Escola Universitària
Politècnica de Mataró

Enginyeria Tècnica en Informàtica de Gestió

FRAMEWORK WEB SERVICES MULTIPROTOCOL

MARC RIGAT LLAURADÓ
JOSEP ROURE ALCOBÉ

TARDOR 2010

A la meva dona, l'Anna, perquè em fa feliç.

Agraïments

Als meus pares, per fer-me com sóc, una part d'ells, i per l'educació que m'han donat.

Als meus germans perquè sempre m'han ajudat, aconsellat i divertit.

A tota la meva família per l'amor que m'han regalat.

Als meus amics per suportar-me, escoltar-me i compartir la seva vida amb mi.

Al meu amic i cunyat, En Pere, per l'ajuda inestimable que m'ha prestat en aquest projecte entre moltes altres coses.

Al meu tutor, En Josep Roure,pel seu recolzament des d'un bon principi.

Resum

Aquest projecte final de carrera es centra bàsicament en la creació d'un Framework, per tal de facilitar la creació i l'ús de serveis web. El destinatari final d'aquesta eina és un programador en llenguatge PHP.

El framework implementarà una sèrie de classes, amb els seus mètodes corresponents, que habilitaran al programador a crear un servei web multiprotocol d'una forma senzilla, clara i ràpida. Concretament, el servei web implementat oferirà la possibilitat de servir la informació desitjada amb els següents protocols: XML-RPC, SOAP, JSON-RPC i PHP Serialitzat-RPC. El Framework permetrà al programador incorporar dins del servei web creat, crides a serveis web externs de tercers amb els mateixos protocols abans esmentats. La implementació es realitzarà amb el llenguatge PHP 5.0 realitzant un ús intensiu de la programació orientada a objectes, facilitant l'escalabilitat del Framework per tal de que es puguin ampliar els protocols i formats suportats.

Resumen

Este proyecto final de carrera se centra básicamente en la creación de un Framework, con el objetivo de facilitar la creación y el uso de servicios web. El destinatario final de esta herramienta es un programador en lenguaje PHP.

El framework implementará una serie de clases con sus métodos correspondientes, que habilitaran al programador a crear un servicio web multiprotocolo de una forma sencilla, clara y rápida. Concretamente el servio web implementado ofrecerá la posibilidad de servir la información deseada con los siguientes protocolos: XML-RPC, SOAP, JSON-RPC i PHP Serializado-RPC. El Framework permitirá al programador incorporar dentro del servicio web creado, llamadas a servicios web externos de terceros usando los mismos protocolos anteriormente mencionados. La implementación se realizará en lenguaje PHP 5.0 realizando un uso intensivo de la programación orientada a objetos, facilitando la escalabilidad del Framework permitiendo ampliar los protocolos y formatos soportados.

Abstract

This End of Degree Project focuses on creating a Framework in order to facilitate the creation and use of web services. The final recipient of this tool is an PHP programmer.

The framework will implement a series of classes with their corresponding methods, which enables the developer to create a simple, quick and clear multi-protocol web service. Specifically, the web Services implemented offered the opportunity to serve the desired information with the following protocols: XML-RPC, SOAP, JSON-RPC and PHP Serialized-RPC. It also will enable the developer to incorporate into the web service created, external web service calls from third parties using the same protocols described above. The implementation will be implemented in PHP 5.0 making extensive use of object-oriented programming, facilitating the scalability of the framework allows to extend the protocols and formats supported.

ÍNDEX

Pàg.

ÍNDEX Figures

Pàg.

1. Introducció.

1.1 Raó i oportunitat del projecte

Durant cinc anys i escaig vaig treballar com a Analista-Programador en una PIME (petita i mitjana empresa) d'informàtica especialitzada en programació en entorn web. Principalment treballava per a altres empreses del món de la informàtica, subcontractada per d'altres habitualment més grans. En els darrers anys, l'empresa estava completament especialitzada en la programació web en entorn lliure, fonamentalment en entorn LAMP (sigles de Linux, Apache i MySql). La plantilla fixa era molt reduïda: un Cap de projectes, un Analista-Programador i dos Programadors, però habitualment s'integraven en l'equip dos o tres programadors externs.

Com a Analista-Programador vaig rebre l'encàrrec d'implementar un servei web per a un client que es dedicava al lloguer de vehicles.
L'empresa client volia oferir el lloguer dels seus vehicles a agències de viatges online. Amb aquest servei web, els clients de l'agència de viatges podrien llogar els vehicles en temps real.

Fins aleshores, la meva experiència no era molt àmplia en el món del serveis web, bàsicament es reduïa a connectar clients a serveis web mitjançant una API.
En el procés de desenvolupament de l'aplicació encarregada, em vaig adonar que hi havia moltes maneres diferents d'implementar els serveis web i que aquesta diversitat és la que s'utilitzava en la majoria de serveis webs comercials. Finalment es va decidir que l'aplicació web només utilitzaria Soap.
Un cop realitzada l'entrega i la validació satisfactòria de l'aplicació, el client va sol·licitar ampliació de les funcionalitats. L'aplicació, a més de gestionar el lloguer de vehicles en temps real, s'havia d'ampliar per tal de servir informació i imatges sobre rutes en tot terreny a l'illa Eivissa. Aquesta nova aplicació ampliada amb les noves funcionalitats mai es va poder realitzar, ja que malauradament, la meva empresa va cessar activitats l'any 2009. Tot i així, jo havia dedicat un munt d'hores en l'ampliació d'aquestes funcionalitats i el desenvolupament de l'aplicació estava força avançat. Per tal d'obtenir la informació

sol·licitada, l'aplicació havia de connectar amb els serveis web del Consell Insular d'Eivissa que oferia la informació sobre les rutes en tot terreny de l'illa. Aquesta aplicació també necessitava connectar amb el servei web de Panoramio, empresa espanyola adquirida per Google l'any 2007, per extreure imatges concretes de les rutes.
Em vaig adonar de la complexitat de les estructures de dades resultants obtingudes de les crides a serveis web de tercers i combinades amb les dades de la base de dades local de l'aplicació a desenvolupar.

Aquesta darrera aplicació, m'ha servit d'inspiració per escollir la temàtica del meu projecte final de carrera, els serveis web.

Tot i que existeixen multitud d'eines, frameworks o toolkits que faciliten les tasques pròpies de l'ús i creació de serveis web, no hi ha cap específica que sigui multiprotocol.

Amb motiu de realitzar el projecte final de carrera m'ha semblat força interessant rescatar la problemàtica que vaig tindre quan estava realitzant l'aplicació que no vaig concloure i plantejar la creació d'un framework per tal de facilitar la connexió i creació de serveis web multiprotocol.

1.2 Objectius

L'objectiu principal d'aquest projecte final de carrera és la creació d'un Framework per donar suport als programadors, concretament programadors de PHP, en tot el referent al món del serveis web.

La idea és que gracies a aquest framework, augmenti la productivitat dels programadors que el facin servir, obtenint uns resultats fiables i de qualitat.

Framework és un terme adoptat de l'anglès i equival a entorn de treball o, també, marc de treball. Defineix, en termes generals, un conjunt estàndard de conceptes, practiques i criteris per enfocar un tipus de problemàtica particular que, serveix com a referència per a enfrontar i resoldre nous problemes d'indole similar. [1]

Aquest Framework, en endavant WSFW(web service framework), permetrà als seus usuaris crear i consumir serveis web mitjançant un conjunt de protocols i formats establerts.

Per tal de decidir a quins protocols i formats concrets es dona suport, al capítol 2, es realitzarà un estudi de la situació actual en el sector dels serveis web.

Concretament, els objectius del framework són tres:

1) Crear serveis web. Les aplicacions creades amb WSFW, crearan serveis web per oferir a tercers. És a dir, es comportaran com a Servidors de serveis web, amb la intenció que altres aplicacions, de tercers, realitzin peticions(clients).

2) Consumir serveis web(client). Les aplicacions creades amb WSFW podran utilitzar informació proveïda per serveis web extern de tercers. Per un servei web extern de tercers, l'aplicació es podrà connectar amb un protocol concret dels suportats pel framework.

3) Donar suport multiprotocol. Els serveis web creats i consumits amb WSFW seran multiprotocol.

1.3 Definició i abast del projecte

WSFW, el framework, oferirà als programadors Php, totes les eines necessàries per crear i consumir serveis web amb SOAP, JSON-RPC,XML-RPC i PHP Serialitzat-RPC, protocols que posteriorment explicarem més endavant.

El framework estarà implementat en llenguatge Php, i estarà format per una sèrie de fitxers .php que contindran classes implementades, i algunes llibreries d'utilitats.

Tenint en compte els objectius d'aquest projecte, citats al apartat anterior, l'abast comprèn tot el cicle de vida del Serveis Web: creació del servei web, ús de serveis web i explotació d'aquests. Al capítol 3, més endavant, el podem veure detalladament.

2. Introducció als Serveis Web.

2.1 Introducció

Abans de realitzar la implementació del nostre framework,en endavant WSFW, explicarem breument de què estem parlant, des d' un punt de vista més general i en concret sobre les tecnologies que són considerades en aquest projecte final de carrera. Explicarem breument, les parts imprescindibles i necessàries per realitzar aquest projecte.

El concepte de servei web(web service) ha evolucionat des de la seva definició inicial. Els inicis dels serveis web els podem trobar en els sistemes distribuïts, amb Corba i Dcom, però cap dels dos sistemes va acabar generalitzant-se degut als problemes d'aplicació en la Internet real.

Més endavant la W3C predefiniria els Serveis Web amb el protocol SOAP, que forma part de la capa de missatgeria. Des de la seva creació s'ha estès ràpidament el seu funcionament i ha estat fonamental per teixir la web 2.0.

Actualment, els serveis SOAP destinats a aplicacions distribuïdes conviuen amb noves formes més simples i no estàndards degut al millor rendiment en vers la implementació de SOAP.

XML-RPC, JSON-RPC, PHP-RPC són els nous mètodes/protocols/formats més destacats que trobem als serveis web a les API de Facebook, Google i Yahoo.

Entre les diverses organitzacions que s'encarreguen de vetllar pels estàndards i arquitectura web destaquen la WWWC (W3C - World Wide Web Consortium) i OASIS (The Organization for the Advancement of Structured Information Standards).

2.2 Que són els serveis web?

La World Wide Web Consortium (W3C) ho defineix com "Un servei web és un sistema de programari dissenyat per a donar suport a la interacció interoperable màquina a màquina sobre una xarxa. Té una interfície descrita en un format processable per màquina, específicament WSDL. Altres sistemes interactuen amb el servei web d'una manera prescrita per la seva descripció usant missatges SOAP, típicament transmès a través d' HTTP amb una serialització XML en conjunció amb altres normes relacionades amb la web" [2].

Es pot definir més clarament com una forma estandarditzada d'integrar aplicacions Web mitjançant l'ús de XML, SOAP, WSDL i UDDI. Aquests permeten la comunicació entre aplicacions o components d'aquestes de forma estàndard mitjançant protocols comuns(típicament HTTP) i de forma totalment independent al llenguatge de programació, plataforma d'implantació, sistema operatiu o format de presentació. L'èxit de la interoperabilitat s'assoleix amb l'adopció de protocols i estàndards oberts.

En un intent d'estandardització global, la W3C proposa la seva visió dels Serveis Web, especificant la Web Services protocol stack – pila de protocols per a serveis web.
Tot i el pes i la importància de la W3C com a organisme d'estandardització en els serveis web, entre d'altres tecnologies, és ben cert que molts fabricants proposen petits canvis a aquesta especificació.
De totes maneres, no hi ha dubte que a grans trets, aquesta especificació de protocols i estàndards, fou i és, un marc de referència imprescindible per entendre i implementar els serveis Web. En el següent apartat, anem a fer una breu especificació d'aquesta especificació de la W3C.

2.3 Web Services Protocol Stack

És un conjunt de protocols i estàndards per a xarxes(Internet, intranet..) utilitzats per definir, localitzar, implementar i fer que un Servei Web interactuï amb altres.

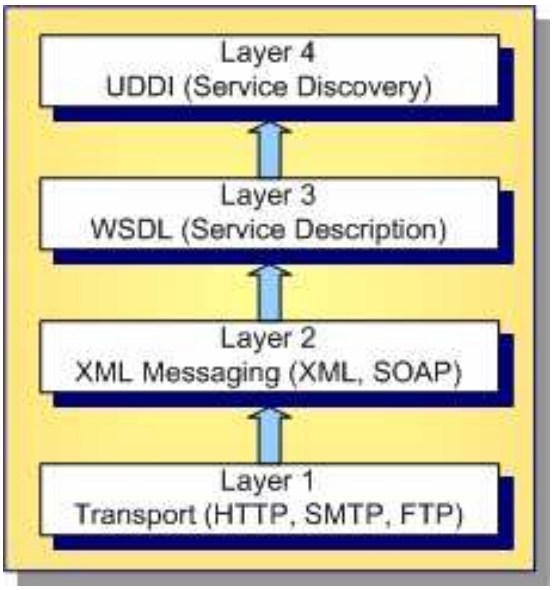

Figura 2. 1 Web Services Protocol Stack

2.3.1 Transport

La capa de transport és responsable del transport dels missatges entre les aplicacions de xarxa i els protocols com HTTP, HTTPS, SMTP, FTP, JABBER, IIOP, BEEP, etc.

A la pràctica, la gran majoria de Serveis Web estan disponibles per http i/o https.

2.3.2 Missatgeria XML

Llenguatge de marques extensible. És un estàndard per descriure dades i crear etiquetes. Les característiques especials són la independència de dades i la separació dels continguts de la seva representació. És un metallenguatge que permet dissenyar un llenguatge propi d' etiquetes per múltiples tipus de documents. Els documents XML estan formats per unitats d'emmagatzematge, anomenades entitats (entities), que contenen dades analitzades (parsed) o sense analitzar (unparsed). Les dades analitzades estan formades per caràcters. Alguns d'aquests caràcters formen les dades del document i la resta formen les etiquetes.

Les etiquetes codifiquen la descripció de l'estructura lògica i d'emmagatzematge del document. XML proporciona un mecanisme per imposar restriccions a l'estructura lògica i de emmagatzematge. XML es va convertir en l'estàndard de les comunicacions per Internet. El que abans es transmetia en formats propietaris ara és fàcilment interoperable gràcies a ell.

En l'especificació de la W3C només es contempla XML i SOAP com a capa de missatgeria(veure figura 2.2), en apartats següents ja explicarem una mica més detalladament XML i SOAP.

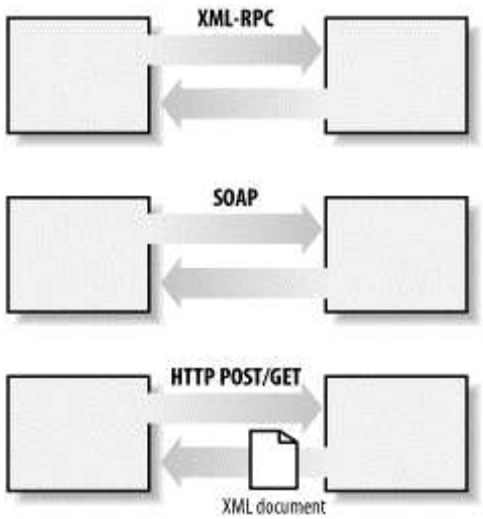

Figura 2 .2 Missatgeria XML

2.3.3 WSDL (Web Service Description Language)

Llenguatge de descripció de Serveis Web.
És una especificació XML per la construcció del document de descripció del Servei Web. Quan diem que és una especificació xml volem recalcar que el document wsdl està escrit en XML.

Identifica els mètodes, funcions i paràmetres necessaris per invocar un determinat Servei. El fitxer WSDL descriu informació crítica que el Client del Servei Web necessita conèixer, com:

- El nom del servei, incloent la seva URN(Uniform Resource Name).
- La localització del Servidor, normalment una adreça URL(Uniform Resource Location) per HTTP.
- Els mètodes disponibles per ser invocats.
- Els paràmetres d'entrada i sortida per a cada mètode.

2.3.4 UDDI (Universal Description Discovery and Integration)

Descripció, descobriment i Integració. UDDI és un element fonamental en el que es recolzen els Serveis Web. Ofereix la possibilitat de que les empreses/organitzacions puguin publicar i trobar altres Serveis Web de tercers. Proveeix d'un mecanisme per a que els "negocis" es descriguin a ells mateixos i els diversos serveis que proporcionen, permetent que es puguin registrar i publicar en un Registre UDDI.

Aquest serveis publicats poden ser cercats, consultats o descoberts per altres usant missatges SOAP.

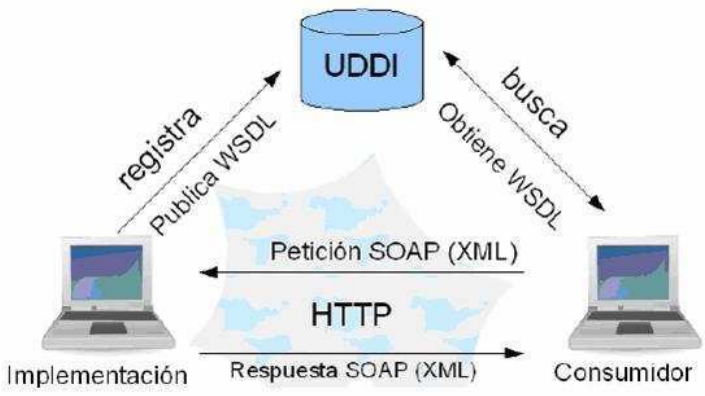

Figura 2 .3 Funcionament de UDDI

Les dades tractades per UDDI es classifiquen en tres categories:

- Pàgines Blanques: Contenen informació general de la Empresa/Organització com nom, descripció, informació de contacte, adreça i telèfon.

- Pàgines Grogues: És molt semblant al seu equivalent telefònic, inclouen categories de catalogació industrial, ubicació geogràfica, etc. Existeixen uns codis i claus preestablerts que faciliten la inscripció al registre i així es facilita a tercers la cerca de serveis mitjançant aquests codis de classificació.

- Pàgines Verdes: Contenen informació tècnica sobre un Servei Web.
  Normalment inclou un punter a la especificació externa i una adreça on invocar el Servei.

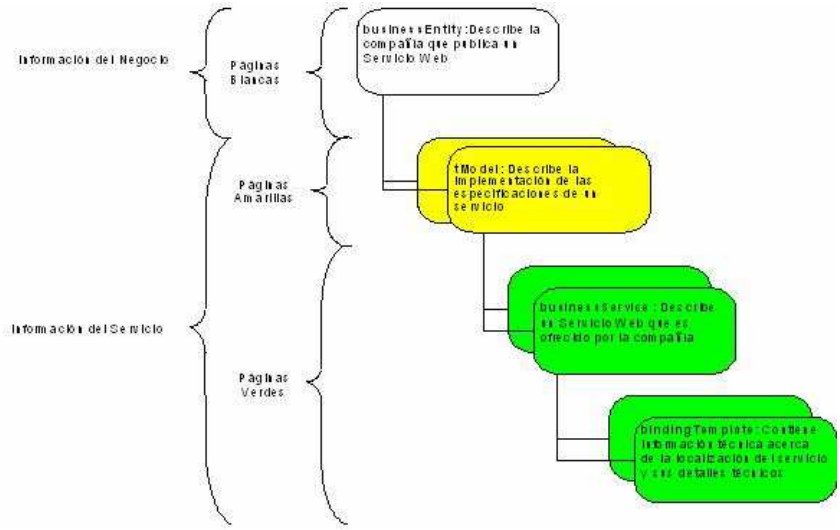

Figura 2. 4 Classificació dades registre UDDI

2.4 Protocols de la capa de missatgeria

Després d'estudiar el Mercat i comprovar els diferents formats i protocols usats en els Serveis Web actuals hem decidit donar suport als més estesos : XML, JSON, SOAP i PHP serialitzat. Formalment, estem parlant de XML-RPC, JSON-RPC, SOAP i PHP serialitzat-RPC.

RPC és un acrònim que respon a les sigles en anglès de Remote Procedure Call, en català, crida a un procediment remot. RPC és un protocol que permet que una aplicació executi codi en una màquina remota sense tenir cura de les comunicacions entre ambdues. Gràcies a això, el programador de l'aplicació només s'ha de centrar en la codificació de la mateixa aplicació, no té perquè diferenciar si el procediment s'executarà de forma remota o local. El programador ja no necessita implementar sockets i es pot oblidar de les comunicacions que ja estan encapsulades dintre de RPC.

2.4.1 SOAP (Simple Object Acces Protocol)

Protocol d' accés simple a objectes. És una especificació XML per a la creació dels missatges intercanviats entre sistemes distribuïts i la xarxa. Aquest protocol es deriva inicialment del XML-RPC. Els missatges estan formats per un sobre(ENVELOPE), la capçalera (HEADER) i el cos (BODY). El sobre embolica el missatge i conté la capçalera i el cos.

La capçalera és opcional i només dona informació per l' enrutament del missatge. Dintre del cos del missatge SOAP és on trobem les dades o un error(fault) etiquetades com XML.

El document XML de la figura 2.5 és un resposta SOAP; en color blau tenim el cos del missatge. En aquest missatge no hi ha capçalera.

```
<?xml version="1.0" encoding="ISO-8859-
1"?><SOAP-ENV:Envelope SOAPENV:
encodingStyle="http://schemas.xmlsoap.org/soap/
encoding/" xmlns:SOAPENV="
http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchemainstance"
xmlns:SOAPENC="http://schemas.xmlsoap.org/soap/encoding/">
<SOAP-ENV:Body><ns1:listar_poblaciones3Response
xmlns:ns1="http://tempuri.org"><return
xsi:type="SOAP-ENC:Array"
SOAPENC:
arrayType="unnamed_struct_use_soapval[2]">
<item><poblacion xsi:type="xsd:string">Dos primeras
files</poblacion></item><item><poblacion
xsi:type="xsd:string">Ultima
fila</poblacion></item></return></ns1:listar_poblacion
es3Response></SOAP-ENV:Body>
</SOAPENV:Envelope>
```

Figura 2. 5 Missatge SOAP

2.4.2 XML-RPC

Especificació del protocol creat per Userland Software i Microsoft l'any 1998.

Format XML

Com ja hem dit anteriorment, és un llenguatge d'etiquetes que proporciona una separació total entre el que són les dades pròpiament dites i el que serà la representació d'aquestes. Realment és un metallenguatge, ja que serveix per a definir-ne d'altres.

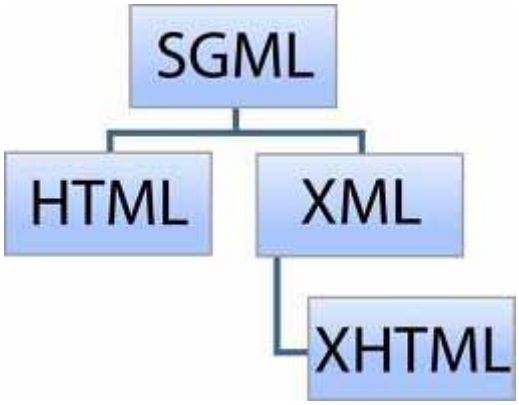

Figura 2.6 Jerarquia llenguatges d'etiquetes.

Està basat en un estàndard anterior, el SGML(Standard Generalized Markup Language) que data dels any 80'. És un subconjunt "petit" d'aquest. L'any 1998, la W3C ja el va postular com a recomanació.

Un fitxer XML s'escriu com un document pla de text, per exemple amb el notepad de Windows o amb el vi de linux. A la capçalera del document s'indica la versió de xml i la codificació de caràcters, opcionalment, també es pot indicar l'existència d'algun fitxer extern que definirà l'estructura de les dades del fitxer XML, per exemple **<?xml version="1.0" encoding="UTF-8" standalone="yes"?>**. A la resta del document s'escriuen etiquetes, niuades unes dintre d'altres <TAG1> ..<TAG2>..</TAG2>..</TAG1>.

Qualsevol etiqueta pot tenir tants atributs com es necessitin <TAG1 atribut1=”valor1” atribut2=”valor2”...>. Per tal de definir d’una manera ordenada les dades que conté un fitxer XML es pot utilitzar o bé DTD (Definition Type Document) o XML Schema.

El DTD és el vocabulari que defineix l’estructura i els elements d’un document XML. Aquesta definició es pot fer dintre del mateix document xml o bé es pot definir externament. Els Usuaris que utilitzin un document xml i vulguin conèixer l’estructura del document necessiten conèixer l’esquema o bé la definició del document. Veiem un exemple de DTD per aclarir-ho.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <!DOCTYPE ARTICLES [
4  <!ELEMENT ARTICLES(ARTICLE*)>
5  <!ELEMENT ARTICLE(ARTICLEDADES*)>
6  <!ELEMENT ARTICLEDADES(TITUL,AUTOR)>
7  <!ELEMENT TITUL(#PCDATA)>
8  <!ELEMENT AUTOR(#PCDATA)>
9  ]>
10 <ARTICLES>
11   <ARTICLE>
12     <ARTICLEDADES>
13       <TITUL>"Resum xml"</TITUL>
14       <AUTOR>"Marc Rigat"</AUTOR>
15     </ARTICLEDADES>
16   </ARTICLE>
17 </ARTICLES>
```

Figura 2.7 Fitxer articlesdtd.xml

A la figura 2.7, podem veure com a la línia 1 indica que es tracta d’un fitxer xml versió 1.0 i utilitza codi de caràcters UTF-8. Entre la línia 3 i 8 descriu l’estructura de les dades.

A les figures següents, podem veure com es podria haver separat el fitxer anterior, la figura 2.8 és el fitxer DTD anomenat articles.dtd i la figura 2.9 és el nou fitxer articles.xml.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <!ELEMENT ARTICLES (ARTICLE*)>
4 <!ELEMENT ARTICLE (ARTICLEDADES*)>
5 <!ELEMENT ARTICLEDADES (TITUL,AUTOR)>
6 <!ELEMENT TITUL (#PCDATA)>
7 <!ELEMENT AUTOR (#PCDATA)>
```

Figura 2.8 Fitxer articles.dtd

```
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <!DOCTYPE ARTICLES SYSTEM "http://localhost/articles.dtd">
4  <ARTICLES>
5    <ARTICLE>
6      <ARTICLEDADES>
7        <TITUL>Resum xml</TITUL>
8        <AUTOR>Marc Rigat</AUTOR>
9      </ARTICLEDADES>
10   </ARTICLE>
11 </ARTICLES>
```

Figura 2.9 Fitxer articles.xml

Podem observar a la línia 3 de la figura 2.9, com el fitxer xml fa referència a un fitxer extern .dtd i a la seva localització.

La principal avantatge de separar els DTD en fitxers separats del xml és que les definicions que contingui podran ser compartides per multitud de fitxer xml diferents.
Com ja hem comentat abans, una altra manera de descriure les dades i l'estructura d'un document xml és amb Schemas (esquemes).

Un Schema és quelcom molt semblant als DTD, defineix els elements d'un document xml, com estan estructurats, els atributs i els tipus de dades dels elements.

Les avantatges XML SCHEMA en front dels DTD, és que cobreixen algunes de les seves limitacions, principalment:

- ✓ XML SCHEMA està escrit en XML, és més fàcil per als usuaris la seva comprensió i a més, és un llenguatge extensible. DTD està escrit en un altre llenguatge anomenat ENBF que té una sintaxi molt diferent a la de Xml, força més complicada per als usuaris.

- ✓ Permet especificar una gran varietat de tipus de dades com enter, double, float, bolea, date, time,etc. DTD només permet especificar tipus de dades molt elementals, principalment text.

- ✓ XML SCHEMA permet l'herència, així podem reutilitzar i personalitzar a partir d'esquemes ja existents.

Per il·lustrar el concepte de XML SCHEMA veiem un exemple d'un fitxer d'esquema i d'un document xml que en faci ús.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
3      <xsd:element name="Llibre">
4          <xsd:complexType>
5              <xsd:sequence>
6                  <xsd:element name="Titol" type="xsd:string"/>
7                  <xsd:element name="Autors" type="xsd:string" maxOccurs="10"/>
8                  <xsd:element name="Editorial" type="xsd:string"/>
9              </xsd:sequence>
10             <xsd:attribute name="preu" type="xsd:double"/>
11         </xsd:complexType>
12     </xsd:element>
13 </xsd:schema>
```

Figura 2.10 Fitxer llibre.xsd

L'element Arrel es diu llibre i té tres fills i un atribut. Els fills són Títol, Editorial i Autors. Tant el Títol com l'editorial han d'aparèixer un sol cop, mentre que Autors pot aparèixer de 1 a 10 vegades. Els tres fill són de tipus string. El fet de que els fills estiguin agrupats en una seqüència implica que apareixeran en ordre, primer el Títol, després els Autors i finalment l'editorial. L'únic atribut de Llibre és el preu i és de tipus double.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Llibre xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3  xsi:noNamespaceSchemaLocation="http://localhost/llibre.xsd" preu="20">
4          <Titol>HTML: The Definitive Guide</Titol>
5          <Autors>Chuck Musciano</Autors>
6          <Autors>Bill Kennedy</Autors>
7          <Editorial>Wiley</Editorial>
8  </Llibre>
9
10
11
```

Figura 2.11 Fitxer llibre.xml

Podem observar com a la línia 3 del document xml de la figura 2.11, fa referència a la ubicació i a l'esquema que utilitza.

Ja per finalitzar aquesta breu explicació sobre XML, només comentar un dels temes que no hem explicat. Hem explicat què és un document xml, com està format, com es poden descriure les seves dades i estructures però en cap moment ens hem ocupat de la presentació final que tindrà la informació del fitxer xml. Per tal de tractar la representació final del contingut del fitxer Xml, es poden utilitzar fulles de estil CSS (Cascade Style Sheets) i/o XSL (extensible Stylesheet Language). Segons la W3C, XSL és "un llenguatge per transformar XML", afegint un vocabulari XML per poder especificar semàntica de formateig de documents.

2.4.3 JSON-RPC (JavaScript Object Notation)

Especificació del protocol creat per Google, el nostre framework compleix amb la versió 2.0 d'aquest protocol.
JSON-RPC simplement és una crida a un procediment remot molt semblant a XML-RPC amb la diferencia que usa el format lleuger de JSON en comptes de XML.

Format JSON

En català, Notació d'objectes JavaScript.

És un format lleuger per a l'intercanviï de dades. Està format per un subconjunt de la notació literal dels objectes de JavaScript que no necessita XML.
Un dels avantatges de JSON enfront XML és ,com a format d'intercanvi de dades, que és molt més senzill escriure un analitzador sintàctic de JSON. D'altre banda la incorporació del processament nadiu de XML dels navegadors actuals ha reduït aquesta avantatge. Per això JSON s'utilitza en entorns on el volum de dades a intercanviar entre el client i el servidor és molt alt (per això Google i Yahoo ja l'estan fent servir).
Tot i ésser freqüent posicionar JSON en front XML, és força habitual trobar-se una aplicació en que coexisteixin ambdós.

Està basat en dues estructures força habituals en qualsevol llenguatge de programació:

- ✓ Una col·lecció de parelles nom/valor
  En molts llenguatges de programació això s'implementa com un objecte, registre, estructura, diccionari, taula de hash, llista o array associatiu. El valor pot ser de qualsevol dels següents tipus : string, null, objecte, boolean, char, number i array.

- ✓ Una llista ordenada de valors
  En molts llenguatges de programació això s'implementa com un array, vector, llista o seqüència.

A la imatge de la figura 2.12, podem veure un exemple per fer-nos idea de la codificació.

```
1    [
2      {"place":"Dos primeres files"},
3      {"place":"Darrera fila"}
4    ]
```

Figura 2. 12 Format JSON

Entre les línies 1 i 4 està representat l'objecte ([.....]).

A les línies 2 i 3 observem una llista ordenada de parelles nom-valor ("place" : "Dos primeres files" i "place" : "Darrera fila"). Veiem que els elements de la llista estan separats per una coma, al final de la línia 2.

Veiem a la figura 2.13, la descodificació de l'exemple anterior, on podem entendreu molt millor la estructura del objecte original.

```
Array (
        [0] => Array ( [place] => Dos primeras files )
        [1] => Array ( [place] => Ultima fila )
    )
```

Figura 2.13 Representació JSON descodificat

2.4.4 PHP serialitzat-RPC

Malauradament no existeix cap especificació estàndard d'aquest protocol però l'estan començant a utilitzar alguns serveis web per l'alt rendiment que ofereix. En el nostre framework hem decidit com es codifiquen els mètodes i els paràmetres de les crides.

Format PHP serialitzat

Serialitzar és el procés de convertir un objecte en una cadena de text per poder utilitzar-lo fàcilment, podent revertir la codificació per obtenir l'objecte original. Típicament s'usa la serialització per transmetre a traves d'una xarxa l'objecte com una cadena de text. Aquesta codificació un cop rebuda, es pot utilitzar per crear un nou objecte idèntic en tot(és un clon) o en part, incloent el seu estat. Serveix per fer persistent un objecte en un arxiu físic o base de dades i distribuir objectes idèntics a varies aplicacions o localitzacions.

El nostre framework, s'implementarà amb el llenguatge de programació PHP que ja explicarem en un altre apartat. Ara només ens fixarem en la part referent a protocols, explicarem el format php serialitzat que nosaltres utilitzarem sobre RPC, és a dir, el protocol serà PHP serialitzat –RPC.

A partir de php 4.0 ja disposem de les funcions imprescindibles necessàries :

- string **serialize** (mixed $valor) per serialitzar.
- mixed **unserialize** (string $cadena) per deserialitzar (desfer la serialització).

Veiem un exemple per fer-nos una idea de la codificació.

```
1  a:2:{
2   i:0;a:1:{
3   s:5:"place";s:18:"Dos primeres files";
4              }
5   i1;a:1:{
6  s:5:"place";s:11:"Darrera fila";
```

Figura 2. 14 Codificació PHP serialitzat

A la línia 1 de la figura 2.14, podem veure que es tracta d'un array amb dos valors (a:2). La posició(índex) zero del array també és un array (i:0;a:1) que conté una parella de strings ("place" i "Dos primeres files") que tenen una llargada de 5 i 18 respectivament (s:5 i s:18).

La posició 1 del array (índex) també és un array (i:1;a:1) que conté una parella de strings ("place" i "Darrera fila") que tenen una llargada de 5 i 11 respectivament (s:5 i s:11).

Com hem explicat, és un array que conté dos arrays, per tant estem parlant d'un array associatiu.

Veiem a la figura 2.15, la deserialització de l'exemple anterior, on podem entendreu molt més fàcilment l'estructura del objecte original.

```
Array (
        [0] => Array ( [place] => Dos primeras files )
        [1] => Array ( [place] => Ultima fila )
      )
```

Figura 2. 15 Representació PHP deserialitzat

3 Definició del Framework (WSFW).

3.1 Els serveis web al Framework

Com a model important de referència tenim el de la W3C, nosaltres el seguirem però intentant fer servir només les parts comuns de tots els protocols que implementarem.

En aquest escenari, del model de la pila de protocols de la W3C necessitarem les tres primeres capes (transport, missatgeria i descripció del servei).

No ens preocuparem de la publicació, registre o descobriment de serveis de UDDI ja que els usuaris del serveis web ja tindran tota la informació necessària proporcionada per el proveïdor dels serveis.

El llenguatge de descripció del servei (Wsdl) només és utilitzat pel protocol SOAP mentre que els altres protocols, als que donarem suport, s'acostumen a descriure mitjançant API's (Application Programming Interface) o documentació similar proporcionada per el propietari del servei web.

API és un terme anglès, que podem traduir al català com Interfície de Programació d'Aplicacions. "És un conjunt de funcions i procediments (o mètodes, a la programació orientada a objectes) que ofereix certa llibreria per ser utilitzat per un altre programari com una capa d'abstracció." [3]

La creació de la documentació o API de l'aplicació creada amb WSFW, serà responsabilitat del programador del Servei Web.

Exemples d'API's

http://code.google.com/intl/es-ES/apis/maps/index.html

En aquesta adreça de Google podem trobar l'Api de Google Maps

http://developer.yahoo.com/

En aquesta adreça web de Yahoo trobarem un accés directe a totes les Api's dels diferents serveis webs oferts.

En el cas de crear un servei web ofert mitjançant el protocol SOAP amb el nostre framework, serà responsabilitat del programador generar el wsdl. Aquesta generació del fitxer wsdl queda fora de l'abast del nostre projecte. El programador pot utilitzar qualsevol eina de les que existeixen en el Mercat, entre elles, pot elegir diverses de lliures.

En el capítol d' Implementació, ja veurem com el programador indica a WSFW el fitxer wsdl del servei web extern a consumir, així com també veurem com indica el fitxer wsdl al WSFW perquè aquest pugui registrar-ho i servir-lo als clients finals que desitgin consumir un servei web SOAP creat amb el nostre framework.

3.2 Arquitectura Client-Servidor

Les aplicacions creades amb WSFW, representen a la perfecció el paradigma Client-Servidor.

A la figura 3.1, sota, podem veure un exemple. Es tracta d'una aplicació creada amb el nostre Framework, destinada a una empresa que desitja oferir un servei web als seus clients.

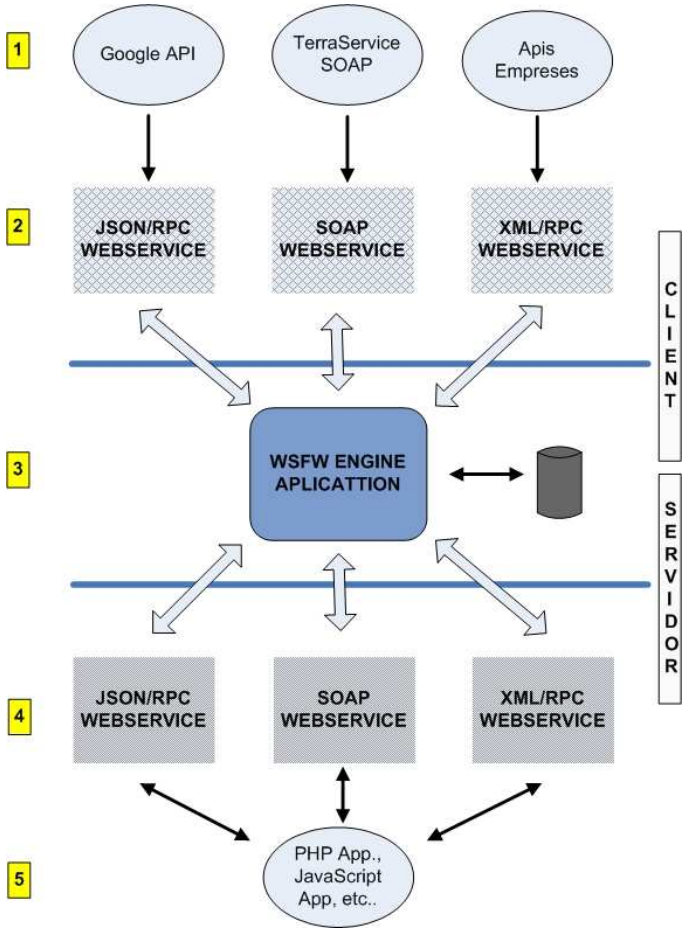

Figura 3.1 Cicle complet aplicació creada amb WSFW

A continuació, podem veure l'explicació de la figura 3.1, que mostra el cicle complet d'una aplicació creada amb el framework, concretament és una aplicació que necessitarà informació addicional d'un servei web extern, de tercers, i que crearà un servei web.

1

En aquesta part veiem empreses (privades o publiques) o organitzacions que ofereixen informació o utilitats mitjançant serveis web, cadascuna fent servir els protocols que ha considerat oportú.

2

Podem observar els serveis webs oferts i els protocols concrets en què es serveix la informació.

3

En aquesta part trobem ja l'aplicació creada amb el nostre framework. L'aplicació creada amb el nostre framework actua de Client respecte a la part superior (2), obtenint la informació desitjada. També interactuarà, en cas d' existir, amb la base de dades local pròpia de l'aplicació que hem creat. Normalment, es tractarà tant la informació de la base de dades local, com la informació obtinguda dels serveis web de tercers, servint finalment la informació consolidada que es desitja servir.
L'aplicació creada actua de servidor respecte a la part inferior (5).

4

Podem observar els serveis webs oferts per l'aplicació final i els protocols concrets en que es serveix la informació.

5

Representa la/les aplicacions que demanen informació al servei web que ofereix l'aplicació creada amb el framework.

Típicament serà un pàgina web, amb Php, Asp, JavaScript,etc, però podria ser qualsevol aplicació amb suport per fer crides a serveis web amb els protocols necessaris. Per exemple, una aplicació de escriptori implementada en C# de Microsoft.

Com hem explicat, l'aplicació creada amb WSFW, presenta una dualitat en el comportament, vegem més detingudament aquest doble comportament: Servidor de serveis web i Client de serveis web. Ambdós comportaments gaudeixen de ser multiprotocol.

a) Servidor de serveis web:

WSFW no només permetrà crear un servei web amb un dels protocols i formats que establirem, sinó que, oferirà els seus serveis mitjançant tots ells a la vegada. Alguns dels proveïdors de serveis web més populars com Yahoo o Google ja contemplen aquesta problemàtica i ofereixen els seus serveis web mitjançant diversos protocols. La figura 3.2, sota, il·lustra una aplicació WSFW multiprotocol actuant de servidor.

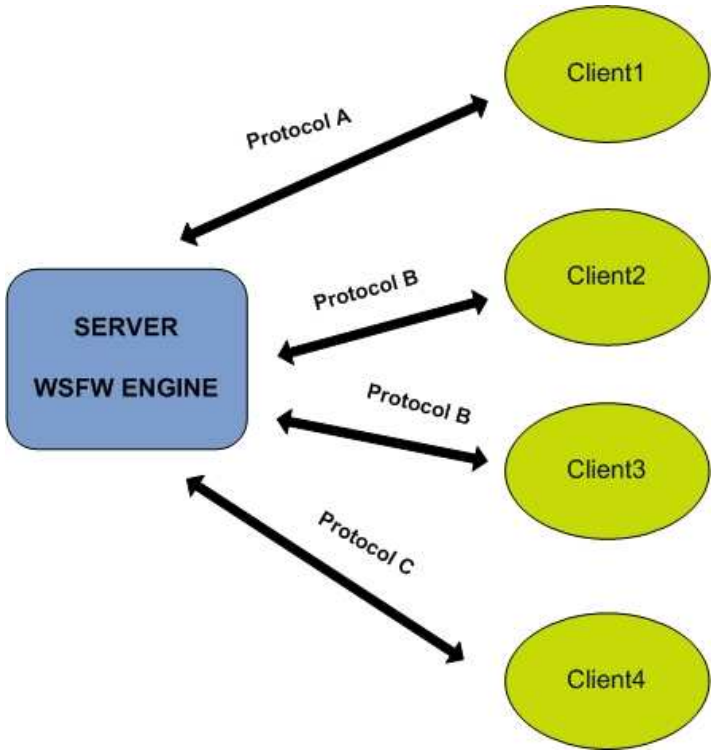

Figura 3.2 Aplicació WSFW - Servidor

A la figura 3.2, dalt, podem veure com un únic servei web de WSFW accepta peticions de quatre clients diferents amb tres protocols (A,B i C) que també són diferents.

Per exemple el client1 pot ser una aplicació web en java que es connecta amb el protocol A, el client2 pot ser una aplicació d'escriptori en VB.net que utilitzi el protocol B, el client3 és una aplicació web en ASP.NET que també utilitza el protocol B i finalment el client4 és una aplicació en PHP que es connecta amb el protocol C.

b) Client de serveis web:

WSFW permet consumir serveis web de tercers amb qualsevol dels protocols suportats pel framework. El client del servei web creat es connectarà a un servei web extern de tercers amb un protocol concret. Un client d'un servei web, tècnicament, només es pot connectar amb un servei web amb un únic protocol. A la figura 3.3, sota, es pot veure més clar.

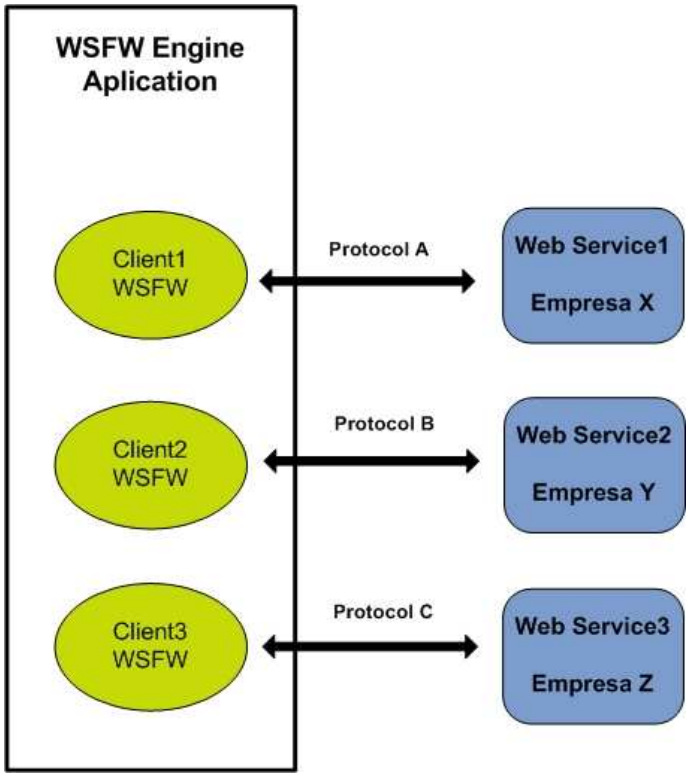

Figura 3.3 Aplicació WSFW - Client

A la figura,dalt, podem veure com una aplicació creada amb WSFW té tres clients diferents que es connecten amb tres serveis web externs de tercers mitjançant tres protocols diferents (A,B i C).
Per exemple el Client1 es connecta amb un servei web de Google utilitzant el protocol A, el Client2, consumeix un servei web de Yahoo connectant amb el protocol B i per últim, el Client3 es connecta a FriendFeed mitjançant el protocol C. [4]

Nota: FriendFeed és un servei web que permet compartir opinions amb els amics a Internet.

4. Anàlisi.

Per tal de desenvolupar qualsevol sistema software és imprescindible la realització d'un conjunt de tasques de forma paral·lela, seqüencial, incremental o una barreja de totes, que inclouen de forma molt resumida, l'Anàlisi, el Disseny i la Implementació.

Un requisit del software pot ser definit com:

- Una capacitat del software necessària per l'usuari per tal de resoldre o obtindré un objectiu.
- Una capacitat del software que ha de posseir un sistema o component del mateix per satisfer un contracte, especificació, estàndard o una altra documentació formal.

Dintre de l'anàlisi és part fonamental la definició dels requisits del sistema a implementar. Un tant per cent molt alt del fracàs en els projectes de software esdevé de la incorrecta elaboració d'aquests o com a resultats de que estan incomplerts.

Per tant, per assolir l'èxit del projecte és imprescindible ,des d'un bon principi, realitzar el document d'especificació de requisits del sistema. L'objectiu d'aquest document és definir i enumerar de forma inequívoca les característiques i funcionalitats del sistema a desenvolupar. Aquest document pot ser força útil com a part del contracte amb el client del projecte.

4.1 Requisits funcionals

Els requisits funcionals els obtenim de l'estudi de mercat d'altres frameworks, toolkits i de les converses mantingudes amb els clients objectiu del framework: programadors web amb eines d'entorn lliure.

1) El framework ha de permetre crear serveis web a l'usuari (programador) només utilitzant les eines que proporciona.

Aquest requisit vol mostrar que els usuaris del framework no necessitaran cap mena d'eina o mecanisme extern per implementar un web service.

2) El framework ha de permetre crear serveis web a l'usuari (programador) en diversos protocols(és multiprotocol).

Concretament, el framework, permetrà crear serveis web en :

- XML-RPC
- SOAP.
- JSON-RPC.
- PHP Serialitzat-RPC.

3) El framework ha de permetre crear serveis web a l'usuari (programador) que consumeixin d'altres serveis web externs.

És a dir, el servei web creat amb el framework podrà actuar com a client de serveis web externs (google, yahoo, etc). Concretament permetrà connectar amb qualsevol servei web que ofereixi serveis web mitjançant XML-RPC, SOAP, JSON-RPC o PHP Serialitzat-RPC.

4) El framework ha de permetre crear serveis web que opcionalment contemplin temes de seguretat com poden ser autentificació i xifrat dels missatges.

4.2 Requisits no funcionals

5) Requisits econòmics.

El nostre framework s'implementarà amb software lliure i la seva filosofia és aquesta, per tant l'usuari del framework, així com l'explotació de les aplicacions que sorgeixin de la seva utilització, poden utilitzar 100% software lliure, no propietari.

6) Usabilitat.

La usabilitat d'un software indica el nivell de facilitat amb que els usuaris poden utilitzar-ho. És un factor clau, perquè encara que l'usuari del framework és un programador, aquest no té perquè estar familiaritzat amb el món dels serveis webs i dels protocols emprats.

Per tant, el framework ha de ser molt fàcil d'entendre amb una corba d'aprenentatge exponencial i curta.

7) Escalabilitat.

El framework ha de ser fàcil de modificar i ampliar. Es poden afegir nous protocols i formats i modificar els existents.
A l'apartat 6.3.1 (Orientació a objectes) i al capítol 9 (Exemple pràctic del framework) s'il·lustra aquest requisit no funcional.

8) Independència del Sistema Operatiu.

El framework i les aplicacions resultants de la seva utilització han de poder córrer en qualsevol sistema operatiu (Windows, Mac OS,Linux, etc..).

9) Independència del Hardware.

El framework i les aplicacions resultants de la seva utilització han de poder funcionar en qualsevol plataforma hardware (Mainframe, HP, IBM, Sun Solaris, Intel x86, etc).

4.3 Restriccions del sistema

10) L'Aplicació resultant de l'ús del framework serà programada obligatòriament amb el llenguatge PHP 5.0 o superior.

11) És necessari que el servidor web que executi l'aplicació final desenvolupada amb el framework suporti PHP 5.0 o superior.

12) El servidor web que executi l'aplicació final on corrin les aplicacions desenvolupades amb el framework suportarà PHP 5.0 o superior. amb les extensions carregades que s'especificaran a l'apartat d'implementació.

4.4 Casos d'ús

4.4.1 Fonaments Casos d'ús

Els casos d'ús són uns dels diagrames més utilitzats del UML (Unified Modeling Language). Són força útils per capturar el comportaments d'un sistema o d'una part específica de ell. Un cas d'ús descriu la interacció dels actors com una seqüència de missatges entre el sistema i un o més actors. En l'etapa de captura de requisits solen ser molt útils per la comunicació entre els usuaris finals del sistema i el personal tècnic encarregat de dissenyar-lo i implementar-lo, que normalment utilitza un vocabulari massa tècnic allunyat del llenguatge dels usuaris finals.

A la figura 4.1, sota, podem veure els elements emprats en els diagrames de Casos d'ús.

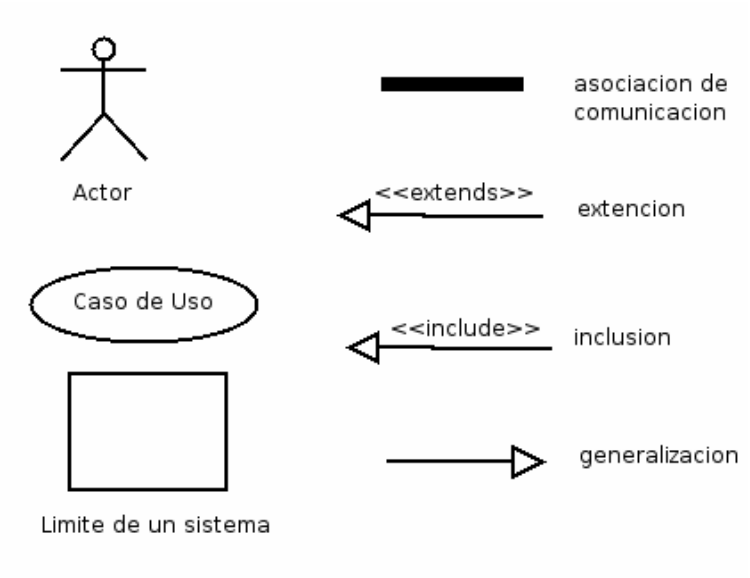

Figura 4.1 Notació Cas d' Ús.

Un Actor és una persona, organització o sistema extern que juga un paper en una o més interaccions amb el sistema.

Cada cas d'ús té un nom que en poques paraules descriu la funcionalitat requerida, es representa amb un oval amb el nom centrat dintre.

Les relacions vistes a la figura 19 són:

"**<<Comunicates>>**" : indica la participació del Actor en un determinat cas d'ús.

"**<<Extends>>"** : indica que un cas d'ús és una especialització de l'altre. La fletxa oberta surt del cas d'ús especialitzat i apunta al cas d'ús bàsic.

"**<<Include>>**" : en termes senzills, quan relacionem dos casos d'ús, el primer (bàsic) inclou el segons. Això vol dir que el segon és part essencial del primer. El cas d'ús basic no podria complir la seva funcionalitat sense el segon. La fletxa oberta surt de la cas d'ús bàsic i apunta al cas d'ús inclòs.

"**Generalization**" : indica l'existència de la herència entre casos d'ús o entre actors. A l'exemple de la figura 4.2 podem veure els dos tipus.

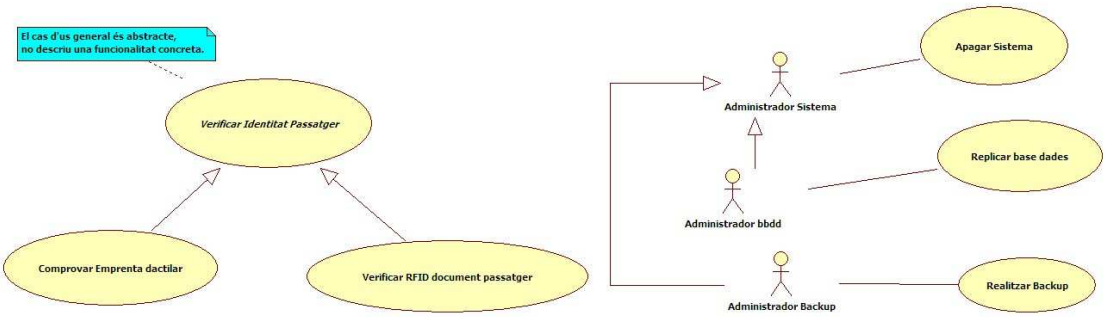

Figura 4.2 Generalització.

4.4.2 Principals casos d'ús del Framework

A la imatge de sota (figura 4.3), podem observar els casos d'ús fonamentals del nostre Framework, WSFW(Web Service Framework). Aquest és merament un exemple ja que aquests casos d'ús dependran del que implementi el programador.

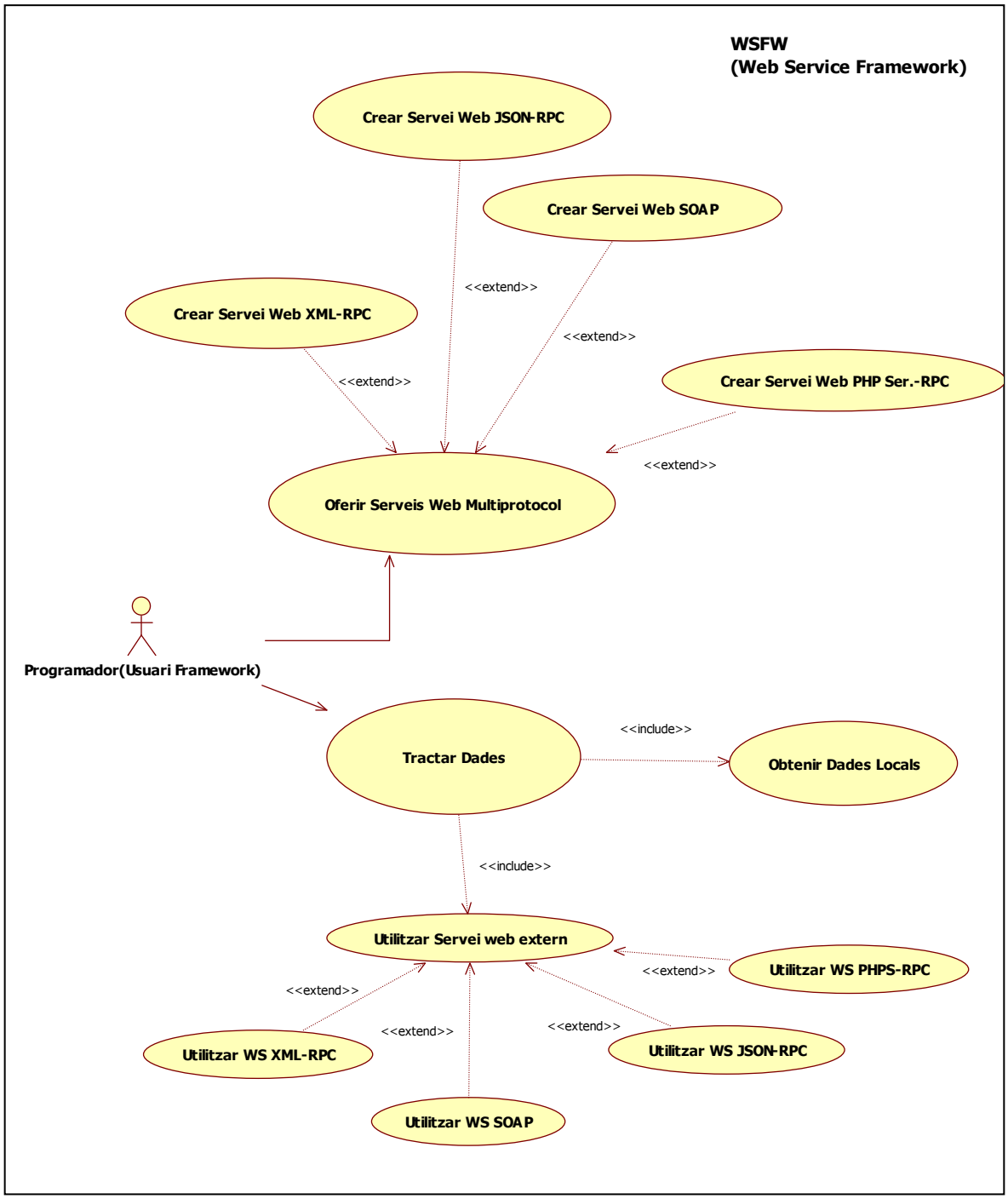

Figura 4.3 Casos d'ús del Framework.

4.4.3 Flux del casos d'ús

En aquest apartat pretenem mostrar el flux lògic d'execució d'una implementació concreta creada usant el Framework WSFW.
Aquest flux és merament un exemple, ja que els fluxos dependran del que implementi el programador. Potser el programador no té base de dades local o potser no necessita obtenir dades d'un servei Web extern, etc. Explicarem el flux que s'entén serà el més habitual per ser el més complet.

1. Utilitzar Servei Web Extern

   És força habitual que el servei web que es vulgui crear necessiti informació proveïda per una altre servei web extern. Actualment tenim molts exemples de serveis web de tercers coneguts ,com per exemple, google search, panoramio o yahoo PlaceFinder.

2. Obtenir Dades Locals

   És força probable que el servei web que es vulgui crear necessiti informació que estigui localitzada a una base de dades local pròpia.

3. Tractar Dades

   Prepararem les dades obtingudes de la base de dades local i del servei web extern per tal de poder-les servir amb el servei web que crearà el programador.

4. Crear Servei Web Multiprotocol

   Aquest cas d'ús, crearà pròpiament el servei web final que vol implementar el programador usuari del WSFW.

5. Disseny.

5.1 Disseny del domini

5.1.1 Fonaments del diagrama de classes

Un diagrama de classes és un dels diagrames de UML que serveix per definir els elements que hi ha d'haver en el sistema a modelar . Es tracta d'un diagrama de tipus estàtic que descriu l'estructura d'un sistema mostrant les seves classes, atributs i les relacions entre ells. Els diagrames de classes s'utilitzen durant el procés d'anàlisi i disseny dels sistemes, on es crea el disseny conceptual de la informació que es manegarà al sistema, i els components que s'encarregaran del funcionament i la relació entre un i altre.
Una classe representa un conjunt de "coses" que tenen un estat i comportament comú. Una classe es representa com una caixa rectangular dividida en tres compartiments: el superior que conté el nom de la classe, el del mig els atributs i el de baix que indica els mètodes o les operacions que la classe pot tenir o dur a terme. Els mètodes poden tenir un qualificador de visibilitat : + (públic), - (privat) , # (protegit) o ~(de paquet).
En aquest diagrama també hi ha representades les relacions entre les diferents classes del sistema. Aquestes relacions són :

- Dependència: és una forma feble de la relació que indica que una classe depèn d'una altra, ja que aquesta la utilitza en algun moment del temps. La dependència existeix si una classe és una variable, paràmetre o variable local d'un mètode d' una altra classe.

- Generalització: indica que una de les dues classes relacionades (el subtipus) és considerada com una forma especialitzada de l'altra (el supertipus).

- Realització: és una relació entre dos elements del model, en què un element del model (el client) realitza (implementa o executa) el comportament que determina l'element d'un altre model (el proveïdor).

Per més informació podeu consultar qualsevol del tres llibres sobre UML indicats a la Bibliografia d'aquesta memòria.

5.1.2 Diagrama de classes del domini

A la imatge de sota (figura 5.1) , podem observar el diagrama de classes del nostre Framework, WSFW.

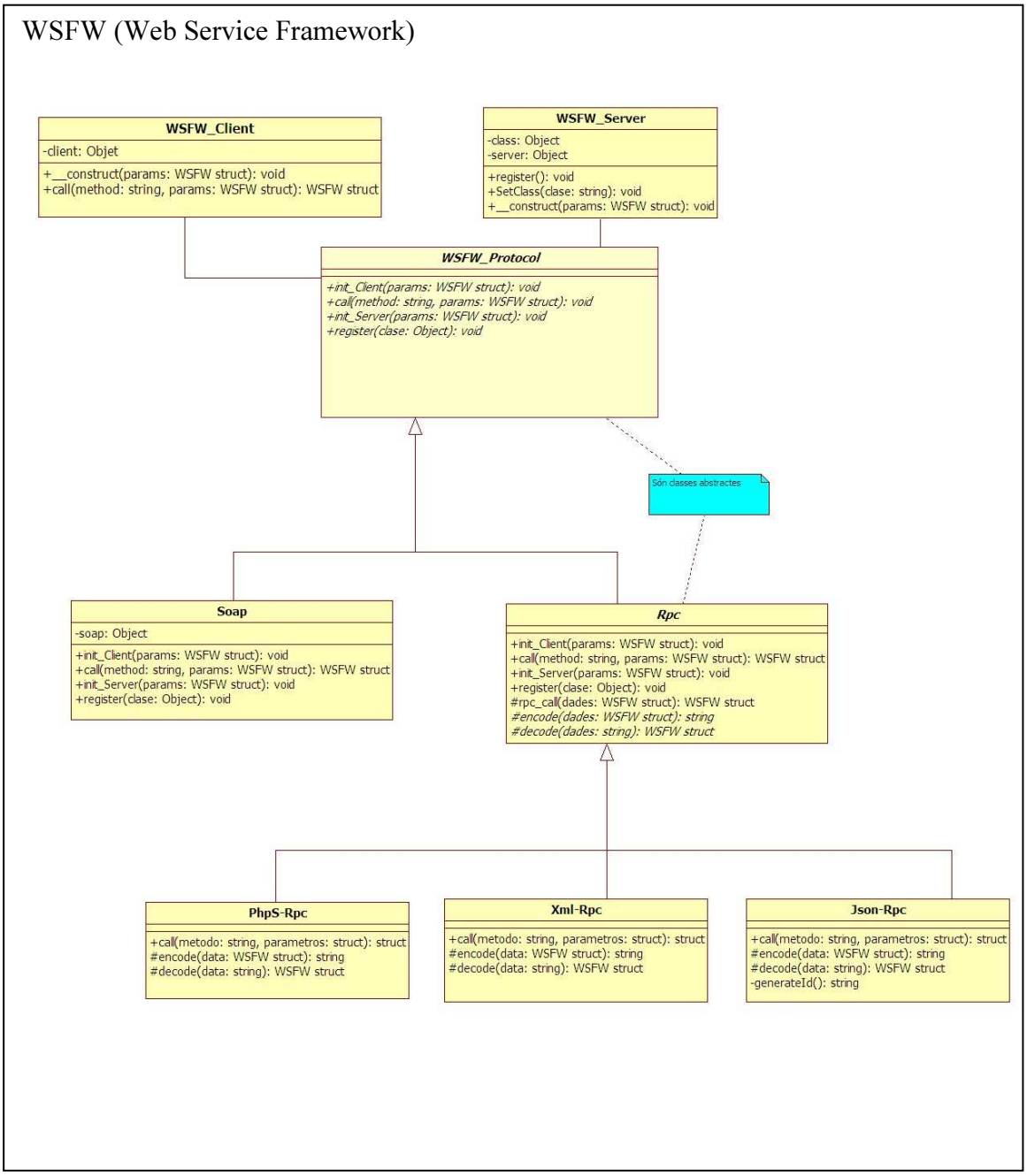

Figura 5.1 Diagrama de Classes del Framework

5.1.3 Explicació de les classes

En aquest apartat explicarem breument cadascuna de les classes del nostre model.

Hem de tenir en compte les següents consideracions sobre els atributs i els tipus de dades que es poden observar en algunes classes del model (veure figura 5-1):

1) El tipus de dades **struct**, és un tipus de dades complex que explicarem més endavant, en l'apartat 6.3.2 del capítol d'implementació.

2) El tipus de dades Object fa referència a un objecte propi del usuari del framework que explicarem més endavant en l'apartat 6.3.1 del capítol d'implementació. Aquest tipus de dades el podem observar en algun atribut o paràmetre d'algun mètode en algunes de les classes.

WSFW_Client

Té com a objectiu principal implementar un Client multiprotocol de serveis web. Ens permetrà realitzar la connexió, de forma homogènia, a un servei web extern de tercers mitjançant diversos protocols.

WSFW_Server

És la classe que implementa el servidor multiprotocol del nostre framework. El programador especificarà la seva pròpia classe, així com els mètodes i serveis que desitgi oferir. La classe realitzarà la inicialització del servidor i prepararà les respostes.

WSFW_Protocol

Aquesta classe, que és abstracta, indica els mètodes necessaris per la implementació dels protocols. Ens mostra els paràmetres i tipus de retorn dels principals mètodes, call i register que corresponen al client i al servidor respectivament. Tots els mètodes que ofereix també són abstractes, per tant seran implementats per les classes filles.

Soap

És l'encarregada d'implementar el protocol SOAP heretat de la classe abstracta WSFW_Protocol . S'encarrega de fer la crida i el registre del serveis consumits o servits. Les definicions de tipus de dades que requereix el protocol SOAP es faran manualment amb el fitxer wsdl que s'especifiqui.

Rpc

La seva missió és implementar el protocol RPC. Es tracta d'una classe abstracta que no implementa cap mètode. La implementació dels mètodes abstractes per part de les classes filles facilitarà l'addició de noves especificacions de serveis web que utilitzin RPC, permeten donar noves funcionalitats al nostre framework WSFW.

PhpS_Rpc

El seu objectiu és especificar la codificació amb el format propi de PHP serialitzat per la classe Rpc de la que hereta. L'ús de PHP serialitzat simplifica el "parser" (parsejador) de PHP incrementant sensiblement el rendiment respecte XML i JSON, ja que inclou paràmetres per facilitar-ne l'accés.

Xml_Rpc

Aquesta classe específica la codificació amb el format de XML per la classe Rpc de la que hereta. Concretament, compleix amb l'especificació de XML-RPC de Microsoft.

Json_Rpc

Classe que especifica la codificació amb el format Json per la classe Rpc de la que hereta. La seva especificació segueix concretament l'especificació Json-RPC creada per Google.

5.2 Estructura del Framework

A la figura 5.2, sota, podem veure una representació del Framework.

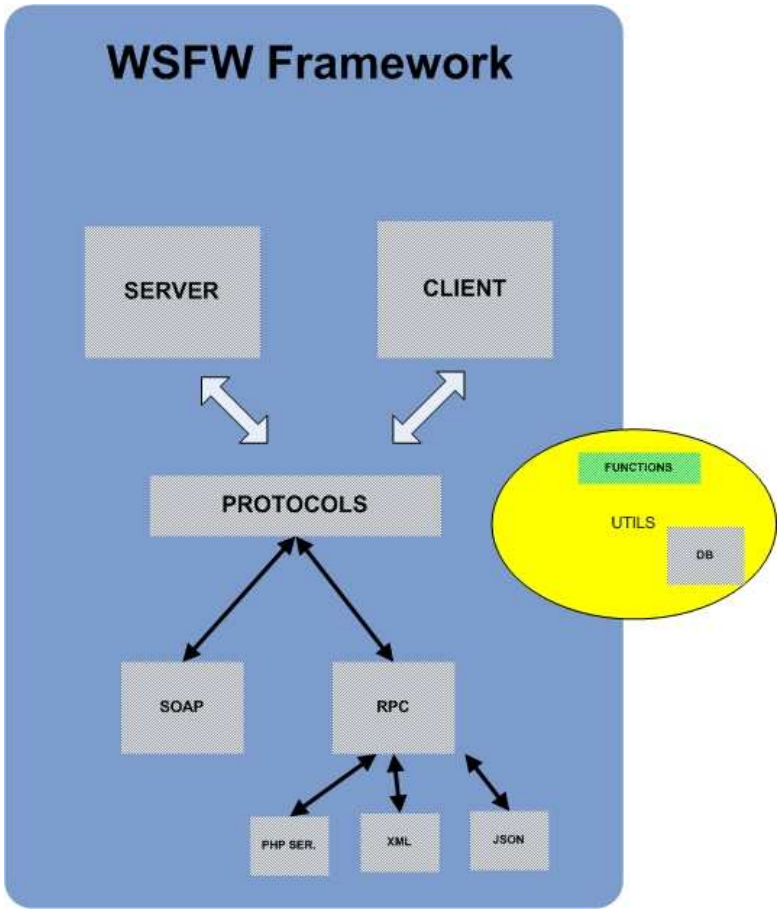

Figura 5.2 Estructura del Framework

El Framework està format per una sèrie de "mòduls", els fonamentals són el Servidor i el Client. Aquests últims, es recolzaran en el mòdul de protocols per funcionar. El mòdul de protocols està format per dos mòduls especialitzats : SOAP i RPC. El mòdul RPC està format per tres, més específics: PHP SER, XML i JSON.

Per últim, el framework (WSFW) proveeix unes utilitats: FUNCTIONS i DB. Aquest mòdul d'utilitats no forma part del Framework en si mateix, però el seu ús pot facilitat molt la explotació d'aquest.

6. Implementació.

6.1 Llenguatges

En aquest apartat s'explicarà breument tot els llenguatges i eines emprades per poder dur a terme aquest projecte.

Cal tenir en compte que encara que l'objectiu d'aquest projecte és la creació del framework WSFW per treballar amb serveis web i que aquest no té res a veure amb la capa de presentació, hem utilitzat llenguatges com el HTML, JavaScript o CSS per realitzar els exemples que demostren el funcionament del mateix.

6.1.1 PHP (Hypertext Prepocessor)

És un llenguatge de programació d'alt nivell, especialment pensat per desenvolupar webs amb continguts dinàmics. És interpretat, de codi obert i gratuït. L' última versió alliberada és la 5.3.4 (10/12/2010).

Un tret important del PHP és el fet de ser un llenguatge "server side", del costat del servidor, és a dir, els fitxer en PHP s' interpreten i executen al propi servidor. La figura 6.1 il·lustra el cicle complet que fa un navegador a una pàgina php.

Tot i que el podem catalogar com a llenguatge d'script es diferencia del JavaScript en que aquest últim s'executa en el client (navegador). El PHP, igual que JavaScript es pot incrustar dintre de pàgines HTML, per fer-ho s'utilitza la següent sintaxi :

<?php ..... ?> o depengen del entorn, <? ..... ?>

La seva sintaxi és molt semblant a C, Java i Perl i es considera fàcil d'aprendre tot i tenir característiques força avançades (orientació a objectes).

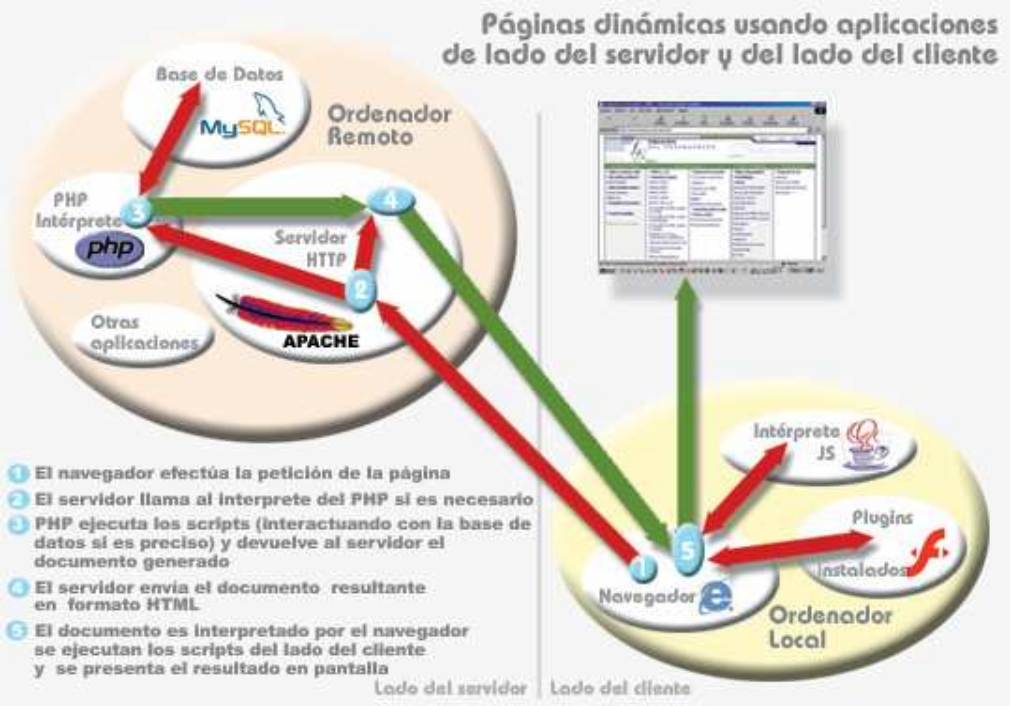

Figura 6.1 Visita a una pàgina php

Una de les característiques clau és el fet de ser suportat per la majoria de sistemes operatius i servidors web del mercat. Es calcula a data d'avui, que està instal·lat en més de 20 milions de "web sites" i en un milió de servidors. [5]

També podem considerar un altre element clau del PHP, el fet de ser el quart llenguatge més popular segons el ranking elaborat mitjançant el Índex Comunitari de programació TIOBE (desembre de 2010), només superat per Java, C i C++. [6]

El framework està implementat 100% amb aquest llenguatge ja que semblava el més oportú perquè el seu ús va destinat a programadors de PHP que hagin de crear o utilitzar serveis web. Concretament està implementat en PHP 5.2.0 i les aplicacions desenvolupades amb ell podran funcionar amb la versió 5.0 o superior.

Per més informació podeu consultar : http://www.php.net/

6.1.2 **JAVASCRIPT** (Hyper Text Markup Language)

És un llenguatge de programació interpretat del costat del client. És un llenguatge de codi obert, fet que l'ha convertit en el llenguatge d'script del costat del client més emprat al món.

Fou inventat per un treballador de la empresa Netscape i va aparèixer per primer cop al navegador Netscape Navigator 2.0. L'any 1997 va ser adoptat com un estàndard ECMA i posteriorment com un estàndard ISO.

Figura 6.2 Logotip JavaScript

Té una sintaxi molt semblat a Java, però a diferencia d'aquest , no està pròpiament orientat a objectes sinó que segueix el paradigma del prototipatge. Les classes a implementar es construeixen a partir de classes base (prototips) ampliant la seva funcionalitat. Per tal de que tots els navegadors actuals el puguin interpretar correctament, aquests es proveeixen d'una implementació del Document Object Model (DOM).

El codi JavaScript s'incrusta dins del codi HTML aportant a la pàgina web dinamisme i interactivitat amb l'usuari. El navegador executa el codi JavaScript de la pàgina html quan un event és executat, per exemple un click en un botó d'un formulari.

El codi JavaScript s'ubica dins del document html entre les etiquetes <body></body> o bé es pot fer una referència a un fitxer extern, indicant <script type="text/javascript" src="codi.js"></script>.

6.1.3 CSS (Cascading Style Sheets)

Les fulles d'estil en cascada és un llenguatge formal per descriure la presentació d'un document estructurat, típicament HTML,XML i els seus derivats. La principal propietat que ens aporta és la separació entre el document escrit i la seva presentació.

Figura 6.3 Logotip CSS

Els principals avantatges que aporta són:

- ✓ Control centralitzat de la presentació d'un lloc web complet amb el que s'agilitza de forma considerable l'actualització del mateix.

- ✓ Els navegadors permeten als usuaris especificar el seu propi full d'estil local que serà aplicat a un lloc web, amb el que augmenta considerablement l'accessibilitat. Per exemple, persones amb deficiències visuals poden configurar el seu propi full d'estil per a augmentar la grandària del text o remarcar més els enllaços.

- ✓ Una pàgina pot disposar de diferents fulls d'estil segons el dispositiu que la mostri o fins i tot a elecció de l'usuari. Per exemple, per a ser impresa, mostrada en un dispositiu mòbil, o ser "llegida" per un sintetitzador de veu.

- ✓ El document HTML en si mateix és més clar d'entendre i s'aconsegueix reduir considerablement la seva grandària .

Per més informació podeu consultar : http://ca.wikipedia.org/wiki/Cascading_Style_Sheets

6.1.4 HTML (Hyper Text Markup Language)

El Hyper Text Markup Language, en català llenguatge de marcat d'hipertext, és un llenguatge d'etiquetes dissenyat per descriure l'estructura i el contingut en format text i amb capacitat per incloure objectes tals com imatges.

Gràcies a Internet i als navegadors més populars com Internet Explorer, Mozilla Firefox, Chrome, Safari,Opera, ha esdevingut el llenguatge predominant en la construcció de documents per la web.

HTML pot ser editat i creat amb qualsevol editor de text senzill com el vi de Linux o el notepad de Windows, tot i que existeixen editors molt avançat per facilitar la construcció visual dels documents.

Dintre del document HTML, es poden incloure llenguatges tipus script, com JavaScript, que poden afectar al comportament del navegador en resposta a qualsevol event.

6.1.5 UML (Unified Modeling Language)

El llenguatge unificat de modelat es va desenvolupar com a resultat d'un esforç en el desenvolupament d'un llenguatge estandarditzat únic de modelatge orientat a objectes. Aquest llenguatge millora i reuneix conceptes de tres mètodes d'anàlisi i disseny, anomenats Booch, OMT i OOSE.[7]

UML capta informació sobre l'estructura estàtica i el comportament dinàmic d'un sistema. Un sistema es modela com un recull d'objectes discrets que interactuen per realitzar un treball que finalment beneficia a un usuari extern.

Aquest llenguatge, amb l'objectiu de construir models, fa ús de diversos diagrames. A la versió UML 2.0 hi ha tretze diagrames i s'agrupen d'acord a una de les següents tipologies: diagrames d'estructura, diagrames de comportament i diagrames d'interacció.

Els diagrames d'estructura es fixen en els elements que hi ha d'haver en el sistema a modelar. Els diferents diagrames són: diagrames de classes, diagrames de components, diagrames d'objectes, diagrames d'estructura composta, diagrames de desplegament i diagrames de paquets.

Els diagrames de comportament es fixen en el què ha d'esdevenir el sistema que volem modelar. Els diferents diagrames són: diagrames d'activitats, diagrames de casos d'ús i diagrames d'estats.

Els diagrames d'interacció són un subconjunt del diagrames de comportament que es fixen en el flux de control i dades entre els diferents elements del sistema modelat. Els diferents diagrames són: diagrames de seqüència, diagrames de comunicació, diagrames de temps i diagrames de vista d'interacció.

En el projecte només s'ha considerat oportú utilitzar dos d'aquest diagrames: diagrama de classes i diagrama de casos d'ús. Al capítol d'anàlisi fem una breu explicació del diagrama de casos d'ús i al capítol de disseny expliquem el diagrama de classes.

6.2 Entorn desenvolupament

6.2.1 Gantt Project

És un software creat per a la gestió i planificació de projectes de tot tipus. Es distribueix sota diverses llicenciés del tipus GPL, és software lliure i de codi obert. Té versions per Windows, MacOS i Linux.

En el projecte s'ha utilitzat per fer els diagrames de Gantt de la planificació del projecte del capítol 7 i en les conclusions, capítol 10, per fer la re planificació del projecte obeint a les variacions temporals que han esdevingut.

Per més informació podeu consultar : http://www.ganttproject.biz/

6.2.2 Start UML

És un programari de codi obert per facilitar el modelatge del software amb UML (llenguatge unifica de modelatge). Només està disponible per Windows i possibilita l'addicció de nous components mitjançant plug-ins.

En el projecte s'ha utilitzat per realitzar tots els diagrames de UML, concretament s'ha fet servir el diagrama de classes i el de casos d'ús.

6.2.3 IDE NetBeans

NetBeans és una plataforma pel desenvolupament d'aplicacions d'escriptori. Està escrit en llenguatge Java i és un dels IDEs més utilitzats per aquesta comunitat. És una eina de codi obert sotmesa a doble llicència: CDDL ("Common Development and Distribution License") i GNU General Public License v.2. Aquesta eina està destinada a programadors per escriure, compilar, depurar i executar aplicacions. Encara que la versió d' aquest IDE més utilitzada és la versió per desenvolupar Java, hi ha versions per altres llenguatges de programació com Php.

En el projecte s'ha utilitzat per fer tota la codificació en php, tant a la implementació del propi framework WSFW , com en l' elaboració del les dues aplicacions de l'exemple.

Per més informació podeu consultar : http://www.netbeans.org

6.2.4 Wamp

WAMP és un acrònim que respon a les sigles de Windows – Apache – MySql i PHP/Python/Perl. Aquest terme fa referència al sistema creat pel conjunt d'aquestes aplicacions lliures (de codi obert) i el sistema operatiu propietari Windows. WAMP proveeix al desenvolupador amb els quatre elements típics d'un servidor Web: un sistema operatiu (Windows), un gestor de bases de dades (MySQL), un programari servidor web (Apache) i un llenguatge de guions (PHP, Python o PERL).

Existeixen diverses agrupacions de programari molt semblants a WAMP, com poden ser MAMP, FAMP o LAMP. La diferència entre elles es pot deduir de la primera lletra del acrònim (el sistema operatiu), els citats responen respectivament a MacOS, FreeBSD i Linux.

En el projecte s'ha utilitzat per realitzar totes les aplicacions. El PHP és el llenguatge utilitzat per implementar el framework WSFW i les dues aplicacions que formen part del exemple del apartat 9. MySQL l'hem utilitzat per emmagatzemar la informació del exemple. Apache és el servidor web emprat per poder córrer les aplicacions web desenvolupades i Windows és el sistema operatiu de l'ordinador portàtil de desenvolupament.

Per més informació podeu consultar : http://www.wampserver.com/en/index.php

6.2.5 PHP Documentor

Conegut informalment com PHPDOC, és l'eina preferida dels programadors php per crear documentació de les aplicacions desenvolupades. Va ser una adaptació del famós documentador Javadoc. Mitjançant una sèrie de marques en el codi font dels fitxers, el programador pot indicar-li diversos paràmetres que influiran a la documentació generada. La documentació generada és de fàcil lectura i ofereix els formats més populars com pdf, CHM (arxius d'ajuda de Windows), HTL o XML.

En el projecte s'ha utilitzat per unificar tota la documentació del codi font escrit i oferir-la amb format HTML en el CD que acompanya aquesta memòria.

Per més informació podeu consultar : http://www.phpdoc.org

6.2.6 Navegador Web

Un navegador web és un programari que permet a l'usuari visualitzar la informació que conté una pàgina web.

En el projecte s'han utilitzat els navegadors web per comprovar el funcionament de les dues aplicacions web implementades al capítol 9 (Exemple pràctic del framework). Majoritàriament s'ha utilitzat Mozilla Firefox perquè es considera que és el navegador amb més capacitat de "debugging", alternativament per comprovar la compatibilitat de les aplicacions desenvolupades s'han fet proves amb Opera, Google Chrome i Internet Explorer.

6.2.7 Programari d'ofimàtica

Pròpiament anomenat OpenOffice.org (no OpenOffice, degut a una disputa de marques). És un projecte comunitari per crear un paquet d'ofimàtica en codi obert (amb llicencia LGPL), procedent d'una versió antiga de StarOffice de Sun Microsystems. Actualment la marca és propietat de l'empresa Oracle. [8].

Aquest paquet d'ofimàtica consta dels següents components: Editor de text ("Writter"), Full de càlcul ("Calc"), Presentacions ("Impress"), Base de dades("Base") i Dibuixos ("Draw").

L'editor de text, permet guardar el document creat en multitud de formats, a més del propi (odt), entre d'ells destaquem word 97/2000/XP i rtf. També permet exportar el document a PDF.

6.2.8 PHPUnit

És un framework per realitzar proves unitàries específicament ideat per llenguatge php. Va ser creat per Sebastian Bergman i alliberat amb llicencia BSD ("Berkeley Software Distribution"). Aquest tipus de llicència és inclús més permissiva que la GPL, ja que permet l'ús del codi font en software no lliure.

Dintre del mètodes de test, utilitzarem mètodes d'asserció com **assertEquals()** per realitzar assercions del valor esperat en cada moment. Altres mètodes disponibles són :

- **void assertTrue(bool $condition):** Retorna error si la condició és falsa.
- **void assertTrue(bool $condition, string $message):** Retorna error i el missatge si la condició és falsa.
- **void assertFalse(bool $condition):** Retorna error si la condició és certa.

.

- **void assertNull(mixed $variable):** Retorna error si la variable no és null.
- **void assertNotNull(mixed $variable):** Retorna error si la variable és null.

- **void assertSame(Object $expected, Object $actual):** Retorna error si l'objecte esperat i la actual no fan referència al mateix objecte.

Per automatitzar les proves de l'aplicació php, simplement estenem la classe principal del framework, que es pot dir de maneres diferents depenent de la versió concreta de PHPUnit.

Per estalviar temps en la codificació del test, recordem que cada un ha de ser totalment independent, la classe principal del framework ens ofereix dos mètodes molt interessants: **setUp()** i **tearDown()**. Bàsicament el que proporcionen és estalvi de línies de codificació en el test.

Per més informació podeu consultar : https://github.com/sebastianbergmann/phpunit/

6.3 Decisions d'implementació

6.3.1 Orientació a objectes

El framework actua sobre la classe definida pel programador, al instanciar el servidor hem d'indicar quina és aquesta classe per tal de poder servir com a servei web tots els mètodes que tingui definits.

Per simplificar la implementació, els mètodes de la classe del programador només necessiten rebre un paràmetre d'entrada, que serà la petició. Aquesta serà un array associatiu on es podran afegir tots els paràmetres que es desitgi. Els mètodes retornaran un altre array associatiu amb els valors de la resposta.

Només s'accepta una classe per instància del servidor, però en cas de necessitar servir més d'una classe, el programador pot crear un objecte servidor per cada classe. En el cas de l'exemple creuRoja, en lloc de crear el servei web com a index.php, s'hauria pogut crear un servei web creuroja.php i definir-ne d'altres per a altres classes.

L'orientació a objectes permet estendre les classes del framework per tal d'adaptar els protocols al format que es necessiti. D'aquesta manera, en cas de desitjar servir dades o connectar a un servei web que utilitzi JSON-RPC que no compleixi amb l'especificació de JSON-RPC 2.0 utilitzada, simplement s'haurà d'estendre la classe JSON-RPC per adaptar-la al format requerit. Podem veure un exemple en la classe Panoramio, en el Capítol 9, Exemple pràctic del framework.

La classe RPC és abstracte, permet implementar altres protocols que utilitzin formats diferents a XML, JSON o serialitzat PHP, simplement extenent-la i implementant els mètodes abstractes de codificació i descodificació.

D'altra banda, la classe abstracte protocol permet definir com s'han de crear nous protocols per tal que es puguin instanciar des de les classes client i servidor.

6.3.2 Arrays associatius

Les consultes habituals que es fan als serveis web utilitzen com a entrada i sortida, una petició i resposta, ambdues solen ser estructures de dades complexes.

En PHP hi ha dues maneres per representar estructures de dades: els objectes i els arrays associatius.

La resposta d'un servei web pot ser un registre o una col·lecció de registres, en cas d'utilitzar objectes tindríem que els tipus de retorn serien arrays d'objectes o objectes, sempre en funció del servei web on connectem.

D'altra banda els conjunts de dades que es retornen en un servei web o les dades de la petició no inclouen mètodes, per tant nosaltres només necessitem utilitzar les dades i la seva estructura.

S'ha decidit que el framework, WSFW, treballarà amb arrays associatius enlloc d'objectes. Tot i que es perden certes funcionalitats, es simplifica considerablement l'ús del mateix

perquè al no haver de definir les estructures de dades en cas de que aquestes ja vinguin definides per una base de dades o per un altre servei web. També s'aprofiten les funcions i les propietats sobre arrays ja definides al propi llenguatge PHP. Amb l'objectiu de simplificar i ajudar al màxim a l'usuari del WSFW, aquest proveeix d'un paquet d'utilitats globals amb diverses funcionalitats complexes com comparar arrays d'arrays associatius, fer-ne la unió o intersecció.

Tots els mètodes i propietats de les classes del framework utilitzen aquestes estructures de dades.

6.3.3 Seguretat

El framework no ofereix per si mateix cap mecanisme específic de seguretat. Tot i així, per inicialitzar-se, tant el Client com el Servidor , utilitzen paràmetres d'entrada, un dels quals és la url del servei web que s'oferirà o del servei web que es consumirà. Aquesta url, pot indicar l'ús de SSL, en el cas que la cadena sigui del tipus "https://xxxx/yyyy..." .

Gràcies a SSL (Secure Socket Layer) es proporciona autentificació i privacitat de la informació entre extrems a Internet. En la majoria dels casos , trobem només autentificació del servidor mentre que la autentificació del client, si es fa, sol ser externament mitjançant login i password. En altres casos, cada cop més habituals, s'utilitza un certificat del client per garantir una comunicació amb total garantia de privacitat.

SSL també ens proporciona xifrat de la informació intercanviada entre els extrems, generant una clau única de sessió per cada transacció utilitzant algorismes de xifrat de clau publica, típicament l'algoritme de hash és MD5.

6.4 Implementació classes del domini

En aquest apartat mostrarem i explicarem com s'han implementat les principals classes del domini i els mètodes més importants. Concretament explicarem les següents classes del framework WSFW: Client, Server, Protocol, Soap, Rpc i Json-Rpc. Les classes no explicades són molt semblants a algunes de la que comentarem, de totes maneres, a l'Annex I (Contingut del CD) podem trobar tot el codi font comentat del framework. En cas de dubte, es pot trobar el codi font corresponent a una classe concreta consultant l'apartat 6.6, Jerarquia de fitxer de WSFW.

WSFW_CLIENT

Aquesta classe és la que utilitzarà el programador de WSFW per implementar un client multiprotocol que pot connectar a un servidor web service extern.

- L'atribut $client és l'objecte que implementa el client del protocol que es vol utilitzar.

- El constructor de la classe, la **funció __construct()**, s'ocupa de crear l'objecte client i crida al mètode **initClient()** de la classe indicant-li com a paràmetre d'entrada un array associatiu ($params) que típicament contindrà la url destí.

- La funció **call()** retorna un array associatiu amb la informació sol·licitada al servei web extern. El mètode a executar ens l'indica el paràmetre d'entrada $method, i l'altre paràmetre d'entrada, $params, que és un array associatiu, representen els paràmetres definits pel programador que necessitarà el mètode cridat del servei web.

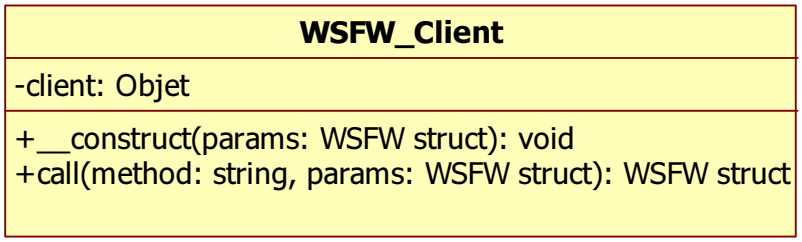

Figura 6.4 Classe Client

A la figura 6.5 podem veure la codificació en PHP d'aquesta classe.

```
class WSFW_Client {

        private $client;


        public function  __construct($params) {
                $this->client = new $params["protocol"]();
                $this->client->initClient($params);
        }


        public function call($method, $params) {
                return $this->client->call($method, $params);
        }

}
```

Figura 6.5 Codificació PHP de WSFW_Client

WSFW_Server

Aquesta classe és la que s'utilitzarà per implementar un servidor web service multiprotocol.

- L'atribut $class és l'objecte que instància la classe pròpia del programador.
- L'atribut $server és l'objecte que implementa el servidor del protocol que es vol utilitzar.
- El constructor de la classe, la funció **__construct()** assigna el protocol a servir que es rebrà per GET i crida a la funció **initServer()** de la classe protocol indicant-li com a paràmetre un array associatiu ($params) que típicament contindrà el fitxer WSDL pel protocol SOAP.
- La funció **setClass()** assigna l'atribut $class de la classe del programador i crea un objecte d'aquesta.

- La funció **register()**, registra els serveis provinents dels mètodes de la classe pròpia del programador.

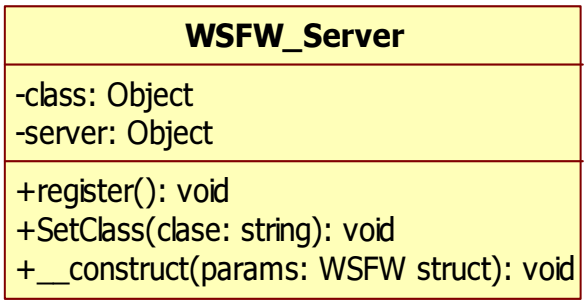

Figura 6.6 Classe Server

A la figura 6.7 podem veure la codificació en PHP d'aquesta classe.

```
class WSFW_Server {

   private $class;
   private $server;



 public function __construct($params) {
                global $_GET;
                $protocol = key($_GET);
                $this->server = new $protocol ();
                $this->server->initServer($params);
 }


  public function setClass($user_class) {
                $this->class = new $user_class();
  }

  public function register() {

                        $this->server->register($this->class);
  }

}
```

Figura 6.7 Codificació PHP de WSFW_Server

WSFW_Protocol

Aquesta classe és abstracta i totes les funcions també ho són. Per tant, simplement està indicant a les seves classes filles quines funcionalitats hauran d'implementar. Els servidors utilitzaran **register()** i **initServer()** mentre que els clients faran ús de **call()** i de **initClient()**

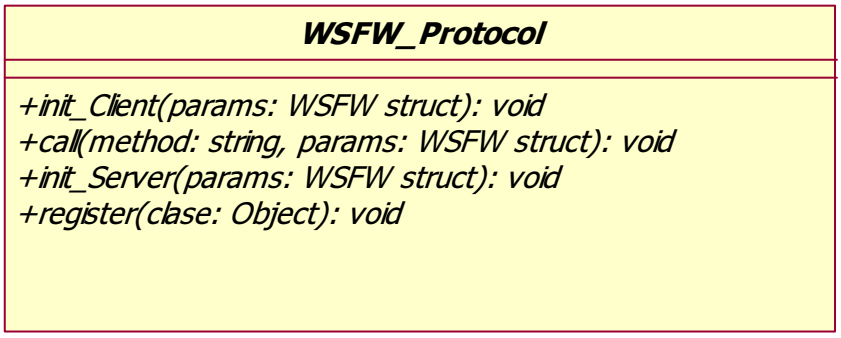

Figura 6.8 Classe Protocol

A la figura 6.9 podem veure la codificació en PHP d'aquesta classe abstracte.

```
abstract class WSFW_Protocol {

        abstract public function initClient($params);

        abstract public function call($method, $params);

        abstract public function initServer($params);

        abstract public function register($class);
}
```

Figura 6.9 Codificació PHP de WSFW_Protocol

Soap

És l'encarregada d'implementar el protocol SOAP, hereta de la classe abstracta WSFW_Protocol . S'encarrega de fer la crida i el registre del serveis consumits o servits.

- L'atribut privat $soap és un objecte, pròpiament és el client o el servidor soap.

- El mètode **initClient()** té com a paràmetre d'entrada $params, que és una array associatiu. A la pràctica, és el constructor de la classe quan es comporta com a client d'un servei web. El paràmetre $params["url"] conté la direcció a on es realitza la crida. La instrucció **$this->soap = new SoapClient($params["url"], array("trace"=> 1, "exceptions"=> 1))** crea i assigna l'objecte a la variable privada $soap, fent ús de la llibreria SOAP pròpia del PHP.

- La funció **call()** implementa la crida a un servei web extern, per tant, estem parlant d'un client soap. Aquesta funció té com a paràmetres d'entrada $method, que és una cadena (string) que conté el nom del servei que volem cridar i $params que és un array associatiu que conté els paràmetres necessaris per realitzar la petició. La instrucció **$this->soap->__soapCall($method, $params)** utilitza la funció baix nivell de l'API de PHP per fer crides Soap. La funció **call()** retorna un array associatiu que conté la resposta del servei web en soap.

- El mètode **initServer()** és a la practica el constructor de la classe quan es tracta d'un servidor d'un servei web. La instrucció **$this->soap = new soap_server($params[$params["wsdl"])** crea i assigna l'objecte a la variable privada $soap, fent ús de la llibreria SOAP pròpia del PHP. El paràmetre $params["wsdl"] és l'arxiu de descripció del serveis que indicarà el programador al instanciar l'objecte WSFW_Server.

- El mètode **register()** s'ocupa de registrar tots els mètodes disponibles a la classe pròpia del programador, que es passa com a paràmetre d'entrada. És a dir, registra tots els serveis disponibles del servidor del servei web i assigna el mètode de la classe del programador al mètode registrat.

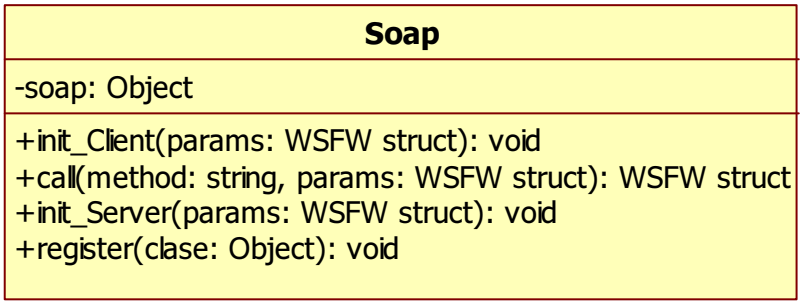

Figura 6.10 Classe Soap

A continuació (figura 6.11) podem veure la codificació en PHP d'aquesta classe.

```
class Soap extends WSFW_Protocol {

        private $soap;

        public function initClient($params) {

                $this->soap = new SoapClient($params[“url”], array("trace"=> 1,
"exceptions"=> 0));                                }

        public function call($method, $params) {
                if (strstr($method, "."))
                $response = $this->soap->__soapCall($method, $params);
                else
                $response = $this->soap->$method($params);
                return objectToArray($response);
        }

        public function initServer($params) {
                $this->soap = new soap_server($params[“wsdl”]);
        }

        public function register($class) {
                $request = file_get_contents('php://input');
                $this->soap->setClass(get_class($class));
                $this->soap->handle($request);
        }

}
```

Figura 6.11 Codificació PHP de Soap

Rpc

Classe que implementa el protocol RPC. Mitjançant el cos de les peticions POST i GET d' HTTP es transporten les estructures de dades de peticions i respostes respectivament. La classe és abstracta ja que es pot especificar la codificació que s'utilitza. En els casos de Xml_Rpc i Json_Rpc a part de la codificació també s'inclouen els paràmetres necessaris per complir amb les especificacions més esteses.

Aquesta classe és abstracta però a diferencia de casos anteriors, no tots el seus mètodes i/o funcions ho són. Al heretar d'una classe abstracta en principi, al menys en llenguatge PHP, es té l'obligació d'implementar (codificar) totes les funcions i mètodes abstractes de la classe mare (superclasse), en aquest cas de WSFW_Protocol. En aquest projecte ens saltem aquesta limitació tècnica simplement reescrivint les que no ens interessa codificar deixant el cos buit, per exemple a les funcions **call()** i **initServer().** Com totes les classes abstractes, el seus mètodes i funcions simplement indiquen a les seves classes filles quines funcionalitats hauran d'implementar.

- El mètode **initClient()** que té com a paràmetre d'entrada $params, que és un array associatiu amb tota la informació necessària, és a la practica com el constructor de la classe quan es comporta com un client d'un servei web. El paràmetre $params["url"] és la direcció a on es realitza la crida. Inicialitza el client, únicament es guarda la direcció per poder fer les crides.

- La funció **rpc_call()** implementa la crida a un servei web extern. Inclou els paràmetres codificats al cos del POST de la petició, fa la crida HTTP i retorna el resultat descodificat en un array associatiu.

- El mètode **register()** s'encarrega de registrar tots el serveis del servidor. Com a paràmetre d'entrada té l'objecte que pertany a la classe del programador. Agafa la petició realitzada pel client, mitjançant POST, i la descodifica. En aquesta petició hi troba el mètode i els paràmetres que utilitza per fer la crida a l'objecte de la classe del programador. El resultat d'aquesta crida, la resposta, es codifica i es retorna.

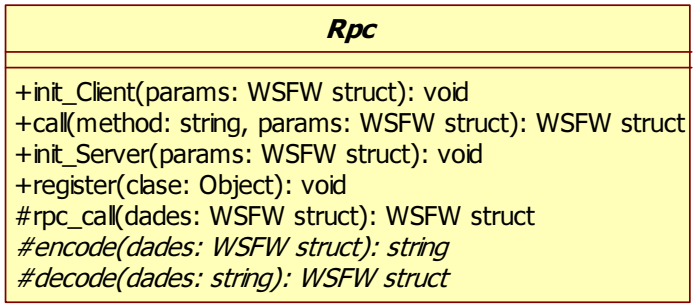

Figura 6.12 Classe Rpc

A la figura 6.13 podem veure la codificació en PHP d'aquesta classe.

```
abstract class Rpc extends WSFW_Protocol {

    protected  $request_url;

    public function initClient($params) { $this->request_url = $params[“url”]; }

    public function call($method, $params){}

    protected function rpc_call ($_data=null) {

      if ($_data) {
            $request = $this->encode($_data);
            $data = stream_context_create(array('http' => array('method' =>
        'POST', 'content' => $request)));
            $response = file_get_contents($this->request_url, false, $data);
                        }
                    else { $response = file_get_contents($this->request_url);}

             if ($response == false) return error("Error en la petición RPC");

            $data = $this->decode($response); return $data;
        }
    public function initServer() {}

    public function register($class) {
                $request =$this->decode(file_get_contents('php://input'),true);
                $response = $class->$request[“method”]($request[“params”]);
                echo $this->encode($response);
        }
      abstract protected function encode($data);
      abstract protected function decode($data);
}
```

Figura 6.13 Codificació PHP de Rpc

Json-Rpc

Classe que especifica la codificació amb el format Json per la classe Rpc, de la que hereta. El format compleix amb l'especificació Json-RPC creada per Google. Implementarà els mètodes abstractes de la classe mare, Rpc, que són **encode()** i **decode()**. També estendrà la funció **call()** per tal d'adaptar-se a l'especificació.

- La funció **call()** implementa la crida a un servei web extern, per tant, estem parlant d'un client Json-Rpc. Aquesta funció té com a paràmetres d'entrada $method, que és una cadena (string) ,que conté el nom del servei que volem cridar i $params que és un array associatiu que conté els paràmetres necessaris per realitzar la petició. La instrucció **$data=$this->rpc_call($data)** utilitza el mètode de RPC per fer crides. La funció **call()** retorna un array associatiu que conte la resposta del servei web.

- La funció privada **generateId()** que té com a valor de retorn una cadena (string), és una utilitat definida a l'especificació Json-RPC per comprovar que el identificador de la crida és el mateix que es rep a la resposta. Aquesta funció l'utilitza la funció **call().**

- La funció protegida **encode()** té com a paràmetre d'entrada un array associatiu i retorna la codificació Json d'aquesta. La funció PHP **json_encode($valor)** només pot treballar amb valors en UTF-8.

- La funció protegida **decode()** té com a paràmetre d'entrada una string codificada amb Json i retorna un array associatiu amb la descodificació. La funció PHP **json_decode($data,true)** només pot treballar amb valors en UTF-8. El segon paràmetre (true), indica que retorni un array associatiu.

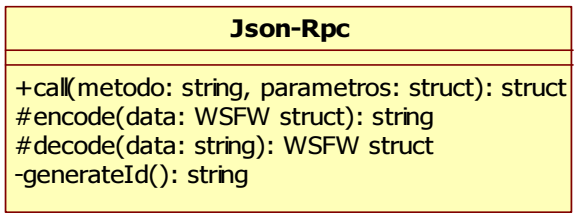

Figura 6.14 Classe Json-Rpc

A la figura 6.15 podem veure la codificació en PHP d'aquesta classe.

```
class Json_rpc extends Rpc {


  public function call($method, $params) {
                parent::call($method, $params);
                $_data["params"] = $params;
                $_data["jsonrpc"] = "2.0";
                $_data["method"] = $method;
                $_data["id"] = $this->generateId();
                $data = $this->rpc_call($_data);
    if ($data["id"] != $id)
                        return error("Error en la ID de la petición JSON");
                return $data;
        }


  private function generateId() {
     $chars = array_merge(range('A', 'Z'), range('a', 'z'), range(0, 9));
     $id = '';
     for($c = 0; $c < 16; ++$c)
        $id .= $chars[mt_rand(0, count($chars) - 1)];
     return $id;
  }


  protected function encode($data) { return json_encode($data);
        }

  protected function decode($data) { return json_decode($data,true);}
```

Figura 6.15 Codificació PHP de Json_rpc

6.5 Implementació utilitats de WSFW

Juntament amb la implementació de les classes del domini, el framework incorpora dos utilitats força interessants per facilitar la feina als seus usuaris: WSFW_Db és una implementació d'una classe per facilitar la connexió i les consultes a una base de dades MySql, i functions que és una llibreria per facilitar-nos treballar amb comoditat amb els arrays associatius que utilitzarem en tot el framework. A continuació farem una breu explicació d'aquestes:

6.5.1 WSFW_Db

Aquesta classe s'encarrega d'implementar la connexió a una base de dades MySQL i les peticions corresponents amb SQL. Ofereix mètodes per realitzar crides a sentencies simples SQL com poden ser selecció (SELECT), actualització (UPDATE), eliminació (DELETE) o inserció (INSERT).

Aquesta classe no forma part de domini, és una classe de cortesia que proveeix el framework per als programadors que utilitzin una base de dades MySQL.

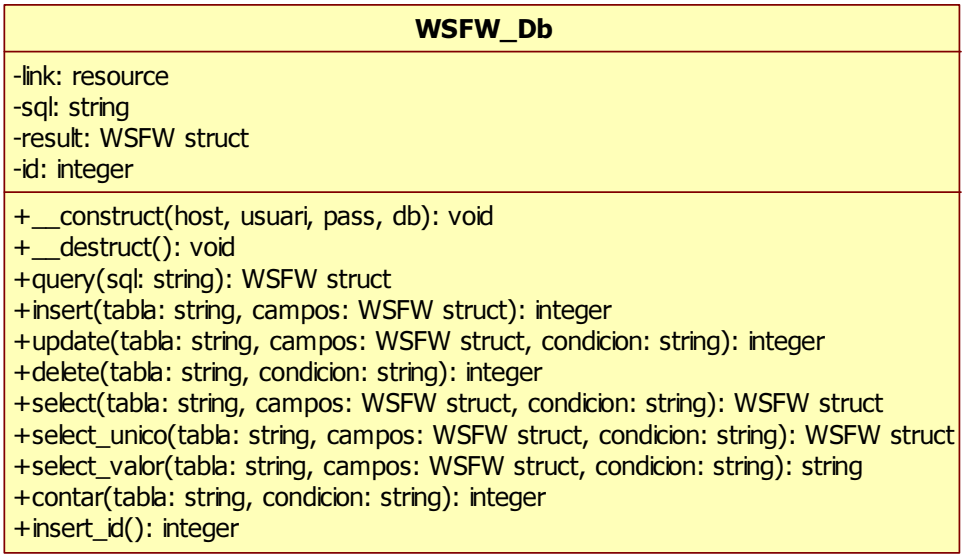

Figura 6.16 Classe WSFW_Db

Atributs:

- Link : aquest atribut és de tipus resource, propi de PHP per connectar amb MySql.
- Sql: és una cadena de text, que contindrà la consulta Sql a executar.
- Result: és un array associatiu que contindrà el resultat d'una consulta Sql.
- Id: és un enter que és l'últim valor generat per un camp autoincremental de la consulta prèvia.

Mètodes

```
function __construct ($host, $port, $user, $pass, $database) {
        $this->link = mysql_connect ($host, $user, $pass);
        if (!$this->link) die ('<b>No se puede conectar a la base de datos</b><br>' . mysql_error());
        mysql_select_db($database) or die('<b>No se puede seleccionar la base de datos</b><br>' . mysql_error());
        mysql_query ("SET NAMES 'utf8'");
    }
```

Figura 6.17 Codi Font constructor WSFW_Db

Aquest mètode, el constructor, és l'encarregat de realitzar la connexió a la base de dades. Per exemple, si els paràmetre de host, port, user, pass i database fossin respectivament 'localhost', '3306', 'marc', 'xyZ023', 'evsplatges', la funció de l'Api de PHP **mysql_connect()** intentaria realitzar una connexió per el port numero 3306 a la base de dades anomenada evsplatges, allotjada a la màquina localhost, proporcionant com a nom d'usuari marc i password xyZ023. En cas de no poder establir la connexió sortim del mètode amb **die()**, semblant al exit() en altres llenguatges de programació.

```
function query ($sql) {
        $this->result = mysql_query ($sql);
        if (!$this->result) die ('<b>Consulta inv&aacute;lida</b><br>La
consulta ha fallado, consulte con el administrador.<br><b>Descripci&oacute;n
del error:</b><br> ' . mysql_error() . '<br><b>Consulta:</b> ' . $sql);
        return $this->result;
    }
```

Figura 6.18 Codi funció query() de WSFW_Db

Aquesta funció ens retorna un array associatiu si el resultat de la consulta té valor. Podem veure, que la consulta l'executa amb la funció de l'Api de PHP **mysql_query()**.

```
function select_unico ($campos, $tabla, $condicion) {
        $valores = null;
        foreach ($campos as $value) {
            $valores .= $value;
            if (next ($campos)) $valores .= ", ";
        }
        if ($condicion) $condicion = "WHERE $condicion";
        $sql = "SELECT $valores FROM $tabla $condicion LIMIT 0,1";
        $this->query($sql);
        if (mysql_num_rows($this->result) == 1)
            return mysql_fetch_assoc($this->result);
        else return false;
    }
```

Figura 6.19 Codi funció query() de WSFW_Db

Aquesta funció retorna un array associatiu amb un únic resultat corresponent a la fila recuperada per la consulta o bé retorna un valor bolea fals cas de que la consulta retorni més d'un resultat.

```
function insert ($tabla, $campos) {
        $valores = null;
        $i = 0;
        foreach ($campos as $key => $value) {
                $value = ereg_replace("\"", "'", $value);
                if ($value == "NOW()") $valores .= "$key=$value";
                else $valores .= "$key=\"$value\"";
                if ($i != count($campos)-1) $valores .= ", ";
                $i++;
        }
        $this->query ("INSERT INTO $tabla SET $valores");
        $this->id = mysql_insert_id ();
        return mysql_affected_rows();
    }
```

Figura 6.20 Codi funció insert() de WSFW_Db

Aquest mètode s'encarrega de fer una inserció d'un registre a la taula de la base de dades activa amb els valor indicat pel paràmetre $campos , que és un array associatiu amb l'estructura següent : Array ("camp1"=>"valor1", "camp2"=>"valor2", "campN"=>"valorN") on camp1, camp2 i campN són diferents camps de la taula i valor1, valor2 i valorN representa el valor que es vol donar a aquests camps.
En el codi, es pot observa la instrucció, pròpia de PHP, ereg_replace (string1,string2,string3) que és una funció que retorna una cadena (string), resultat de cerca al string3 el patró del string1 i substituir-ho per la cadena string2.

La resta de mètodes de la classe WSFW_Db són molt semblants als ja comentats, es pot consultar el codi font al CD entregat amb aquest projecte a la carpeta del framework. També es pot veure tota la documentació del codi generada amb Php Documentor, a la carpeta Documentació.

6.5.2 Functions

Es tracta d'un fitxer (functions.php) on es recullen un conjunt de mètodes que poden resultar molt útils per treballar amb arrays associatius. Els mètodes principals implementats són :

- **function** array_union($campos, $cmp, $ws, $db)

  El seu objectiu és la unió de dos arrays associatius per un camp coincident.

- **function** array_inter($campos, $cmp, $ws, $db)

  El seu objectiu és la intersecció de dos arrays associatius per un camp coincident.

- **function** array_filters($args, $campos, $replace = **null**)

  El seu objectiu és filtrar els camps a un array associatiu.

- **function** array_filters_assoc($args, $campos, $replace = **null**)

  El seu objectiu és aplicar el filtre a un array associatiu.

- **function** array_replace_assoc($from, $to, $array, $assoc = **null**)

  El seu objectiu és la unió de dos arrays associatius per un camp coincident.

- **function** array_replace_assoc($from, $to, $array, $assoc = **null**)

  El seu objectiu és substituir els camps d'un arrays associatiu.

- **function** objectToArray($object)

  El seu objectiu és convertir un objecte a array associatiu. Aquesta funció l'utilitzarem força ja que en Php molts dels serveis webs cridats, ens retornen un objecte stdClass.

El codi font d'aquesta utilitat es troba al CD entregat amb aquest projecte a la carpeta del framework. També es pot veure tota la documentació del codi generada amb Php Documentor, a la carpeta Documentació.

6.6 Mòduls necessaris de PHP

Per defecte, a la primera instal·lació de php 5.3.5 per Windows, es carregen les extensions de php mostrades a la figura 6.21.

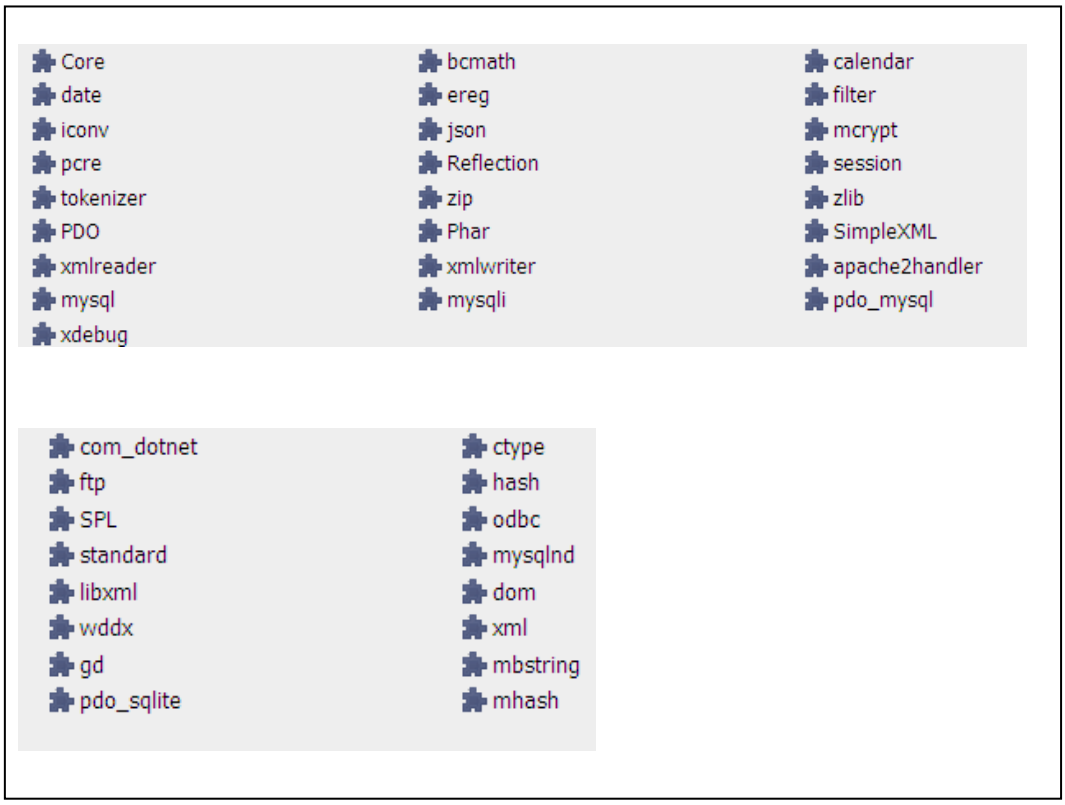

Figura 6.21 Informació de http://localhost

La informació de la figura 6.21, l'obtenim des de qualsevol navegador, es suposa que es tracta de l'ordinador on està funcionant el servidor web (WAMP) i que aquest té tots els serveis iniciats.

Les extensions de php concretes referents a protocols que s'utilitzaran en el framework WSFW són: libxml, json, soap i xmlrpc. Es pot observar que no estan carregades totes les que es precisen. Falten per carregar soap i xmlrpc. També s'observa que no està carregada l'extensió referent a seguretat ssl.

Per carregar les extensions necessàries només caldrà editar el fitxer de configuració php.ini i anular els comentaris a les línies on es fa referència a las seves dll corresponents.

En aquest cas, són : php_soap.dll (protocol Soap), php_xmlrpc (protocol Xml-Rpc) i php_openssl.dll (corresponent a temes de criptografia i accés per https). Els comentaris en el fitxer php.ini apareixen després d'un punt i coma, provocant que cap directiva o comanda posterior en una mateixa línia tingui cap efecte. Veure la Figura 6.22 (sota).

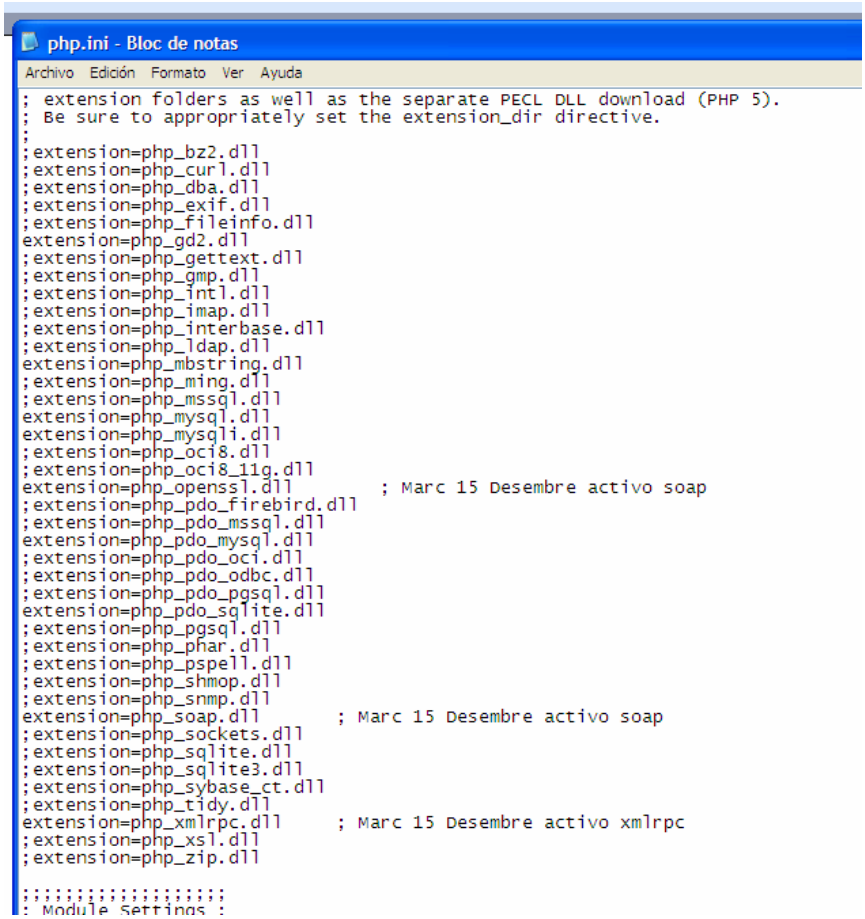

Figura 6.22 Fitxer de configuració de PHP

Es pot veure com simplement hem esborrat el punt i coma de les línies corresponent a les extensions que es volen activar, per precaució, és aconsellable enregistrar amb un comentari la data de qualsevol canvi i la persona responsable d'ells.

Totes les extensions utilitzades de Php en el framework WSFW estan disponibles per als desenvolupadors de php a partir de la versió Php 5.0. En tot cas, suposant que s'hagués extraviat alguns dels fitxers dll necessaris, només caldria obtenir-los per Internet,de la

pàgina oficial, o qualsevol altre medi, i copiar-los a la carpeta d'extensions del servidor Wamp. La carpeta per defecte, en aquest cas és "C:\wamp\bin\php\php5.3.5\ext".

Finalment només s'ha de salvar el fitxer php.ini amb els canvis efectuats i reiniciar els serveis del servidor. Es torna a comprovar les extensions carregades i el resultat es pot veure a la figura 6.23, sota.

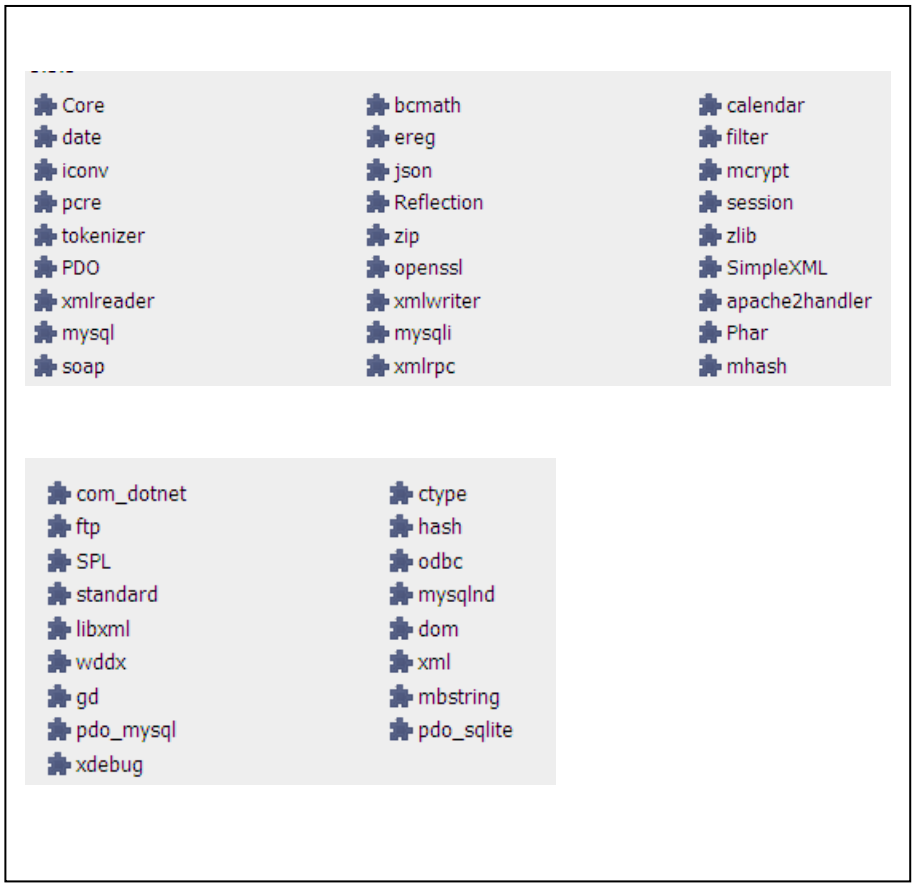

Figura 6.23 Informació de http://localhost

6.7 Jerarquia de fitxers de WSFW

L'estructura de fitxers del framework implementat es mostra a la figura 6.24, sota.

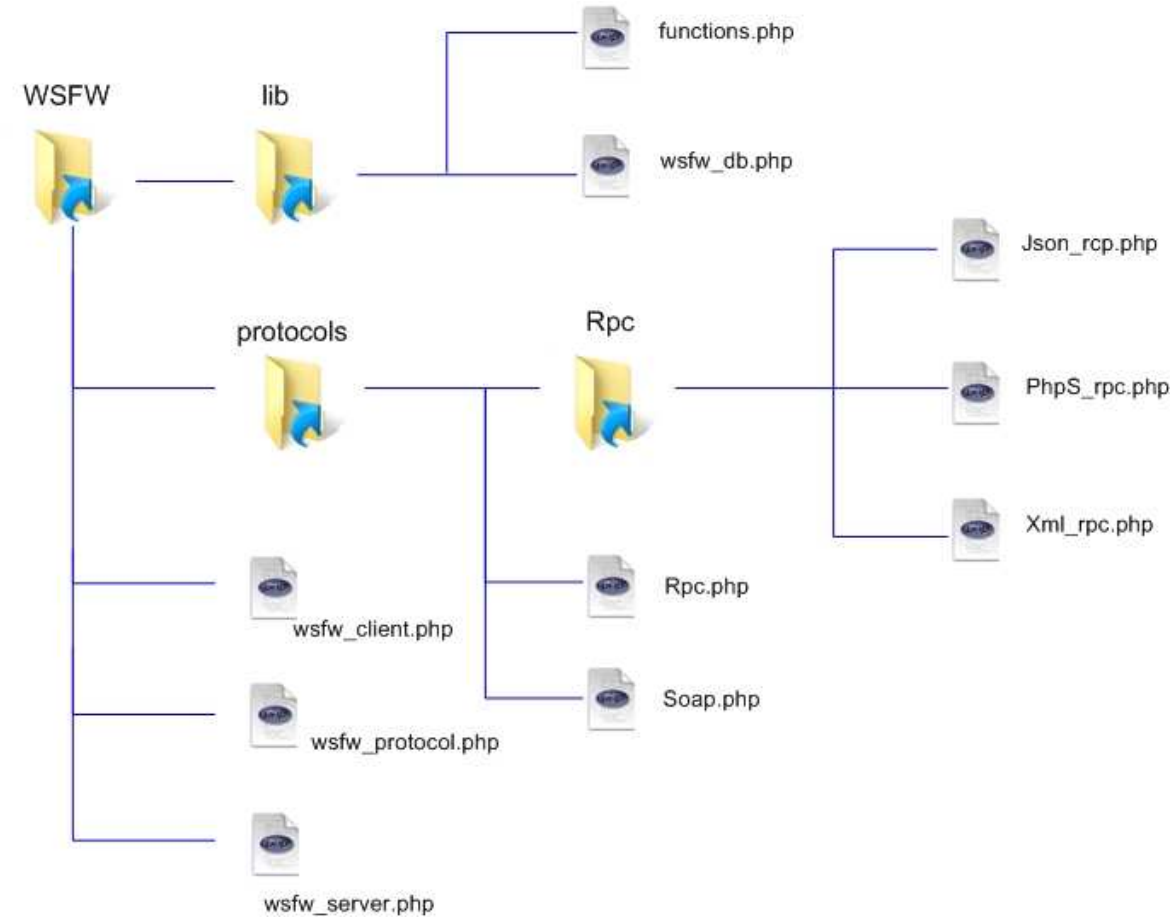

Figura 6.24 Jerarquia de fitxers del framework

Aquesta estructura, és la que realment té el framework, però pot ser modificada pel programador si ho considera oportú. Només tindrà que actualitzar les referències d'inclusió dintre dels fitxers php.

7. Planificació.

La planificació del projecte s'emmarca dins de la gestió de projectes i la seva realització és imprescindible. Gràcies a ella es pot quantificar el cost econòmic i temporal del projecte que es vol desenvolupar. L'estimació obtinguda permet conèixer el temps a invertir i el pressupost necessari pel desenvolupament del projecte. Qualsevol de les dos variables obtingudes pot ser la causa per desestimar o acceptar la realització del projecte.

7.1 Fases del projecte, descripció i "timing"

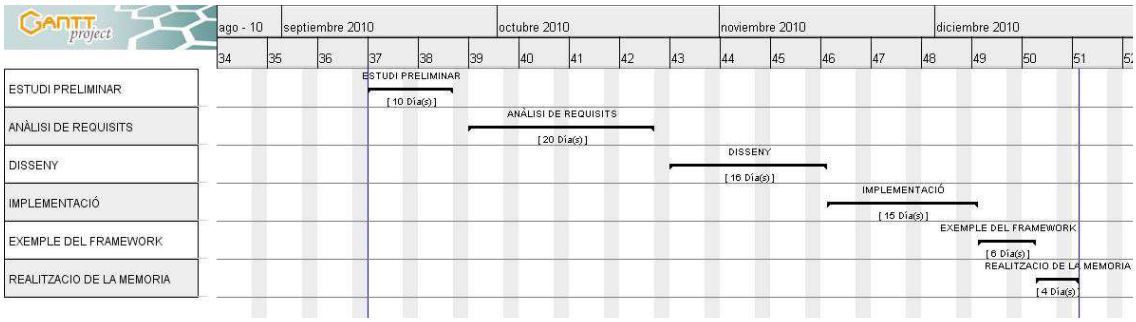

Figura 7.1 Gantt Projecte complet

Com es pot veure dalt, a la figura 7.1, s'ha subdividit el projecte en sis fases : Estudi preliminar, Anàlisi de requestis, Disseny, Implementació, Exemple del Framework i Realització de la Memòria. En aquesta figura també es pot veure els dies totals que consumeixen cadascuna d'elles, la data d'inici del projecte (començament de la setmana 37 de 2010 – 13/09/2010) i la data final (primer dia de la setmana 51 de 2010 – 20/12/2010).

Cal tenir en compte, que els recursos de personal assignats a cada fase és el mateix, un únic recurs que és un Analista-Programador. També cal saber que l'horari d'aquest treballador és de dilluns a divendres, amb una dedicació diària de quatre hores, treballant tots els festius.

Segons les premisses de disponibilitat de recursos de personal, es pot concloure que el projecte es podrà realitzar en 70 dies laborals, que es correspon a 280 hores de feina.

Per facilitar el desenvolupament, seguiment i control del projecte, aquestes fases es desglossaran en un conjunt de tasques més petites i/o específiques.

A continuació es farà una breu explicació i es detallarà cadascuna de les fases.

Estudi Preliminar

En aquesta fase es realitza un estudi previ de l'estat de l'art del serveis web, després la planificació del projecte i finalment es documenta tota la informació obtinguda.

A la figura 7.2, sota, es pot veure com aquesta fase es compon de tres tasques: Investigació Serveis Web, Planificació i Pressupost i documentar.

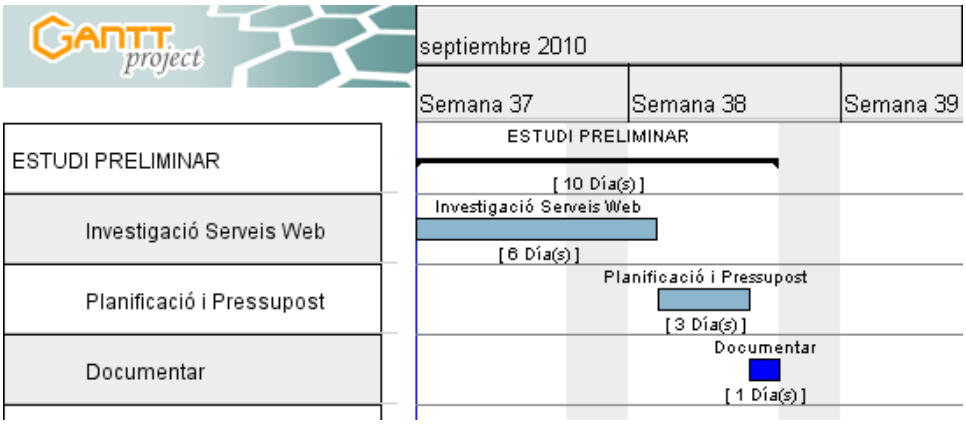

Figura 7.2 Gantt ESTUDI PRELIMINAR

Es pot observar el temps dedicat a cadascuna de les tasques, així com la data d'inici i fi de cadascuna. En resum, el temps total invertit és de 10 dies que es correspon a 40h laborals de l'únic recurs assignat (Analista-Programador).

Anàlisi de requisits

En aquesta fase, es continua aprofundint en la investigació dels serveis web, s'identifiquen tots els requisits del projecte i es realitzen els diagrames de casos d'ús més significatius. Finalment cal documentat tota la informació, tant la obtinguda com la generada.

A la figura 7.3, sota, es pot veure com aquesta fase es compon de quatre tasques: Investigació Serveis Web, Planificació i Pressupost i documentar.

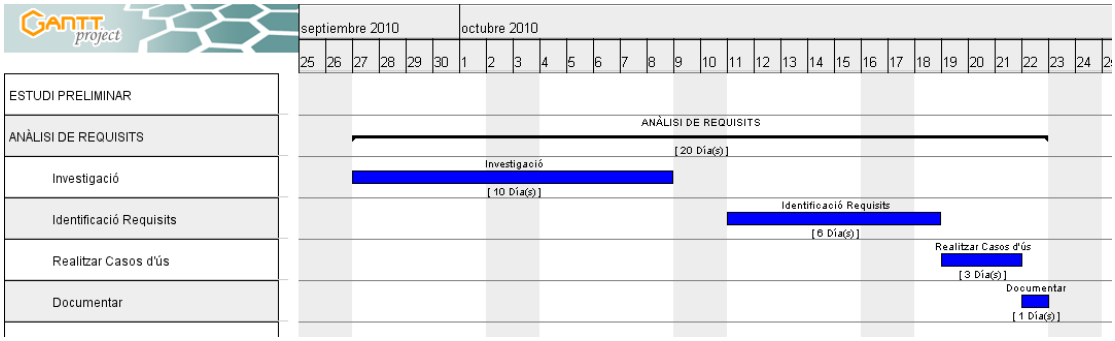

Figura 7.3 Gantt ANÀLISI DE REQUISITS

Es pot observar el temps dedicat a cadascuna de les tasques, així com la data d'inici i fi de cadascuna. En resum, el temps total invertit és de 20 dies que es correspon a 80h laborals de l'Analista-Programador.

Disseny

En aquesta fase ja es té l'anàlisi del sistema i ja es pot començar el desenvolupament. El disseny del sistema comporta analitzar com serà l'arquitectura i l'estructura, dissenyant cadascuna de les parts o mòduls que el conformen i llurs interaccions. S'establirà el disseny general del framework, es realitzarà el Diagrama de classes del Domini i finalment es documentarà tota la informació.

A la figura 7.4, sota, es pot veure com aquesta fase es compon de quatre tasques: Establir Disseny general, Realitzar Diagrama de classes del Domini i finalment documentar.

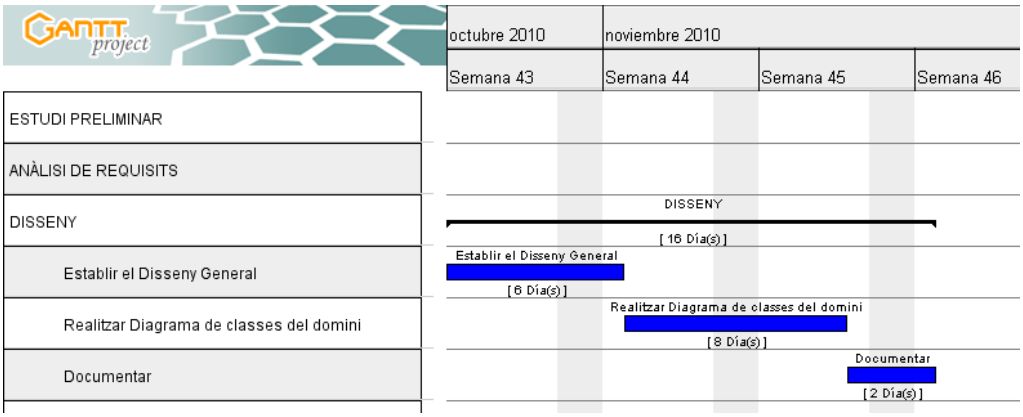

Figura 7.4 Gantt DISSENY

Es pot observar el temps dedicat a cadascuna de les tasques, així com la data d'inici i fi de cadascuna. En resum, el temps total invertit és de 16 dies que es correspon a 64h laborals de l'Analista-Programador.

Implementació

En aquest punt, ja s'ha fet tot l'Anàlisi i el Disseny del Framework i és el moment de realitzar la implementació de totes les funcionalitats definides. Primerament s'haurà de decidir qüestions tècniques pròpies del llenguatge emprat, seguidament s'escriurà el codi font, es realitzaran les proves necessàries i finalment es documentarà tota la informació de la fase.

A la figura 7.5, sota, es pot veure com aquesta fase es compon de quatre tasques: Estudi i decisions d'implementació, Codificació en PHP del Framework WSFW, Proves i Documentació.

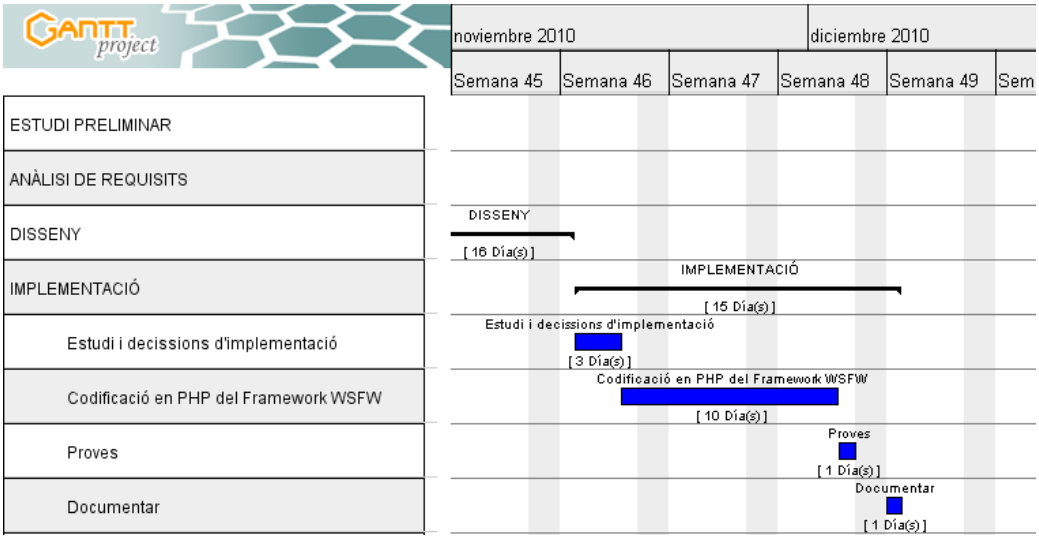

Figura 7.5 Gantt IMPLEMENTACIÓ

Es pot observar el temps dedicat a cadascuna de les tasques, així com la data d'inici i fi de cadascuna. En resum, el temps total invertit és de 15 dies que es correspon a 60h laborals de l'Analista-Programador.

Exemple del Framework

En aquesta fase es vol demostrar la utilitat del framework que ja està implementat i provat. Per tal de poder mostrar el funcionament del sistema, s'implementarà una aplicació creada amb ell. Aquesta aplicació (CRE) serà un aplicació servidora d'un servei web. Amb l'objectiu de poder demostrar el funcionament d'aquesta aplicació, es crearà una altra, molt simple, que consumeixi el servei web. S'aprofitarà que tenim aquestes dues aplicacions implementades i funcionant per realitzar proves que completaran les ja realitzades en la fase anterior.

A la figura 7.6, sota, es pot veure com aquesta fase es compon de quatre tasques: Codificació PHP Aplicació amb WSFW (CRE), Codificació PHP aplicació Client CRE, Proves aplicacions CRE i Client CRE i per últim, Documentar.

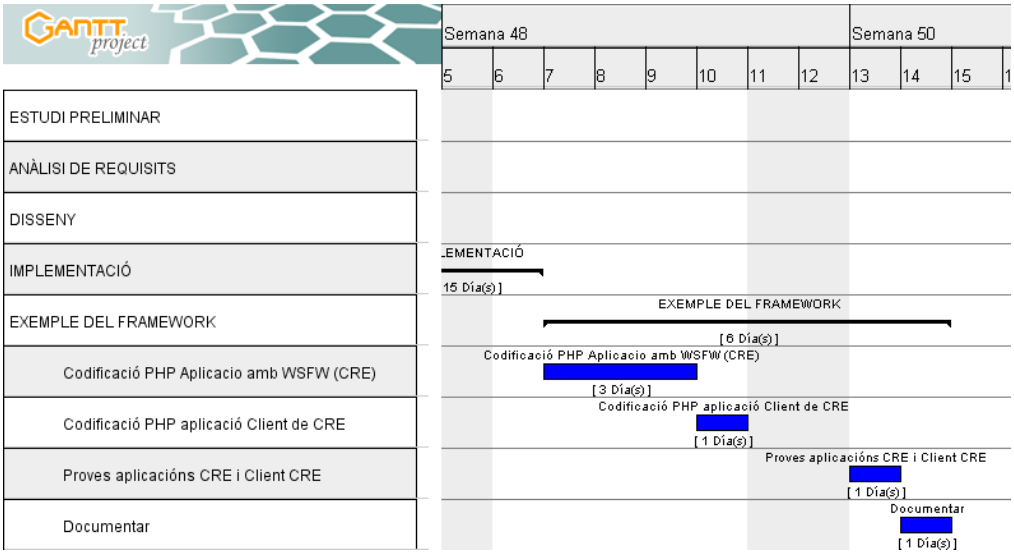

Figura 7.6 Gantt EXEMPLE DEL FRAMEWORK

Es pot observar el temps dedicat a per cadascuna de les tasques, així com la data d'inici i fi de cadascuna. En resum, el temps total invertit és de 6 dies que es correspon a 24h laborals del Analista-Programador.

Realització de la Memòria.

Aquesta fase es correspon a l'elaboració d'aquest document, el projecte final de carrera. Gràcies a la documentació realitzada en totes les fases anteriors, s'ha pogut executar ràpidament les tasques que la conformen: Integració de tota la documentació , Revisió i Modificació final.

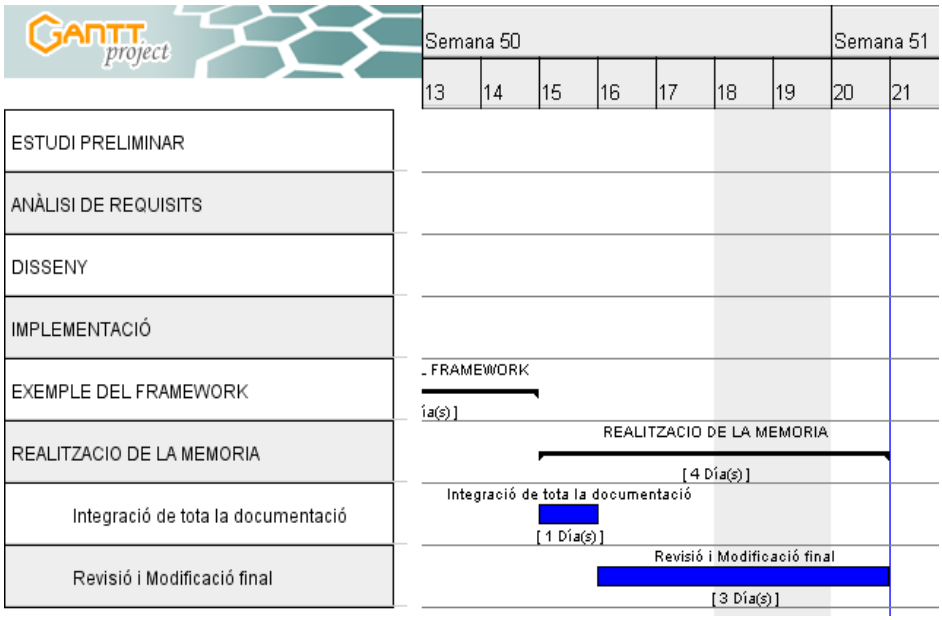

Figura 7.7 Gantt REALITZACIO DE LA MEMORIA

Es pot observar el temps dedicat a cadascuna de les tasques, així com la data d'inici i fi de cadascuna. En resum, el temps total invertit és de 3 dies, que es correspon a 12h laborals.

7.2 Estudi econòmic

El cost total del projecte, el pressupost, és l'altre variable estimada de la planificació, cabdal per decidir l'acceptació o no de qualsevol projecte. Aquesta variable contempla tots els costos que són imputables a la realització del projecte, es divideix en: costos de software, de hardware i costos de personal.

Costos de Software

No hi ha costos de Software imputables a aquest projecte, el cost es pot considerar de 0€. Tot el programari utilitzat per el desenvolupament del projecte és software lliure, amb diferents tipus de llicència GNU/GPL o semblants. A més, la majoria són "Open Source", és a dir, de codi obert. L'única excepció que podem trobar és el sistema operatiu del ordinador utilitzat, es tracta d'un Windows Xp Home edition. De totes maneres, no s'imputa cap cost en aquest apartat, ja que es tracta d'una llicencia OEM que es subministra juntament amb el hardware per part del fabricant.

Costos de Hardware

Els elements hardware emprats per realitzar el projecte són:

I. Ordinador portàtil. Cost de compra 335 €.
   Es tracta d'un HP Mini Note 2140. Les seves característiques principals són: 1GB de memòria RAM, processador Intel Atom N270 a 1,6 Ghz i pantalla TFT 11".

II. Teclat i Ratolí sense fils. Es tracta d'un pack de Logitech. Cost de compra 25 €.

III. Monitor LED 24". És un SAMSUNG BX 2335. Cost de compra 170 €.
   Es tracta d'un HP Mini Note 2140. Les seves característiques principals són: 1GB de memòria RAM, processador Intel Atom N270 a 1,6 Ghz i pantalla TFT 12".

El cost total de compra del hardware (sense iva), és 530€. Considerant una amortització lineal de 3 anys d'aquest elements hardware, i imputant el seu ús a quatre mesos

aproximadament, s'obté que el pressupost total en concepte de hardware del projecte és, aproximadament, 59€. El càlcul realitzat és : (Total preu compra / TA) * TU = (530/36) * 4 on TA és el temps d'amortització i TU és el temps d'ús del hardware, ambdós expressats en unitats de temps mes.

Costos de Personal

El projecte s'ha realitzat amb un únic recurs, un Analista-Programador. Per simplificar, i obeint a la realitat del projecte, no s'ha discriminat quan aquest recurs exerceix el rol de Analista/Cap de projectes o el de Programador.

El càlcul de la tarifa horària d' Analista s'ha comptabilitzat a 45€/h i la de programador a 25€/h. La tarifa única que s'imputarà en aquest projecte és de 35€/h que és la mitjana de les tarifes horàries del dos perfils esmentats.

Es pot concloure que el pressupost total, imputable al personal, és de 9.800€. El càlcul realitzat és el producte de la tarifa horària establerta i les hores previstes de dedicació al projecte (calculades al apartat 7.1).

Finalment, s'obté el Pressupost total previst del projecte fent la suma directa del costos de software, hardware i personal imputables al projecte:

Software	0 €
Hardware	59€
Personal	9.800€
==========	==========
TOTAL	9.859€

8.Proves.

En aquest projecte no s'ha planificat una fase de proves independent, sinó que s'han integrat dintre de la fase de Implementació i Exemple del Framework. Al tractar-se d'un framework, una eina per ajudar a desenvolupar aplicacions, ha resultat de vital importància per tal de aplicar els test unitaris, la creació de les dues aplicacions del apartat següent, apartat 9, Exemple pràctic del Framework .

Principalment s'han realitzat proves unitàries i d'integració. Durant la fase d'implementació del framework s'han realitzat proves unitàries per cada component individualment i a mesura que s'anaven implementant nous components s'ha pogut anar realitzant les proves d'integració. En cada test unitari s'ha comprovat que tots els cassos possibles quedessin coberts dins del codi.

Finalment, després de realitzar les dues aplicacions d'exemple, s'ha pogut realitzar les proves unitàries i d'integració sobre elles.

Per realitzar les proves s'ha utilitzat una eina d'entorn lliure, el PHP Unit. A l'apartat del capítol 6, concretament a 6.2.1, ja s'ha fet una breu explicació sobre aquest Framework que serveix per realitzar suites de test. S'han detallant alguns mètodes concrets, alguns d'ells emprats per codificar el nostre fitxer de test.

Per tal de poder utilitzar eficientment aquesta eina, s'ha tingut que codificar un fitxer propi en php que realitzes totes les proves que desitjaven (test.php), podem localitzar-lo al CD que acompanya aquesta memòria, juntament amb la documentació en format HTML resultant de les proves efectuades.

A continuació es mostrarà una part d'aquest fitxer, i un detall d'un mètode per realitzar una prova concreta. Veure Figura 8.1 i 8.3 a la pàgina següent.

```
6   class MyTest extends PHPUnit_Framework_TestCase {
7           protected $fixture;
8
9           public function testSoapServer() {...}
21
22          public function testJson_rpcServer() {...}
30
31          public function testPhpS_rpcServer() {...}
39
40          public function testXml_rpcServer() {...}
48
49          public function testSoapClient() {...}
61
62          public function testPhpS_rpcClient() {...}
75
76          public function testXml_rpcClient() {...}
86
87          public function testJson_rpcClient() {...}
100
101
102         public function testCreuRojaList() {...}
107
108
109         public function testCreuRojaInfo() {...}
114
115         protected function setUp() {
116                 $this->resultado["informacio_platja"] = unserialize(stripslashes(htmlspecialchars_decode
117                                                         (file_get_contents("tests/informacio_platja.result"))));
118                 $this->resultado["beaches_list"] = unserialize(stripslashes(htmlspecialchars_decode
119                                                         (file_get_contents("tests/beaches_list.result"))));
120         }
121
122
```

Figura 8.1 Fitxer de proves

Tal i com es pot veure a la figura 8.1, dalt, a la línia 6 apareix la classe MyTest que estén la classe del Framework PHPUnit. S'han preparat 10 proves, i es comproven els mètodes **informacio_platja(...)** i **beaches_list()**. A la línia 115 es veu com s'ha utilitzat la funció protegida **setUp()** per preparar els resultats esperats i compartir-los per totes les proves que es fan. A les línies 116 i 118 es pot veure que els resultats esperats estan en format Php serialitzat i a més que estan emmagatzemats en dos fitxers externs : informacio_platja.result i beaches_list.result. S'han utilitzat fitxers de text pla externs, en comptes d'escriure el resultat en el propi fitxer php perquè contenen milers de caràcters que representen la informació obtinguda del servei web en format Php serialitzat (veure figura 8.2).

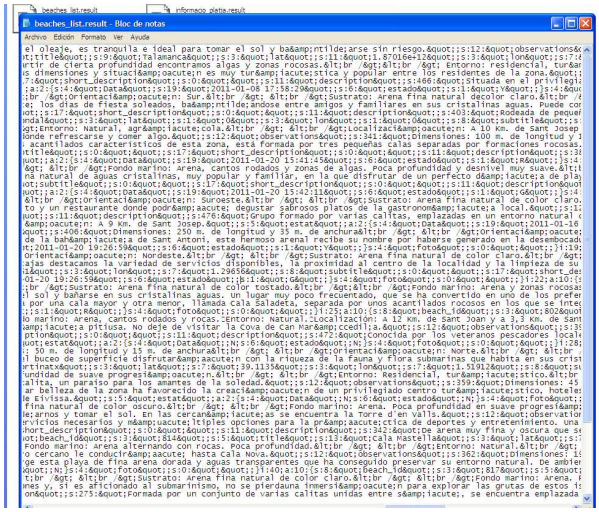

Figura 8.2 Fitxer resultat extern

A la figura 8.3, es veu la codificació d'un test dels deu implementats.

```
101
102         public function testCreuRojaList() {
103                 $creuRoja = new creuRoja();
104                 $response = $creuRoja->beaches_list();
105                 $this->assertEquals(Array(), array_diff_assoc($this->resultado["beaches_list"],$response));
106         }
107
108
109         public function testCreuRojaInfo() {...}
114
115         protected function setUp() {
116                 $this->resultado["informacio_platja"] = unserialize(stripslashes(htmlspecialchars_decode
117                                                         (file_get_contents("tests/informacio_platja.result"))));
118                 $this->resultado["beaches_list"] = unserialize(stripslashes(htmlspecialchars_decode
119                                                         (file_get_contents("tests/beaches_list.result"))));
120         }
121
```

Figura 8.3 Codificació prova testCreRojaList

La funció **testCreuRojaList()** crea un objecte creuRoja i fa la crida al mètode de la classe que retorna un array associatiu amb tota la informació de totes les platges. A la línia 105 es fa la asserció que compara l'array obtingut amb el resultat que ens preparat amb la funció protegida **setup()**.

A la figura 8.4, es veu la codificació d'un altre test dels deu implementats.

```
76          public function testXml_rpcClient() {
77                  $client = new WSFW_Client(
78                          array(
79                                  "url" => 'http://localhost/CRE/index.php?Xml_rpc',
80                                  "protocol" => 'Xml_rpc'
81                          )
82                  );
83                  $response = $client->call('informacio_platja', array('beach_id' => '302'));
84                  $this->assertEquals(Array(), array_diff_assoc($this->resultado["informacio_platja"],$response));
85          }
```

Figura 8.4 Codificació prova testXml_rpcClient

La funcio **testXml_rpcClient()** crea un objecte client de la classe client del framework, concretament es pot veure que el client realitzarà la petició mitjançant el protocol Xml_rpc.

A la línia 83 es guarda en la variable $response el resultat de la crida al mètode informacio_platja() que és un mètode de la classe creuRoja que retorna un array associatiu amb tota la informació disponible de la platja indicada per el paràmetre d'entrada (beach_id => 302).

A la línia 84 es fa la asserció que compara l'array obtingut amb el resultat que ens preparat amb la funció protegida **setup()** (corresponent a $resultado[informacio_platja]). Aquest resultat preparat conté la resposta en format php serialitzat corresponent al array retornat per informacio_platja() quan se li ha passat la platja amb identificador 302. A la figura 8.5 es pot veure aquest array preparat deserialitzat que correspon a la platja amb identificador 302.

```
Array
(
    [beach_id] => 302
    [title] => Talamanca
    [lat] => 1.87016e+12
    [lon] => 1.46028
    [subtitle] =>
    [short_description] =>
    [description] => Su cercanía al casco urbano de Eivissa la hace especialmente
práctica. Larga y bien protegida del viento, se encuentra en el interior de una bahía
cerrada. Poco masificada, dispone de rampas de acceso para minusválidos y es adecuada
para niños por su escasa profundidad. Cuenta con una amplia oferta de servicios y es
ideal para la práctica de la vela ligera.
    [observations] => Dimensiones: 930 m de longitud y 30 m de anchura.

 Orientación: sureste. Vientos del sur, de mar a tierra.

 Composición (sustrato): arena fina natural.

 Fondo marino: predomina la arena; a partir de cierta profundidad encontramos algas y
zonas rocosas.

 Entorno: residencial, turístico.

Localización: a 1,8 km de la ciudad de Eivissa

    [estat] => Array
        (
            [Data] => 2011-01-08 18:00:10
            [estado] => G
        )

    [foto] => Array
        (
            [upload_date] => 14 April 2008
            [owner_name] => Dejah
            [photo_id] => 9363990
            [longitude] => -72.607527
            [height] => 375
            [width] => 500
            [photo_title] => Marble Cave
            [latitude] => -46.647138
            [owner_url] => http://www.panoramio.com/user/947917
            [owner_id] => 947917
            [photo_file_url] => http://mw2.google.com/mw-
panoramio/photos/medium/9363990.jpg
            [photo_url] => http://www.panoramio.com/photo/9363990
        )

)
```

Figura 8.5 Array comparat

9. Exemple pràctic del framework.

En aquest capítol, es realitza un exemple complet per demostrar la utilitat del framework implementat. Es tracta d'un exemple fictici centrat en les qüestions pròpies que ens afecten: els serveis web.

El desenvolupador de l'aplicació utilitzarà el framework, WSFW, per il·lustrar el seu funcionament. En aquest exemple el desenvolupador realitzarà una aplicació per la Creu Roja d'Eivissa, veure més detall veure apartat següent(9.1).

Tal i com s'ha comentat a l'apartat 2.2.2 del capítol Introducció als Serveis Web, aquests es centren en el intercanvi d'informació màquina – màquina, per tant, per poder veure el resultat del desenvolupament de la aplicació de la Creu Roja, es necessita crear un altre aplicació amb entorn visual que mostri els resultats del Servei Web creat. Aquesta aplicació serà una pàgina web que pertany a un associat o col·laborador de la Creu Roja. Aquesta pàgina simplement consumirà el Servei Web ofert per mostrar la informació formatejada amb un aspecte visual agradable i funcional. Es detallarà aquesta aplicació client a l'apartat 9.2.

9.1 Aplicació Servei Web Creu Roja Eivissa (CRE)

9.1.1 Introducció

La Creu Roja d'Eivissa, ha decidit oferir un servei web que proveeixi informació sobre l'estat de les platges a l'illa. La informació clau a oferir és l'estat actual de les platges, a saber: senyera vermella – Prohibició de banyar-se , senyera groga – Precaució i senyera verda – Bany autoritzat. La creu roja és l'encarregada d'informar diàriament l'estat actual de les platges de tota l'illa i actualment ja disposa de la infraestructura necessària per proporcionar aquesta informació. Inicialment, la Creu Roja, no té clar a quins socis o simpatitzants donarà accés al servei web, però ja entreveu l'oportunitat d'oferir informació addicional de les platges, a més de la disponibilitat per realitzar les activitats pròpies de bany. Gracies a les converses mantingudes amb el personal de la Creu Roja i amb un grup

d'associats i col·laboradors d'aquesta, s'ha decidit quina és la informació interessant a oferir mitjançant el servei web de la aplicació, en endavant CRE.

La informació susceptible d' interès és:

- Nom de la platja.
- Localització geogràfica, és a dir, la longitud i latitud on està situada la platja.
- Estat. Indica l'ultima situació registrada a la platja, pot ser senyera vermella, groga o verda.
- Últim registre. Indica la data en que s'ha realitzat l'ultima comprovació de l'estat en que es troba una platja. Encara que normalment la Creu Roja es compromet a realitzar la catalogació del estat de les platges de forma diària, no sempre és possible l'accés al registre en temps real.
- Imatge de la platja.

També s'ha constatat que els llocs web dels socis i col·laboradors que han mostrat interès per el futur servei web de la CRE, ja estan fent servir serveis web però utilitzant diferents protocols.

9.1.2 Estudi realitzat

Finalment s'ha pogut definir quina informació es desitja servir per l'aplicació CRE. Realitzant l'estudi tècnic ens trobem que la Creu Roja gairebé no té informació sobre les platges susceptible d'oferir en un servidor d'Internet. Actualment té un servidor connectat a Internet amb una base de dades MySql que bàsicament conté el nom de les platges i l'estat d'aquestes a una determinada data. Es planteja crear una nova base de dades MySql que contingui tota la informació desitjada. Inicialment el plantejament és extreure les dades del sistema d'informació propi de la Creu Roja i carregar-les a la nova base de dades. El problema principal que es presenta és la no disponibilitat de tota la informació desitjada, apart del cost addicional de crear una base de dades nova i efectuar la carrega de les dades.

Sortosament, gracies als contactes de la Creu Roja amb el Consell Insular d'Eivissa, s'ha aconseguit accés a un Servei Web ofert pel portal de turisme oficial d'Eivissa. Es tracta de la web http://www.ibiza.travel.

Aquest servei web del Consell Insular d'Eivissa, en endavant CIE, ofereix informació detallada de les platges de l'illa mitjançant el protocol SOAP. Per tant es decideix no crear una base de dades nova sinó servir la informació desitjada de les platges dinàmicament actuant com a client del servei web del portal de turisme i afegir la informació de l'estat de les platges consultant la base de dades local de la Creu Roja. Amb l'objectiu de poder relacionar la informació de les platges de la base de dades local amb la informació del servei web del CIE simplement s'ha afegit un identificador a una de les taules que correspon al identificador de la platja que utilitza CIE.

En aquest punt es detalla tota la informació disponible que es podrà servir amb el servei web encarregat per la Creu Roja:

- **Nom de la platja.**

(Disponible a la base de dades local de la Creu Roja i al servei web de CIE)

- **Localització geogràfica,** és a dir, la longitud i latitud on està situada la platja.

(Disponible com a client del servei web de CIE)

- **Estat.** Indica l'ultima situació registrada a la platja, pot ser senyera vermella, groga o verda.

(Disponible només a la base de dades local de la Creu Roja)

- **Últim registre.** Indica la data en que s'ha realitzat l'ultima comprovació de l'estat en que es troba una platja. Encara que normalment la Creu Roja es compromet a

realitzar la catalogació del estat de les platges de forma diària, no sempre és possible l'accés al registre en temps real.

(Disponible només a la base de dades local de la Creu Roja)

- **Descripció de la platja.** El servei web ofert pel CIE inclou una breu descripció de totes les platges.

(Disponible com a client del servei web de CIE)

- **Observacions de la platja.** El servei web ofert pel CIE inclou unes observacions de la majoria de les platges que solen indicar l'amplada i la llargada en metres.

(Disponible com a client del servei web de CIE)

- **Imatge de la platja.**

(No disponible)

En aquest instant es té tota la informació que es desitjava oferir a l'aplicació CRE, a excepció de la imatge de la platja. Aquest problema s'ha resolt gracies a un servei web molt conegut disponible a Internet, es tracta de Panoramio. Aquest servei web ofereix, de forma gratuïta, imatges properes a un punt geogràfic mitjançant les coordenades latitud i longitud, informació de la que si es disposa a traves de CIE.

S'ha estudiat les diverses opcions de l'API de Panoramio i s'ha constatat que pot servir informació mitjançant json. També s'ha constatat que el protocol Json-Rpc de Panoramio no segueix cap estàndard reconegut, és propi de ell.

L'API de Panoramio es pot consultar a http://www.panoramio.com/api/data/api.html

En aquest punt, ja s'ha aconseguit tota la informació necessari a servir, però finalment la CRE ha decidit que vol oferir el servei web amb diferents protocols i així atendre al màxim nombre de socis i col·laboradors.

Gracies a WSFW no haurà cap esforç addicional en la implementació necessària per escometre aquest últim requisit (multiprotocol).

9.1.3 Base de dades local de CRE.

A continuació, s'expliquen les dues taules que interessen de la base de dades local de l'aplicació de la Creu roja. Es tracta d'una base de dades MySQL ja existent. Com ja s'ha comentat anteriorment, s'afegirà un camp a la taula playas (id_beach) que és l' identificador de les platges propi del Consell Insular D'Eivissa.

Taula playas

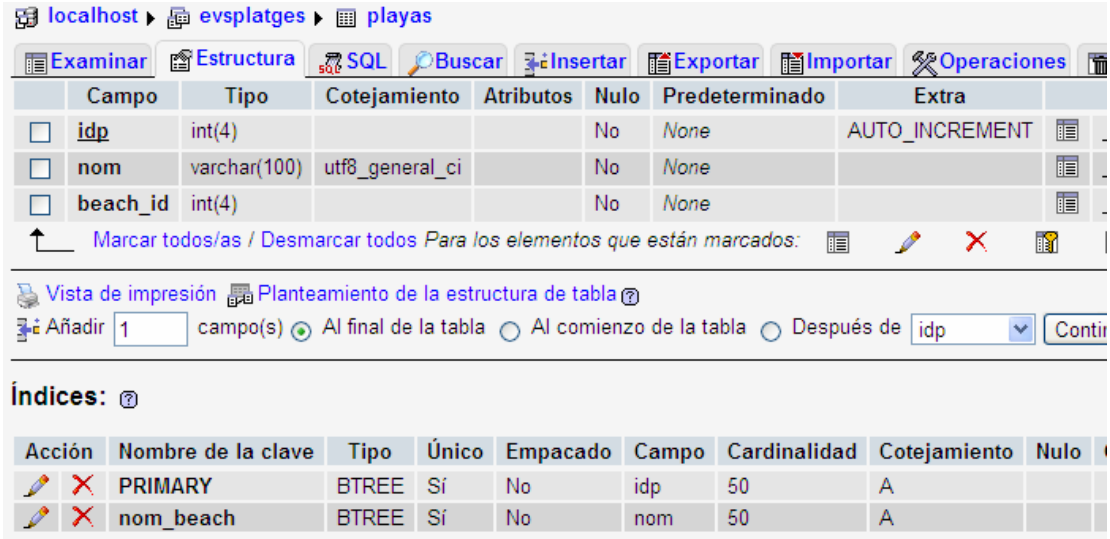

Figura 9.1 Taula playas

La taula de la figura 9.1 (dalt), té com principal objectiu emmagatzemar les dades de totes les platges d'Eivissa.

Els atributs d'aquesta taula són:

- idp : és un camp enter auto numèric.
- nom : és un camp de tipus varchar de 100 caràcters.
- bach_id: és camp de tipus enter.

El idp indica el nombre total de platges a la base de dades. És la clau primària d'aquesta taula.

El camp nom, és el nom amb el que es coneix la platja.

El camp beach_id indica l'identificador de la platja, com el primer camp, però en aquest cas es tracta d'un identificador d'una aplicació externa a Creu Roja, el servei web de CIE.

Taula lectures

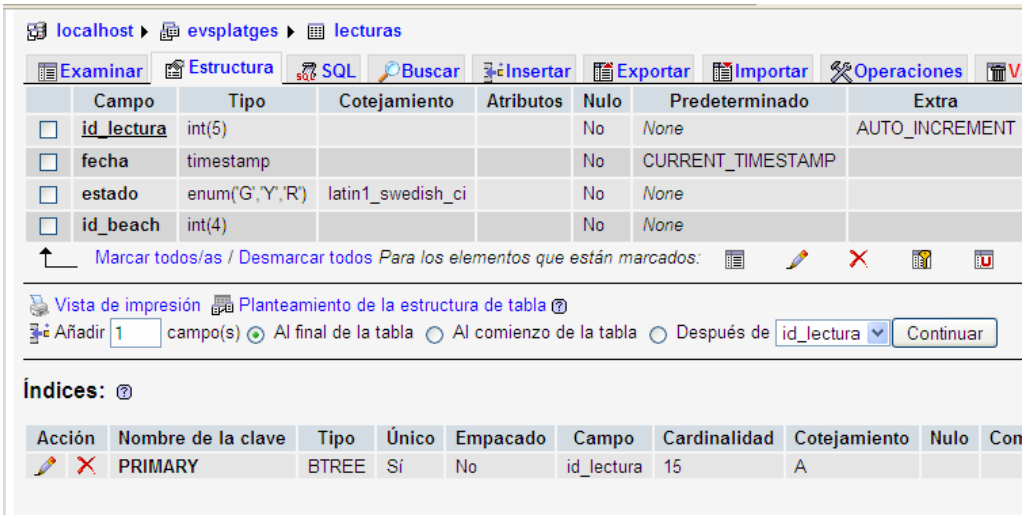

Figura 9.2 Taula lecturas

La taula de la figura 9.2, té com principal objectiu emmagatzemar l'estat de les platges en una data determinada. El personal de la Creu Roja que fa la inspecció física de la platja, té els mitjans necessaris per inserir la informació en aquesta taula, normalment en temps real.

Els atributs d'aquesta taula són:

- id_lectura : és un camp enter auto numèric.
- fecha : és un camp de tipus data, que per defecte agafa l'hora actual del sistema.
- estado: és un camp de tipus enum accepten els valors G,Y,R.
- id_beach: és un camp enter que no accepta valors nuls.

El id_lectura indica el nombre total de lectures realitzades a les platges d'Eivissa. És la clau primària d'aquesta taula.

El camp fecha ens indica quan s'ha realitzat la lectura del estat d'una determinada platja.

El camp estat indica la avaluació que fa la Creu Roja de la platja. G vol dir apte per el bany, Y vol dir precaució i R prohibit banyar-se.

El camp id_beach és un identificador de la platja i és clau forana. Es correspon al camp idp de la taula playas.

9.1.4 Desenvolupament del Servei Web CRE.

Després de l'estudi realitzat en els apartats anteriors, la implementació sembla obvia utilitzant el framework desenvolupat al projecte, WSFW.

La implementació es resumeix en:

a) Crear la classe creuRoja, amb tots el mètodes/operacions que desitgem oferir.

b) Crear dos clients de serveis web externs i obtenir la informació desitjada. Aquests dos serveis web externs de tercers són Panoramio i el servei web del CIE. Per servir la informació utilitzen respectivament els protocols Json-Rpc propi i SOAP.

c) Crear finalment el servei web CRE (Servidor) per oferir tota la informació desitjada als socis i simpatitzants de la Creu Roja. Inicialment es dona accés al públic en general sense cap restricció d'accés ja que es consideren totes les dades servides de interès públic.

S'ha decidit crear una classe Panoramio per implementar el protocol Json-Rpc propi de Panoramio i d'aquesta manera poder reutilitzar-ho en altres desenvolupaments de Serveis Web. Vegem el codi amb detall.

```
11  */
12  class Panoramio extends Json_rpc {
13
14
15      public function call($method, $params) {
16                  $this->request_url .= "?".http_build_query($params);
17                  $data = $this->rpc_call();
18                  return $data;
19          }
20
21  }
```

Figura 9.3 Codificació php Classe Panoramio

Simplement s'ha estès la classe Json_rpc del nostre framework i s'ha sobreescrit la funció **call()** ja que el Panoramio no envia la petició codificada en JSON, sinó que utilitza el mètode GET sense codificar per passar els paràmetres.

A continuació s'ha representat la classe creuRoja per facilitar la codificació en PHP.

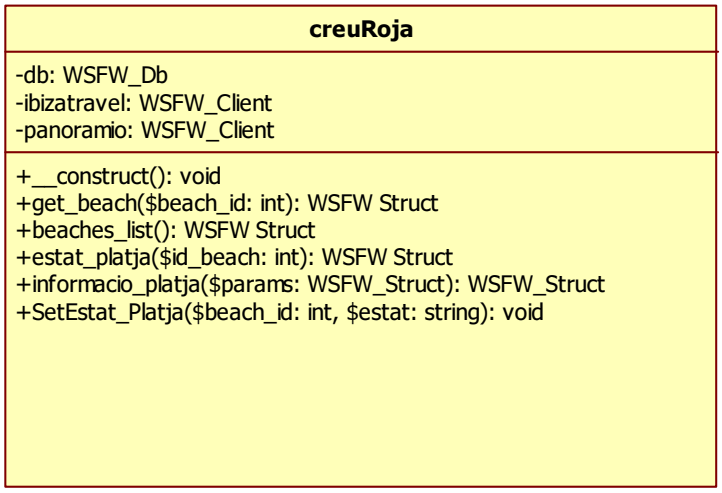

Figura 9.4 Classe CreuRoja

L'objecte WSFW_Db és una de les utilitats ofertes pel framework, serveix per realitzar la connexió a la base de dades local MySql de la Creu Roja i ofereix mètodes per realitzar diferents operacions SQL (INSERT, DELETE, UPDATE i SELECT).

A continuació, s'explica breument que fan els mètodes d'aquesta classe i es mostra la codificació php dels més significatius. Els mètodes són:

get_beach(..)

Aquesta funció retorna un array associatiu amb tota la informació d'una platja que ens pot donar el servei web del CIE. El paràmetre $beach_id és l'identificador de la platja corresponen al CIE. L'array associatiu retornat no conté informació sobre l'estat de la platja.

beaches_list()

Aquesta funció retorna un array associatiu amb tota la informació de totes les platges, que inclou l'estat de la platja i la data de l'ultima avaluació realitzada.

__construct()

S'encarrega d'inicialitzar la classe. Això implica realitzar la connexió amb la base de dades local de la CRE i crear els dos clients de serveis web externs.

```
function __construct(){
        $this->db = new WSFW_Db('localhost', '3306', 'usuari', 'pasword', 'evsplatges');
        $this->ibiza_travel = new WSFW_Client(array(
                "url" => 'http://webservice.ibiza.travel/soap/index.php?wsdl',
                "protocol" => "Soap")
        );
        $this->panoramio = new WSFW_Client(array(
                "url" => 'http://www.panoramio.com/map/get_panoramas.php',
                "protocol" => 'Panoramio')
        );
}
```

Figura 9.5 Codi font del constructor

Es pot veure a la figura 9.5 com a la segona línia crea una connexió amb la base de dades evsplatges que es troba a localhost, per accedir indica el nom d'usuari el password. A la següent línia, crea el client soap indicant com a paràmetres l'adreça del fitxer wsdl del servei web del CIE i el protocol de connexió. A la línia 7, crea el Client de Panoramio indicant com a paràmetres l'adreça web del servei a consumir i el protocol de connexió.

estat_platja(..)

Aquesta funció retorna un array associatiu amb l'estat i la data de l'ultima lectura efectuada d'una platja que té com identificador el paràmetre indicat.

```
function estat_platja($id_beach){

        $idp=$this->db->select_unico(array("idp"),"playas","beach_id='$id_beach'");
        $id=$idp['idp'];
        $cs=array ("max(fecha) as Data","estado");
        $tabla="lecturas,playas";$busqueda="lecturas.idp='$id'";
        $beach=$this->db->select_unico($cs, $tabla, $busqueda);
        return $beach;
}
```

Figura 9.6 Codi font estat_platja(..)

Es pot veure a la figura 9.6 com primer fa una consulta a la base de dades local de la CRE per obtenir l'identificador corresponent al camp idp de la taula playas, ja que el paràmetre d'entrada correspon al identificador de la platja del servei web CIE que té el mateix valor que el camp beach_id de la taula. Això és necessari perquè cal recordar que en aquesta taula hi ha emmagatzemats idp com identificador exclusiu de la CRE i beach_id que és l' identificador que usa el servei web del CIE.

informacio_platja(..)

Aquesta funció retorna un array associatiu amb tota la informació d'una platja. Realitza una petició a Panoramio per veure si aquest li retorna alguna foto corresponent a la platja indicada com a paràmetre d'entrada.

```
function informacio_platja($params) {
        $params = objectToArray($params);
        $data = $this->ibiza_travel->call("beach.get",
                array("beach_id" =>$params['beach_id'],"lang_id"=>"es", "id" => "IDEN"));


        $info=$data;

        $info['estat']=$this->estat_platja($params['beach_id']);


        $data = $this->panoramio->call(null, array(
                "set" => "full",
                "from" => "0",
                "to" => "1",
                "minx" => $info["lat"]-0.5,
                "miny" => $info["lon"]-0.5,
                "maxx" => $info["lat"]+0.5,
                "maxy" => $info["lon"]+0.5,
        ));
        if ($data["count"]) $info["foto"] = $data["photos"][0];

        return $info;
}
```

Figura 9.7 Codi font informacio_platja(..)

Podem veure a la figura 9.7 com recull tota la informació que ofereix d'una platja el servei web del Consell Insular d'Eivissa a la variable $data. Per fer la crida al servei, especifica l'operació desitjada, beach.get i l'indica una sèrie de paràmetres que són : l'identificador de la platja, el llenguatge de la resposta que és espanyol i un "id" que apareix a la imatge amb el valor IDEN. Aquest últim paràmetre, és l'identificador únic per a cada client que s'ha de sol·licitar al CIE per tenir accés al servei web.

SetEstat_platja(..)

Aquest mètode s'ocupa de inserir una nova lectura del estat de la platja a la taula lectures de la base de dades local de la Creu Roja.

```
function SetEstat_Platja($beach_id,$estat) {
        $cs=array ("id_beach"); $tabla="playas";$busqueda="id_beach=$beach_id";
        $platja=$this->db->select_unico($cs, $tabla, $busqueda, null, null);

        if ($platja) {
                $campos=array('id_beach'=>$beach_id,'estat'=>$estat);
                $this->db->insert("lecturas", $campos);
        } else
                echo "No existeix aquest identificador de platja";
}
```

Figura 9.8 Codi font SetEstat_platja(..)

El mètode rep com a paràmetres d'entrada l'identificador de la platja corresponent al identificador de CIE, per això realitza una consulta per obtenir l'identificador de la CRE, que es correspon amb el camp idp de la taula playas. Cas de trobar correspondència entre els identificadors, insereix una nova lectura a la base de dades de la CRE.

A continuació es mostra el codi font (del servidor) del Servei Web de la Creu Roja que es correspon amb el fitxer index.php de l'aplicació creada amb el framework del projecte, WSFW.

```
5
6   <?php
7
8   include ("WSFW/wsfw_server.php");
9   include ("creuRoja.php");
10
11  $servidor=new WSFW_Server(
12          array(
13                  "wsdl" => "index.wsdl"
14          )
15  );
16
17  $servidor->SetClass("creuRoja");
18
19  $servidor->Register();
20
21  ?>
```

Figura 9.9 Codi font servidor

A la figura 9.9 es pot veure la simplicitat de la creació del servidor del servei web. Es crea una instancia de WSFW_Server(...) amb certs paràmetres. En aquest cas s'indica que el fitxer wsdl es diu index.wsdl i es pot deduir que es troba en el mateix directori on es troba el servidor (index.php). Aquest servidor implementat és multiprotocol, en funció de la petició del client pot servir la informació en els quatre protocols suportats, exemple que s'implementa al següent apartat (9.2).

La única feina addicional que ha fet el programador ha estat el fitxer wsdl. En el cas de no haver tingut interès en oferir el servei web mitjançant SOAP, el programador només hauria d'indicar un null a la llista de paràmetres ("$servidor= **new** WSFW_Server(**null**)").

A la línia 17, de la figura 9.9, veiem com s'assigna la classe desenvolupada pel programador i dos línies després veiem com es registren tots els serveis que s'oferiran.

9.2 Client final del Servei Web Creu Roja Eivissa

El client final implementat, consumidor del servei web ofert per la Creu Roja, és simplement una pàgina web implementada en php que mostra amb una interfície funcional i atractiva la informació consumida del servei web creat amb WSFW.

Aquest Client, s'ha programat en llenguatge php per seguir el tarannà de tot el projecte però es podria haver realitzat en qualsevol llenguatge de programació que suportes qualsevol dels protocols que suporta WSFW: Json-Rpc, Xml-Rpc, Php Serialitzat-Rpc i SOAP. De fet l'aplicació client podria ser qualsevol, per exemple una aplicació d'escriptori, implementada amb Visual Basic 6.0.

A continuació es mostra l'aplicació client tal com es veu a un navegador web i posteriorment s'explica com es comporta.

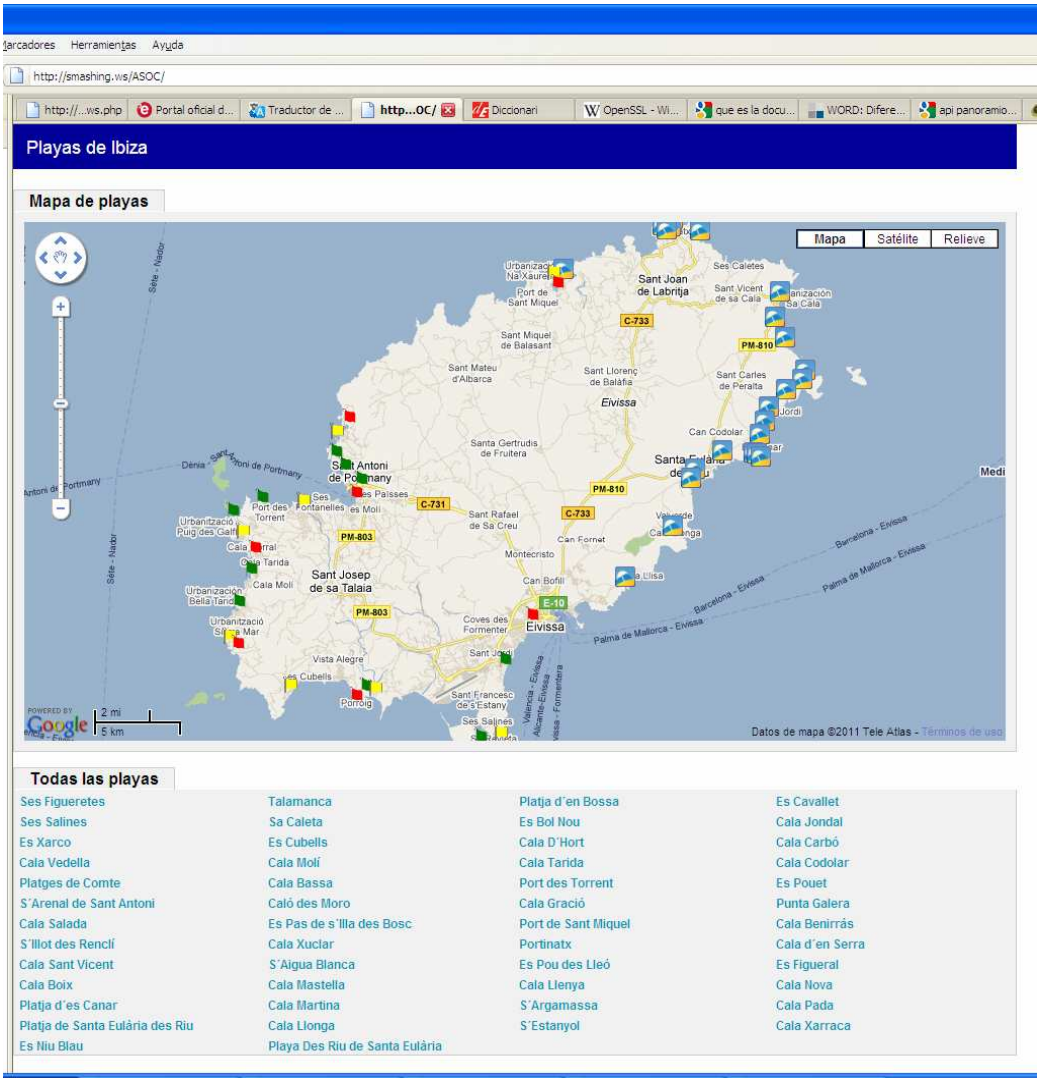

Figura 9.10 Navegació inicial

La imatge de la figura 9.10 és el punt d'entrada a la web client. A la part superior es pot veure un plànol de la illa d'Eivissa. El programador d'aquesta web ha incrustat un plànol de google. Aquest servei que ofereix google no és el que entenem com a Servei Web, simplement, s'incrusta codi JavaScript que carrega el plànol amb multitud d'opcions que es poden indicar. Es pot veure senyalat al plànol un munt de senyeres de diferents colors (Vermell, Groc i Verd) així com uns icones genèrics de platges. Sota el plànol també s'observa un llistat amb totes les platges de l'illa. Tota aquesta informació l'obté consumint el servei web de la Creu Roja.

A continuació, veiem la part del codi que interessa, de fet en les línies mostrades estan les úniques crides als serveis web oferts.

```
<?
$beach_id = $_GET["beach_id"];

$client = new SoapClient("http://10.0.1.1/CRE/index.wsdl");

if ($beach_id) {
        $playa = $client->informacio_platja(array('beach_id' => $beach_id));
} else {
        $list = $client->beaches_list();
}

?>
```

Figura 9.11 Codi font estat_platja(..)

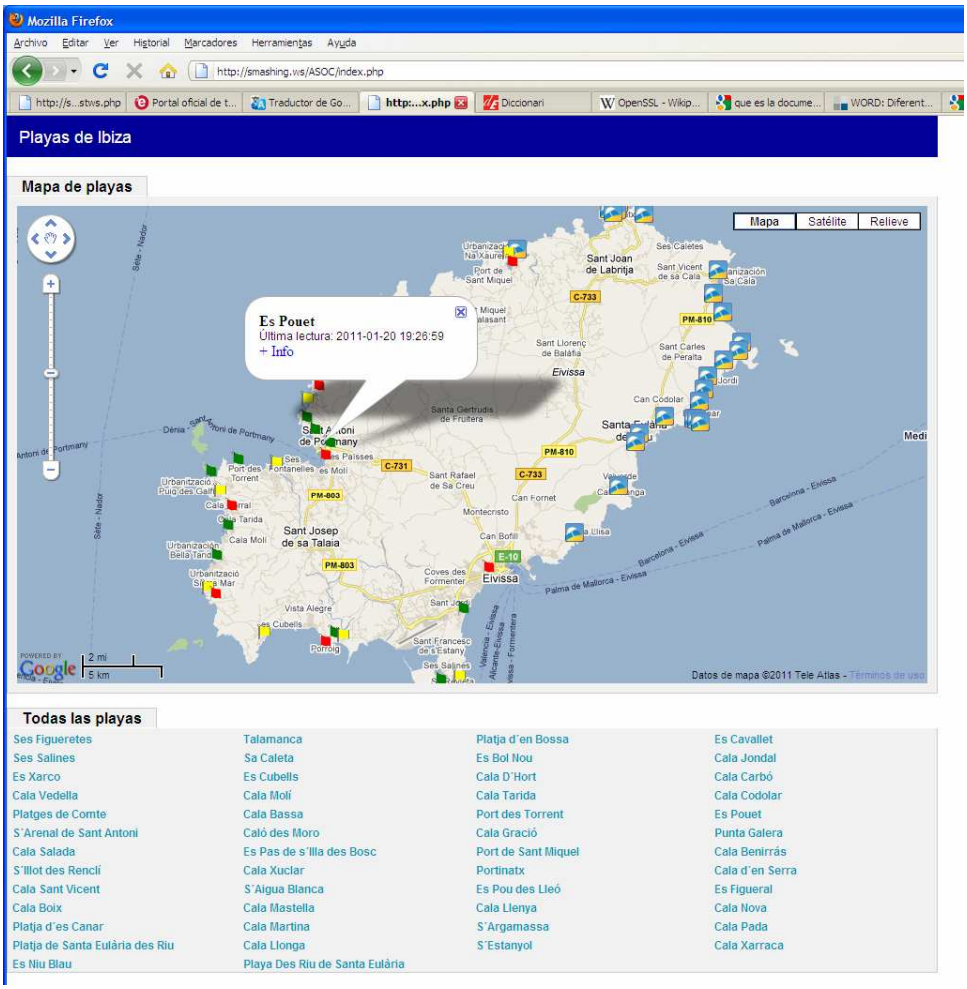

Figura 9.12 Navegació inicial + click plànol

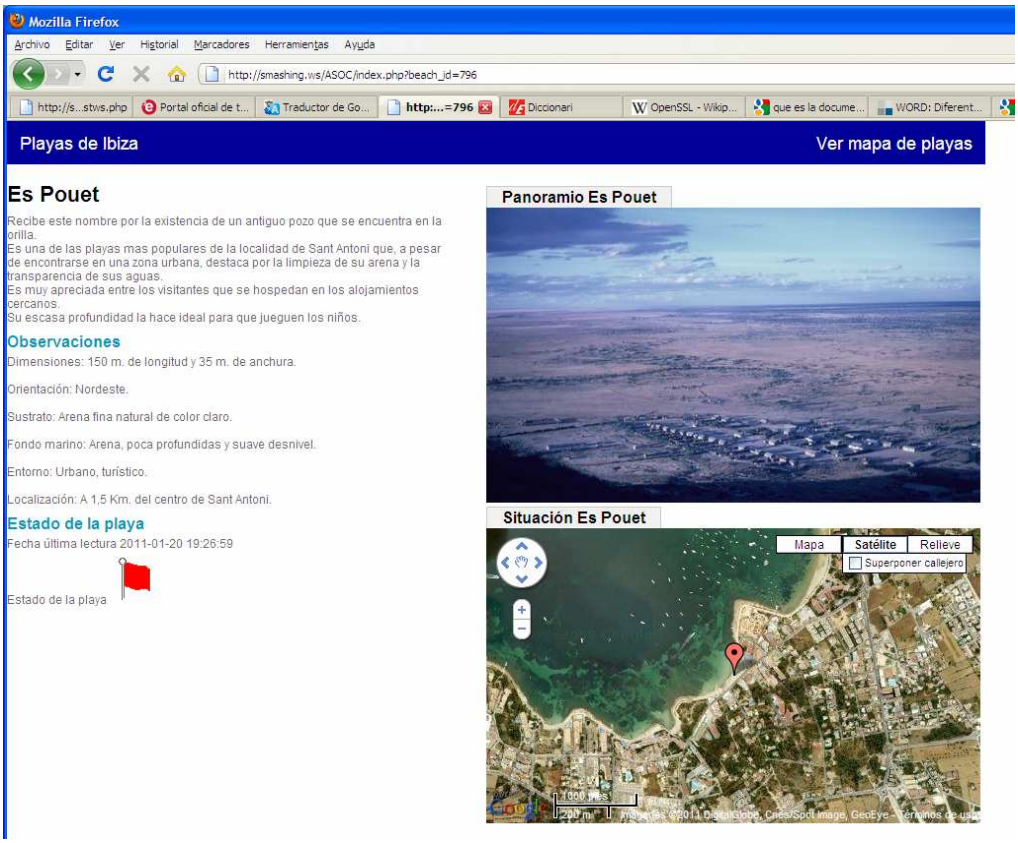

Figura 9.13 Navegació fitxa platja.

Totes les icones del plànol són sensibles a click del ratolí, així com totes les platges de la llista "Todas las playas". Per exemple si fem click sobre d'una icona del plànol, una platja, apareix un globus informatiu que conté la data de l'ultima lectura efectuada i un link amb un enllaç que mostra + Info (veure figura 9.12). Si es fa click sobre aquest darrer vincle la web ens porta a una plana informativa amb tota la informació disponible i ens mostra tota la informació de la platja (Nom, Descripció, Observacions, Estat de la platja i última lectura realitzada) a més de mostrar-nos una imatge i la seva localització en un petit plànol de google (veure figura 9.13).

En cas de fer click a Ver mapas de playas des de la figurà 9.13, es torna a la pàgina inicial (figura 9.10). Estan a la pàgina inicial si es fa click a qualsevol de les platges de la llista Todas las playas s'accedeix directament a la pàgina que conte tota la informació de la platja seleccionada (figura 9.13).

10. Conclusions.

10.1 Revisió de la planificació

Variable temps

Durant tot el desenvolupament del projecte s'ha anat revisant l'estat de les fases del projecte i de les tasques de cada una d'elles. En la etapa de Implementació es va detectar importants desajustos en la predicció del temps necessari per realitzar les tasques de Codificació en PHP del Framework WSFW i Proves. Concretament es va calcular que hi havia un retràs de 80h, repartides entre la codificació (64h) i les proves (16h). L'únic recurs disponible, l'Analista-Programador, no tenia la possibilitat d'augmentar la seva dedicació horària de quatre hores laborals. Atenen a aquestes circumstancies i a la data d'entrega màxima del projecte (25/01/2011), és va decidir no aplicar cap mesura correctora ja que existia suficient marge temporal per realitzar l'entrega satisfactòriament.

A la figura 10.1 es pot observar aquest retràs i l'execució temporal final del projecte.

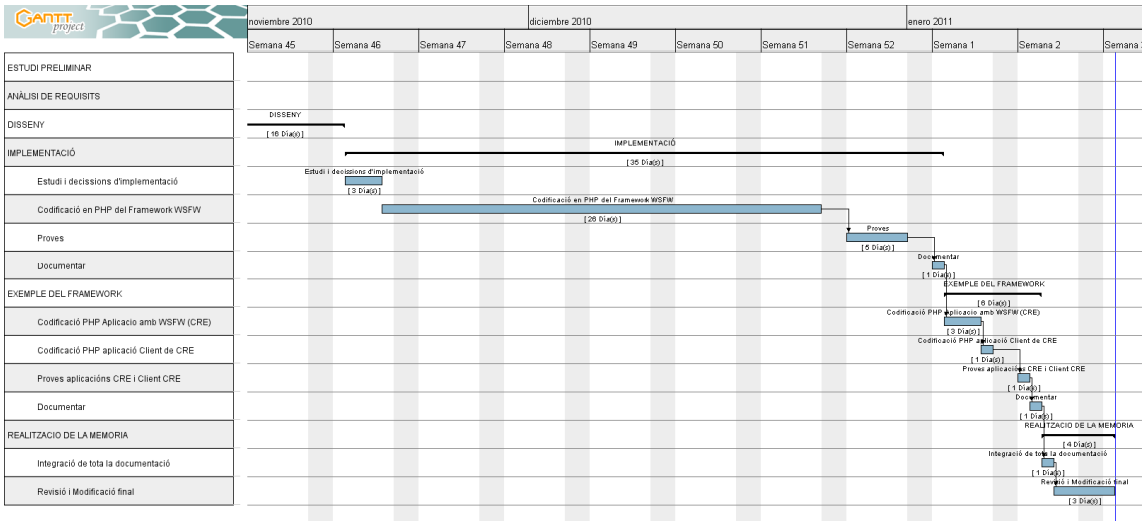

Figura 10.1 Gantt final del projecte

Es pot veure que les fases posteriors a la Implementació s'han desplaçat 20 dies hàbils corresponents a 80h de desviació de la predicció dividit entre 4h laborals de l'únic recurs en que es disposa. Degut al retràs en la planificació la nova data de finalització és el 17/01/2011.

Variable econòmica

No s'imputarà cap increment econòmic per l'ús de software i hardware.

El increment del pressupost del projecte serà exclusivament el causat per la variació en les hores dedicades de l'Analista-Programador. Aquest increment d'hores s'ha vist que suposa 20 dies laborals corresponents a 80h del recurs. Per calcular l'augment econòmic del projecte simplement es fa el producte entre les hores no previstes i la tarifa horària del recurs. En aquest cas: 80 x 35= 2.800€.

El cost econòmic total del projecte serà el planificat (capítol 7, apartat 2) més l'augment citat anteriorment. En aquest cas: 9.859 + 2.800= 12.649€

En resum, com podem veure a la figura 10.2 el projecte ha presentat un desviació negativa del 28,57% en temps i 28,40% en unitats monetàries.

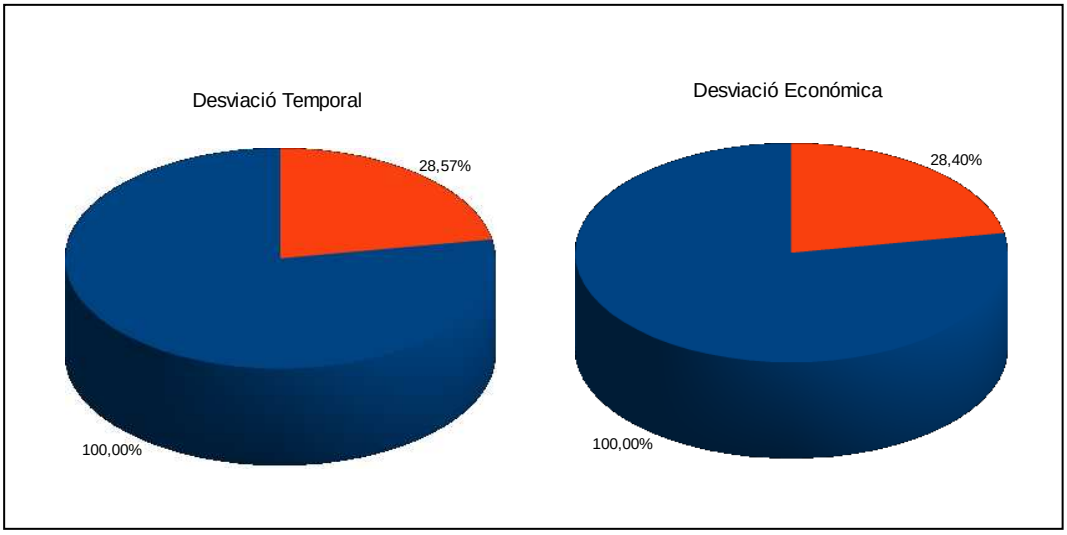

Figura 10.2 Increments % econòmics i temporals

Es pot observar que l'augment percentual de ambdós variables és pràcticament el mateix, normal ja que la imputació econòmica de hardware i software es menyspreable comparat amb la mà d'obra (59 € de 9.859€ planificats).

Aquestes desviacions econòmiques i temporals demostren una mala planificació, possiblement deguda a la insuficient experiència de l'Analista-Programador en tot el referent als Serveis Web.

10.2 Assoliment d'objectius

L'objectiu principal d'aquest projecte final de carrera és la creació d'un Framework per donar suport als programadors, concretament programadors de PHP, en tot el referent al serveis web.

Aquest objectiu ha estat assolit per complet i per poder demostrar-ho s'han realitzat dues aplicacions web. La primera és l'aplicació creada amb el nostre framework (WSFW). La segona és una pagina web que es "beneficia" d'aquesta consumint el servei web ofert. El framework havia de permetre crear i consumir serveis web i a més, utilitzant diversos protocols ("multiprotocol"). També, amb la aplicació desenvolupada amb WSFW, hem demostrat que permet consumir serveis web de tercers (el Consell Insular d'Eivissa i Panoramio) amb protocols diferents (SOAP i JSON-RPC) i oferir-los als clients interessats, per exemple, la pagina web consumidora.

El framework s'ha sotmès a un conjunt de proves on a acomplert tots els requisits funcionals esperats citats a la fase d'anàlisi. De cara als requisits no funcionals s'ha dissenyat el framework per ser usable i escalable.

L'escalabilitat es podria definir informalment com "propietat desitjable en un procés que indica la seva habilitat per poder créixer sense perdre la qualitat del seus serveis".[9]

Les circumstancies canviants en el panorama actual del món d'Internet, i per extensió dels serveis web, ens obliga a implementar qualsevol eina de programació pensant amb la seva escalabilitat. WSFW ha estat desenvolupat amb el paradigma d'orientació a objectes i gracies a l'abstracció i a l'herència és fàcilment extensible.

A l'aplicació d'exemple, "Aplicació Servei Web Creu Roja Eivissa (CRE)" podem observar com el desenvolupador ha realitzat una extensió del protocol JSON-RPC per poder consumir el servei web ofert per Panoramio que utilitza un protocol JSON-RPC propi, no estàndard.

Creant una estructura definida d'herència d'objectes i simplificant les estructures de dades utilitzades s'ha cercat l'usabilitat per tal de fer més simple l'aprenentatge.

S'ha desenvolupat tot el projecte aconseguint complir el requisit no funcional respecte a temes econòmics, ja que tot el software emprat es lliure. També s'ha demostrat, amb l'exemple, la possibilitat d'explotar els productes creats amb framework WSFW en un entorn 100% lliure, sense cap cost econòmic de llicencies.

Finalment podem concloure que aquest projecte final de carrera ha assolit amb escreix els objectius enunciats, complint els requisit establerts i a temps per ser lliurats dins del termini temporal establert.

10.3 Línies de treball

1) Tractament d' Errors:

El tractament d'errors, per la manca de temps no s'ha implementat acuradament, a més, caldria estudiar detingudament els errors característics que es poden donar a cada protocol concret. També s'hauria de comprovar els errors generats pels protocols atenen al comportament específic de les funcions de baix nivell emprades de l'Api de PHP.

2) Depurador

La depuració de serveis web és un problema important, identificar errors pot ser complicat quan l'error es pot generar tant al client com al servidor, i en cadascun d'aquests, l'error pot derivar de la transmissió, de la codificació de l'especificació, de la programació del servidor o de les peticions del client.

S'ha implementat la depuració a la banda del client on es mostra les peticions de baix nivell que s'intercanvien els protocols, però s'ha d'implementar un sistema per poder depurar el servidor i el client en l'àmbit del servei web, sense haver de recórrer a crear classes de test per instanciar les altres i mostrar la depuració.

3) Millora la Seguretat del Serveis Web:

La seguretat de les aplicacions generades amb WSFW, dependran exclusivament del programador i del serveis web consumits i/o oferts, assignant la url destí com http:// (sense seguretat) o https:// (amb seguretat ssl). Es força probable que el programador implementi addicionalment qualsevol tipus de mecanisme de seguretat, com punt d'entrada amb login/password o amb l'autentificació "HTTP Basic access authentication" pròpia d'HTTP. Degut a la multitud d'opcions no estàndards que hi ha al mercat, no hem entrat en l'estudi del tractament de la seguretat que pugui aportar les especificacions de cada protocol en particular. La W3C (World Wide Web Consortium) mitjançant la WS-Security sembla estar guanyant seguidors en la seva proposta per gestionar la seguretat en el serveis web.

4) Utilitat per generar documentació de l'API:

La documentació de la API (Interfície de Programació d'Aplicacions) és una eina indispensable per poder utilitzar una aplicació de tercers sobre la que no tenim cap mena de control. A l'apartat 3.1 ja hem comentat breument el tema i hem indicat alguns exemples coneguts com Google i Yahoo. Per oferir interoperabilitat entre diferents programaris, propis o de tercers, és necessari que el desenvolupador que vol que la seva aplicació interactuí amb un programari propietat de tercers precisarà la documentació bàsica per comunicar els diferents sistemes. Aquesta documentació pot ser més formal o menys, en suport paper, o qualsevol altre mitja.

WSFW implementa moltes funcionalitats dels Serveis Web, ja comentades anteriorment, estalviant molta feina de programació i possibilitant un marc únic de treball per els programadors, però no proporciona dos elements que es podem considerar útils: el fitxer WSDL (llenguatge de descripció de serveis web) del servei web ofert i la documentació de l'API. Generar el fitxer WSDL automàticament com a utilitat del framework comporta una

sèrie de consideracions i problemàtiques que ha fet descartar-ho i a més, és específic del protocol SOAP.

WSFW,en un futur proper, podria oferir documentació de l'API dels serveis web oferts de forma que el programador podrà parametritzar quina documentació ofereix, es a dir, quins serveis desitja oferir. La generació de la documentació es farà automàticament, en format web i opcionalment és podrà elegir en quina adreça web és publica.

5) Distribució a Internet, millora continua:

Després de realitzar parcialment o per complet les millores anteriors, la intenció del autor del framework WSFW és distribuir-lo gratuïtament a la comunitat d'Internet, possiblement sota llicencia GPL versió 3.

L'única manera de millorar i corregir errors d'un programari és mitjançant l'ús més o menys intensiu que un grup de persones realitzin d'aquest, per això considero important poder transmetre el framework a la comunitat de software lliure.

6) Implementació del framework per Java:

Resultaria força interessant traspassar l'experiència obtinguda en aquest projecte per realitzar una versió per Java, llenguatge més utilitzat just per sobre de C, C++ i PHP. Naturalment, aquesta versió Java és realitzaria en el propi llenguatge i l'aportació que oferiria seria idèntica al WSFW però atenen al particular tractament que fa Java de la temàtica de Serveis Web.

Annex I.

En aquest annex detallem el contingut del CD adjuntat amb la memòria del projecte.

/DOCUMENTACIO/ MarcRigatLlauradoArticle.doc
MarcRigatLlauradoResum.doc
Memoria_MarcRigatLlaurado.pdf

/APLICACIO/FRAMEWORK/WSFW
Conté tot el codi font del framework desenvolupat.

/APLICACIO/EXEMPLE_CRE /
Conté tot el codi font del exemple creat amb el framework

/APLICACIO/CLIENT/
Conté tot el codi font del exemple client del EXEMPLE_CRE

/PHPDOC/
Conté tota la documentació generada del codi font.

/PHPUNIT/PROVES/
Conté els fitxers utilitzats per fer les proves unitàries i d'integració.

/PHPUNIT/RES/
Conté els resultats generats per PhpUnit

/GANTT/
Conté els fitxers diagrama de gantt planificació prevista i la real.

Bibliografia.

Referències

[1] http://es.wikipedia.org/wiki/Framework

[2] http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/#whatis

[3] http://es.wikipedia.org/wiki/Interfaz_de_programaci%C3%B3n_de_aplicaciones

[4] http://www.friendfeed.com

[5] http://es.wikipedia.org/PHP

[6] http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html

[7] UML for the Impatient Martin Gogolla University of Bremen, FB 3
Computer Science Department. Postfach 330440, D-28334 Bremen, Germany
gogolla@informatik.uni-bremen.de

[8] http://ca.wikipedia.org/wiki/OpenOffice.org

[9] http://www.alegsa.com.ar/Dic/escalabilidad.php

Llibres

Pro PHP XML and Web Services
Robert Richards
Apress 2006
ISBN 1-59059-633-1

Web Services Essentials Distributed Applications with XML-RPC, SOAP, UDDI & WSDL
O'Reilly - February 2002
ISBN: 0-596-00224-6

Programming Web Services with XML-RPC
O'Reilly - June 2001
ISBN: 0-596-00119-3

RESTful Web Services
O'Reilly – May 2007
Leonard Richardson and Sam Ruby
ISBN-10: 0-596-52926-0

Programming Web Services with SOAP
Doug Tidwell - James Snell - Pavel Kulchenko
O'Reilly - First Edition December 2001
ISBN: 0-596-00095-2

The Unified Modeling Language Reference Manual
James Rumbaugh - Ivar Jacobson - Grady Booch
ADDISON-WESLEY - 1999
ISBN 0-201-30998-X

UML 2.0 in a Nutshell
Dan Pilone - Neil Pitman
O'Reilly - June 2005
ISBN: 0-596-00795-7

Database Design for Smarties: Using UML for Data Modeling
Robert J. Muller
Morgan Kaufmann Publishers © 1999
ISBN: 1558605150

Localització URL

http://zenon.etsii.urjc.es/grupo/docencia/as/material/tema6.pdf
http://www.scourdesign.com/articulos/tutoriales/php/tutoriales-php-mysql-servicios-web-soap.php
http://en.wikipedia.org/wiki/Web_service
http://es.wikipedia.org/wiki/JSON
http://estebanfuentealba.wordpress.com/2010/04/25/creando-un-mini-webservice-en-php-json-jsonp/
http://www.marlonj.com/blog/2009/05/webservice-en-php-con-nusoap/
http://www.oclipa.com/university/nusoap/3.php
http://www.scottnichol.com/nusoapprogwsdl.htm
http://es.wikipedia.org/wiki/AJAX
http://atc.ugr.es/pedro/tutoriales/cursos/xml
http://www.desarrolloweb.com/articulos/1545.php
http://csharpcomputing.com/XMLTutorial/
http://www.desarrolloweb.com/articulos/producir-json-desde-php.html
http://www.json.org/
http://tarjuccino.com/tutoriales/programacion-web/introduccion-a-json/