. Introducció**

### ***1.1. Justificació: context, punt de partida i aportació del TFC***

L'evolució dels SIG ha estat espectacular en els darrers gairebé 50 anys. S'ha passat del desenvolupament gairebé exclusiu de complexos sistemes per part de departaments governamentals, militars, universitaris i empreses especialitzades que requerien dissenys i implementacions específiques propietàries i d'alt cost a les actuals alternatives de programari lliure, codi obert i SIG web que permeten un ampli accés a aquests tant per la reducció dels costos com per la seva major possibilitat de compartir dades amb altres aplicacions.

És precisament dins el context de l'alternativa tecnològica que representen els SIG web, on tant les dades geogràfiques com els processos que sobre aquestes s'han d'executar estan localitzats en servidors als quals els usuaris poden accedir des de la interfície dels seus navegadors, que es desenvolupa aquest TFC.

Tot i que aquests SIG web no siguin tan complets en funcionalitats com els SIG d'escriptori el cert és que s'han estès molt perquè són capaços d'arribar al gran públic dels internautes sense cap cost de cartografia ni llicències i es preveu que dominin en la següent dècada.

En aquest sentit cal fer especial esment de Google Maps que, dins el món Web 2.0, permet la creació de mashups per integrar les seves funcionalitats geogràfiques amb dades provinents d'altres fonts per crear SIG webs a mida.

Aquesta tecnologia esdevé el punt de partida d'aquest treball que ha permès aportar el desenvolupament d'un SIG web per la representació dels jaciments de Molins de Rei amb Google Maps, de manera que els usuaris poden consultar el fons documental del Museu de Molins de Rei accedint a la informació de documents i imatges lligats a la localització de cada jaciment.

### ***1.2. Objectius***

Es destaquen aquí dos tipus d'objectius diferents. D'una banda estarien els objectius generals i específics de l'assignatura del TFC per permetre desenvolupar una aplicació SIG web, i d'una altra els objectius propis del SIG web concret a desenvolupar.



### 1.2.1. Objectius generals i específics del TFC

Objectius generals:

- Conèixer l'arquitectura conceptual i els components necessaris per desenvolupar aplicacions SIG en entorn web.
- Conèixer els beneficis d'ús de la tecnologia de fer mashups de diferents proveïdors de dades.

Objectius específics:

- Utilitzar diferents serveis i components d'informació geogràfica a l'abast de tothom.
- Creuar informació generada de diverses fonts d'informació.
- Conèixer els llenguatges de programació per al desenvolupament d'aplicacions riques en Internet.

### 1.2.2. Objectius del SIG web a desenvolupar

L'objectiu principal del treball és desenvolupar una aplicació web que permeti mostrar sobre Google Maps tant el museu com els jaciments del terme municipal de Molins de Rei de manera que els usuaris puguin veure les dades, els documents i les imatges de cada jaciment quan hi passin per sobre amb el ratolí.

En concret, l'aplicació ha de permetre:

- Mostrar els límits municipals de Molins de Rei amb Google Maps de manera que l'usuari no en pugui sortir d'aquests.
- Mostrar el museu i els jaciments amb Google Maps. Els jaciments s'han de mostrar en funció del seu període i l'usuari ha de poder triar de quins períodes vol veure els jaciments.
- Mostrar un menú quan l'usuari es posicioni a sobre d'un jaciment que li permeti triar la informació a veure: dades del jaciment, imatges del jaciment i els documents disponibles d'aquest.
- Mostrar un plànol del museu amb la distribució dels diversos jaciments d'aquest quan l'usuari s'hi posicioni a sobre.
- Proposar una ruta per arribar des d'un punt d'origen triat per l'usuari fins al museu o fins a qualsevol jaciment.

### 1.2.3. Abast i possibles ampliacions

L'abast del treball ve determinat pels objectius abans comentats i que s'han de poder executar en local, és a dir, podent albergar tant l'aplicació web com la informació i les imatges en l'ordinador en que es desenvolupi la implementació.

Es deixa com a possibles ampliacions l'accés remot a l'aplicació i qualsevol altra funcionalitat no indicada en els objectius com ara:

- Registrar usuaris.
- Permetre que els usuaris registrats editin informació com dades i imatges.
- Optimització de rutes.
- Etc.

### 1.3. Enfocament i mètode seguit

En primer lloc s'han estudiat les principals característiques tant dels SIG en general com dels SIG web en particular per conèixer millor l'àrea temàtica en què s'havia de desenvolupar el treball. Posteriorment ha calgut centrar-se en els coneixements de l'entorn de treball per poder desenvolupar l'aplicació. En aquest aspecte s'han seguit els següents passos:

- Estudiar la metodologia d'inclusió de mashups de Google Maps per afegir mapes en pàgines web.
- Estudiar l'*API* de *Javascript* de Google Maps.
- Estudiar la creació de bases de dades en un servidor web amb MySQL per gestionar les dades dels jaciments i del fons documental del museu.
- Estudiar com les aplicacions web accedeixen a les dades mitjançant PHP.
- Estudiar la tecnologia AJAX per permetre l'actualització asíncrona de les dades presentades en una web, és a dir, sense haver de recarregar o actualitzar la pàgina web.
- Estudiar XML com a format de comunicació de dades entre el servidor web i l'aplicació web.
- Estudiar codis d'exemple de Javascript per gestionar mapes: afegir continguts, calcular rutes, etc.

Un cop estudiat l'entorn de treball s'ha pogut preparar aquest i centrar-se en l'adquisició de les dades necessàries, tant geogràfiques com documentals, per fer la representació del museu i dels jaciments.

Finalment s'ha realitzat l'anàlisi i disseny de les funcionalitats i la interfície de l'aplicació, així com de la base de dades que aquesta havia d'utilitzar, per passar després a la seva implementació i comprovació.

## 1.4. Planificació

Per la consecució del TFC caldrà assolir els següents ítems bàsics:

- Adquisició dels coneixements necessaris de l'entorn de treball abans esmentats.
- Preparació de l'entorn del treball.
- Adquisició de tota la informació necessària per fer la representació del museu i dels jaciments: dades, documents i imatges.
- Anàlisi i disseny de les funcionalitats i la interfície de l'aplicació, així com de la base de dades que aquesta haurà d'utilitzar.
- Desenvolupament de la interfície i de les diferents funcionalitats de l'aplicació.
- Proves unitàries de les funcionalitats.
- Redacció de la memòria del projecte i preparació de la seva presentació virtual.

Tenint això en compte es preveuen les següents fases amb la conseqüent divisió en tasques.

### 1.4.1. Fase d'aprenentatge i preparació de l'entorn del treball

Aquesta fase inclou les següents tasques i subtasques:

- Cerca d'informació i aprenentatge sobre la tecnologia de desenvolupament de SIG webs: mashups, API de Google Maps, Javascript, MySQL, PHP, AJAX i XML.
- Preparació de l'entorn de treball.
  - Instal·lació d'*Apache/Tomcat* o similar com a servidor d'aplicacions web i de bases de dades.
  - Instal·lació d'*Aptana* o *IDE* similar com a entorn de desenvolupament en Javascript amb suport per PHP i AJAX.
  - Obtenció d'una clau de Google Maps.
  - Test de l'entorn de treball creant un mashup senzill.

Les tasques d'aprenentatge i preparació de l'entorn de treball es poden executar en paral·lel, és a dir, al mateix temps ja que van molt lligades, però la subtasca de test de l'entorn de treball es preveu executar després de les subtasques anteriors.

Temps previst de dedicació: 48 hores.      Nombre de planes dedicades al treball: 7.

### 1.4.2. Fase d'adquisició de dades per la representació

Aquesta fase inclou les següents tasques i subtasques:

- Cerca de dades geogràfiques per limitar el terme municipal de Molins de Rei i ubicar el museu i els jaciments.

- Cerca d'informació documental del museu i dels jaciments.
  - Cerca del plànol del museu i de la distribució dels seus jaciments.
  - Cerca d'informació dels jaciments: dades del jaciment, imatges del jaciment i els documents disponibles d'aquest.

Les tasques i subtasques de cerca es preveu executar-les de manera seqüencial, és a dir, en sèrie, una rera l'altra.

Temps previst de dedicació: 48 hores.      Nombre de planes dedicades al treball: 3.

### **1.4.3. Fase d'implementació de l'aplicació**

Un cop superades les fases anteriors es pot iniciar el desenvolupament de l'aplicació que inclou les següents tasques i subtasques:

- Anàlisi dels requeriments funcionals de l'aplicació.
- Disseny de la base de dades de l'aplicació.
- Disseny de la interfície de l'aplicació.
- Implementació de les diferents funcionalitats de l'aplicació.
  - Mostrar els límits municipals de Molins de Rei.
  - No permetre a l'usuari sortir d'aquests límits.
  - Mostrar el museu i els jaciments.
  - Permetre seleccionar els jaciments a mostrar segons el període.
  - Detectar quan el ratolí es posa a sobre d'un jaciment i mostrar un menú que permeti triar la informació a veure: dades del jaciment, imatges del jaciment i els documents disponibles d'aquest.
  - Detectar quan el ratolí es posa a sobre del museu i mostrar la distribució dels diversos jaciments d'aquest.
  - Calcular rutes des de qualsevol punt d'origen triat per l'usuari fins al museu o fins a qualsevol jaciment.
- Proves unitàries de cada funcionalitat.
- Integració de les funcionalitats amb la interfície.

La tasca de disseny de la interfície es pot fer abans d'iniciar la implementació de l'aplicació tot i que, al ser recomanable implementar l'aplicació desenvolupant una funcionalitat rere l'altra (amb la realització de la prova unitària de cada funcionalitat abans d'iniciar la implementació de la següent), es pot anar completant el disseny de la interfície mentre es realitza la integració de cada funcionalitat amb aquesta.

Temps previst de dedicació: 120 hores.      Nombre de planes dedicades al treball: 30.

### **1.4.4. Fase de documentació de l'aplicació**

En aquesta fase no cal esperar a tenir enllestida la fase anterior per iniciar tota la documentació que pot abastir les següents tasques:

- Documentació tècnica resultant de la implementació de l'aplicació.

- Manual d'usuari per l'execució de l'aplicació.
- Memòria del treball.
- Presentació virtual del treball.

Mentre la documentació tècnica es pot redactar durant la implementació, la redacció del manual d'usuari serà més adient realitzar-la al final del desenvolupament d'aquesta. D'una altra banda, la memòria del treball, tot i ser un producte final d'aquest, també es pot anar elaborant durant l'elaboració del projecte a partir de les experiències i observacions obtingudes al llarg d'aquest. Finalment, la presentació virtual es deixa pel final per recollir allò realment essencial del treball i la seva memòria.

Temps previst de dedicació: 72 hores.

Nombre de planes dedicades al treball: 20. Aquest nombre de planes, al formar la documentació tècnica part resultant de la fase anterior, es refereix als apartats de manual d'usuari i altres parts de la memòria com la introducció, la valoració econòmica, l'extracció de conclusions, glossari i bibliografia.

#### 1.4.5. Calendari

El lliurament del pla de treball es va realitzar el 13/03/2010 i la data de lliurament de l'aplicació i la documentació (memòria i presentació virtual) és el 07/06/2010. Això implica, sense comptar aquestes dates, disposar de 12 setmanes al llarg del semestre amb la següent previsió de disponibilitat horària setmanal del desenvolupador:

- Dilluns, dimarts, dimecres i divendres: 3 hores diàries.
- Dijous: 0 hores diàries doncs la feina no li permet.
- Dissabtes i diumenges: 6 hores diàries.

Això implica un total de 24 hores per setmana, és a dir, 288 hores al llarg del període abans comentat.

La distribució que d'aquestes 288 hores s'ha fet per a les diferents fases abans comentades es pot observar al diagrama de Gantt (fig. 1.1.).

#### 1.4.6. Anàlisi de riscos i camí crític

De cara a la planificació del treball també serà important tenir en compte els possibles riscos i imprevistos d'aquest.

- **Manca d'experiència.** Aquest es considera el principal risc. Tot i que el desenvolupador té certa experiència en programació tant estructurada com en orientada a objectes, encara ha d'adquirir molts dels coneixements necessaris per al desenvolupament del treball en aquest entorn, com ara obtenció de dades geogràfiques i georeferenciació, Javascript, creació de mashups amb Google Maps, etc.

Pla de contingència a aplicar: aquest risc cal tenir-lo en compte per intentar ser el més realista possible i per això s'ha volgut evitar una planificació massa optimista. En aquest aspecte, i tal com es podrà observar al diagrama de Gantt (fig. 1.1.), s'han dedicat 48 hores repartides en 2 setmanes per a la fase d'aprenentatge i preparació de l'entorn de treball, i 48 hores més durant altres 2 setmanes per a la fase d'adquisició de dades, de manera que hi hagi marge per poder reajustar la planificació si fos necessari i sigui possible fer un esforç d'hores extra per pal·liar aquest risc.

- **Manca temporal de dades.** Aquest risc consisteix en la possibilitat de no disposar en el temps previst de les dades que ha d'utilitzar l'aplicació, tant geogràfiques com dels jaciments i del fons documental.

Pla de contingència a aplicar: es preveu que aquest risc no tingui cap impacte important en el desenvolupament del treball ja que pot afectar poc en el disseny de la base de dades al tenir una bona definició dels requeriments i, a més, és possible avançar en alguns aspectes de la implementació de l'aplicació sense aquestes o bé fins i tot es podrien emprar dades fictícies fins que no es disposés de les dades reals. D'aquesta manera la fase d'adquisició de dades es podria estendre i córrer paral·lelament a la fase d'implementació (fig. 1.1.).

- **Imprevistos.** Un altre risc esdevé per possibles imprevistos que puguin sorgir durant el semestre en el que es desenvoluparà el treball. Atenent a allò que s'ha comentat en l'anterior apartat del calendari, això implica disposar de 85 dies al llarg del semestre. La disponibilitat horària del desenvolupador podria no ser sempre la indicada llavors durant tots aquests dies per motius obvis: pics de feina a la seva feina habitual, malalties, no disponibilitats temporals d'Internet o de maquinari o programari, etc.

Pla de contingència a aplicar: aquest risc fa pensar en una planificació no excessivament ajustada i flexible, doncs el camí més crític que es preveu és l'establiment de l'entorn del treball i l'adquisició de dades i dels coneixements necessaris que permetin iniciar la programació de les funcionalitats. Un cop assolit aquest fet per al qual al diagrama de Gantt (fig. 1.1.) s'hi preveuen les 4 setmanes inicials, la implementació de les diferents parts es preveu assolible tot i que sigui difícil estimar el temps necessari per la implementació de cada funcionalitat de l'aplicació. Al diagrama de Gantt es pot observar que a la fase d'implementació es dediquen les 5 setmanes següents i les 3 setmanes posteriors es dediquen a la documentació. Així doncs, algunes tasques poden admetre reajustaments a mesura que es vagi seguint el pla de treball.

En aquest sentit cal notar que el camí crític vindria donat per les dates de lliurament de les diferents PAC al llarg del semestre, el contingut de les quals es detalla al següent apartat.

### 1.4.7. Fites

Atenent a allò comentat sobre el camí crític a l'apartat anterior, es fa la següent proposta dels productes a lliurar:

- Pla de treball (13-03-2010): document de planificació del treball.
- PAC2 (13-04-2010): document descriptiu de la consecució tant de la fase d'aprenentatge i preparació de l'entorn del treball com de la fase d'adquisició de dades per la representació.
- PAC3 (18-05-2010): document descriptiu de les funcionalitats assolides de la fase d'implementació.
- Lliurament final (07-06-2010): l'aplicació, la seva memòria i la presentació virtual.
- Debat virtual (21-06-2010).
- Fi del treball (23-06-2010).

Una setmana abans de la PAC2, la PAC3 i el lliurament final es preveu lliurar un esborrany al consultor per poder incloure les correccions que siguin convenientes.

### 1.4.8. Diagrama de Gantt

Atenent a les consideracions comentades als punts anteriors es presenta el diagrama de Gantt (fig. 1.1.) amb les seves fases, tasques, subtasques i fites.

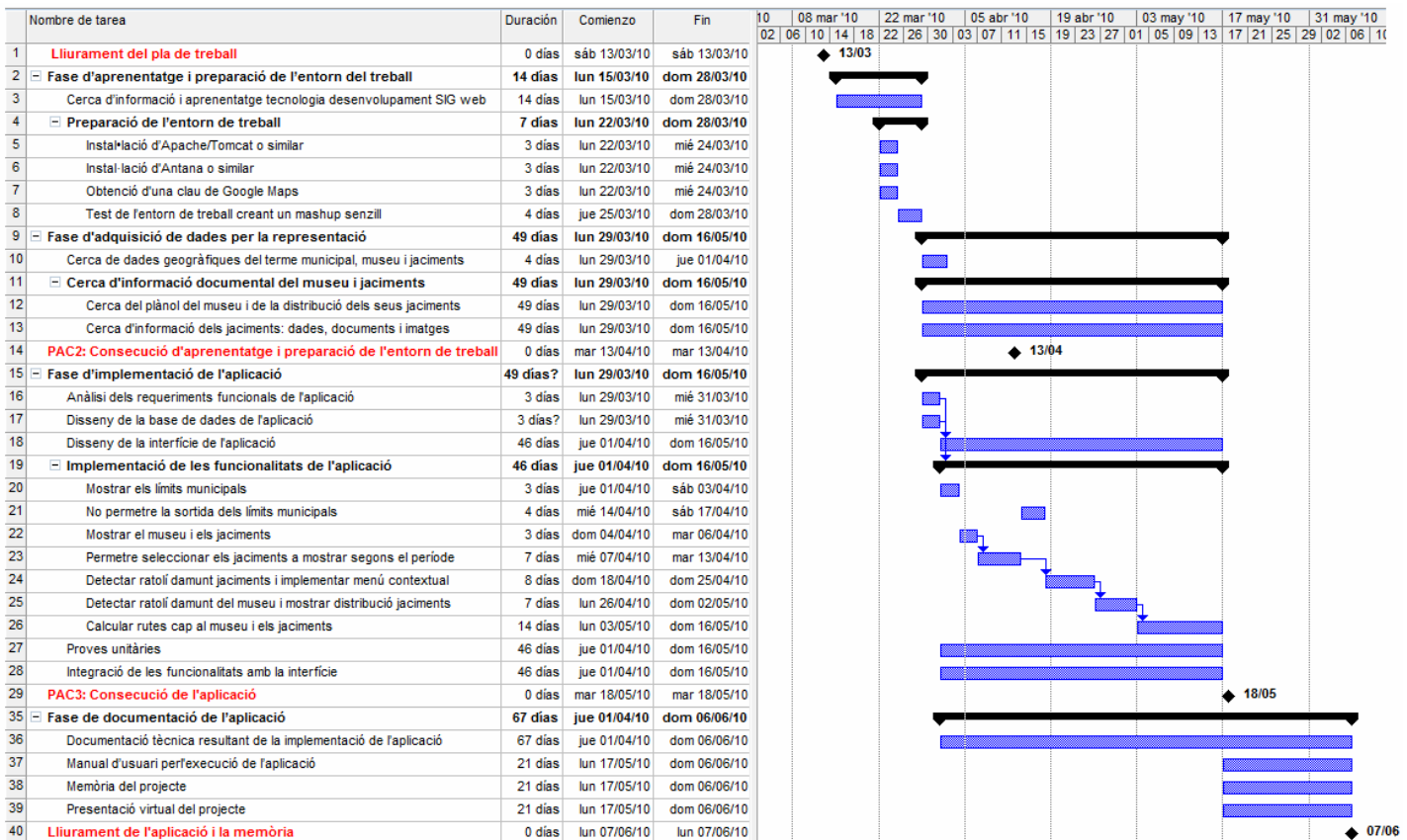


Figura 1.1. – Diagrama de Gantt amb la planificació del TFC

### 1.4.9. Productes obtinguts

El principal producte obtingut ha estat la implementació de l'aplicació web, objecte d'aquest TFC, que permet mostrar sobre Google Maps tant el museu com els jaciments del terme municipal de Molins de Rei de manera que els usuaris puguin veure les dades, els documents i les imatges de cada jaciment i el plànol del museu quan hi passin per sobre amb el ratolí, així com el càlcul de rutes, complint amb els objectius marcats al punt 1.2.2 i amb l'anàlisi de requeriments de l'apartat 4.1 d'aquesta memòria.

Com a documentació del desenvolupament del treball i implementació de l'aplicació, també s'ha obtingut aquesta memòria i una presentació en vídeo que el resumeix.

### 1.4.10. Descripció dels capítols de la memòria

La memòria s'estructura en els següents capítols:

- **Capítol 1 – Introducció.** Es presenta el treball a realitzar, els seus objectius, metodologia i planificació.
- **Capítol 2 – Preparació de l'entorn del treball.** Es detalla la tecnologia a emprar en el desenvolupament de l'aplicació al mateix temps que es mostren els passos seguits per preparar l'entorn de treball i fer-lo operatiu fins arribar a la creació d'un mashup senzill amb Google Maps.
- **Capítol 3 – Adquisició de les dades.** Es detallen els processos de conversió de les dades geogràfiques que han estat necessaris i l'obtenció dels documents i imatges del fons documental del museu.
- **Capítol 4 – Anàlisi i disseny de l'aplicació.** Es presenta l'anàlisi dels requeriments funcionals que ha de complir l'aplicació i s'elabora la proposta de disseny de la base de dades i de la interfície per donar resposta als requeriments.
- **Capítol 5 – Implementació de l'aplicació.** En primer lloc es mostra el procés seguit per crear tant la base de dades com el sistema d'arxius de l'aplicació, per després passar a detallar la implementació de cadascuna de les seves funcionalitats.
- **Capítol 6 – Manual d'usuari per l'execució de l'aplicació.** Instruccions bàsiques per l'explotació de l'aplicació.
- **Capítol 7 – Valoració econòmica.** Es fa una valoració dels recursos necessaris i una estimació del cost per dur a terme el treball.
- **Capítol 8 – Conclusions.** Es comenten les conclusions obtingudes a partir de l'elaboració del treball.
- **Capítol 9 – Glossari.** Petit recull dels conceptes més destacats que van sorgint en la redacció de la memòria.
- **Capítol 10 – Bibliografia.** Relació de les fonts consultades.



## 2. Preparació de l'entorn de treball

### 2.1. Instal·lació del servidor web Apache

L'aplicació web ha de córrer en local i per això cal instal·lar i configurar un servidor web a l'equip en que s'ha de desenvolupar aquesta. S'ha optat per instal·lar el paquet **AppServ 2.5.10** que inclou:

- Apache 2.2.8 com a servidor web.
- MySQL 5.0.51b com a sistema gestor de bases de dades.
- PHP 5.2.6 com a llenguatge d'scripts per permetre accedir a les bases de dades des de les planes web.
- phpMyAdmin 2.10.3 com a administrador de les bases de dades del servidor.

Des del lloc <http://www.appservnetwork.com/> es pot baixar el fitxer d'instal·lació **appserv-win32-2.5.10.exe**. La instal·lació és molt senzilla, acceptant les característiques per defecte que es van demanant durant el procés i indicant en els quadres de diàleg adients (fig. 2.1.) el nom del servidor (<http://localhost> per emprar-lo en local) i una contrasenya per l'usuari **root** del servidor MySQL.

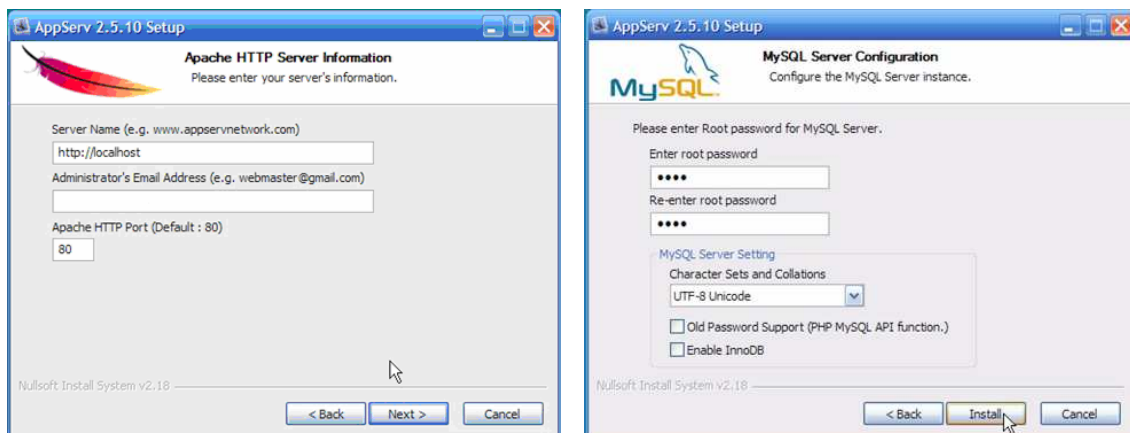


Figura 2.1. – Instal·lació d'Apache

La contrasenya que s'ha emprat en aquesta instal·lació és **sigweb**.

Al final de la instal·lació es demana iniciar Apache i MySQL. Cal acceptar i ja es pot comprovar la correcta instal·lació si a al posar la URL <http://localhost> al nostre explorador s'obté la resposta de la imatge (fig. 2.2.). Ara, les planes web que hagi de servir el servidor hauran d'estar ubicades dins la carpeta **C:\AppServ\www** o subcarpetes que hi penguin d'aquesta.

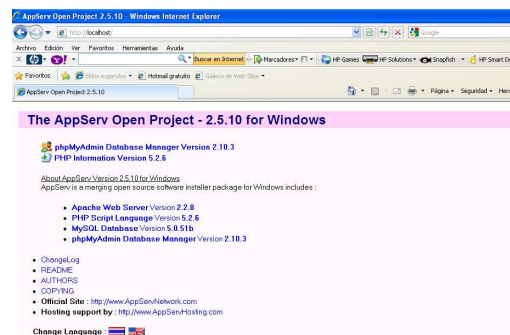


Figura 2.2. – Comprovació d'instal·lació d'Apache

## 2.2. Creació de la base de dades amb MySQL

Un cop instal·lat el servidor web cal crear la base de dades que emprerà l'aplicació.

A <http://localhost> es fa clic a l'enllaç **phpMyAdmin Database Manager Version 2.10.3** i es requerirà autenticació. El nom d'usuari és **root** i la contrasenya **sigweb**. Amb això es mostra el panell que permet entrar el nom de la nova base de dades que serà **sigweb** (fig. 2.3.).

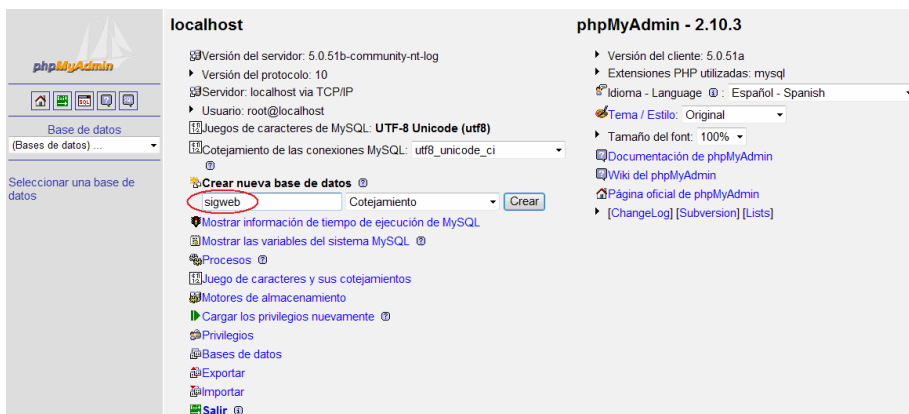


Figura 2.3. – Creació de la base de dades amb MySQL

Amb això és té creada la base de dades que emprerà l'aplicació. Falta encara definir les taules i entrar les dades. Això es veurà al capítol 4, dins l'apartat 4.2 d'anàlisi i disseny de la base de dades.

## 2.3. Instal·lació de l'IDE Aptana de Javascript

Per crear les pàgines web amb scripts de Javascript i PHP es pot fer amb un editor de textos senzill com el bloc de notes però amb un IDE adient es tenen avantatges com la creació de projectes o visualitzar millor el codi a l'emprar-se colors per diferenciar etiquetes HTML, paraules reservades de Javascript, etc.

Des del lloc <http://www.aptana.com/> es pot baixar el fitxer d'instal·lació **Aptana\_Studio\_Setup\_2.0.2.exe**. La instal·lació és molt senzilla, acceptant les opcions per defecte.

## 2.4. Obtenció d'una clau de Google Maps

Per poder inserir mapes de Google Maps a l'aplicació web a desenvolupar cal tenir una clau que es pot obtenir a <http://code.google.com/intl/es/apis/maps/signup.html>. Per obtenir la clau només cal indicar la URL del lloc web on es vol fer servir. Per emprarla en un lloc web local, com és el cas del treball, la URL a indicar és <http://localhost>.

La clau obtinguda per a aquest treball és:

**ABQIAAAA5zTB9ijfJ0ZGn6xjLVGTiRT2yXp\_ZAY8\_ufC3CFXhHIE1NvwkxSU  
sile5aTwWsajB3TDWzChCl4xeA.**

## 2.5. Creació d'un mashup senzill

Per provar l'entorn de treball s'ha anat creant un mashup senzill però complet que consisteix en mostrar un mapa de Google Maps de la zona de Molins de Rei i un requadre i un botó de manera que quan es passi per sobre del requadre o quan es faci clic a sobre del botó es mostri al mapa el 1r jaciment i al requadre les seves dades (fig. 2.4.).



Figura 2.4. – Mashup senzill

Els fitxers que componen aquest mashup són:

- **mashup.html**: és la plana web que l'usuari visualitzarà al seu explorador i conté el codi HTML i Javascript de l'API de Google Maps.
- **conexio.php**: conté el codi PHP que permet connectar-se a la base de dades.
- **jaciment.php**: conté el codi PHP que, un cop establerta la connexió amb la base de dades, obté d'aquesta les dades del 1r jaciment i les retorna formatades en XML.
- **ajax\_functions.js**: conté codi Javascript de les funcions que permeten utilitzar la tecnologia AJAX al mashup.

Al capítol 4 es comenta amb detall el disseny de la base de dades i al capítol 5 com han estat creades les seves taules. En aquest apartat es detalla més com s'obtenen les dades amb el codi PHP i AJAX. A continuació es mostren els passos següents:

### 2.5.1. Visualitzar el mapa

Es mostra aquí el codi necessari per visualitzar el mapa a la web. Bàsicament:

- A la capçalera del document **mashup.html**, és a dir, entre les etiquetes `<head>` i `</head>`, es col·loca:
  - l'script per accedir a l'API de Google Maps amb la clau obtinguda.
  - l'script amb la variable *map* com a contenidora del mapa que, un cop es cridi a la funció *load()* i aquesta comprovi que el navegador és compatible, emprarà dues funcions de l'API de Google Maps:
    - la funció *GMap2()* per obtenir el contenidor del mapa i situar-lo dins la capa del document amb l'identificador *map*.
    - la funció *setCenter()* per indicar la latitud i longitud a la qual s'ha de centrar el mapa, així com el seu nivell de zoom (de l'1 al 19).

- Al cos del document **mashup.html**, s'observa a l'etiqueta `<body>` les accions que es vol que es produeixin, respectivament, al carregar-se i tancar-se la pàgina web, i que són la càrrega (mitjançant la crida a la funció `load()` comentada abans) o descàrrega del mapa. També es pot observar al cos del document l'etiqueta `<div>` amb l'identificador `map` que és la capa on es mostrarà el mapa.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" dir="ltr" lang="en">
<head> <title>Mashup</title>
<script src="http://maps.google.com/maps?file=api&v=2&sensor=true_or_false&
amp;key=ABQIAAAA5zTB9ijfJ0ZGn6xjLVGTiRT2yXp_ZAY8_ufC3CFXhHIE1NvwkxSUSile5aTwWsaJB3TDWzCh
Cl4xeA"
type="text/javascript"></script>

<script type="text/javascript">
var map;
function load() {
  if (GBrowserIsCompatible()) {
    map = new GMap2(document.getElementById("map"));
    map.setCenter(new GLatLng(41.427776, 2.0256945), 13);
  }
}
</script>
</head>

<body onload="load();" onunload="GUnload();">
<table align="center">
  <td> <div id="map" style="width: 500px; height: 500px"></div> </td>
</table>
</body>
</html>
```

## 2.5.2. Cridar la funció per obtenir les dades

El codi HTML del cos del document **mashup.html** es completa introduïnt una capa amb l'identificador `dadesjaciment` i un formulari amb un botó. La capa `dadesjaciment` és un requadre on es mostraran les dades que s'obtindran del jaciment. Per tal d'aconseguir les dades es cridarà a la funció `obtenirJaciment()` tant quan es faci clic al botó (`onsubmit`) com quan el ratolí es mogui per la capa (`onmouseover`).

```
<div align="center">
  <h1>Mashup senzill amb Google Maps, JavaScript, PHP, MySql, XML i Ajax</h1>
  <p align="center">Mou el ratolí pel requadre de sota o prem el botó.<br/>
  Veuràs la informació d'un jaciment i la seva posició al mapa.</p>
  <div id="dadesjaciment" class="displaybox" onmouseover="obtenirJaciment();" ></div>
</div>

<center>
  <form name="consulta" action="" onsubmit="obtenirJaciment(); return false">
    <label> <input type="submit" value="Mostrar el jaciment al mapa" /> </label>
  </form>
</center>
```

La funció `obtenirJaciment()` està implementada al document **ajax\_functions.js** de manera que a la capçalera de **mashup.html** caldrà incloure el següent script.

```
<script src="ajax_functions.js" type="text/javascript"></script>
```

### 2.5.3. Emprar la tecnologia AJAX

La resposta que s'obtingui del servidor web amb aquesta tecnologia s'emmagatzemarà a l'objecte *myReq* que es crea al següent script que s'ubicarà a la capçalera de **mashup.html**.

```
<script type="text/javascript">  
var myReq = getXMLHttpRequest();  
</script>
```

La resta de la tecnologia AJAX està implementada al document **ajax\_functions.js** amb el següent codi on es pot observar:

- La funció *getXMLHttpRequest()* crea l'objecte *XMLHttpRequest* de Javascript, segons el navegador que s'utilitzi, que permetrà connectar-se amb el servidor web i obtenir dades d'aquest sense haver de recarregar la pàgina web. Un cop creat l'objecte es passarà a *myReq*.
- A la funció *obtenirJaciment()* l'objecte *myReq* invocarà l'execució del codi del fitxer **jaciment.php** que és el que farà la consulta MySQL a la base de dades i recollirà les dades de la consulta mitjançant la cadena retornada per la funció *theHTTPResponse()*.
- A la funció *theHTTPResponse()* s'observa que la resposta són dades del jaciment que es desen a les variables *latitud*, *longitud*, *nom* i *ubicació*. La resposta s'obté en format XML de manera que cal indicar en quina etiqueta de la resposta XML es trobarà cada valor mitjançant la funció *getElementByTagName()*. Al final es mostren les dades obtingudes a la capa *dadesjaciment* de **mashup.html** actualitzant el seu atribut *innerHTML* i es crida a la funció *mostraPunt()*, implementada a **mashup.html** i encarregada de mostrar el jaciment al mapa mitjançant les seves dades de latitud i longitud.

```
function getXMLHttpRequest() {  
    var req = false;  
    try { /* for Firefox */  
        req = new XMLHttpRequest();  
    } catch (err) {  
        try { /* for some versions of IE */  
            req = new ActiveXObject("Msxml2.XMLHTTP");  
        } catch (err) {  
            try { /* for some other versions of IE */  
                req = new ActiveXObject("Microsoft.XMLHTTP");  
            } catch (err) {  
                req = false;  
            }  
        }  
    }  
    return req;  
}  
  
function obtenirJaciment() {  
    myReq.open("GET", 'jaciment.php', true);  
    myReq.onreadystatechange = theHTTPResponse;  
    myReq.send(null);  
}
```

```
function theHTTPResponse() {  
  if (myReq.readyState == 4) {  
    if(myReq.status == 200) {  
      var latitud = myReq.responseXML.getElementsByTagName("latitud")[0];  
      var longitud = myReq.responseXML.getElementsByTagName("longitud")[0];  
      var nom = myReq.responseXML.getElementsByTagName("nom")[0];  
      var ubicacio = myReq.responseXML.getElementsByTagName("ubicacio")[0];  
      document.getElementById('dadesjaciment').innerHTML =  
nom.childNodes[0].nodeValue + " - " + ubicacio.childNodes[0].nodeValue + " - <br> " +  
latitud.childNodes[0].nodeValue + ", " + longitud.childNodes[0].nodeValue;  
      mostraPunt(latitud.childNodes[0].nodeValue, longitud.childNodes[0].nodeValue);  
    }  
  }  
}
```

A continuació es mostra el codi que cal afegir a l'script de les funcions de l'API de Google Maps dins **mashup.html** per implementar la funció *mostraPunt()* on, bàsicament, es crea un marcador del punt que s'afegeix com una capa al mapa amb el mètode de l'API *addOverlay()*.

```
function createMarker(point) {  
  var marker = new GMarker(point);  
  return marker;  
}  
function mostraPunt(latitud, longitud){  
  var point = new GLatLng(latitud,longitud);  
  map.addOverlay(createMarker(point));  
}
```

#### 2.5.4. La consulta MySQL

La consulta MySQL a la base de dades s'implementa al fitxer **jaciment.php** que inclou el fitxer **conexio.php**.

El codi de **conexio.php** es connecta a la base de dades **sigweb** del servidor local emprant l'usuari **root** i la seva contrasenya **sigweb**.

```
<?  
$conexio = mysql_connect("localhost","root","sigweb");  
mysql_select_db("sigweb",$conexio);  
?>
```

Un cop establerta la connexió amb la base de dades **jaciment.php** selecciona tots els elements de la taula *jaciments* i obté el 1r element o registre, i retorna el contingut dels seus camps en formant XML.

```
<?php  
include ("conexio.php");  
$sql=mysql_query("SELECT * FROM jaciments");  
$row = mysql_fetch_array($sql);  
  
header('Content-Type: text/xml');  
echo "<?xml version='1.0' ?>  
  <jaciment>  
    <nom>".$row['nom']. "</nom>  
    <ubicacio>".$row['ubicacio']. "</ubicacio>  
    <latitud>".$row['latitud']. "</latitud>  
    <longitud>".$row['longitud']. "</longitud>  
  </jaciment>";  
?>
```

## 2.6. Resum de la tecnologia a emprar

La tecnologia descrita a l'anterior apartat serà la base pel desenvolupament del treball. A continuació es mostra una comparació d'aquest model respecte el model clàssic d'aplicació web (fig. 2.5.). Es pot observar com el motor AJAX de Javascript s'encarrega de la comunicació asíncrona amb el servidor rebent les dades en format XML.

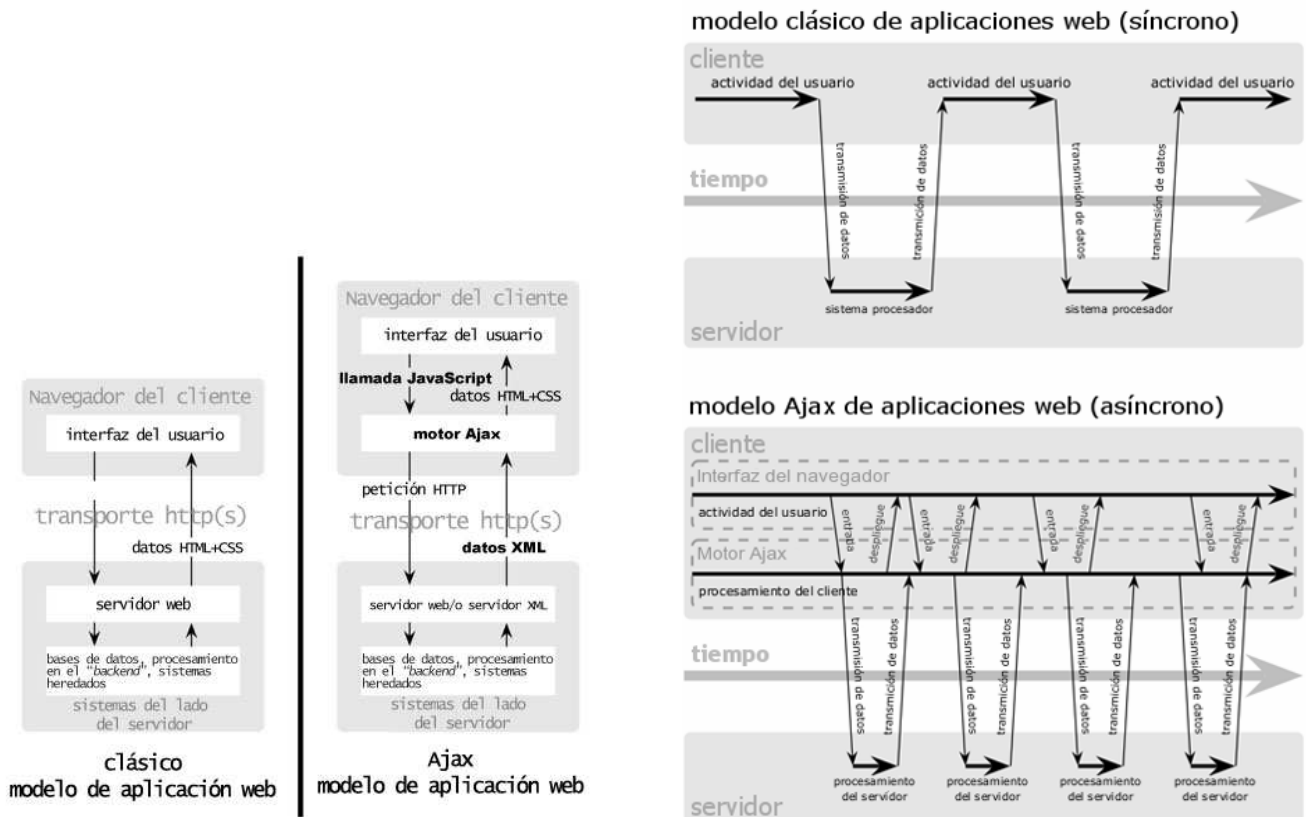


Figura 2.5. – Comparació entre el models clàssic i AJAX d'aplicacions web

### 3. Adquisició de les dades

#### 3.1. Preparació de les dades geogràfiques

Les dades geogràfiques dels jaciments han estat facilitades pel consultor en coordenades geodèsiques *UTM 31N* del *datum ED50*. Com que Google Maps empra coordenades geogràfiques de latitud i longitud amb graus decimals del datum *ETRS89*, cal realitzar la corresponent conversió.

Per veure com s'ha procedit per realitzar aquesta conversió es mostra, a mode d'exemple, el procés seguit pel cas del jaciment del Canyet, a Catellbisbal.

- A la web <http://www.icc.cat> de l'Institut Cartogràfic de Catalunya s'entren les coordenades UTM facilitades (E 415.710,0 m - N 4.590.029 m). Aquesta posició serà el centre del mapa mostrat (fig. 3.1.).

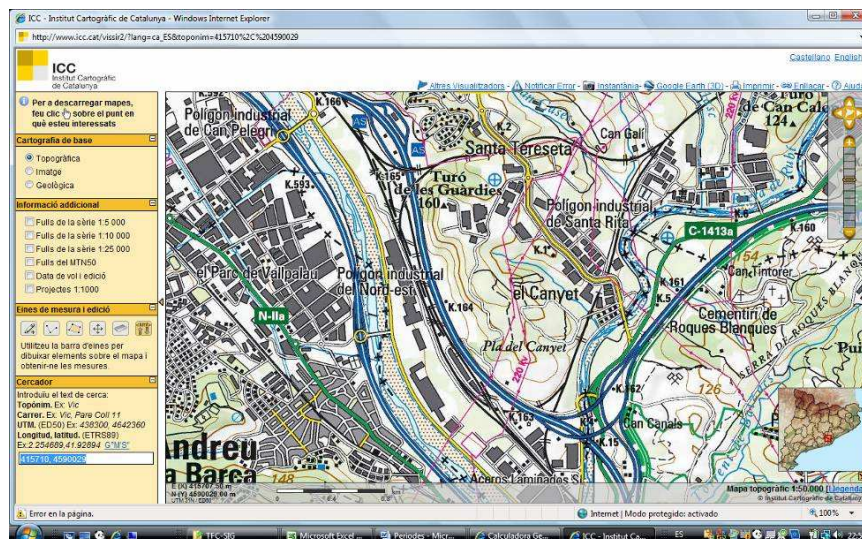


Figura 3.1. – Comprovació de coordenades UTM 31N ED50 amb la web de l'ICC

- S'empra la calculadora geodèsica de l'Institut Geogràfic Nacional <http://www.ign.es/ign/home/calculadora/UTM2LL.jsp> per convertir les coordenades geodèsiques a geogràfiques ED50 (fig. 3.2.).

x UTM	415710	m
y UTM	4590029	m
HEMISFERIO	N	
Huso	31	
ELIPSOIDE	<input checked="" type="radio"/> Internacional <input type="radio"/> SGR80	
<b>Coordenades Geodèsicas</b>		
LONGITUD	1 ° 05 ' 26.92736 " E	
LATITUD	41 ° 27 ' 24.11405 " N	
Anamorfosis	0.999687296	m
Convergència	0,° 40' 8.982011"	

Figura 3.2. – Conversió de coordenades geodèsiques UTM 31N ED50 a geogràfiques



- Ara s'empra la calculadora geodèsica de l'Institut Geogràfic Nacional <http://www.ign.es/ign/home/calculadora/ED2WGS.jsp> per convertir les coordenades geogràfiques ED50 a coordenades geogràfiques ETRS89 (fig. 3.3.).

LONGITUD ED50	1	0	59	'	26.9273	"	E
LATITUD ED50	41	0	27	'	24.1140	"	N
<b>ETRS89 Coordenadas</b>							
LONGITUD ETRS89	1	0	59	'	22.7746	"	E
LATITUD ETRS89	41	0	27	'	20.0745	"	N

Figura 3.3. – Conversió de coordenades geogràfiques d'ED50 a ETRS89

- Finalment s'empra la calculadora de la Federal Communications Commission <http://www.fcc.gov/mb/audio/bickel/DDMMSS-decimal.html> per passar les coordenades geogràfiques ETRS89 a la notació decimal que empra Google Maps, obtenint-se els valors de latitud 41.455576 i longitud 1.989659 (fig. 3.4.).

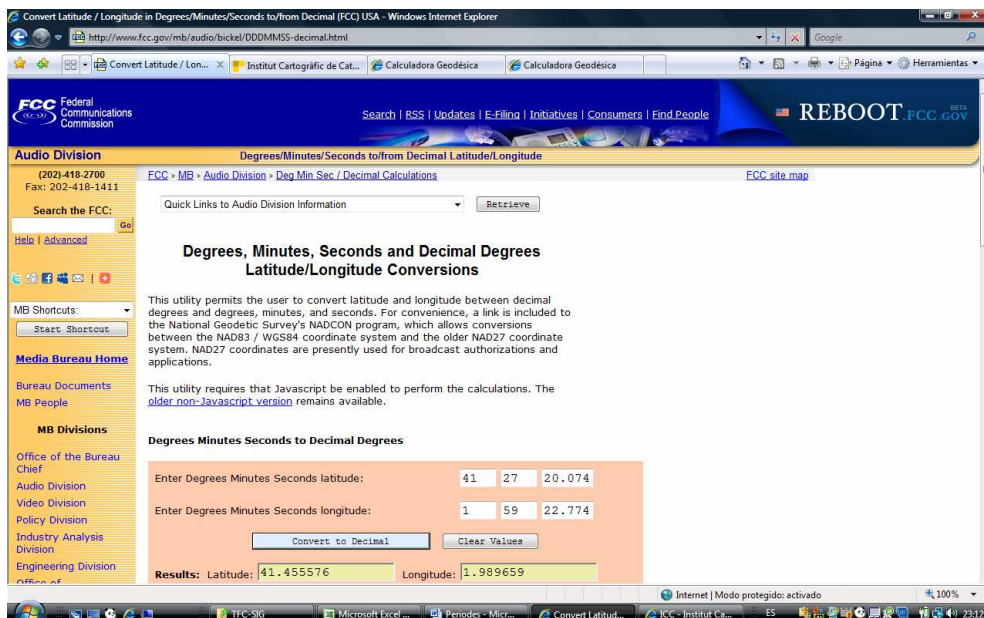


Figura 3.4. – Conversió de coordenades geogràfiques ETRS89 a notació decimal

- Entrant aquests valors a Google Maps aquest mostra el mateix punt que mostrava l'Institut Cartogràfic de Catalunya (fig. 3.5.).

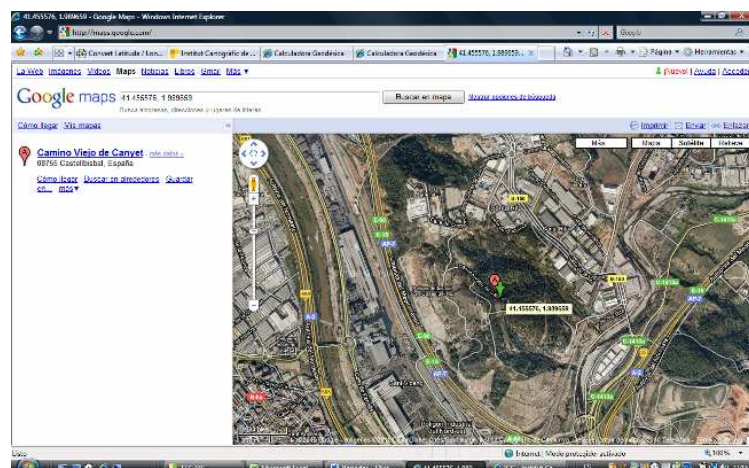


Figura 3.5. – Comprovació de coordenades geogràfiques ETRS89 amb Google Maps

### 3.2. Les dades del fons documental del museu

Les dades del fons documental del museu consisteixen en una breu descripció de cada període de cada jaciment i algunes imatges. Es mostra com a exemple la informació disponible per al jaciment de Ca n'Albareda (fig. 3.6.).

**Paleolític**

Ca n'Albareda.  
Sant Feliu de Llobregat  
(80/90000 a.C.)

*El jaciment data del Paleolític, principalment trobem indústria lítica que mostra les característiques del període. Són pobles nòmades que utilitzaven eines rudimentàries com les que veiem. Tenien una economia basada en la caça i la recol·lecció*



*Indústria lítica*  
*Es tracta d'eines destinades a activitats domèstiques i de caça.*

Figura 3.6. – Fons documental d'un jaciment

Aquesta informació servirà per crear un document **PDF** sobre el jaciment i obtenir les seves imatges en format **JPG**. L'usuari tindrà accés tant als documents com a les imatges de cada jaciment quan es situï a sobre d'aquests amb el ratolí al mapa.

Al llarg dels propers capítols es detallen tant el disseny com la implementació de la base de dades que inclou tant les dades geogràfiques com les del fons documental. També es detallarà llavors la implementació de les funcionalitats que les utilitzen.

## 4. Anàlisi i disseny de l'aplicació

### 4.1. Anàlisi dels requeriments

Els requeriments funcionals de l'aplicació són:

- Mostrar els límits municipals de Molins de Rei i no permetre a l'usuari sortir d'aquests. S'entén que sempre s'ha de visualitzar aquest mapa a l'aplicació.
- Mostrar el museu i els jaciments permetent a l'usuari seleccionar el període dels jaciments que vol veure o bé si els vol veure tots. S'entén que es vol diferenciar el museu dels jaciments de manera que es pugui triar també entre mostrar o no el museu. Els jaciments que es mostrin al mapa s'han d'etiquetar i acompanyar d'un llistat d'aquests dins l'aplicació web de manera que es pugui identificar a quin jaciment concret correspon cada punt mostrat al mapa. El museu es mostrarà al mapa d'un altre color per diferenciar-lo ràpidament dels jaciments.
- Mostrar un menú contextual al passar el ratolí per sobre de cada jaciment que permeti seleccionar la informació a consultar: dades del jaciment, imatges del jaciment i els documents disponibles d'aquest. S'entén que aquesta possibilitat es faci a la manera típica de Google Maps quan al passar amb el ratolí per sobre de punts d'interès dels mapes es mostra una finestra d'informació que, en aquest cas, es pot implementar amb pestanyes: una per les dades del jaciment, una altra per les imatges i una altra pels documents. Quan l'usuari seleccioni una imatge o un document aquest s'ha de poder veure en un espai adient de manera que això es farà obrint-lo en una nova finestra del navegador.
- Mostrar el plànol del museu amb la distribució dels seus jaciments d'exposició quan es passi el ratolí per sobre d'aquest. Aquest plànol es mostrarà reduït en una finestra d'informació típica de Google Maps associada al punt del museu i serà, al mateix temps, enllaç que porti a veure'l a major grandària en una nova finestra del navegador.
- Calcular rutes des de qualsevol punt d'origen triat per l'usuari fins al museu o fins a qualsevol jaciment. Es permetrà a l'usuari entrar la posició origen mitjançant un quadre de text d'un formulari o bé fent clic a sobre d'un punt al mapa. Quant a la posició destí aquesta la podrà seleccionar d'un llistat amb els jaciments i museu que s'estiguin mostrant en cada moment.

Les imatges i documents dels jaciments s'emmagatzemaran en local. A tal efecte es preveu crear, dins la carpeta on s'allotgi l'aplicació, una subcarpeta per les imatges i una altra pels documents.

## 4.2. Disseny de la base de dades

Per tal de poder complir amb els requeriments funcionals es proposa un model relacional de base de dades (fig.4.1.).

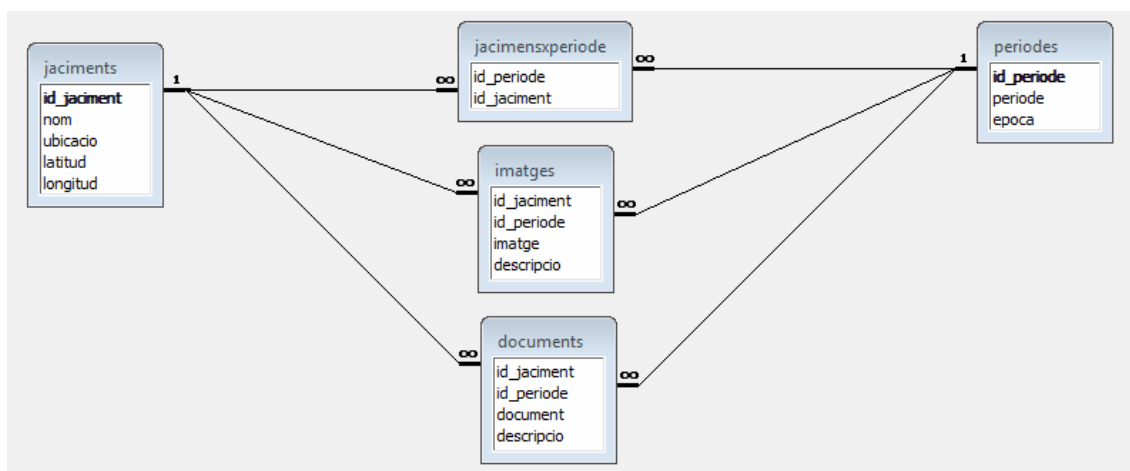


Figura 4.1. – Model relacional de la base de dades

- Les dades a emmagatzemar a la taula **jaciments** són el nom, ubicació (localitat) i posició (latitud i longitud) de cada jaciment desats com a text. El camp *id\_jaciment* és un identificador numèric que serà camp clau en aquesta taula.
- La taula **periodes** és necessària ja que un mateix jaciment pot pertànyer a més d'un període. El camp *id\_periode* és un identificador numèric que serà camp clau en aquesta taula. Les dades a emmagatzemar són el nom del període (per exemple, *Edat de Bronze*) i la seva època (per exemple, *2500 a.C.*).
- La relació entre **jaciments** i **periodes** es materialitza amb la taula **jacimentsxperiode**.
- La taula **documents** contindrà el nom de cada document que pertanyi a un jaciment i període concrets, emprant *id\_jaciment+id\_periode* com a clau forana.
- La taula **imatges** contindrà el nom de cada imatge que pertanyi a un jaciment i període concrets, emprant *id\_jaciment+id\_periode* com a clau forana.

No es preveu necessari per al desenvolupament de l'aplicació que la informació del museu hagi d'estar recollida a la base de dades doncs pot formar part de l'aplicació sense necessitat d'obtenir-la de cap consulta. Aquesta decisió es justifica pel fet que la informació que cal tenir en compte (nom, ubicació, telèfon, posició i imatge del seu plànol) només representaria un registre en una taula aïllada i, per tant, a efectes de desenvolupament de seria indiferent implementar-la a la base de dades o directament al mòdul de l'aplicació que s'encarregaria de donar aquesta resposta des del servidor.

### 4.3. Disseny de la interfície

Es proposa un disseny clar, diàfan i intuïtiu amb 3 zones ben diferenciades (fig. 4.2):

- Capçalera amb el títol de l'aplicació i els formularis que mostrin i permetin les diferents funcionalitats de l'aplicació:
  - Mostrar o no el museu, emprant botons de ràdio.
  - Mostrar cap jaciment o bé mostrar-los tots o només els d'un període determinat, emprant botons de ràdio.
  - Calcular rutes, donant la possibilitat d'introduir l'origen bé escrivint-lo en quadre de text o bé fent clic directament sobre un punt al mapa. El destí es triarà d'una llista de selecció que contingui els jaciments o museu que s'estiguin mostrant en cada moment al mapa.
- Meitat esquerra del cos de la pàgina en que el mapa estigui sempre visible. Es preveu mostrar-lo inicialment a un nivell de zoom que permeti localitzar la zona dels jaciments, emmarcada dins un requadre vermell, respecte a la ciutat de Barcelona. Aquest zoom serà variable i permetrà veure la zona amb més detall. En cas, però, es permetrà sortir de la visió de la zona dels jaciments.
- Meitat dreta del cos de la pàgina amb espais per mostrar diferents informacions:
  - Espai que indiqui que la informació del museu i el seu plànol de la seva distribució per jaciments es mostrarà quan es mostri el museu al mapa.
  - Espai per llistar els jaciments que es mostrin al mapa i que informi també que les dades del fons documental de cada jaciment seran accessibles quan l'usuari posicioni el ratolí a sobre del jaciment al mapa.
  - Espai per mostrar la informació d'una ruta calculada.

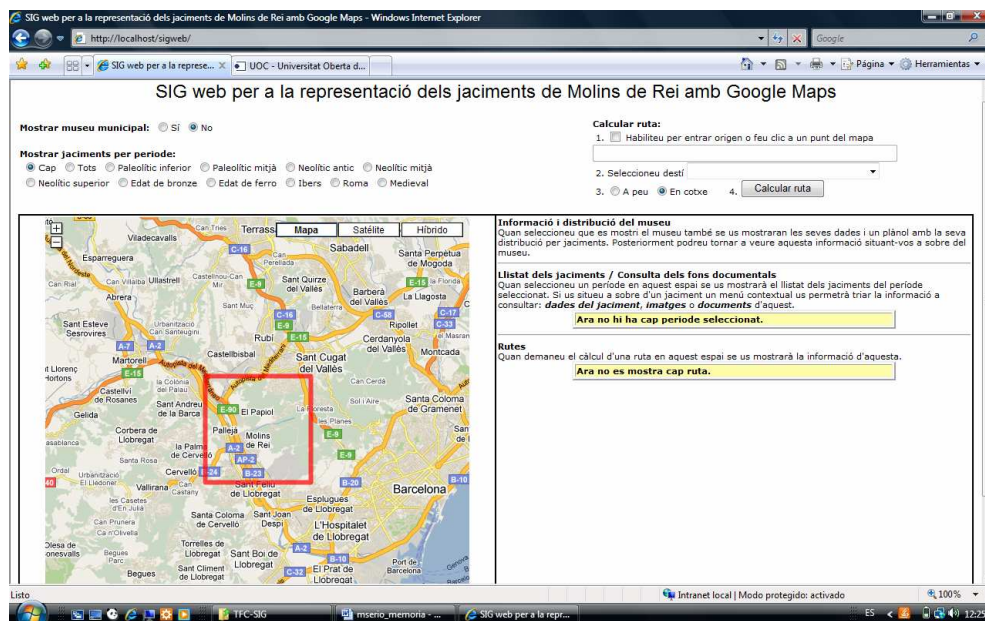


Figura 4.2. – Disseny de la interfície

El disseny s'ha optimitzat per resolucions de pantalla de 1024x768 píxels i superiors, tant per Internet Explorer com per Mozilla Firefox i Google Chrome.

## 5. Implementació de l'aplicació

### 5.1. Implementació de la base de dades

Ja al punt 2.2 s'ha comentat com crear la base de dades **sigweb** amb el servidor MySQL. El procediment seria equivalent a executar l'script: `CREATE DATABASE `sigweb` ;`

A continuació es mostra com s'han creat les taules dissenyades al punt 4.2. En concret s'observarà el procediment per crear la taula **jaciments** des del panell phpMyAdmin, l'script MySQL equivalent i la resta d'scripts per la resta de les taules.

Un cop creada la base de dades **sigweb** aquesta té 0 taules encara i acte seguit es permet introduir el nom i nombre de camps de la nova taula (fig. 5.1.).

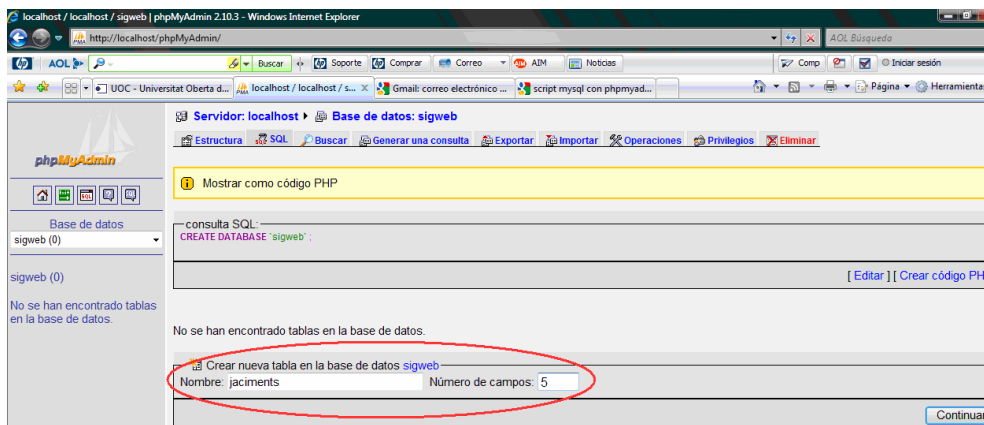


Figura 5.1. – Creació de taula jaciments de la base de dades

Es continua introduint el nom, tipus i longitud dels camps de la taula. El camp *id* es marca com a camp clau en una columna que hi ha més a la dreta i que no es veu a la figura (fig. 5.2.).

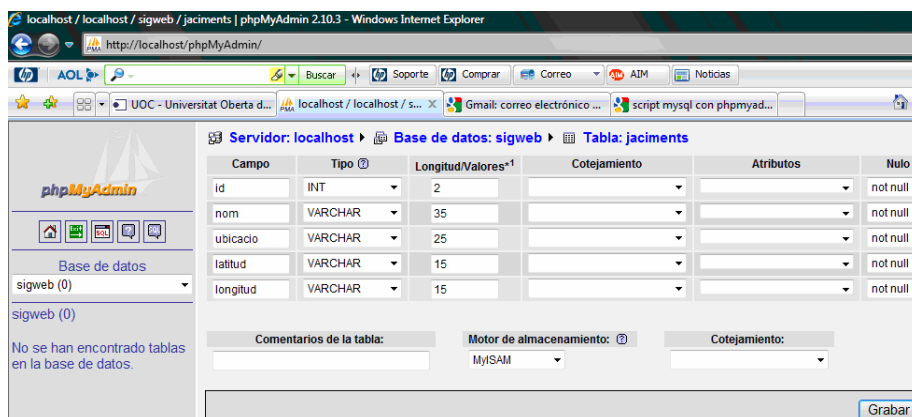


Figura 5.2. – Definició dels camps de la taula jaciments

Al grabar la taula es mostra l'script equivalent:

```
CREATE TABLE `jaciments` (  
  `id` INT( 2 ) NOT NULL ,  
  `nom` VARCHAR( 35 ) NOT NULL ,  
  `ubicacio` VARCHAR( 25 ) NOT NULL ,  
  `latitud` VARCHAR( 15 ) NOT NULL ,  
  `longitud` VARCHAR( 15 ) NOT NULL ,  
  PRIMARY KEY ( `id` )  
 ) ENGINE = MYISAM ;
```

Els scripts per crear la resta de les taules són:

```
CREATE TABLE `perioades` ( `id_periode` INT( 2 ) NOT NULL , `periode` VARCHAR( 25 ) NOT NULL ,  
  `epoca` VARCHAR( 35 ) NOT NULL , PRIMARY KEY ( `id_periode` )  
 ) ENGINE = MYISAM ;  
  
CREATE TABLE `jacimentsxperiode` ( `id_periode` INT( 2 ) NOT NULL , `id_jaciment` INT( 2 ) NOT NULL  
 ) ENGINE = MYISAM ;  
  
CREATE TABLE `imatges` ( `id_jaciment` INT( 2 ) NOT NULL , `id_periode` INT( 2 ) NOT NULL , `imatge` VARCHAR(  
  25 ) NOT NULL , `descripcio` VARCHAR( 100 ) NOT NULL  
 ) ENGINE = MYISAM ;  
  
CREATE TABLE `documents` ( `id_jaciment` INT( 2 ) NOT NULL , `id_periode` INT( 2 ) NOT NULL , `document`  
  VARCHAR( 50 ) NOT NULL , `descripcio` VARCHAR( 100 ) NOT NULL  
 ) ENGINE = MYISAM ;
```

Per inserir registres en una taula des del panell phpMyAdmin es selecciona aquesta i es fa clic sobre la pestanya *Inserir*. Llavors el panell mostra els quadres de text per poder omplir amb els valors dels camps. Es mostra un exemple per a un jaciment (fig. 5.3.) i el seu script MySQL equivalent.

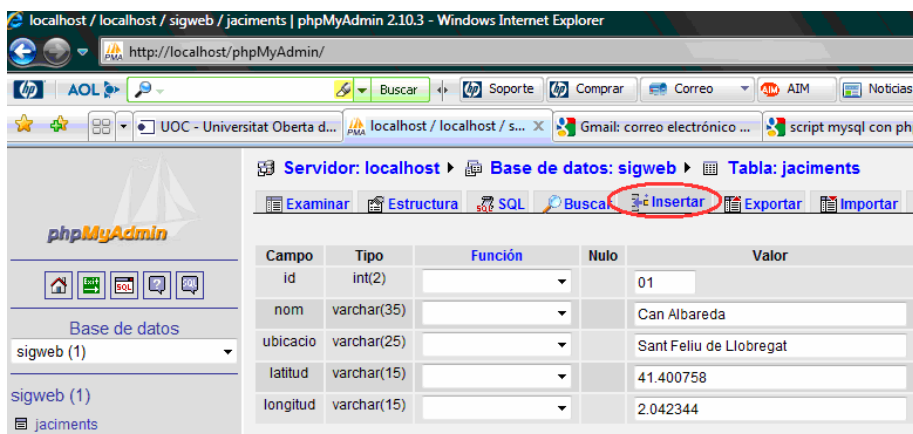


Figura 5.3. – Inserció de registres a la taula jaciments

```
INSERT INTO `sigweb`.`jaciments` (  
  `id` ,  
  `nom` ,  
  `ubicacio` ,  
  `latitud` ,  
  `longitud`  
 )  
 VALUES (  
  '01', 'Can Albareda', 'Sant Feliu de Llobregat', '41.400758', '2.042344'  
 );
```

## 5.2. Implementació del sistema d'arxius

El servidor desa els fitxers de la base de dades **sigweb** creada a la carpeta **C:\AppServ\MySQL\data\sigweb**. En quant al sistema d'arxius de l'aplicació, aquest s'ubica a **C:\AppServ\www\sigweb** i consta dels fitxers i carpetes següents (fig. 5.4.).

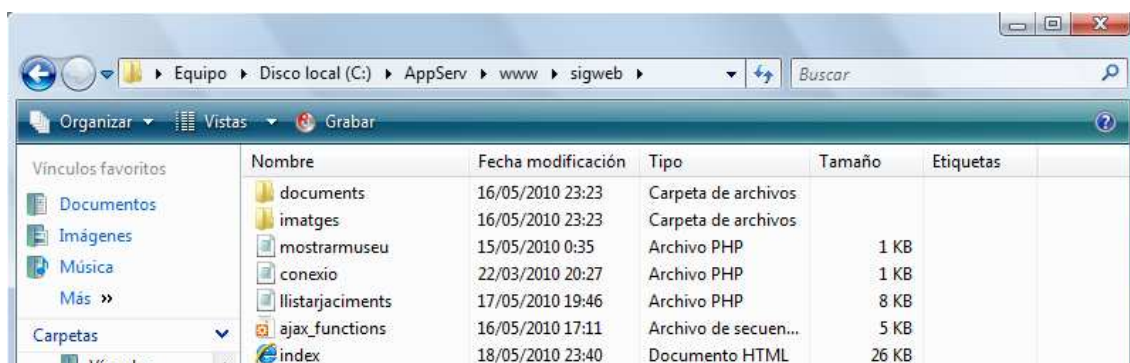


Figura 5.4. – Sistema d'arxius

- **index.html**: és la plana web que l'usuari visualitzarà al seu explorador i conté el codi HTML i Javascript de l'API de Google Maps.
- **conexio.php**: conté el codi PHP que permet connectar-se a la base de dades.
- Resta de fitxers **php**: contenen el codi PHP que, un cop establerta la connexió amb la base de dades, implementen diferents consultes i obtenen d'aquesta les dades requerides per cada funcionalitat, retornant-les formatades en XML.
- **ajax\_functions.js**: conté codi Javascript de les funcions que permeten utilitzar la tecnologia AJAX al mashup.
- Carpetes **documents** i **imatges**: en aquestes carpetes es desaran respectivament els documents i imatges del fons documental del museu.

## 5.3. Implementació de les funcionalitats

### 5.3.1. Mostrar els límits municipals

Per mostrar els jaciments de Molins de Rei i voltants, però pertanyents tots al Baix Llobregat, cal observar una sèrie de característiques a l'hora de mostrar aquests límits:

- La zona a mostrar ha d'incloure els jaciments més extrems:
  - Jaciment del Canyet a Castellbisbal [Lat: 41.455576 - Lon: 1.989659]. És el situat a l'extrem superior (major latitud).
  - La Palma de Cervelló [Lat: 41.413948 - Lon: 1.979057]. És el situat a l'extrem esquerra (menor longitud).
  - Puig Castellar a Torre Vileta [Lat: 41.397267 - Lon: 1.995311]. És el situat a l'extrem inferior (menor latitud).
  - Ermita de Santa Creu d'Olorda a Santa Creu d'Olorda [Lat: 41.415043 - Lon: 2.059305]. És el situat a l'extrem dret (major longitud).



- El nivell de zoom ha de permetre llegir clarament la toponímia de les seves principals poblacions.
- La grandària del mapa ha de permetre que sigui sempre visible a la web tot deixant espai suficient per mostrar la resta d'informacions.

A tal efecte s'ha observat suficient, mitjançant proves experimentals, centrar el mapa al punt [Lat: 41.427776 - Lon: 2.0256945] amb un nivell de zoom 13 de Google Maps per poder observar tots els jaciments (fig. 5.5.). Als següents apartats s'explica amb detall el codi per mostrar el museu i els jaciments al mapa.

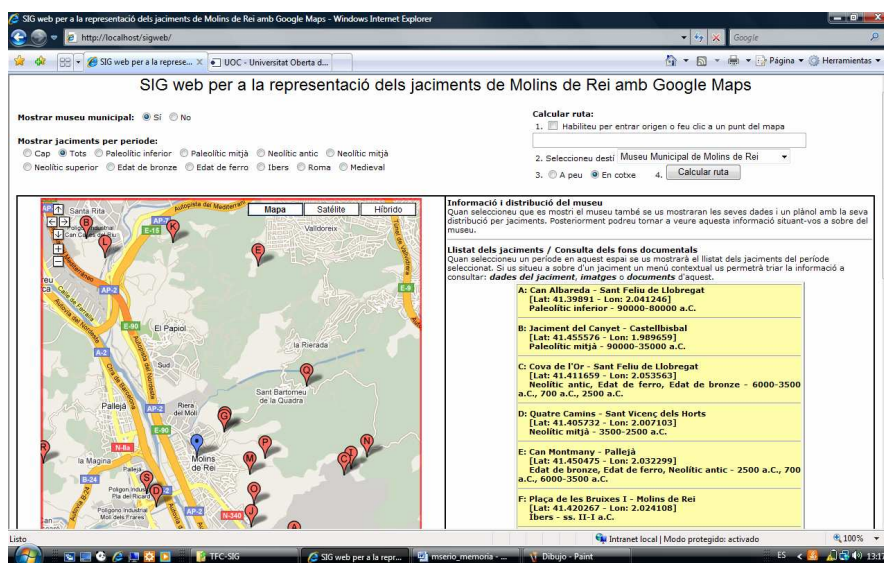


Figura 5.5. – Mapa centrat a nivell de zoom 13

Tot i que amb el nivell de zoom 13 es poden observar tots els jaciments també es considera interessant incorporar els següents aspectes a l'hora de mostrar els límits:

- Mostrar inicialment el mapa amb un nivell de zoom inferior per tal que la zona dels jaciments resulti millor localitzada per l'usuari. Triant un nivell de zoom 11 es pot observar clarament la situació de la zona respecte la ciutat de Barcelona.
- El zoom ha de ser variable de manera que es permeti a l'usuari apropar-se convenientment a la zona dels jaciments i, a partir del nivell de zoom 13 i fins al 16, afegir al mapa un control de zoom que també li permeti el desplaçament amunt, avall, a dreta i a esquerra per poder apropar-se i veure amb major detall els elements del mapa del seu interès. El desplaçament, però, serà limitat dins la zona dels jaciments. Aquesta funcionalitat es detallarà a l'apartat 5.3.4.
- Emmarcar la zona dels jaciments dins un requadre vermell que permeti a l'usuari ubicar millor la zona, sobretot en els nivells de zoom inferiors a 13.
- Informar l'usuari quan intenti accedir a un nivell de zoom inferior a 11 o superior a 16, i quan intenti sortir de la zona dels jaciments, sense permetre-li.
- Afegir controls al mapa que permetin, a més de triar el nivell de zoom, triar també el tipus de mapa –com el de vista satèl·lit per exemple–, i mostrar un regle d'escala per orientar les distàncies en quilòmetres.

Abans de detallar el codi per aconseguir això es mostra el resultat d'implementar aquestes característiques (fig. 5.6. a 5.8.) tot mostrant, a mode d'exemple, la situació del museu. A més, a la cantonada superior esquerra es situa el control del zoom que permet ampliar-lo o reduir-lo (el desplaçament és possible a partir del nivell de zoom 13), a la cantonada superior dreta es veu el pallet per seleccionar el tipus de mapa i al marge inferior s'observa el control d'escala.

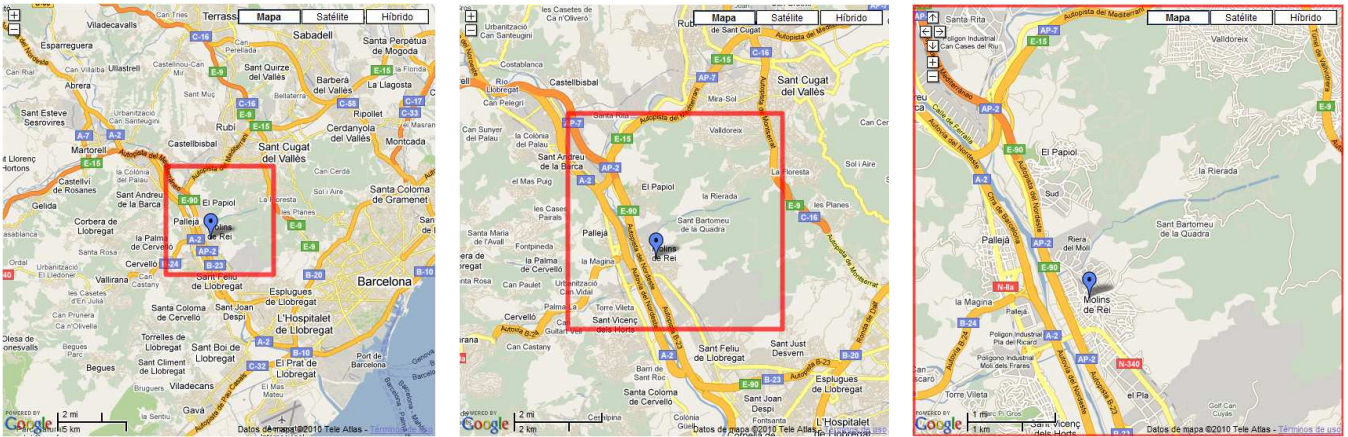


Figura 5.6. – Nivells de zoom 11, 12 i 13

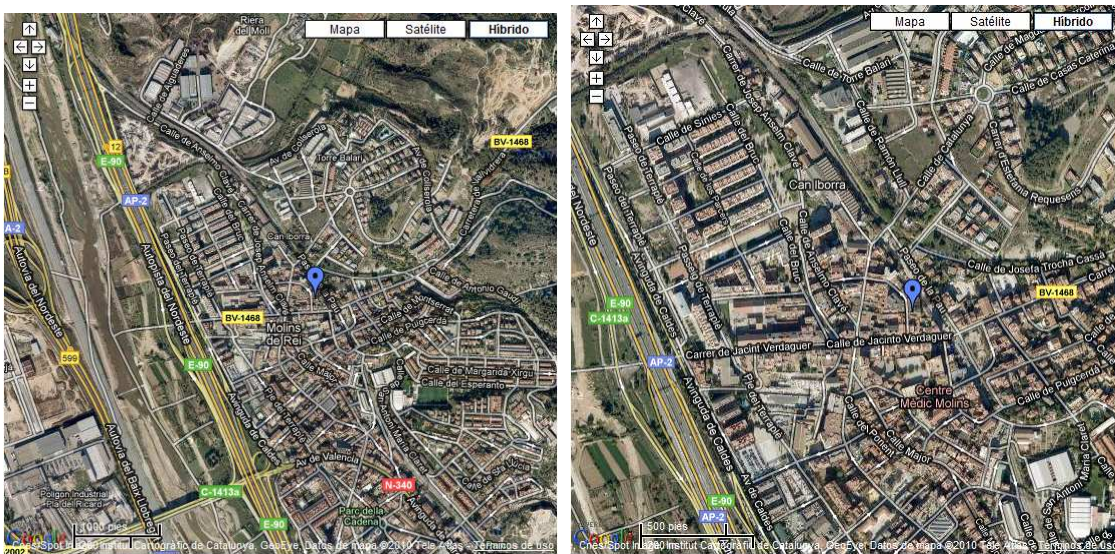


Figura 5.7. – Nivells de zoom 15 i 16 sobre mapes de tipus híbrid



Figura 5.8. – Alertes a l'intentar excedir els nivells de zoom o els límits

Per afegir els controls de zoom, tipus de mapa i escala s'ha afegit a **index.html** el següent codi a l'espai de l'script de les funcions de l'API de Google Maps dins la funció *load()*.

```
var zoom;
var puntZona1;
var puntZona2;
var puntZona3;
var puntZona4;
var zona;
var control;

function load() {
  if (GBrowserIsCompatible()) {
    map = new GMap2(document.getElementById("map"));
    map.setCenter(new GLatLng(41.427776, 2.0256945), 11);
    map.disableDragging();
    map.disableDoubleClickZoom();
    map.addControl(new GMapTypeControl());
    control = new GSmallZoomControl();
    map.addControl(control);
    map.addControl(new GScaleControl());
    zoom = map.getZoom();
    puntZona1 = new GLatLng(41.4631833369799, 2.0729827880859375);
    puntZona2 = new GLatLng(41.392392833843616, 2.0729827880859375);
    puntZona3 = new GLatLng(41.392392833843616, 1.9785690307617187);
    puntZona4 = new GLatLng(41.4631833369799, 1.9785690307617187);
    zona = new GPolyline([puntZona1, puntZona2, puntZona3, puntZona4,
puntZona1], "#FF0000", 6, 0.7);
    map.addOverlay(zona);

    GEvent.addListener(map, "zoomend", function (oldZoom, newZoom){
      if(newZoom < 11){
        zoom = oldZoom;
        map.setZoom(zoom);
        alert("No es permet un nivell de zoom inferior.");
      }
      if(newZoom > 16){
        zoom = oldZoom;
        map.setZoom(zoom);
        alert("No es permet un nivell de zoom superior.");
      }
      if(newZoom >= 11 && newZoom <=16)
        zoom = newZoom;
      if(zoom > 12){
        map.removeControl(control);
        control = new GSmallMapControl();
        map.addControl(control);
      }
      else {
        map.removeControl(control);
        control = new GSmallZoomControl();
        map.addControl(control);
      }
      if(zoom <= 12)
        map.setCenter(new GLatLng(41.427776, 2.0256945), zoom);
    });
  }
  . . .
}
```

Aquest codi es comenta a continuació.

Es pot observar com s'afegeixen el control de tipus de mapa amb *GMapTypeControl* i d'escala amb *GScaleControl*.

Quant al control de zoom, aquest ha de ser diferent si el nivell de zoom és inferior a 13 que si és igual o superior a 13. Pel primer cas es presentarà el control *GSmallZoomControl* consistent en només 2 botons per ampliar o reduir el zoom, mentre que pel segon cas es mostrarà el control *GSmallMapControl* que afegeix 4 botons per fer petits desplaçaments cap al Nord, Est, Sud o Oest, ja que aquests nivells de zoom mostren amb més detall la zona dels jaciments i han de permetre certa mobilitat. Aquesta mobilitat, però, no ha d'excedir els límits de la zona però aquest aspecte es tracta més endavant a l'apartat 5.3.4.

Per afegir el control de zoom caldrà, llavors, la variable *control* que a la funció *load()* li assignarà un objecte de control *GSmallZoomControl* ja que es carrega el mapa amb nivell de zoom 11. Aquesta variable s'utilitzarà després al mètode *addListener()* de l'objecte *GEvent* de l'API. En concret aquest mètode, amb el paràmetre *zoomend*, detectarà quan s'acabi de fer un canvi de zoom al mapa. En el moment que el nivell de zoom sigui superior a 12 es canviarà el control a *GSmallMapControl* i, si es tornés a un zoom inferior a 13 es tornaria a canviar a *GSmallZoomControl*.

El mètode *GEvent.addListener()* pel cas de detecció de *zoomend* aporta les variables *oldZoom* i *newZoom* que contenen el valor del nivell de zoom abans i després d'haver-se realitzat un canvi de zoom. Com que no es vol permetre nivells de zoom inferiors a 11 ni superiors a 16, s'empren aquestes variables per actualitzar la variable *zoom* i informar a l'usuari amb la funció *alert()* si aquest intenta excedir aquests nivells de zoom. A més, si el nivell de zoom és inferior a 13 el mapa es centra al punt central de la zona dels jaciments [Lat: 41.427776 - Lon: 2.0256945] que s'ha comentat al principi d'aquest apartat.

A la funció *load()* també es pot observar com s'impossibiliten la realització de zooms amb doble clic i la mobilitat arrossegant el ratolí sobre el mapa mitjançant els mètodes *disableDragging()* i *disableDoubleClickZoom()*, ja que per a un major control això es vol realitzar amb els controls comentats anteriorment.

Finalment, també es pot observar com s'emmarca la zona dels jaciments, mitjançant la creació d'un objecte *GPolyline* que s'assigna a la variable *zona*. Aquest objecte construeix una polilínea que forma un rectangle partint de les línies definides entre els punts de les cantonades de la zona definits a les variables de *puntZona1* a *puntZona4*. Les coordenades d'aquests punts es corresponen amb les cantonades del mapa quan aquest es mostra a un nivell 13 de zoom.

Els paràmetre “#FF0000”, 6 i 0.7 de la polilínea defineixen, respectivament, el seu color (vermell en codi **RGB**), el gruix en píxels i el valor de transparència que ha de ser entre 0 (totalment transparent) i 1 (totalment opac).

### 5.3.2. Mostrar el museu i els jaciments

Per mostrar o ocultar el museu s'implementa a la pàgina web **index.html** un formulari (fig. 5.9.) amb el següent codi.

Mostrar museu municipal:  Sí  No

Figura 5.9. – Formulari museu

```
<form name=museu>
<span style="font-weight: bold;">Mostrar museu municipal:</span>
&nbsp;&nbsp;&nbsp;<input name="mostrarmuseu" value="si" type="radio" onclick="ajax_function = 2;
noMostrarMuseuEnMapa(); mostrarMuseu();">S&iacutecute;
&nbsp;&nbsp;&nbsp;<input checked="checked" name="mostrarmuseu" value="no" type="radio"
onclick="noMostrarMuseuEnMapa();">No
</form>
```

El nom del formulari és *museu* i consta de dos botons de radio pertanyents al mateix bloc *mostrarmuseu* de manera que només un dels dos pot estar seleccionat al mateix temps. Per defecte, l'opció de valor *no* és la que està amb l'atribut *checked* activat de manera que és l'opció que es veu activada quan es carrega la pàgina web. Quan l'usuari faci clic a sobre de qualsevol dels botons es recollirà l'event *onclick* i s'executarà el codi associat en cada cas.

Quan es faci clic sobre l'opció  Sí es crida primer a la funció *noMostrarMuseuEnMapa()* que força l'eliminació del museu del mapa i després es crida a la funció *mostrarMuseu()* que s'encarrega d'obtenir les dades del museu i mostrar aquest al mapa amb les seves dades en una típica finestra d'informació de Google Maps (fig. 5.10.). El fet que es cridi primer a *noMostrarMuseuEnMapa()* és per evitar que la senyalització del museu es torni a dibuixar sobre ella mateixa en cas que l'usuari premi repetidament sobre l'opció.

Cal observar la missió de la variable *ajax\_function* que, tal i com es veurà més endavant, serveix com a indicador a la funció *theHTTPResponse()* d'*ajax\_functions.js* per saber quina resposta retornar. En aquest cas, quan es vulgui retornar les dades del museu, aquest valor ha de ser 2.

Quan es faci clic sobre l'opció  No es crida a la funció *noMostrarMuseuEnMapa()*.

A continuació es mostra la implementació de les funcions *noMostrarMuseuEnMapa()* i *mostrarMuseu()*.

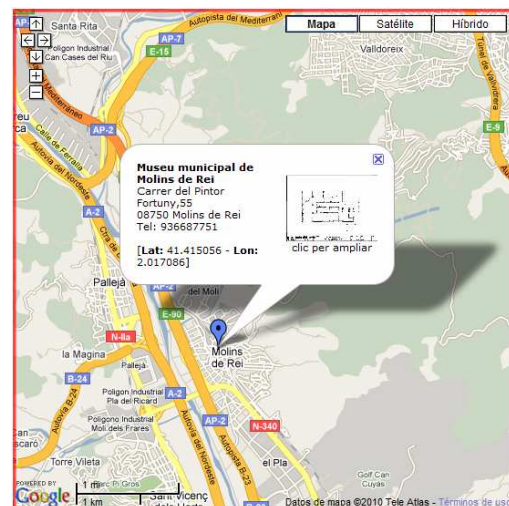


Figura 5.10. – Mostrar el museu i les seves dades

### 5.3.2.1. *noMostrarMuseuEnMapa()*

El seu codi s'afegeix a l'espai de l'script de les funcions de l'API de Google Maps dins **index.html** i consisteix en primer lloc en treure del mapa qualsevol finestra d'informació que hi pugui haver visible de qualsevol marcador amb el mètode *closeInfoWindow()* de l'API. Després s'empra el mètode *removeOverlay()* de l'API per eliminar qualsevol marcador del museu que s'hagués afegit abans al mapa. Com que es controla l'existència d'aquests marcadors afegint-los al vector *markersMuseu* quan es creen, ara és l'oportunitat d'eliminar-los d'aquest vector amb el mètode *pop()*.

```
function noMostrarMuseuEnMapa(){
    map.closeInfoWindow();
    for(i=markersMuseu.length-1; i>=0; i--){
        map.removeOverlay(markersMuseu[i]);
        markersMuseu.pop();
    }
}
```

### 5.3.2.2. *mostrarMuseu()*

Aquesta funció es troba dins **ajax\_functions.js** i l'objecte *myReq* invocarà l'execució del codi del fitxer **mostrarmuseu.php** que retorna les dades del museu mitjançant la cadena retornada per la funció *theHTTPResponse()*.

```
function mostrarMuseu() {
    myReq.open("GET", 'mostrarmuseu.php', true);
    myReq.onreadystatechange = theHTTPResponse;
    myReq.send(null);
}
```

El codi de **mostrarmuseu.php** retorna a *myReq* les dades del museu en format XML.

```
<?php
header('Content-Type: text/xml');
echo "<?xml version='1.0' ?><dadesMuseu>";
echo "<museu><nom>Museu municipal de Molins de Rei</nom>
    <ubicacio>Carrer del Pintor Fortuny,55</ubicacio>
    <municipi>08750 Molins de Rei</municipi><telefon>Tel: 936687751</telefon>
    <planol>planol</planol><latitud>41.415056</latitud><longitud>2.017086</longitud>
    </museu></dadesMuseu>";
?>
```

Dins les possibles respostes del servidor implementades a la funció *theHTTPResponse()* d'**ajax\_functions.js** s'observa que, tal i com s'ha comentat anteriorment, quan *ajax\_function* val 2 la resposta són les dades del museu. La resposta s'obté en format XML de manera que cal indicar en quina etiqueta de la resposta XML es trobarà cada valor mitjançant la funció *getElementByTagName()*. D'aquesta manera es passen les dades a la funció *mostrarMuseuEnMapa()*, implementada a **index.html** i encarregada de mostrar el museu al mapa mitjançant les seves dades de latitud i longitud.

```
if(ajax_function == 2) {
    mostrarMuseuEnMapa(myReq.responseXML.getElementsByTagName("nom")[0].childNodes[0].nodeValue,
        myReq.responseXML.getElementsByTagName("ubicacio")[0].childNodes[0].nodeValue,
        myReq.responseXML.getElementsByTagName("municipi")[0].childNodes[0].nodeValue,
        myReq.responseXML.getElementsByTagName("telefon")[0].childNodes[0].nodeValue,
        myReq.responseXML.getElementsByTagName("planol")[0].childNodes[0].nodeValue,
        myReq.responseXML.getElementsByTagName("latitud")[0].childNodes[0].nodeValue,
        myReq.responseXML.getElementsByTagName("longitud")[0].childNodes[0].nodeValue); }
}
```

Es pot observar a continuació com el codi de *mostrarMuseuEnMapa()*, que s'afegeix a l'espai de l'script de les funcions de l'API de Google Maps dins **index.html**, crea el punt amb les coordenades i afegeix al mapa una nova capa amb el seu marcador obrint una finestra d'informació amb les seves dades<sup>1</sup> amb el mètode de l'API *openInfoWindowHtml()*, eliminant prèviament qualsevol finestra d'informació que hi pugui haver visible amb *closeInfoWindow()*.

El marcador es crea cridant la funció *createMarkerMuseu()* que el crea amb una icona a partir de la imatge [http://www.google.com/intl/en\\_us/mapfiles/ms/micons/blue-dot.png](http://www.google.com/intl/en_us/mapfiles/ms/micons/blue-dot.png), de color blau i adequant les seves característiques de grandària, ombra, punt d'ancoratge al mapa i punt d'ancoratge de la finestra d'informació del marcador. Com que a la funció se li han passat les dades del museu aquestes s'afegeixen a la finestra d'informació del marcador que es mostrarà quan s'hi passi per sobre amb el ratolí, emprant l'API, mitjançant el mètode *addListener()* de l'objecte *GEvent*. S'observa que és aquí quan, un cop creat el marcador, aquest s'afegeix al vector *markersMuseu*.

```
var markersMuseu = new Array();

. . .

function createMarkerMuseu(point, nom, ubicacio, municipi, telefon, planol, latitud, longitud) {
    var baseIcon = new GIcon(G_DEFAULT_ICON);
    baseIcon.shadow = "http://www.google.com/mapfiles/shadow50.png";
    baseIcon.iconSize = new GSize(32, 34);
    baseIcon.shadowSize = new GSize(52, 34);
    baseIcon.iconAnchor = new GPoint(14, 34);
    baseIcon.infoWindowAnchor = new GPoint(52, 34);
    var blueIcon = new GIcon(baseIcon);
    blueIcon.image = "http://www.google.com/intl/en_us/mapfiles/ms/micons/blue-dot.png";
    markerOptions = { icon:blueIcon };
    var marker = new GMarker(point, markerOptions);
    GEvent.addListener(marker, "mouseover", function() {
        var myHtml = "<table width='270' border='0'><td><b>" + nom + "</b><br/>" +
        ubicacio + "</b><br/>" + municipi + "</b><br/>" + telefon + "<br/><br/>[<b>Lat: </b>" + latitud
        + " - <b>Lon: </b>" + longitud + "]"</td><td><a href=\"documents/" + planol + ".pdf\"
        target='_blank'><img src=\"imatges/" + planol + ".jpg\" border='0' width='100'
        heigh='50'></a><br/><center>clíc per ampliar</center></td></table>";
        map.openInfoWindowHtml(point, myHtml);
    });
    markersMuseu.push(marker);
    return marker;
}

function mostrarMuseuEnMapa(nom, ubicacio, municipi, telefon, planol, latitud, longitud){
    var point = new GLatLng(latitud,longitud);
    map.addOverlay(createMarkerMuseu(point, nom, ubicacio, municipi, telefon, planol,
    latitud, longitud));
    map.closeInfoWindow();
    var myHtml = "<table width='270' border='0'><td><b>" + nom + "</b><br/>" + ubicacio +
    "</b><br/>" + municipi + "</b><br/>" + telefon + "<br/><br/>[<b>Lat: </b>" + latitud + " -
    <b>Lon: </b>" + longitud + "]"</td><td><a href=\"documents/" + planol + ".pdf\"
    target='_blank'><img src=\"imatges/" + planol + ".jpg\" border='0' width='100'
    heigh='50'></a><br/><center>clíc per ampliar</center></td></table>";
    map.openInfoWindowHtml(point, myHtml);
}
```

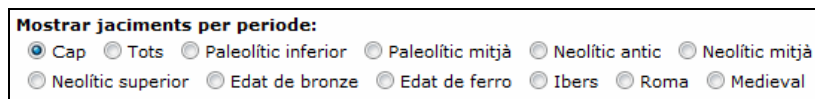
---

<sup>1</sup> Cal notar que dins aquestes dades que es preparen a la variable *myHtml* per presentar a la finestra d'informació s'inclou el codi HTML adient per a la seva presentació. Aquesta consisteix en una taula amb les dades de text a l'esquerra i la imatge del plànol a la dreta, la qual és al mateix temps enllaç cap a una nova finestra de navegador que permetrà veure'l en una grandària major en un document PDF.

Fins aquí s'ha vist el procediment per mostrar o no el museu. Pel cas dels jaciments és molt semblant tot i que amb petites diferències que es comenten al següent apartat ja que en aquest cas, a més, cal poder mostrar els jaciments segons el seu període.

### 5.3.3. Mostrar els jaciments per període

Per ocultar o mostrar tots els jaciments o només els jaciments segons el seu període s'implementa a **index.html** un formulari (fig. 5.11.) amb el següent codi.



Mostrar jaciments per període:  
 Cap  Tots  Paleolític inferior  Paleolític mitjà  Neolític antic  Neolític mitjà  
 Neolític superior  Edat de bronze  Edat de ferro  Ibers  Roma  Medieval

Figura 5.11. – Formulari *jacimentsxperiode*

```
<form name=jacimentsxperiode>
<span style="font-weight: bold;">Mostrar jaciments per període:</span> <br>
&nbsp;<input type="Radio" name="periode" value="Cap" checked="checked"
onclick="ajax_function = 1; noMostrarJacimentsEnMapa();
document.getElementById('dadesjaciment').innerHTML = '<b>Ara no hi ha cap període
seleccionat.</b>';">Cap
&nbsp;<input type="Radio" name="periode" value="Tots" onclick="ajax_function = 1;
l·listarJaciments(document.jacimentsxperiode.periode[1].value);">Tots
&nbsp;<input type="Radio" name="periode" value="Paleolític inferior"
onclick="ajax_function = 1;
l·listarJaciments(document.jacimentsxperiode.periode[2].value);"> Paleolític
inferior
. . .
&nbsp;<input type="Radio" name="periode" value="Medieval" onclick="ajax_function = 1;
l·listarJaciments(document.jacimentsxperiode.periode[10].value);"> Medieval
</form>
```

El nom del formulari és *jacimentsxperiode* i consta de botons de ràdio pertanyents al mateix bloc *periode*. L'opció de valor *cap* és la que està activada per defecte. Quan es selecciona aquesta opció es crida a *noMostrarJacimentsEnMapa()* i s'actualitza el contingut de la capa *dadesjaciment*. Aquesta capa és l'espai de la web on es mostrarà la informació del llistat dels jaciments quan aquests siguin visibles o bé s'indicarà que no s'estan mostrant quan aquests no ho siguin.

Es pot observar el codi de *noMostrarJacimentsEnMapa()* semblant al de *noMostrarMuseuEnMapa()* vist a l'anterior apartat, però eliminant aquí els possibles marcadors dels jaciments que hi hagi al vector *markersJaciments*.

```
function noMostrarJacimentsEnMapa(){
  map.closeInfoWindow();
  for(i=markersJaciments.length-1; i>=0; i--){
    map.removeOverlay(markersJaciments[i]);
    markersJaciments.pop();
  }
}
```

Si es fa clic sobre tots o qualsevol període s'assigna el valor 1 a *ajax\_function* perquè ara voldrem obtenir les dades dels jaciments de *theHTTPResponse()*, i es crida a la funció *l·listarJaciments()* per obtenir les dades d'aquests, mostrar-los al mapa i llistar les seves dades. Per tal de poder-ho fer pel període seleccionat cal passar-li el valor del botó de ràdio seleccionat amb *document.jacimentsxperiode.periode[i].value*.



Es mostra un exemple de com es veurien els jaciments i el seu llistat per al període concret de l'edat de ferro (fig. 5.12.), amb els marcadors al mapa i les dades llistades a la capa *dadesjaciment*.

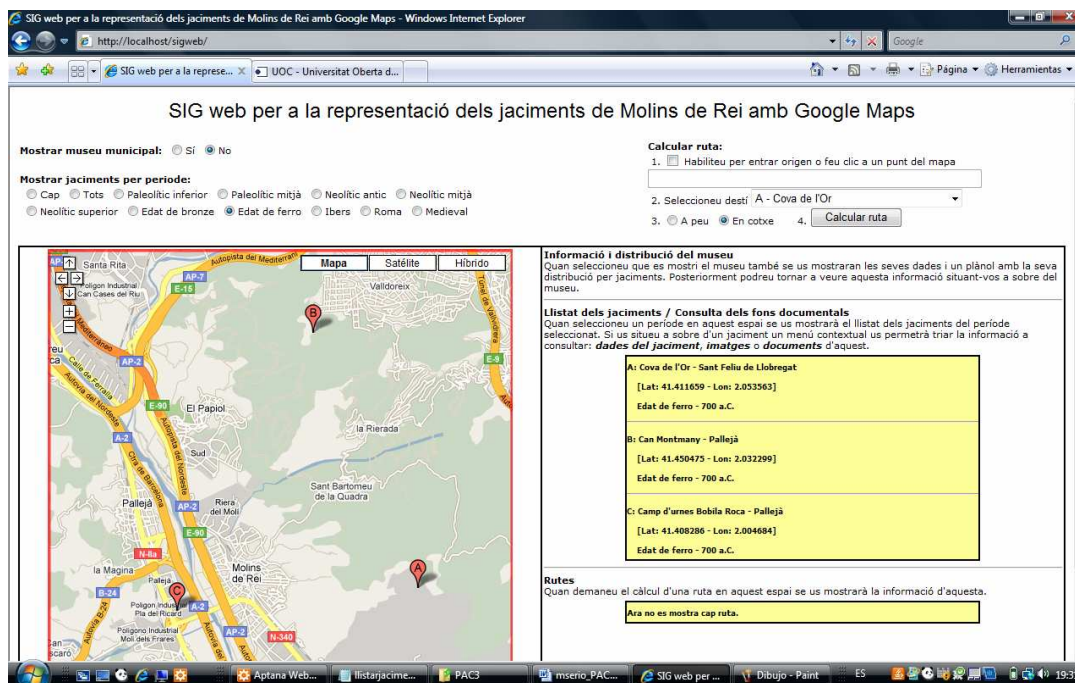


Figura 5.12. – Mostrar jaciments per període i les seves dades

A continuació es mostra la implementació de la funció *l·listarJaciments()*.

### 5.3.3.1. *l·listarJaciments()*

Aquesta funció d'*ajax\_functions.js* recull el valor del període del formulari anterior i l'objecte *myReq* invoca l'execució del codi del fitxer *l·listarjaciments.php* passant-li aquest període com a paràmetre.

```
function l·listarJaciments(període) {  
    var thePage = 'l·listarjaciments.php';  
    var theURL = thePage + "?període="+període;  
    myReq.open("GET", theURL, true);  
    myReq.onreadystatechange = theHTTPResponse;  
    myReq.send(null);  
}
```

El codi de *l·listarjaciments.php* inclou a *conexio.php* per connectar-se a la base de dades *sigweb* emprant la contrasenya *sigweb* del root i, un cop establerta la connexió fa la consulta MySQL per obtenir tots els jaciments del període i retornar a *myReq* les dades d'aquests en format XML.

Primer es mostra el codi de *conexio.php*.

```
<?  
$conexio = mysql_connect("localhost","root","sigweb");  
mysql_select_db("sigweb", $conexio);  
?>
```

I a continuació es mostra el codi de **l·listarjaciments.php** on es pot observar com s'obté el valor del període seleccionat a la variable `$periode` amb l'ordre `$_GET[]` que recull el paràmetre `periode`, així com les diferents consultes que cal fer a les taules de la base de dades per obtenir les dades dels jaciments.

Abans de mostrar el codi es comenten breument les accions a fer depenent de si s'ha seleccionat mostrar-los tots o només els d'un període concret:

- Si s'ha seleccionat mostrar els jaciments de tots els períodes:
  - Es seleccionen tots els camps de tots els jaciments de la taula **jaciments** i es passen a la consulta `$sql_jaciments`.
  - Per cada jaciment de la consulta `$sql_jaciments` es seleccionen de la taula **jacimentsxperiode** els identificadors dels períodes que estiguin relacionats amb l'identificador del jaciment i es passen a la consulta `$sql_jacimentsxperiode`.
  - Per cada identificador de període de la consulta `$sql_jacimentsxperiode` es seleccionen de la taula **periodes** tant el període com l'època corresponents i es passen a la consulta `$sql_periodes`.
  - Per cada parell d'identificadors de jaciment i període concrets es seleccionen de les taules **imatges** i **documents** els noms dels fitxers de les imatges i documents corresponents i la seva descripció, passant-los a les consultes `$sql_imatges` i `$sql_documents`.
  
- Si s'ha seleccionat mostrar només els jaciments d'un període concret:
  - Es seleccionen tots els camps del període de la taula **periodes** i es passen a la consulta `$sql_periodes`.
  - Pel període obtingut de la consulta `$sql_periodes` es seleccionen de la taula **jacimentsxperiode** els identificadors dels jaciments que estiguin relacionats amb l'identificador del període i es passen a la consulta `$sql_jacimentsxperiode`.
  - Per cada identificador de jaciment de la consulta `$sql_jacimentsxperiode` es seleccionen de la taula **jaciments** el nom, la ubicació i la latitud i longitud corresponents i es passen a la consulta `$sql_jaciments`.
  - Per cada parell d'identificadors de jaciment i període concrets es seleccionen de les taules **imatges** i **documents** els noms dels fitxers de les imatges i documents corresponents i la seva descripció, passant-los a les consultes `$sql_imatges` i `$sql_documents`.

Després d'aquests comentaris es mostra, a mode d'exemple, la part del codi de **l·listarjaciments.php** corresponent al segon cas degut a que el codi complet és molt llarg i la dinàmica és semblant però seguint les accions comentades pel primer cas.

Es pot observar també l'ús de la funció `parseToXML()` que s'empra per parsejar la resposta a donar, és a dir, reemplaçar alguns caràcters com accents, dièresis i *c* trencades que es puguin obtenir de la base de dades pel codi HTML que els representa i que serà l'adient a l'hora de mostrar la resposta a la capa `dadesjaciment`.

```
function parseToXML($htmlStr)
{
$xmlStr=str_replace('à','&agrave;',$htmlStr);
$xmlStr=str_replace('À','&Agrave;',$xmlStr);
. . .
$xmlStr=str_replace('ç','&ccedil;',$xmlStr);
$xmlStr=str_replace('Ç','&Ccedil;',$xmlStr);
return $xmlStr;
}

<?php
include ("conexio.php");

$periode = $_GET['periode'];

if($periode=="Tots"){
. . .
}
else {
    $sql_periodes = mysql_query("SELECT * FROM periodes WHERE periode='".$periode."'");
    $row_periodes = mysql_fetch_array($sql_periodes);
    mysql_free_result($sql_periodes);
    $sql_jacimentsxperiode = mysql_query("SELECT * FROM jacimentsxperiode WHERE
                                         id_periode='".$row_periodes['id_periode']."'");

    header('Content-Type: text/xml');
    echo "<?xml version='1.0' ?><llistatJaciments>";
    while($row_jacimentsxperiode = mysql_fetch_array($sql_jacimentsxperiode)) {
        $sql_jaciments = mysql_query("SELECT * FROM jaciments WHERE
                                     id_jaciment='".$row_jacimentsxperiode['id_jaciment']."'");
        $row_jaciments = mysql_fetch_array($sql_jaciments);
        mysql_free_result($sql_jaciments);
        echo "<jaciment><nom>". parseToXML($row_jaciments['nom'])."</nom>
              <ubicacio>". parseToXML($row_jaciments['ubicacio'])."</ubicacio>
              <latitud>". $row_jaciments['latitud']."</latitud><longitud>". $row_jaciments['longitud']."</longitud>
              <periode>". parseToXML($row_periodes['periode'])."</periode>
              <epoca>". parseToXML($row_periodes['epoca'])."</epoca><imatges>";

        $sql_imatges = mysql_query("SELECT * FROM imatges WHERE
                                   id_jaciment='".$row_jacimentsxperiode['id_jaciment']."' AND
                                   id_periode='".$row_periodes['id_periode']."'");
        while($row_imatges = mysql_fetch_array($sql_imatges)) {
            echo "<imatge>". parseToXML($row_imatges['imatge'])."</imatge>
                  <descripcio>". parseToXML($row_imatges['descripcio'])."</descripcio>";
        }
        mysql_free_result($sql_imatges);
        echo "</imatges><documents>";
        $sql_documents = mysql_query("SELECT * FROM documents WHERE
                                     id_jaciment='".$row_jacimentsxperiode['id_jaciment']."' AND
                                     id_periode='".$row_periodes['id_periode']."'");
        while($row_documents = mysql_fetch_array($sql_documents)) {
            echo "<document>". parseToXML($row_documents['document'])."</document>
                  <descripcio>". parseToXML($row_documents['descripcio'])."</descripcio>";
        }
        mysql_free_result($sql_documents);
        echo "</documents></jaciment>";
    }
    echo "</llistatJaciments>";
    mysql_free_result($sql_jacimentsxperiode);
}
?>
```

En ambdós casos, a mesura que es van obtenint les consultes s'extreu la informació adient de cada registre per poder retornar la resposta amb el següent format XML<sup>2</sup>.

---

<sup>2</sup> Cal notar que pel cas de mostrar tots els jaciments els elements <periode> i <epoca> inclouen tots els possibles períodes i èpoques de cada jaciment separats per comes.



```
var documents = myReq.responseXML.getElementsByTagName("documents")[i];
var documents_document = new Array();
var documents_descripcio = new Array();
for (j = 0; j < documents.getElementsByTagName("document").length; j++){
documents_document.push(documents.getElementsByTagName('document')[j].childNodes[0].nodeValue);
documents_descripcio.push(documents.getElementsByTagName('descripcio')[j].childNodes[0].nodeValue);
}

mostrarJacimentEnMapa(i+1,
myReq.responseXML.getElementsByTagName("nom")[i].childNodes[0].nodeValue,
myReq.responseXML.getElementsByTagName("ubicacio")[i].childNodes[0].nodeValue,
myReq.responseXML.getElementsByTagName("periode")[i].childNodes[0].nodeValue,
myReq.responseXML.getElementsByTagName("epoca")[i].childNodes[0].nodeValue,
myReq.responseXML.getElementsByTagName("latitud")[i].childNodes[0].nodeValue,
myReq.responseXML.getElementsByTagName("longitud")[i].childNodes[0].nodeValue,
imatges_imatge, imatges_descripcio, documents_document, documents_descripcio);
}
```

Es pot observar com el codi de *mostrarJacimentEnMapa()*, que s'afegeix a l'espai de l'script de les funcions de l'API de Google Maps dins **index.html**, crea el punt amb les coordenades del jaciment i afegeix al mapa una nova capa amb el seu marcador creat amb *createMarkerJaciment()*. El codi d'aquesta funció crea el marcador amb una icona per defecte de Google Maps, vermella, però amb una lletra per identificar-la a partir de la base <http://www.google.com/mapfiles/marker> segons el valor de la variable *letter* que depèn del número de jaciment (al 0 li correspon l'A, a l'1 li correspon la B, etc.) mitjançant el mètode *String.fromCharCode()*.

Com que a la funció se li han passat no només les dades del jaciment referents al seu nom, ubicació, període, època, latitud i longitud, sinó també els vectors d'imatges i documents, aquestes informacions del fons documental també s'afegeixen a la finestra d'informació del marcador<sup>3</sup>. Un cop creat aquest s'afegeix al vector *markersJaciments*.

```
var markersJaciments = new Array();
.
.
function createMarkerJaciment(point, i, nom, ubicacio, periode, epoca, latitud,
longitud, imatges_imatge, imatges_descripcio, documents_document, documents_descripcio)
{
    var tab2Content = "<p align=justify>";
    for(j=0; j<imatges_imatge.length; j++)
        tab2Content = tab2Content + "<a href=\"imatges/\" + imatges_imatge[j] +
\"\" target='_blank'>\" + (j+1) + \". \" + imatges_descripcio[j] + \"</a><br>\";
    tab2Content = tab2Content + "</p>";
    var tab3Content = "<p align=justify>";
    for(j=0; j<documents_document.length; j++)
        tab3Content = tab3Content + "<a href=\"documents/\" +
documents_document[j] + \"\" target='_blank'>\" + (j+1) + \". \" + documents_descripcio[j]
+ \"</a><br>\";
    tab3Content = tab3Content + "</p>";
    var tab1 = new GInfoWindowTab("Dades", "<div style='width: 270px;'><p
align=justify><b>Nom del jaciment: </b>\" + nom + \"<br><b>Ubicaci&oacute;: </b>\" +
ubicacio + \"<br>[<b>Lat: </b>\" + latitud + \" - <b>Lon: </b>\" + longitud +
\"]<br><b>Per&iacute;ode: </b>\" + periode + \"<br><b>&Egrave;poca: </b>\" + epoca +
\"</p></div>\";
}
```

<sup>3</sup> En aquest cas la finestra d'informació es crea mitjançant el mètode *openInfoWindowTabsHtml()* i es crea a partir de pestanyes d'informació de la classe *GInfoWindowTab*. D'aquesta manera es preparen tres pestanyes (variables *tab1*, *tab2* i *tab3*) dins la finestra: una per mostrar les dades del jaciment, una altra per presentar enllaços a les seves imatges i la tercera per presentar enllaços als seus documents.

```
var tab2 = new GInfoWindowTab("Fotos","<div style='width: 270px;'>" +
tab2Content + "</div>");
var tab3 = new GInfoWindowTab("Documents","<div style='width: 270px;'>" +
tab3Content + "</div>");
var baseIcon = new GIcon(G_DEFAULT_ICON);
var letter = String.fromCharCode("A".charCodeAt(0)+i-1);
var letteredIcon = new GIcon(baseIcon);
letteredIcon.image = "http://www.google.com/mapfiles/marker" + letter + ".png";
markerOptions = { icon:letteredIcon };
var marker = new GMarker(point, markerOptions);
marker.value = i;
GEvent.addListener(marker, "mouseover", function() {
    map.openInfoWindowTabsHtml(point, [tab1, tab2, tab3]);
});
markersJaciments.push(marker);
return marker;
}
function mostrarJacimentEnMapa(i, nom, ubicacio, periode, epoca, latitud, longitud,
imatges_imatge, imatges_descripcio, documents_document, documents_descripcio){
    var point = new GLatLng(latitud,longitud);
    map.addOverlay(createMarkerJaciment(point, i, nom, ubicacio, periode, epoca,
latitud, longitud, imatges_imatge, imatges_descripcio, documents_document,
documents_descripcio));
    map.closeInfoWindow();
}
}
```

### 5.3.4. No permetre la sortida dels límits municipals

A l'apartat 5.3.1 ja es va comentar com quan el zoom està a un nivell entre 13 i 18 es permeten moviments pel mapa però sense excedir els límits de la zona dels jaciments, per la qual cosa s'ha afegit a **index.html** el següent codi a l'espai de l'script de les funcions de l'API de Google Maps dins la funció *load()*. Ara es torna a emprar el mètode *addListener()* de l'objecte *GEvent* de l'API però amb el paràmetre *moveend* per detectar quan s'acabi de fer un moviment al mapa, moment en el qual es recull mitjançant les variables *bounds*, *southWest* i *northEast* les noves latituds i longituds extremes i, segons el nivell de zoom en curs, es comprova que no s'excedeixin d'uns certs límits observats experimentalment. En cas que hi hagi un excés s'informa a l'usuari i es torna a centrar el mapa amb el centre que tenia abans del moviment. Per a això es pot observar com a la variable *center* es recull la posició del centre del mapa tant quan es carrega aquest com després de finalitzar un moviment.

```
var center;
function load() {
    if (GBrowserIsCompatible()) {
        map = new GMap2(document.getElementById("map"));
        map.setCenter(new GLatLng(41.427776, 2.0256945), 11);
        . . .
        center = map.getCenter();
        . . .
        GEvent.addListener(map, "moveend", function () {
            var bounds = map.getBounds(); var southWest = bounds.getSouthWest();
            var northEast = bounds.getNorthEast();
            if (zoom==13)
                if (northEast.lat()>41.49854964120817 ||
northEast.lng()>2.1200180053710937 || southWest.lat()<41.35696864091227 ||
southWest.lng()<1.9311904907226562){
                alert("No es permet sortir de la zona dels jaciments del Baix Llobregat.");
                map.setCenter(new GLatLng(center.lat(), center.lng()), zoom);
            }
            . . .
            center = map.getCenter();
        });
    }
}
```

### 5.3.5. Implementar el menú contextual dels jaciments

A l'anterior punt 5.3.3.1. s'ha vist l'obtenció de les dades, imatges i documents dels jaciments, així com la seva utilització a *createMarkerJaciment()* per afegir-les per pestanyes a les finestres d'informació de cada marcadors de cada jaciment.

Igualment es podia observar en aquella funció com les finestres d'informació dels marcadors seran mostrades quan s'hi passi per sobre d'aquests amb el ratolí, emprant l'API, mitjançant el mètode *addListener()* de l'objecte *GEvent*.

Aquestes finestres amb pestanyes constitueixen el menú contextual que permet accedir a les dades, imatges i documents dels jaciments al passar-hi per sobre amb el ratolí. A continuació es mostra un exemple d'interacció amb aquest menú pel cas del jaciment de Quatre Camins del Neolític mitjà (fig. 5.13. i 5.14.).

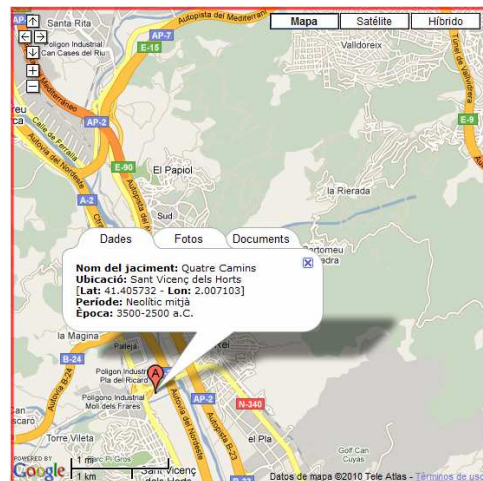


Figura 5.13. – Menú contextual dels jaciments: accés a dades

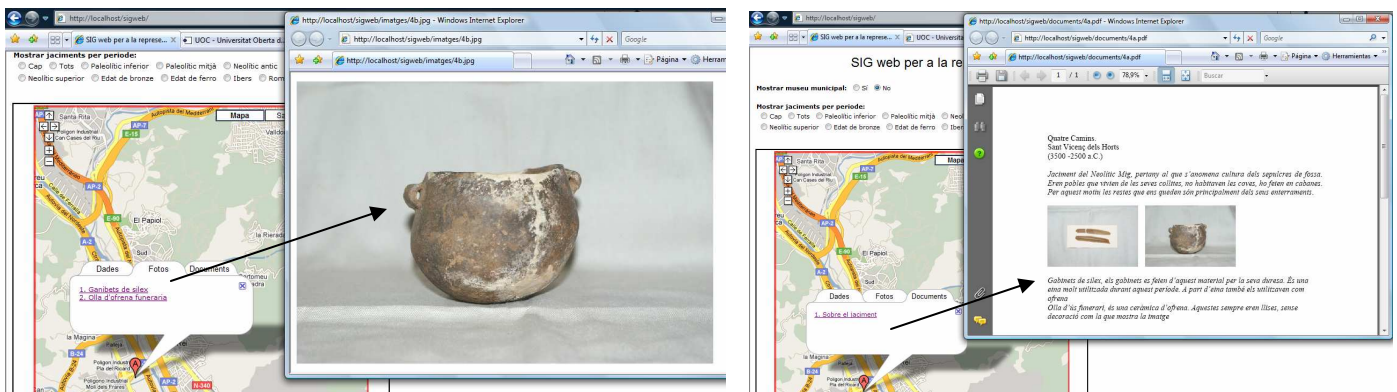


Figura 5.14. – Menú contextual dels jaciments: accés a imatges i documents

### 5.3.6. Mostrar el plànol del museu amb la distribució per jaciments

A l'anterior punt 5.3.2.2. en que s'explica la funció *mostrarMuseu()* i tots els procediments que es deriven, es va veure l'obtenció de les seves dades i la imatge del seu plànol, així com la utilització final d'aquestes a *createMarkerMuseu()* per afegir-les a la finestra d'informació del seu marcadors.

Ja llavors es va mostrar com la imatge de la finestra d'informació era també enllaç cap a una nova finestra de navegador on veure-la a major grandària dins un document PDF. Aquí es mostra un exemple d'interacció amb aquesta (fig. 5.15.).





- La llista de selecció *rutadesti* per a que l'usuari pugui seleccionar el punt d'arribada de la ruta d'un llistat amb els jaciments i/o museu que s'estiguin mostrant a cada moment.
- Dos botons de ràdio pertanyents al mateix bloc *apeuoencotxe* de manera que només un dels dos pot estar seleccionat al mateix temps. Per defecte, l'opció de valor *cotxe* és la que està amb l'atribut *checked* activat de manera que és l'opció que es veu activada quan es carrega la pàgina web. D'aquesta manera es podrà triar entre obtenir una ruta per anar a peu o un ruta per anar en cotxe.
- El botó *calcularruta* per indicar a l'aplicació que comenci a cercar una ruta entre el punt d'origen i el de destí o bé que elimini una ruta anterior.

A continuació es comenta el codi relacionat amb cadascun dels elements del formulari.

### 5.3.7.1. Casella de verificació - *habilitarOrigen()*

Quan l'usuari faci clic a sobre de la casella de verificació *habilitaorigen* s'executa la funció *habilitarOrigen()*. El seu codi es troba a l'espai de les funcions de Javascript dins *index.html* i es comenta a continuació.

```
function habilitarOrigen(){
    if (document.ruta.rutaorigen.disabled) {
        document.ruta.rutaorigen.disabled = false;
        document.ruta.rutaorigen.value = "";
        for(i=markersOrigen.length-1; i>=0; i--){
            map.removeOverlay(markersOrigen[i]);
            markersOrigen.pop();
        }
        gdir.clear();
        document.getElementById('dadesruta').innerHTML = '<b>Ara no es mostra
cap ruta.</b>';
        document.ruta.calcularruta.value='Calcular ruta';
    }
    else {
        document.ruta.rutaorigen.disabled = true;
        document.ruta.rutaorigen.value = "";
    }
}
```

Si al fer clic sobre la casella de verificació el quadre de text *rutaorigen* es troba deshabilitat llavors s'habilita i es buida per a que s'hi pugui escriure, esborrant del mapa i buidant del vector *markersOrigen* qualsevol marcador previ que pogués haver com a punt d'origen, la qual cosa podria passar si, per exemple, l'usuari hagués fet clic abans a sobre d'un punt al mapa. També s'elimina del mapa qualsevol ruta anterior que pugui haver mitjançant el mètode *clear()* de l'objecte *gdir* que es comentarà amb detall al punt 5.3.7.6, s'actualitza la capa *dadesruta* indicant que no s'està mostrant cap ruta i es restableix el valor del botó *calcularruta* per permetre el càlcul de la nova ruta que s'introdueixi.

En cas contrari es deshabilita i es deixa en blanc.

### 5.3.7.2. Quadre de text - *detectarTeclaEnter()*

Quan l'usuari escriu al quadre de text *rutaorigen* apareix el problema que quan es prem INTRO l'efecte és que es recarrega la pàgina i per tant es perd l'estat actual que aquesta pogués tenir. És per això que cal detectar quan es prem aquesta tecla per a que no faci res. Això s'aconsegueix cridant la funció *detectarTeclaEnter()* cada cop que es prem una tecla. El codi d'aquesta funció es troba a l'espai de les funcions de Javascript dins **index.html** i es comenta a continuació.

```
function detectarTeclaEnter(e) {  
    tecla = (document.all) ? e.keyCode : e.which;  
    return (tecla!=13);  
}
```

A la funció se li passa l'event que s'ha produït de prémer una tecla i detecta si el seu codi ASCII és 13, el codi de la tecla INTRO. Per a Internet Explorer es fa servir l'atribut *keyCode* de l'event i per a Firefox es fa servir l'atribut *which*. Si la tecla té el codi 13 llavors retornarà *fals* i no tindrà efecte la seva pulsació.

### 5.3.7.3. Llista de selecció

La llista de selecció *rutadesti*, que es mostra buida al carregar-se la pàgina, és on l'aplicació ha d'afegir els jaciments que l'usuari demani que es mostrin en cada moment i el museu<sup>4</sup> si també l'usuari ha demanat que aquest es mostri. Així doncs, cal augmentar el codi de les funcions *mostrarJacimentEnMapa()*, *noMostrarJacimentsEnMapa()*, *mostrarMuseuEnMapa()* i *noMostrarMuseuEnMapa()* que ja s'han vist en apartats anteriors, per tal que cada vegada que s'hagi d'afegir o treure un marcador del mapa també s'afegeixi o es tregui l'element que l'ha d'identificar a la llista. A continuació es mostra el codi afegit en aquestes funcions.

A *mostrarJacimentEnMapa()* es mira la longitud de la llista *rutadesti*. Si la seva longitud és 0 vol dir que el jaciment que s'acaba de mostrar al mapa és el primer que s'ha d'afegir a la llista i per tant s'hi afegeix creant un nou objecte *Option* amb l'índex<sup>5</sup> *i-1*. Però si la longitud de la llista ja és més gran que 0 llavors poden passar dues coses, o el primer element de la llista *rutadesti.options[0]* és el museu o ho és un altre jaciment, i això es té en compte a l'hora d'afegir el nou jaciment mitjançant l'índex adient *i* o *i-1*, respectivament, ja que si el museu ja és a la llista aquest ja estarà ocupant l'índex 0.

Cal notar que a l'hora d'inserir el nom d'un jaciment els caràcters amb accents, dièresi i *c* trencades cal passar-los a la codificació unicode adient que entén Javascript.

---

<sup>4</sup> El museu, en cas que es mostri al mapa quan ja s'hi estan mostrar jaciments, sempre s'afegeix a la llista com a 1r element. Això serveix per a un major control en la programació d'afegiments i eliminació d'elements de la llista tal i com es pot observar en els comentaris dels codis al llarg d'aquest subapartat.

<sup>5</sup> A *mostrarJacimentEnMapa()* se li passa el paràmetre *i* que val 1 per al primer jaciment, 2 per al segon, etc. Com que el 1r element de la llista ha de ser el 0, l'índex ha de ser *i-1*.

```
function mostrarJacimentEnMapa(i, nom, ubicacio, periode, epoca, latitud, longitud,
imatges_imatge, imatges_descripcio, documents_document, documents_descripcio){
    . . .

    nom = nom.replace("&agrave;", "\u00e0");
    . . .
    nom = nom.replace("&cedil;", "\u00c7");

    if(document.ruta.rutadesti.length>0)
        if(document.ruta.rutadesti.options[0].text=="Museu Municipal de Molins de
Rei")
            document.ruta.rutadesti[i] = new
Option(String.fromCharCode("A".charCodeAt(0)+i-1) + " - " + nom, i);
        else
            document.ruta.rutadesti[i-1] = new
Option(String.fromCharCode("A".charCodeAt(0)+i-1) + " - " + nom, i-1);
        else
            document.ruta.rutadesti[i-1] = new
Option(String.fromCharCode("A".charCodeAt(0)+i-1) + " - " + nom, i-1);
}
```

L'objectiu de *mostrarMuseuEnMapa()* és fer que el museu aparegui com a 1r element de la llista. Si la longitud d'aquesta és 0 s'hi afegeix a l'índex 0. En cas contrari es comprova que no estigui incorporat ja a la llista i, si no ho està, s'hi afegeix movent tots els jaciments que hi hagi entre l'índex 0 i l'índex *rutadesti.length-1* a les posicions entre l'índex 1 i *rutadesti.length*, amb ajut de l'array temporal *tmpArray* de manera que l'índex 0 quedi lliure per inserir el museu.

```
function mostrarMuseuEnMapa(nom, ubicacio, municipi, telefon, planol, latitud,
longitud){
    . . .

    if(document.ruta.rutadesti.length==0)
        document.ruta.rutadesti[document.ruta.rutadesti.length] = new
Option("Museu Municipal de Molins de Rei", document.ruta.rutadesti.length);
    else{
        if(document.ruta.rutadesti.options[0].text!="Museu Municipal de Molins de
Rei"){
            var tmpArray = new Array(document.ruta.rutadesti.length);
            for(i=0; i<document.ruta.rutadesti.length;i++)
                tmpArray[i] = new
Option(document.ruta.rutadesti.options[i].text, i);
            document.ruta.rutadesti[0] = new Option("Museu Municipal de
Molins de Rei", 0);
            for(i=document.ruta.rutadesti.length;i>=1;i--)
                document.ruta.rutadesti[i] = new Option(tmpArray[i-
1].text, i);
        }
    }
}
```

A *noMostrarJacimentsEnMapa()* s'eliminen tots els elements de la llista deixant només el primer si aquest conté el museu.

```
function noMostrarJacimentsEnMapa(){
    . . .

    if(document.ruta.rutadesti.length>0)
        if(document.ruta.rutadesti.options[0].text!="Museu Municipal de Molins de
Rei")
            for(i=document.ruta.rutadesti.length-1;i>=0;i--)
                document.ruta.rutadesti[i] = null;
        else
            for(i=document.ruta.rutadesti.length-1;i>=1;i--)
                document.ruta.rutadesti[i] = null;
}
```

A *noMostrarMuseuEnMapa()* s'elimina el primer element de la llista si aquest conté el museu.

```
function noMostrarMuseuEnMapa(){  
    . . .  
  
    if(document.ruta.rutadesti.length>0)  
        if(document.ruta.rutadesti.options[0].text=="Museu Municipal de Molins de  
Rei")  
            document.ruta.rutadesti[0] = null;  
}
```

#### 5.3.7.4. Botons de ràdio

El grup de botons de ràdio *apeuoencotxe* permet seleccionar l'opció de valor *peu* o l'opció de valor *cotxe* que emprarà la funció *calcularRuta()* per calcular una ruta que serà diferent si es tria anar a peu que si es tria anar amb cotxe.

#### 5.3.7.5. *calcularRuta()*

Quan l'usuari faci clic a sobre el botó *calcularruta* es cridarà a la funció *calcularRuta()* on es comprova si el quadre de text *rutaorigen* i/o la llista de selecció estan buits, la qual cosa es sap, respectivament, si el vector *markersOrigen* està buit i si la longitud de la llista *rutadesti* és 0, per alertar l'usuari d'aquest fet indicant que no es pot fer cap càlcul de ruta si no es tenen tant el punt de partida com el d'arribada i restablir adientment el valor del botó *calcularruta*.

En cas que la llista de selecció *rutadesti* no estigui buida es tenen en compte els següents casos:

- Si el vector *markersOrigen* està buit però no ho està el quadre de text *rutaorigen* serà perquè l'usuari hi haurà escrit el punt de partida. Llavors es comprova quin és l'element seleccionat de la llista *rutadesti* amb l'atribut *selectedIndex*:
  - És el museu albergat a *markersMuseu[0]* si l'element seleccionat és el primer (*selectedIndex* val 0) i el text de l'opció *rutadesti.options[0].text* indica que es tracta del museu.
  - És el jaciment albergat a *markersJaciments[...selectedIndex-1]* si l'element seleccionat no és el primer (*selectedIndex* no és 0) i el text de l'opció *rutadesti.options[0].text* indica que és el museu.
  - És el jaciment albergat a *markersJaciments[...selectedIndex]* si l'element seleccionat no és el primer (*selectedIndex* no és 0) i el text de l'opció *rutadesti.options[0].text* indica que no és el museu.
- Si el vector *markersOrigen* no està buit serà perquè l'usuari haurà fet abans clic en algun punt del mapa per indicar el punt de partida. Llavors es fan les mateixes comprovacions que abans respecte l'element seleccionat de la llista *rutadesti*.

Sigui com sigui, en qualsevol dels casos anteriors es tindrà el punt de partida i el d'arribada i només quedarà calcular la ruta entre aquests emprant l'API de Google Maps. La petició s'ha de fer sobre una instància de la classe *GDirections* que es comenta al següent punt 5.3.7.6. Com que aquesta instància es va crear amb el nom *gdir*, el càlcul de la ruta es demana amb el mètode *gdir.load()* passant-li com a paràmetres els punts d'origen i de destí adients segons el cas, així com el paràmetre *travelMode* que pot ser *G\_TRAVEL\_MODE\_WALKING* o *G\_TRAVEL\_MODE\_DRIVING*, depenent de si el botó de ràdio seleccionat del grup *apeuoencotxe* representa l'opció d'anar-hi a peu o en cotxe, respectivament.

A continuació es mostra el codi acabat de comentar corresponent al primer cas degut a que el codi complet és molt llarg i la dinàmica és semblant pel segon cas.

```
function calcularRuta(){
    if(document.ruta.rutadesti.length==0){
        alert("Per calcular una ruta heu d'entrar tant el punt d'origen com el
d'arribada"); document.ruta.calcularruta.value='Calcular ruta';
    }
    else
        if(markersOrigen.length==0 && document.ruta.rutaorigen.value==""){
            alert("Per calcular una ruta heu d'entrar tant el punt d'origen
com el d'arribada"); document.ruta.calcularruta.value='Calcular ruta';
        }
        if(markersOrigen.length==0 && document.ruta.rutaorigen.value!="")
            if(document.ruta.rutadesti.length>0)
                if(document.ruta.rutadesti.selectedIndex==0 &&
document.ruta.rutadesti.options[0].text=="Museu Municipal de Molins de Rei")
                    if(document.ruta.apeuoencotxe[0].checked)
                        gdir.load("from: " +
document.ruta.rutaorigen.value + " to: " + markersMuseu[0].getLatLng(), { "locale":
"es", "travelMode" : G_TRAVEL_MODE_WALKING });
                    else
                        gdir.load("from: " +
document.ruta.rutaorigen.value + " to: " + markersMuseu[0].getLatLng(), { "locale":
"es", "travelMode" : G_TRAVEL_MODE_DRIVING });
                    else
                        if(document.ruta.rutadesti.selectedIndex!=0 &&
document.ruta.rutadesti.options[0].text=="Museu Municipal de Molins de Rei")
                            if(document.ruta.apeuoencotxe[0].checked)
                                gdir.load("from: " +
document.ruta.rutaorigen.value + " to: " +
markersJaciments[document.ruta.rutadesti.selectedIndex-1].getLatLng(), { "locale":
"es", "travelMode" : G_TRAVEL_MODE_WALKING });
                            else
                                gdir.load("from: " +
document.ruta.rutaorigen.value + " to: " +
markersJaciments[document.ruta.rutadesti.selectedIndex-1].getLatLng(), { "locale":
"es", "travelMode" : G_TRAVEL_MODE_DRIVING });
                            else
                                if(document.ruta.apeuoencotxe[0].checked)
                                    gdir.load("from: " +
document.ruta.rutaorigen.value + " to: " +
markersJaciments[document.ruta.rutadesti.selectedIndex].getLatLng(), { "locale": "es",
"travelMode" : G_TRAVEL_MODE_WALKING });
                                else
                                    gdir.load("from: " +
document.ruta.rutaorigen.value + " to: " +
markersJaciments[document.ruta.rutadesti.selectedIndex].getLatLng(), { "locale": "es",
"travelMode" : G_TRAVEL_MODE_DRIVING });
                                if(markersOrigen.length>0)
                                    . . .
map.setZoom(zoom);
}
```

### 5.3.7.6. La classe *GDirections* de l'API de Google Maps

S'acaba de comentar al punt anterior la utilització de l'objecte *gdir* per al càlcul de les rutes, sent aquest objecte una instància de la classe *GDirections*. La creació d'aquest objecte, així com la dels mètodes *GEvent.addListener()* pels casos de detecció dels seus events de càrrega *load* i de producció d'errors *error*, s'aconsegueix mitjançant el següent codi afegit a *index.html* dins l'espai de l'script de les funcions de l'API de Google Maps dins la funció *load()*.

```
function load() {  
  if (GBrowserIsCompatible()) {  
    ...  
    gdir = new GDirections(map, document.getElementById("dadesruta"));  
    ...  
    GEvent.addListener(gdir, "load", function() {  
      document.getElementById("dadesruta").innerHTML =gdir.getSummaryHtml();  
    });  
    ...  
  }  
}
```

Per la creació de *gdir* es passa com a paràmetres l'objecte *map* contenidor del mapa i la capa *dadesruta* on es vol que es mostrin els resultats de les rutes. La crida al mètode *load()* de *gdir* per calcular una ruta ja s'ha comentat al punt anterior però ara, mitjançant la detecció d'aquest event, a més s'obtindrà un resum de la ruta amb el mètode *getSummaryHtml()* que mostrarà la seva distància i el temps aproximat per fer-la.

Es mostra un exemple (fig.5.17.) de ruta calculada en cotxe entre un punt del carrer de Sínies de Molins de Rei i el museu. Es pot observar tant l'efecte de la crida al mètode *load()* de *gdir* (la polilínia de la ruta al mapa i les seves dades a la capa *dadesruta*) encerclats en blau com el resum obtingut per *getSummaryHtml()* encerclat en vermell.

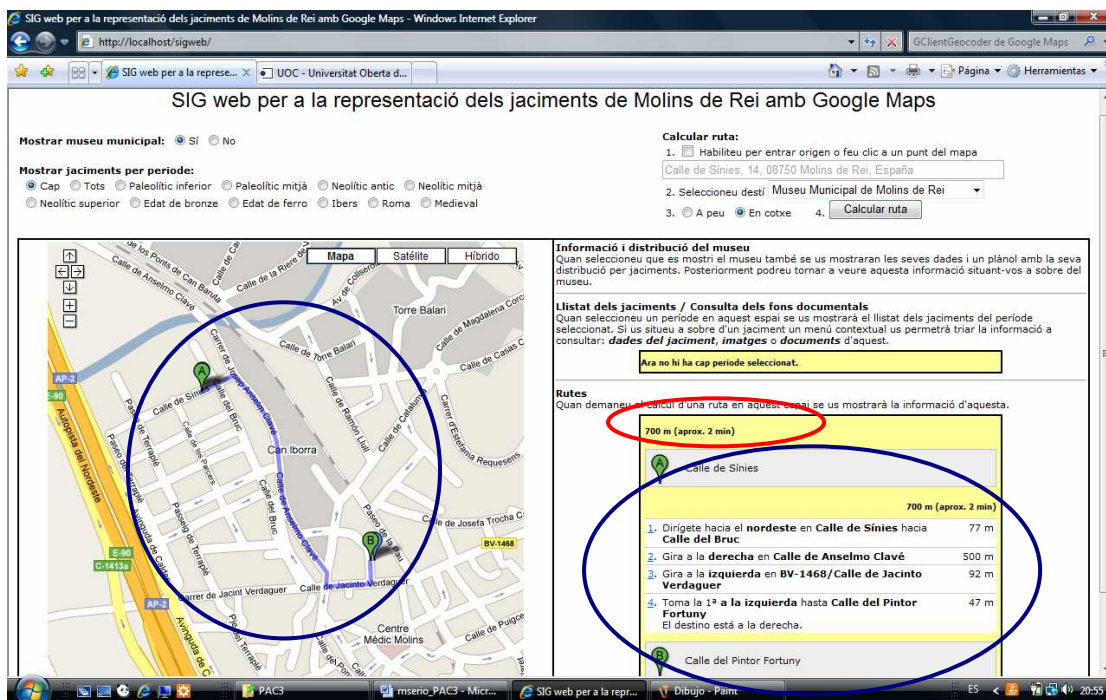


Figura 5.17. – Exemple de ruta

Respecte a la detecció dels errors, aquests poden ser alguns dels tipificats a l'API de Google Maps, els quals s'obtenen de l'atribut *getStatus().code* i es poden mostrar a l'usuari alertes adients indicant els motius dels errors (fig. 5.18.).

```
GEvent.addListener(gdir, "error", function(){  
    if (gdir.getStatus().code == G_GEO_UNKNOWN_ADDRESS)  
        alert("No s'ha trobat alguna de les adreces indicades. Poden ser  
massa recents o incorrectes.");  
    else if (gdir.getStatus().code == G_GEO_SERVER_ERROR)  
        alert("El servidor no ha pogut processar correctament la  
sol·licitud de ruta.");  
    ...  
    else alert("Error desconegut.");  
    document.getElementById("dadesruta").innerHTML = '<b>Ara no es mostra  
cap ruta.</b>'; document.ruta.calcularruta.value='Calcular ruta';  
});  
...  
}
```

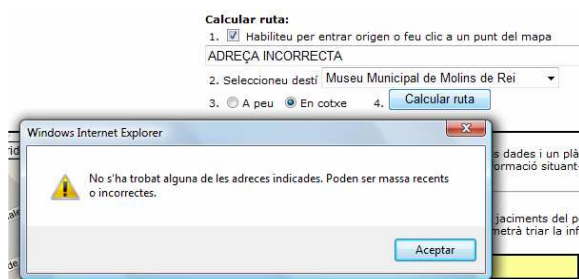


Figura 5.18. – Alerta d'error de ruta per origen incorrecte

### 5.3.7.7. Detecció del punt de partida quan l'usuari faci clic en un punt del mapa

Ja s'ha comentat anteriorment que l'usuari també ha de tenir la possibilitat d'entrar el punt d'origen d'una ruta fent clic al punt desitjat damunt el mapa. A tal efecte cal implementar el mètode *GEvent.addListener()* pel cas de detecció de l'event *click*. Es pot observar al següent codi com es crea un marcador amb la seva icona verda que es desa al vector *markersOrigen* i com es realitza la seva identificació geogràfica, és a dir, el pas de coordenades geogràfiques a una adreça típica<sup>6</sup> de Google Maps mitjançant la instància *geocoder* de la classe *GClientGocoder* i el seu mètode *getLocations()*.

Si la resposta *response* d'aquest mètode és correcta (valor 200 del seu atribut *Status.code*), s'obté d'aquesta resposta l'adreça que es passa al quadre de text *rutaorigen*, es deshabilita aquest perquè l'usuari no hi pugui escriure a sobre, es desmarca la casella de verificació *habilitarorigen* per si estava marcada, al marcador creat del punt se li afegeix l'escoltador de l'event *mouseover* per a que quan es passi a sobre amb el ratolí es mostri una alerta<sup>7</sup> tant amb la seva adreça com amb les seves coordenades geogràfiques, i aquesta mateixa alerta (fig. 5.19.) es mostra també en aquest moment en que s'acaba de crear el marcador.

<sup>6</sup> Una adreça típica de Google Maps és del tipus *carrer, número, codi postal i localitat, i país*. Exemple: *Calle del Pintor Fortuny, 55, 08750 Molins de Rei, España*.

<sup>7</sup> S'ha optat per emprar una alerta de Javascript en lloc d'una finestra d'informació de Google Maps per evitar alguns desplaçaments no desitjats del mapa que aquestes finestres poden produir de vegades.

```

function load() {
    if (GBrowserIsCompatible()) {
    ...
        GEvent.addListener(map, "click", function (overlay, point){
            var baseIcon = new GIcon(G_DEFAULT_ICON);
            ...
            var greenIcon = new GIcon(baseIcon);
            greenIcon.image = "http://www.google.com/intl/en_us/mapfiles/ms/micons/green-dot.png";
            markerOptions = { icon:greenIcon };
            if (point){
                geocoder = new GClientGeocoder();
                geocoder.getLocations(point, function(response){
                    if (!response || response.Status.code != 200) {
                        alert("No es pot obtenir l'adreça del punt marcat. " + response.Status.code);
                    } else {
                        var markerOrigen = new GMarker(point, markerOptions);
                        for(i=markersOrigen.length-1; i>=0; i--){
                            map.removeOverlay(markersOrigen[i]);
                            markersOrigen.pop();
                        }
                        map.addOverlay(markerOrigen); place = response.Placemark[0];
                        document.ruta.rutaorigen.value=place.address;
                        document.ruta.rutaorigen.disabled = true;
                        document.ruta.habilitarorigen.checked = false;
                        alert("Punt de partida: " + place.address + "\n[Lat, Lon]: " + point.y + ", " + point.x);
                        GEvent.addListener(markerOrigen, "mouseover", function() {
                            alert("Punt de partida: " + place.address + "\n[Lat, Lon]: " + point.y + ", " + point.x);});
                        markersOrigen.push(markerOrigen);
                        document.ruta.calcularruta.value='Calcular ruta';
                    } } });
            } } };
    ...
}
    
```

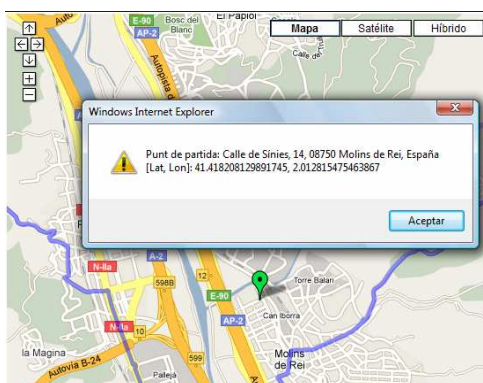


Figura 5.19. – Alerta del punt d'origen

### 5.3.7.8. eliminarRuta()

Quan es mostri una ruta al mapa el valor del botó *calcularruta* serà 'Eliminar ruta' i si es fa clic al botó en aquest moment es cridarà a la funció *eliminarRuta()* on es neteja la ruta del mapa, s'elimina qualsevol marcador de punt de partida i es restaura el formulari *ruta* i la capa *dadesruta*.

```

function eliminarRuta(){
    gdir.clear();
    for(i=markersOrigen.length-1; i>=0; i--){
        map.removeOverlay(markersOrigen[i]);
        markersOrigen.pop();
    }
    document.ruta.apueoencotxe[1].checked=true; document.ruta.habilitarorigen.checked=false;
    document.ruta.rutaorigen.value = ''; document.ruta.rutaorigen.disabled = true;
    document.getElementById('dadesruta').innerHTML = '<b>Ara no es mostra cap ruta.</b>';
}
    
```



## 6. Manual d'usuari per l'execució de l'aplicació

S'ha intentat fer un disseny de l'aplicació intuïtiu (fig. 6.1.) en el que tots els elements i funcionalitats siguin clarament visibles.

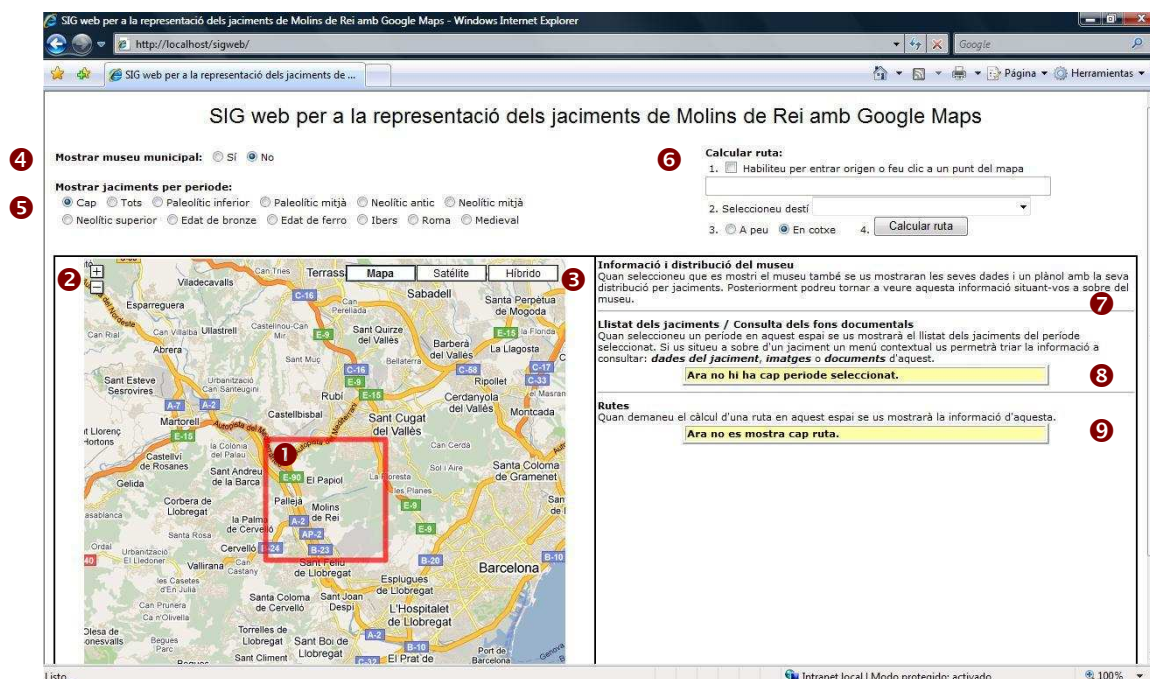


Figura 6.1. – Parts de l'aplicació

- 1 Zona dels jaciments.** Tant el museu com els jaciments a mostrar estaran ubicats en aquest espai emmarcat en vermell.
- 2 Zoom.** Amb aquest control podeu apropar-vos a la zona i, a partir d'un cert nivell de zoom, el control us permetrà també moure-us per dins d'aquesta (fig. 6.2.).
- 3 Tipus de mapa.** A més del mode **Mapa** podeu triar també **Satèl·lit** o **Híbrido** que combina la imatge satèl·lit amb la cartografia (fig. 6.2.).
- 4 Mostrar museu municipal.** Al fer clic sobre l'opció **Sí** apareix al mapa una icona blava indicant la ubicació del museu amb una finestra d'informació amb les dades d'aquest i una imatge que és un enllaç cap al detall del seu plànol amb la seva distribució (fig. 6.3.). Si tanqueu la finestra d'informació podreu obrir-la de nou situant-vos a sobre de la icona.
- 5 Mostrar jaciments per període.** Fent clic sobre l'opció d'un període apareixen al mapa les icones vermelles que ubiquen els jaciments del període al mapa, mostrant-se el seu llistat a la zona **3**. Passant el ratolí per sobre la icona vermella d'un jaciment s'obre la finestra d'informació d'aquest des de la qual hi podreu accedir a les seves dades, imatges i documents (fig. 6.4.).

- 6 **Calcula ruta.** Podeu calcular rutes entre qualsevol punt d'origen i qualsevol jaciment o el museu que s'estiguin mostrant al mapa. El punt d'origen el podeu introduir fent clic al punt a sobre del mapa o bé habilitant el quadre de text del formulari per entrar la seva adreça des del teclat. El museu o el jaciment de destí l'heu de seleccionar de la llista desplegable. La ruta es calcula per defecte en cotxe però també podeu triar el càlcul a peu. Al fer clic sobre el botó **Calcular ruta** apareix la ruta calculada al mapa (fig. 6.5.) entre les icones verdes d'origen i destí, mostrant-se la informació de detall de la ruta a la zona 9. Quan una ruta és visible el botó **Calcular ruta** passa a l'estat **Eliminar ruta** per si voleu eliminar-la.
- 7 8 9 **Zones d'informació.** Aquestes zones informen, respectivament, de com i on visualitzar el museu, els jaciments, les rutes i les seves informacions.

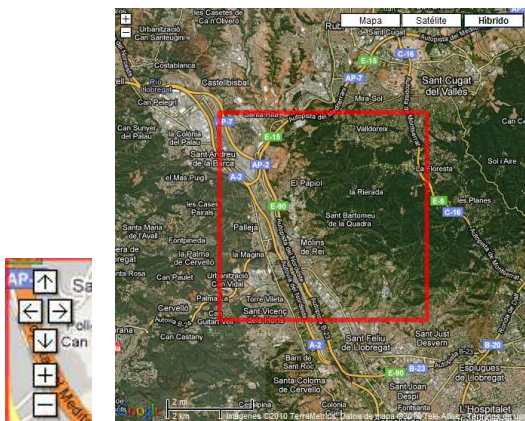


Figura 6.2. – Controls zoom i tipus de mapa

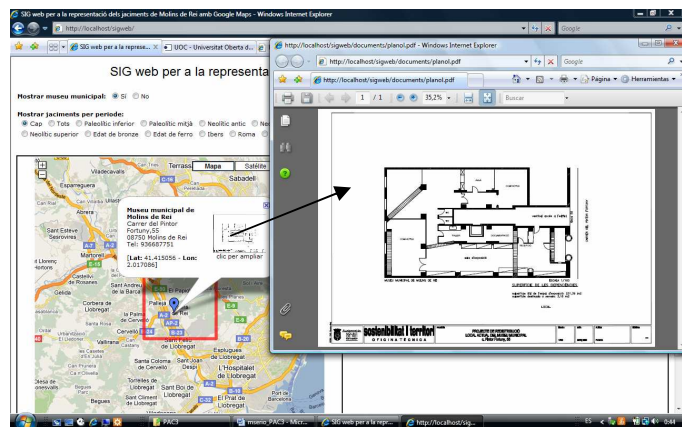


Figura 6.3. – Museu i accés al plànol

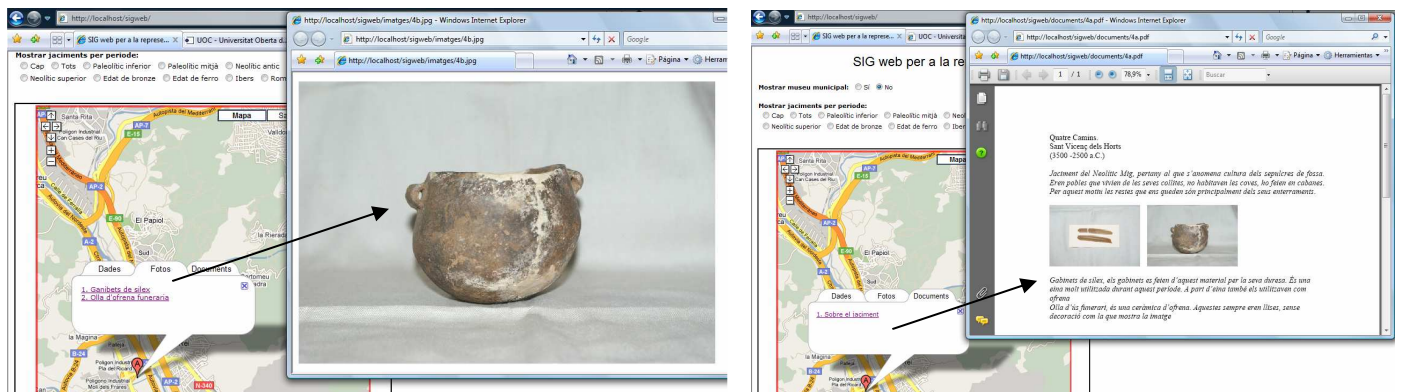


Figura 6.4. – Jaciments i accés a imatges i documents

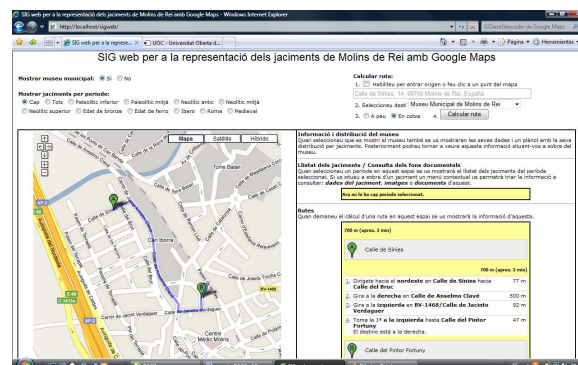


Figura 6.5. – Ruta amb la seva informació

## 7. Valoració econòmica

Es fa primer una avaluació dels recursos que han estat necessaris per la realització del treball per després fer una valoració del cost econòmic d'aquest.

### 7.1. Recursos necessaris

Els recursos personals han correspost exclusivament a l'alumne que ha desenvolupat l'aplicació, és a dir, no es tracta d'un treball realitzat per un equip, tot i que sí s'ha disposat de les orientacions necessàries del consultor del TFC així com del coneixement d'experts en entorns de treball SIG web mitjançant llibres, pàgines web, fòrums, etc.

En quant als recursos materials aquests han consistit en un maquinari propi de l'alumne amb les característiques habituals del punt de treball de la UOC (en aquest cas ha estat un HP Notebook PC Intel Celeron 550 a 2 GHz con 1 GB de RAM), amb sistema operatiu Windows Vista Home Basic i amb el programari lliure de desenvolupament Aptana, el servidor web Apache i un compte de Google Maps.

### 7.2. Valoració del cost econòmic

El cost econòmic seria zero si tenim en compte els recursos abans comentats i la no necessitat de llicències, a excepció d'alguna despesa que hagi pogut sorgir puntualment de l'adquisició d'alguna documentació o la consulta puntual d'algun expert.

No obstant això, sí que es podria fer un càlcul del cost que com a tècnic hagués de preparar l'entorn de treball i després implementar l'aplicació per encàrrec d'un tercer, en aquest cas, per exemple, de l'ajuntament de Molins de Rei. Aquí es consideraria un cost de 4032 € corresponents a uns honoraris de 14 € per hora tenint en compte la previsió que s'ha fet de 288 hores necessàries d'inversió en el treball. Aquest cost inclou una previsió de possibles desviacions, les quals no s'han patit en aquest cas, degudes a imprevistos, temps d'implantació de l'aplicació i aprenentatge dels usuaris, etc., que impliqui un promig de 4 hores diàries de dedicació de manera que l'estimació temporal podria haver arribat fins a les 340 hores (un coixí del 18% més del previst inicialment) sense cap cost addicional.

Al cost indicat caldria sumar 100 € més en documentació (adquisició de llibres) i el cost d'hostatge de l'aplicació en algun servidor però s'entén que aquest ja no l'ha de donar el desenvolupador sinó que se n'encarregaria el mateix client.

Així doncs, la valoració econòmica final seria:

Implementació de l'aplicació	4032 €
Adquisició de documentació	100 €
<b>TOTAL</b>	<b>4132€</b>

## 8. Conclusions

### Quant als objectius proposats

L'objectiu principal del treball ha estat plenament assolit. S'ha desenvolupat una aplicació web que permet mostrar sobre Google Maps tant el museu com els jaciments del terme municipal de Molins de Rei de manera que els usuaris poden accedir a les seves dades, documents, imatges i plànol del museu quan hi passen per sobre amb el ratolí, a més de calcular rutes entre qualsevol punt del mapa i el museu i els jaciments.

Cal destacar que no només s'han complert amb els requeriments funcionals sinó que, a més, s'ha tingut en compte un disseny de la interfície clar, diàfan i intuïtiu intentant seguir l'estil de senzillesa de les aplicacions de Google, i s'han establert certes millores en les funcionalitats com ara la possibilitat de zoom i el control d'aquest i de la mobilitat del mapa que permeten a l'usuari una bona localització referencial de la zona sense possibilitat de perdre-s'hi, la possibilitat d'entrar un punt d'origen d'una ruta bé per teclat o bé fent clic directament sobre un punt al mapa, la possibilitat d'eliminar rutes, el control dels seus possibles errors, calcular-les a peu o en cotxe, la possibilitat de canviar el tipus de mapa, etc.

El treball ha permès conèixer tant l'arquitectura com els components necessaris per a desenvolupar un SIG web emprant concretament els beneficis de la tecnologia de mashups amb l'API de Google Maps i la tecnologia AJAX que amb Javascript, PHP, MySQL i XML han possibilitat crear informació de diferents fonts d'informació.

A més, la planificació ha estat correcta i s'ha seguit sense problemes.

### Quant a la tecnologia emprada

L'aposta de Google per popularitzar les aplicacions SIG es veu clarament en l'API de Google Maps que es mostra com un servei fàcil d'emprar i molt potent al mateix temps, amb moltes funcionalitats que el fan avui dia el més estès a l'hora de crear mashups SIG convertint-se així en un estàndard de facto per al desenvolupament de SIG webs.

En aquest aspecte cal destacar no només la gratuïtat actual del servei i la gran quantitat de documentació que aporta, sinó també molt especialment la seva gran fiabilitat doncs s'han comprovat les coordenades geogràfiques de tots els jaciments, comparant-los amb l'Institut Cartogràfic de Catalunya tal i com s'ha vist en el seu procés de conversió comentat a l'apartat 3.1, i els resultats obtinguts s'han mostrat al major nivell de zoom sense errors de posicionament.

També cal fer esment de la idoneïtat de la tecnologia AJAX que ha permès una comunicació asíncrona amb el navegador i una resposta ràpida a les seves peticions.

### Quan a les possibles línies de continuació

Al punt 1.2.3. s'han comentat algunes possibilitats com l'accés remot a l'aplicació o ampliar funcionalitats: registre d'usuaris per editar la informació, optimitzar rutes, etc.

## 9. Glossari

**AJAX** (*Asynchronous Javascript And XML*): tecnologia que permet desenvolupar aplicacions web interactives que s'executen al navegador de l'usuari i es comuniquen de manera asíncrona amb el servidor de manera que es poden realitzar modificacions a les pàgines web sense necessitat de recarregar-les.

**Apache/Tomcat**: Apache és un servidor web de codi obert i multiplataforma. La seva arquitectura modular permet estendre les seves funcionalitats com, per exemple, fer ús del mòdul Tomcat per al suport de pàgines web dinàmiques en Java. Actualment Apache és el servidor web de la plataforma d'aplicacions XAMP (Servidor web multiplataforma **X** format pel servidor web **A**pache, gestor de base de dades **M**ySQL i intèrprets pels scripts dels llenguatges **P**HP i **P**erl).

**Aptana**: IDE o entorn de desenvolupament per aplicacions web escrites amb Javascript i Ajax.

**API** (*Application Programming Interface*): conjunt de funcions, procediments o mètodes que permeten la comunicació amb un programari de manera que es poden aprofitar les seves funcionalitats sense haver-les d'implementar des de zero.

**Datum geodèsic**: conjunt de punts de referència de la superfície terrestre que defineixen l'orientació d'un el·lipsoide (model matemàtic simple que millor s'aproxima a la forma de la Terra) concret en la superfície terrestre.

**ED50** (*European Datum 1950*): antic sistema de referència geodèsic europeu que emprava l'el·lipsoide internacional de Hayford de 1924. Aquest sistema s'ha substituït pel datum ETRS89 que a Espanya s'empra a partir del 2008.

**ETRS89** (*European Terrestrial Reference System 1989*): sistema de referència geodèsic europeu basat en l'el·lipsoide GRS80. Aquest sistema està lligat a la part estable de la placa continental europea i és consistent amb els sistemes de navegació actuals com ara GPS.

**IDE** (*Integrated Development Enviroment*): eina informàtica que presenta un entorn que facilita el desenvolupament de programari.

**Javascript**: llenguatge basat en objectes que es pot integrar amb el codi HTML per aconseguir pàgines web dinàmiques.

**Mashup**: aplicació web que integra components web i dades de més d'un origen.

**MySQL** (*Structured Query Language*): sistema gestor de bases de dades relacional multiusuari àmpliament utilitzat en aplicacions web.

**PDF** (*Portable Document Format*): format portable d'emmagatzemament de documents desenvolupat per Adobe Systems i que permet que els documents quedin definitius en quant la maquetació per impressió de manera que no es modifica l'aspecte en cap dels principals sistemes operatius (Windows, Unix/Linux o Mac).

**PHP** (*PHP Hypertext Pre-processor*): llenguatge de programació interpretat emprat per la creació de pàgines web dinàmiques.

**RGB** (*Red Green Blue*): model de color emprat en pàgines web i altres programes en que els colors es defineixen indicant una quantitat de vermell, una altra de verd i una altra de blau, mitjançant un codi del tipus #XXXXXX on cada parell d'XX és un valor hexadecimal per cadascun d'aquests 3 colors que va des de l'absència de color 00 al valor més intens d'aquest FF.

**SIG** (*Sistema d'Informació Geogràfica*): sistema d'informació per capturar i emmagatzemar dades referenciades geogràficament i realitzar tasques d'anàlisi, gestió i planificació de territoris.

**Unicode**: estàndard de codificació de caràcters desenvolupat per facilitar el tractament informàtic dels caràcters en múltiples llengües.

**UTM** (*Universal Transversal de Mercator*): sistema de coordenades universal transversal de Mercator, desenvolupat per l'exèrcit dels EEUU a la dècada del 1940 i és una variant de la projecció de Mercator realitzada per aquest geògraf flamenc el 1659. Es tracta d'una projecció cartogràfica cilíndrica conforme (conserva els angles) que transforma les coordenades geogràfiques de latitud i longitud en planes X (longitud) i Y (latitud) en metres. Aquest sistema no és exclusiu d'un datum concret sinó que es pot aplicar a qualsevol de manera que pot haver coordenades UTM ED50 i UTM ETRS89, per exemple. La projecció en l'eix X és composta, representant-se la esfera de la Terra en bocins, anomenats fusos, de 6° de longitud. Respecte a la projecció en l'eix Y, al tractar-se d'una projecció conforme, provoca una separació major a mesura que els paral·lels s'allunyen de l'equador de manera que només es representa la regió entre els paral·lels 84°N i 80°S. La península Ibèrica està ubicada entre els fusos 29 i 31 a l'hemisferi Nord. Per tant, un punt referenciat com UTM 31N (415.710, 4.590.029) vol dir que es troba a 415.710 m a l'est de l'origen del fus 31 i a 4.590.029 m al nord respecte l'equador.

**Web 2.0**: tecnologia web que permet als usuaris passar de ser consumidors d'informació a ser creadors de continguts de manera que la web es fa ara amb la participació i intercanvi de dades entre usuaris.

**XML** (*Extensible Markup Language*): estàndard d'intercanvi d'informació entre plataformes diferents mitjançant un document de marques estructurat. Una marca és una etiqueta que identifica i conté un bocí concret d'informació. El document XML estructura la informació amb una jerarquia d'etiquetes.

## 10. Bibliografia

### 10.1. Documents

**P. Ballard, M. Moncur.** *Programación: Ajax, JavaScript y PHP.* ANAYA Multimedia.

**E. Gutierrez.** *JavaScript: conceptos básicos y avanzados .* Cornellà de Llobregat: ENI.

**A. Pérez, A. Bataller, R. Beneito, N. Sáenz, R. Vidal** (2008). *Treball final de carrera.* Barcelona: FUOC.

**A. Pérez, A. Botella, A. Muñoz, R. Olivella, J.C. Olmedillas, J. Rodríguez** (2009). *Sistemas d'informació geogràfica i geotelemàtica.* Barcelona: FUOC.

**D. Sawyer.** *JavaScript.* ANAYA Multimedia.

**L. Welling, L. Thomson** (2009). *Desarrollo web con PHP y MySQL.* Madrid: ANAYA Multimedia.

Treballs de final de carrera publicats a la biblioteca de la UOC.

### 10.2. Enllaços a Internet

Calculadores geodèsiques:

<http://www.ign.es/ign/home/calculadora/UTM2LL.jsp>

<http://www.ign.es/ign/home/calculadora/ED2WGS.jsp>

<http://www.fcc.gov/mb/audio/bickel/DDDMSS-decimal.html>

Diccionari de l'Institut d'Estudis Catalans: <http://www.iec.cat>

Documentació bàsica d'aptana: <http://www.aclibre.org/archives/Aptana/Aptana.html>

Documentació del desenvolupador de l'API de Google Maps:

<http://code.google.com/intl/es/apis/maps/documentation/index.html>

<http://www.desarrolloweb.com/manuales/desarrollo-con-api-de-google-maps.html>

Documentació de Javascript:

<http://www.webestilo.com/javascript/>

[http://javascripts.astalaweb.com/Formularios%20V/1\\_Formularios%20V.asp](http://javascripts.astalaweb.com/Formularios%20V/1_Formularios%20V.asp)

Documentació del Servidor HTTP AppServ 2.5.10: <http://httpd.apache.org/docs/2.0/es/>

Exemple de mashup: <http://www.estrelateyarde.org/discover/mashup-google-maps-php>

Institut Cartogràfic de Catalunya: <http://www.icc.cat>

Museu de Molins de Rei: <http://www.molinsderei.cat/museu/>