Sistema de validació amb proveïdor de identitat extern per a serveis de Video On Demand

Xavier Segura Gonzalez

ETIS

Carlos Ares Angulo

18 de Gener de 2009

Dedicat a Mariana y al seu MacBook. Pel suport que m'han donat, i per la seva fiabilitat quan els he necessitat

Resum

El TFC s'enmarca en l'àrea de seguretat informàrtica. Mitjançant l'ús de misstges signats i canals de comunicació segurs, garanteix a un usuari, accedir a recusos restingits d'un proveïdor, sense que aquest necessiti dispossar de cap dada de l'usuari.

Al desenvolupament interactuen 3 tipus d'actor:

- El Client, que demana una sèrie de recursos on-line, i s'identifica amb el seu DNIe.
- El Proveïdor de servei, que proporciona cert recursos a aquells clients que tinguin accés permès, però desconeix totalment les identitats d'aquests clients.
- El Proveïdor d'idenditat, el responsable de gestionar els permisos d'accés per al recursos del Proveïdor de servei, i és qui coneix la identitat del Client.

Aquest entorn s'ha de caracteritzar per la total privacitat del Client envers al Proveïdor de servei.

Aplicació proposada

A partir d'aquesta definició, es proposa l'aplicació a una empresa de Video on demand (VODBUSTERS). Aquesta empresa, dóna accés per visualitzar vídeos durant un temps finit als clients que realitzin un pagament per a la seva visualització. Mitjançant la utilització del DNIe, el Client podrà accedir als vídeos llogats des de qualsevol equip connectat a internet, durant el temps que duri el lloguer.

El Proveïdor de servei seria una o més empresses Proveïdores de vídeo streaming (i.e. YOUSTREAM), que degut a la gran volum de recursos que demana aquesta tecnologia, es decideix fer-la externa a la plataforma pròpia la empresa de VOD.

Amb el sistema proposat, la informació del Client es trobarà centralitzada als servidors de la empresa de VOD (Proveïdor d'identitat), sent totalment transparent per als servidors de vídeo streaming (Proveïdor de servei). D'aquesta forma es poden incrementant en nombre de servidors i/o Proveïdors al anar creixent el volum d'accessos, i així garantir la qualitat del servei, sense haver de compartir les dades dels clients de la empresa de VOD.

Índex de continguts

Introducció

Fonaments i estat de l'art

Actualment el sistema de Video on Demand, no està gaire estès, especialment fora dels EEUU. Ens trobem amb sistemes de lloguer de pel·lícules i programes de televisió com el que utilitza iTunes Store d'Apple, on les recursos es descarreguen a un equip, i per poder reproduir-los a un altre equip s'ha de realitzar una transferència utilitzant el sistema FairPlay DRM, per garantir que no es realitza una copia il·legal al dispositiu. Un cop passat el temps de lloguer el fitxer és destruït. Tant per a la descàrrega com per a la reproducció necessita del software iTunes. Un sistema semblant és l'utilitzat per Amazon, anomenat Unbox, on els recursos són també descarregats i destruïts un cop hagi passat un més des de el lloguer o 24 hores des de la primera visualització completa. Aquest sistema també precisa d'un reproductor específic per a la visualització del recurs adquirit.

En aquest projecte proposa un sistema de lloguer i compra mitjançant video streaming, on els clients no han d'estar lligats a un dispositiu per poder gaudir d'aquells recursos que s'han comprat o llogat, ni precisen software específic. Únicament es demana una prova de la identitat del Client, en aquest cas el seu DNIe, perquè aquest pugui accedir a tots els recursos que ha comprat o llogat, des de qualsevol dispositiu amb accés a internet. Però per garantir un servei correcte, es permet distribuir els servidors de serveis, sense haver de distribuir la BBDD de clients ni les seves dades personals, garantint així la privacitat de les dades del Client davant el/els Proveïdor/s de servei.

Per fer possible aquest accés als recursos de forma segura, s'utilitzen certificats digitals X.509 per signar els missatges que s'enviaran entre els servidors i el Client. Aquests missatges tenen format XML i es signen mitjançant l'estàndard XMLDSig. La implementació utilitzada per aquesta signatura és la pròpia de la última versió del JDK de Java, la 6.0. El Client utilitza el certificat incorporat al seu dni electrònic, que pot ser validat mitjançant una autoritat de validació online pública, que implementa el protocol OCSP. Degut a la necessitat de demostrar que la petició del recurs es fa dintre del temps de lloguer establert, la signatures poden portar un segell de temps utilitzant la plataforma TrustedX.

Tant a la comunicació amb la plataforma TrustedX, com la que realitza el Client per validar la seva identitat, es realitza mitjançant Web Services, utilitzant el protocol SOAP (Simple Object Access Protocol). La creació d'aquests missatges es fa utilitzant l'API específica Axis2/Java d'Apache.

Objectius del TFC

Es plantegen com a objectius generals, aquells relacionats amb el producte i el TFC:

- Implementar 2 entitats que permetin a un Client, l'accés d'una manera segura a uns recursos audiovisuals (pel·lícules o documentals), sense ser necessari mostrar la seva identitat al proveïdor que li proporciona el recurs.
- Proporcionar una eina per a l'empresa de VOD, perquè pugui gestionar els seus clients i la seva informació confidencial d'una manera segura. A la vegada, l'eina facilita tenir un o més Proveïdors de video streaming, sense que sigui necessari proporcionar a aquests Proveïdors cap informació dels seus clients.
- Desenvolupar una memòria que reculli les tasques realitzades, el disseny del producte, les presses de decisions, els resultats i conclusions, així com la documentació de instal·lació i ús del producte.
- Realitzar una presentació on resumeixi clarament en que ha consistit el projecte.

Els objectius a assolir a nivell personal són:

- Aplicar a un projecte de software els coneixements adquirits a les assignatures de la carrera, especialment les més relacionades amb la seguretat ('Criptografia' i 'Seguretat en xarxes de computadors').
- Adquirir nous coneixements en desenvolupament i/o configuració de tecnologies i components de sofware desconeguts fins ara.
- Adquirir coneixements teòrics i experiència a la gestió d'un projecte de principi a fi.
- Conèixer i treballar amb noves fonts d'informació sobre seguretat informàtica.
- Conèixer tecnologies i/o components relacionats amb seguretat informàtica que pugui aplicar a la meva vida professional.

Experiència prèvia

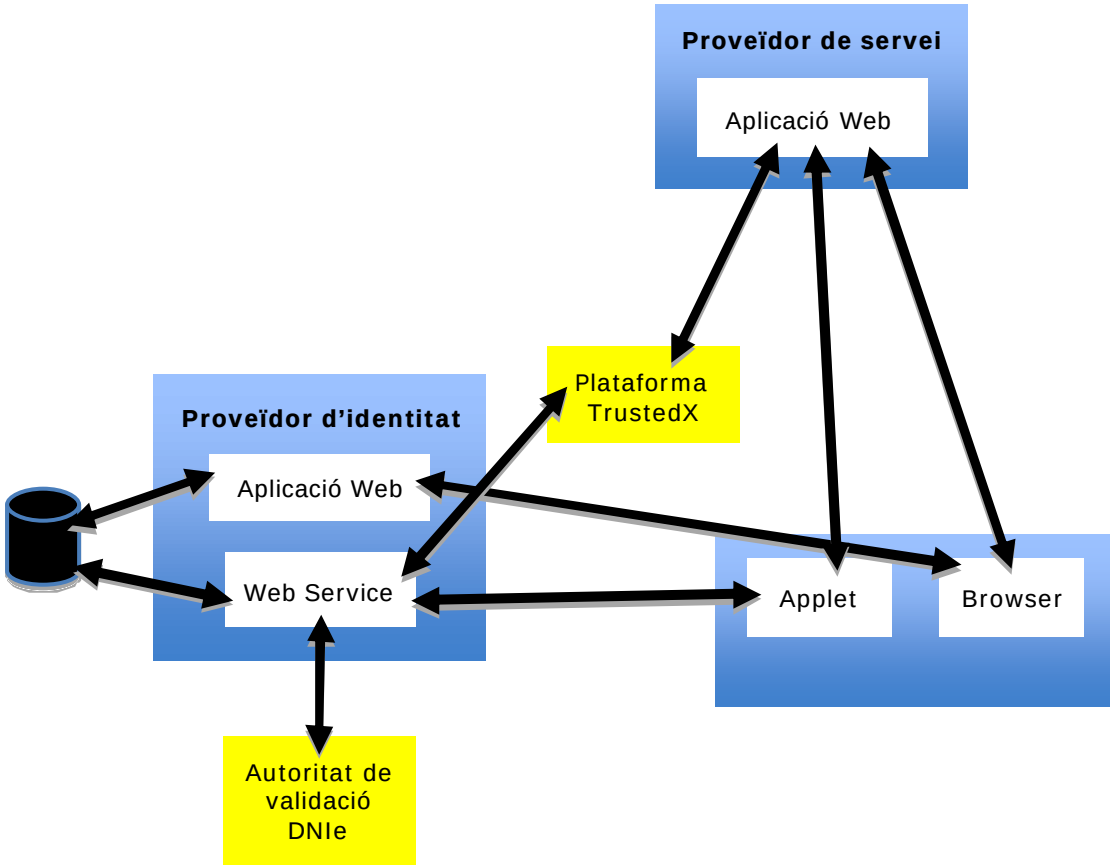
El meu contacte amb el món de la seguretat informàtica, es redueix, per la banda acadèmica, a les assignatures de 'Criptografia' i 'Seguretat en xarxes de computadors' cursades a la UOC, i per la banda professional, al disseny i implementació d'un Single Sign On, per accedir a 7 plataformes de Proveïdors diferents des d'una comunitat de fidelització.

Respecte a desenvolupaments en entorns web, tinc experiència de 7 anys en els diferents perfils que intervenen en un projecte informàtic (de programador a cap de projecte). Les tecnologies amb les que estic més familiaritzat son Java, Struts i Oracle.

Visió general de l'arquitectura

El producte a realitzar es composa de 2 aplicacions web, 1 Web Service i 1 applet, tots ells desenvolupats utilitzant la tecnologia J2EE 6.0. S'ha escollit la versió J2EE de Java, per la facilitat que proporciona per desenvolupaments web i client-servidor.

El següent diagrama mostra els diferents components del producte i de quina manera es comuniquen.



Les aplicacions web es desenvolupen utilitzant Spring Framework 2.5.5, amb el paradigma de programació orientada a aspectes (AOP). La BBDD utilitzada és MySQL que és accessible mitjançant el gestor de persistència Hibernate 3.3.1. Totes dues aplicacions funcionen sobre servidors Tomcat 6 (v. 6.0.18), amb certificats per les connexions SSL. En el cas del servidor del Proveïdor de servei, el certificat haurà de ser confiable per la plataforma TrustedX. Per al Proveïdor d'identitat aquesta premissa no serà necessària.

A continuació es defineix, a gran trets, quines són les responsabilitats de cada un dels components i com les porten a terme.

Aplicació web del Proveïdor d'identitat. S'encarrega de la gestió dels clients i els permisos d'accés als serveis.

Per una banda realitza una tasca d'auto-registre, on els clients es poden donar d'alta al sistema, introduint les seves dades personals i bancaries.

La segona tasca és gestionar la compra de serveis (lloguer o compra de pel·lícules). El Client escull un servei d'un llistat i posteriorment al pagament, es realitza la persistència de l'assignació del Client a aquest servei per un temps determinat.

Aplicació web del Proveïdor de servei. Rep la petició d'accés a un servei, i el dóna després de validar que el Client te accés a aquell servei en aquell precís instant.

Al rebre la petició, es crea l'identificador de petició que es torna signat amb altres dades, i amb un segell de temps a la signatura, generat per la plataforma TrustedX. També s'envia al Client un applet de Java per a la comunicació amb el Web Service del Proveïdor d'identitat.

Després de les gestions dels altres components, rebrà la comunicació del Proveïdor d'identitat donant o denegat l'accés al servei demanat. Un cop realitzades validacions de la signatura i les dades del missatge, donarà o denegarà l'accés al Client.

Applet del Client. S'encarrega de fer de pont entre els 2 Proveïdors, signant les dades que envia.

Per a la petició al Web Service, és necessari demanar al Client quin dels seus certificats és el que vol utilitzar per signar les dades i demanar el PIN corresponent. Per a l'accés al certificat del DNIe és necessari l'accés mitjançant un suport físic. En aquest cas un USB on es connecta el lector amb el DNIe.

Un cop es rep la resposta del Web Service, s'encarrega de reenviar-la perquè arribi al Proveïdor de servei.

Web Service del Proveïdor d'identitat. Dóna o rebutja l'accés un cop comprovada les signatures i el permís per accedir al recurs demanat.

Per a la validació externa de l'estat del certificat del Client, es connecta a la autoritat de validació del DNIe, mitjançant el Protocol OCSP.

També es realitza una validació de la data, utilitzant la plataforma TrustedX.

Com es pot veure al diagrama, al component del Proveïdor d'identitat, tant l'aplicació web com el Web Service ataquen la BBDD, que es on s'emmagatzemen els permisos de que disposa cada Client per accedir al serveis, i quan hi té accés.

Millores opcionals fora la planificació. Es contempla la possibilitat, en cas que per alguna raó es disposi d'un temps extra a la planificació, d'ampliar l'aplicació web del Proveïdor d'identitat.
Es realitzarien eines de manteniment de la BBDD per als administradors: funcionalitats d'alta, baixa, consulta i modificació de les diferents entitats de la BBDD (clients, serveis i permisos amb un rang de dates).

Tasques a realitzar

Les tasques a realitzar per l'execució del TFC es llisten per grups a continuació:

1. Documentació.

- Pla de treball. Document on es plasma el treball que s'ha de portar a terme, objectius i temporització de tasques.
- Memòria
  - o Introducció. 1er capítol.
  - o Nucli. Resta de capítols.
  - o Conclusions. Últim capítol.
- Presentació on-line. Document en Power Point amb el resum del desenvolupament del projecte.

2. Instal·lació de l'entorn de treball.

- IDE Eclipse - JDK Java 6.0 Update 7.
- Generació de certificats. S'utilitza l'eina *OpenSSL* i keytool de Java.
- Instal·lació i configuració de 2 servidors Tomcat amb els certificats corresponents.
- Instal·lació de llibreries:
  - o Framework Spring (serv. del Proveïdor d'identitat i de servei)
  - o Implementació de SOAP Axis2 (serv.del Proveïdor d'identitat)
  - o Hibernate (serv.del Proveïdor d'identitat)
- BBDD: MySql + pluggins per accés a BBDD des de Eclipse.

3. Cerca de documentació sobre les tecnologies a utilitzar.

- Web services – SOAP
- Framework Spring
- Hibernate
- XMLDSig
- Protocol OCSP

4. Desenvolupament (disseny i implementació):

- Aplicació web del Proveïdor d'identitat. Funcionalitats:
  - Auto-registre de clients. Alta de clients al sistema.
  - Compra de serveis.
- Aplicació web del Proveïdor de servei. Funcionalitats:
  - DemanaRecurs. Genera dades de la petició signades amb un segell de temps.
  - AccedeixRecurs. Es denega o es concedeix l'accés al recurs.
- Applet del Client.
  - Generació de la petició al Web Service. Accés al certificat mitjançant l'USB.
  - Recollida de la petició al Web Service. Reenviament al Proveïdor de servei.
- Web service del Proveïdor d'identitat.
  - Verificació de les 2 signatures.
  - Validació certificat del Client (OCSP).
  - Validació de la data (TrustedX).
- Proves unitàries. A nivell de component.
- Proves funcionals. D'interacció entre els diferents components.
- Correccions posterior a les proves.

5. Millores opcionals fora la planificació:

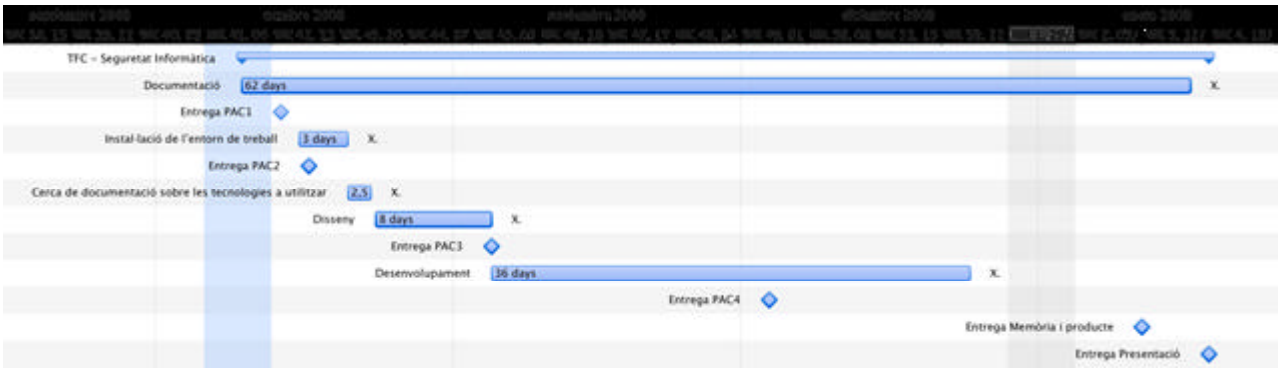
- Aplicació web del Proveïdor d'identitat
  - Manteniments (alta, baixa, consulta i modificació) de:
    - Clients
    - Serveis
    - Permisos clients-recursos amb un rang horari (o de dates)

Planificació

A l'execució d'aquest TFC tenim 6 fites:

- 13/10/08 – PAC1. Entrega no definitiva del Pla de Treball.
- 17/10/08 – PAC2. Introducció de la memòria, amb el Pla de Treball definitiu.
- 5/10/08 – PAC3. Disseny del producte.
- 4/12/08 – PAC4. Entrega de 2 dels 4 mòduls del producte. Podrien ser 'Aplicació web del Proveïdor d'identitat' i 'Aplicació web del Proveïdor de servei'.
- 12/01/09 – Entrega del la memòria i el producte.
- 19/01/09 – Entrega de la presentació virtual.

Per asolir aquests objectius de lliuraments, serà necessaria la planificació del següent diagrama de Gantt.



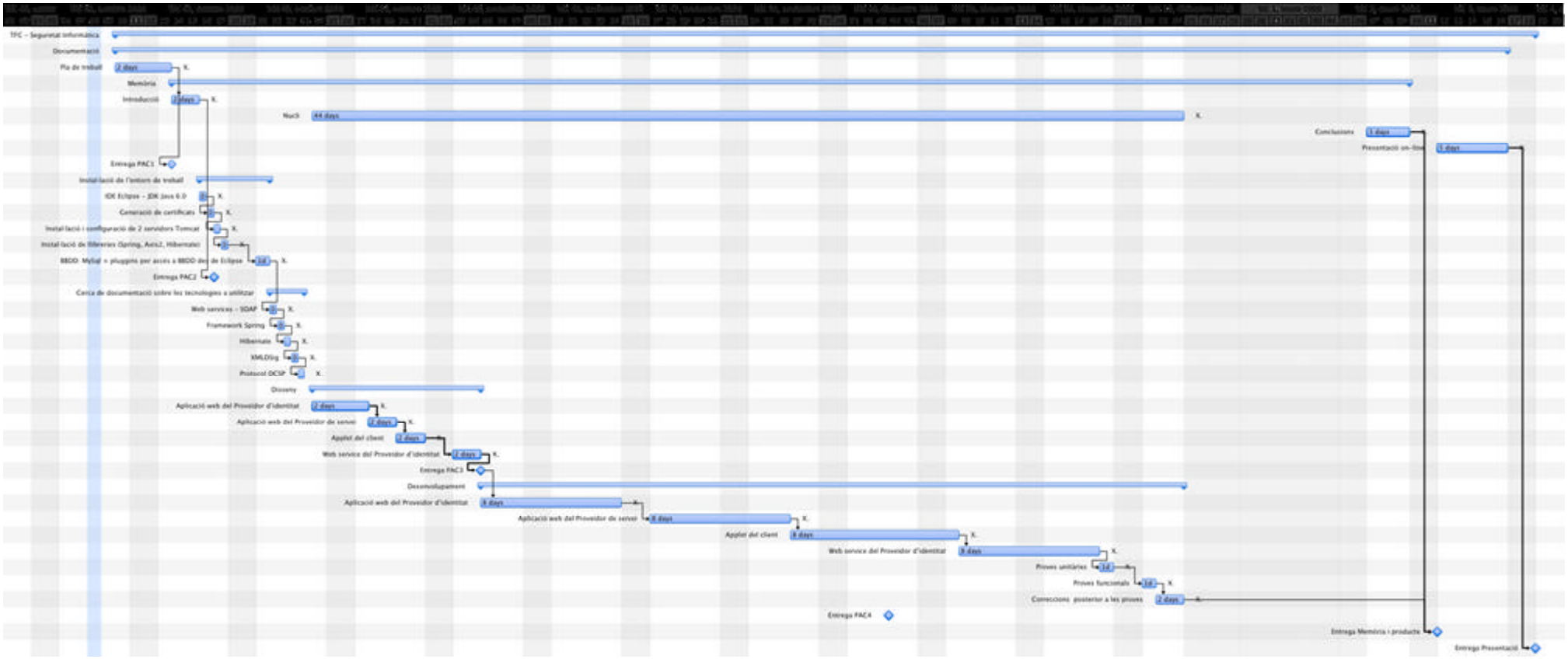
A aquesta taula es mostra el llistat de tasques a realitzat amb la seva temporització en dies, així com els prerequisit necessaris per la realització de cada una d'elles.

Id tasca	Tasca	Data inici	Data fi	Treball	Tasca prèvia
0	TFC - Seguretat Informàtica	10/10/08	19/01/09		
48	Documentació	10/10/08	16/01/09		
49	Pla de treball	10/10/08	13/10/08	2d	
51	Memòria	14/10/08	09/01/09		
52	Introducció	14/10/08	15/10/08	2d	49
54	Nucli	24/10/08	24/12/08		
56	Conclusions	07/01/09	09/01/09	3d	
58	Presentació on-line	12/01/09	16/01/09	5d	
62	Entrega PAC1	13/10/08	13/10/08		49
12	Instal·lació de l'entorn de treball	16/10/08	20/10/08		
13	IDE Eclipse - JDK Java 6.0	16/10/08	16/10/08	0,5d	
15	Generació de certificats	16/10/08	16/10/08	0,5d	13
17	Instal·lació i configuració de 2	17/10/08	17/10/08	0,5d	15

	servidors Tomcat				
19	Instal·lació de llibreries (Spring, Axis2, Hibernate)	17/10/08	17/10/08	0,5d	17
21	BBDD: MySql + pluggins per accés a BBDD des de Eclipse	20/10/08	20/10/08	1d	19
60	Entrega PAC2	17/10/08	17/10/08		52
1	Cerca de documentació sobre les tecnologies a utilitzar	21/10/08	23/10/08		
2	Web services – SOAP	21/10/08	21/10/08	0,5d	21
4	Framework Spring	21/10/08	21/10/08	0,5d	2
6	Hibernate	22/10/08	22/10/08	0,5d	4
8	XMLDSig	22/10/08	22/10/08	0,5d	6
10	Protocol OCSP	23/10/08	23/10/08	0,5d	8
23	Disseny	24/10/08	04/11/08		
24	Aplicació web del Proveïdor d'identitat	24/10/08	27/10/08	2d	
26	Aplicació web del Proveïdor de servei	28/10/08	29/10/08	2d	24
28	Applet del Client	30/10/08	31/10/08	2d	26
30	Web service del Proveïdor d'identitat	03/11/08	04/11/08	2d	28
32	Entrega PAC3	05/11/08	05/11/08		30
33	Desenvolupament	05/11/08	24/12/08		
34	Aplicació web del Proveïdor d'identitat	05/11/08	14/11/08	8d	32
36	Aplicació web del Proveïdor de servei	17/11/08	26/11/08	8d	34
38	Applet del Client	27/11/08	08/12/08	8d	36
40	Web service del Proveïdor d'identitat	09/12/08	18/12/08	8d	38
42	Proves unitàries	19/12/08	19/12/08	1d	40
44	Proves funcionals	22/12/08	22/12/08	1d	42
46	Correccions posterior a les proves	23/12/08	24/12/08	2d	44
61	Entrega PAC4	04/12/08	04/12/08		
63	Entrega Memòria i producte	12/01/09	12/01/09		46; 56
64	Entrega Presentació	19/01/09	19/01/09		58

Indicador de fita

Diagrama de Gantt amb el detall de la planificació per tasques.



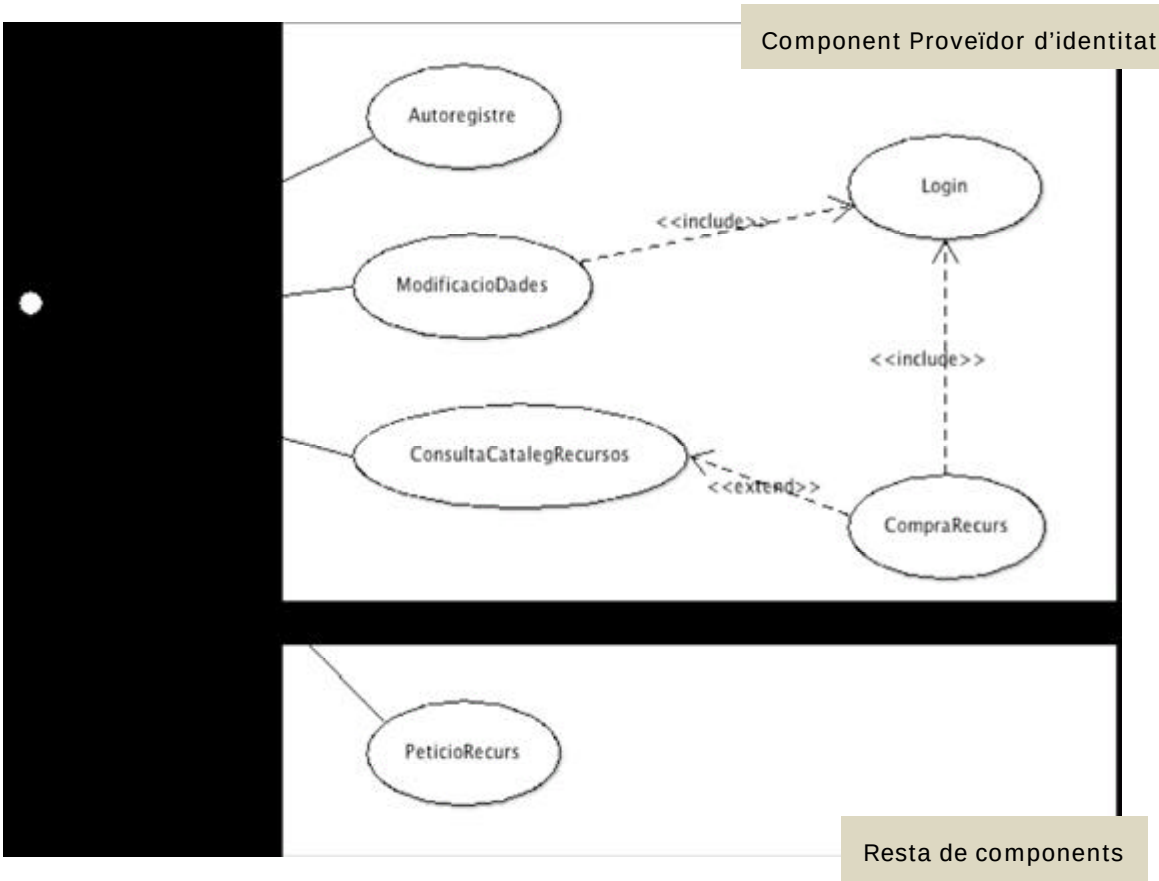
Disseny

Diagrama de casos d'ús

El diagrama de casos d'ús és una eina utilitzada per modelitzar les funcions d'un software des del punt de vista d'interaccions amb l'exterior.

Cal destacar que a aquest diagrama trobem per una banda els actors, que són els elements externs que interactuen amb el sistema, i per una altra banda els casos d'ús, que serien les diferents funcions que realitza el sistema.

Aquest és el diagrama del projecte, on podem veure que l'únic actor és el client, i el sistema es divideix en 2: component PI i resta de components. Al detall del cas d'ús *PeticioRecurs* es poden trobar aquesta resta de components.



Casos d'ús

Cas d'us: Autorregistre

Actor: Client

Descripció: L'actor s'autorregistra al sistema, introduint les seves dades personals i així com les dades de pagament.

Seqüència d'esdeveniments:

Accions de l'actor	*Respostes del sistema*
1. L'actor inicia el cas d'ús quan es vol registrar com a usuari dins de la plataforma.	
2. L'actor envia les seves dades personals i de pagament	
	3. El sistema valida que hi són totes les dades necessàries i són factiblement correctes
	4. El sistema guarda al repositori les dades de l'actor i comunica el correcte emmagatzemament.

Alternatives i excepcions:

3. Error al validar les dades de l'actor.

4. Error al guardar les dades al repositori.

Cas d'us: Login

Actor: Client

Descripció: L'actor s'autentifica al sistema, introduint el seu usuari y contrasenya.

Seqüència d'esdeveniments:

Accions de l'actor	*Respostes del sistema*
1. L'actor inicia el cas d'ús quan es vol accedir als recursos de la plataforma.	
2. L'actor envia les seves dades d'usuari i contrasenya	
	3. El sistema valida les dades i dóna accés a la plataforma.

Alternatives i excepcions:

3. Error al validar les dades de l'actor. No dóna accés a la plataforma. En cas de no estar registrat a la plataforma, s'hauria de executar el cas d'ús Autorregistre.

Cas d'us: ModificacióDades

Actor: Client

Descripció: L'actor modifica les seves dades introduïdes al sistema

Seqüència d'esdeveniments:

Accions de l'actor	*Respostes del sistema*
4. L'actor inicia el cas d'ús quan es vol modificar les seves dades introduïdes al sistema.	
5. L'actor s'autentifica al sistema. (Veure el cas d'ús login)	
	6. El sistema cerca les dades de l'actor les envia.
7. L'actor envia la modificació de les seves dades personals i de pagament	
	8. El sistema valida que hi són totes les dades necessàries i són factiblement correctes
	9. El sistema guarda al repositori les dades de l'actor i comunica el correcte emmagatzemament.

Alternatives i excepcions:

6. Error al validar les dades de l'actor.

7. Error al guardar les dades al repositori.

Cas d'us: ConsultaCatàlegRecursos

Actor: Client

Descripció: L'actor accedeix al catàleg de recursos complet

Seqüència d'esdeveniments:

Accions de l'actor	*Respostes del sistema*
1. L'actor inicia el cas d'ús quan es vol visualitzar el catàleg de recursos de la plataforma.	
	2. El sistema cerca el catàleg del repositori i l'envia a l'actor.

Alternatives i excepcions:

3. Error al cercar la llista del repositori.

Cas d'us: CompraRecurs

Actor: Client

Descripció: L'actor compra l'accés a un recurs per un temps definit.

Seqüència d'esdeveniments:

Accions de l'actor	*Respostes del sistema*

1. L'actor inicia el cas d'ús quan vol tenir accés a un recurs de pagament de la plataforma.	
2. L'actor escull el recurs que desitja comprar i el temps que hi vol tenir accedir.	
3. L'actor s'autentifica al sistema. (Veure el cas d'ús login)	
	4. El sistema realitza el pagament.
	5. Es persisteix el permís de l'actor per accedir al recurs pel temps demanat.
	6. Es comunica a l'usuari la correcta finalització de la transacció i la URI per accedir al recurs.

Alternatives i excepcions:

4. Error al realitzar el pagament. L'actor no te capacitat de pagament o la plataforma de externa on es realitza el pagament està inoperable.

5. Error al persistir les dades al repositori. S'ha de deixar constància que el pagament ha sigut realitzar.

Cas d'us: Petició Recurs

Actor: Client

Descripció: L'actor demana al proveïdor de serveis accés a un recurs, i aquest l'aprova o el rebutja a partir de la resposta del proveïdor de identitat

Seqüència d'esdeveniments:

Accions de l'actor	*Respostes del proveïdor de servei (PS)*	*Respostes del component del client*	*Respostes del proveïdor de identitat (PI)*
1. L'actor inicia el cas d'ús quan vol accedir als recursos que ha comprat de la plataforma.			
2. Demana al PS l'identificador de petició d'un recurs			
	3. Genera l'identificació de petició i signa conjuntament amb l'id del recurs i l'id de l'aplicació		

	4. Actualitza la signatura amb un segell de temps (autoritat externa)		
	5. Torna les dades signades amb la signatura i un component perquè es connecti l'actor amb el PI		
		6. Recull les dades rebudes del PS	
		7. Accedeix als suports físics per cercar certificats disponibles	
8. Indica el certificat amb el que vol signar la petició i el seu PIN			
		9. Signa les dades rebudes del PS amb el certificat escollit per l'actor, i les envia al PI	
			10. Verifica les signatures i el permisos de l'usuari per accedir al recurs.
			11. Genera una resposta permetent o denegant l'accés
			12. Signa i envia la resposta al component
		13. Reenvia la resposta rebuda del PI al PS	
	14. Verifica la signatura del PI i la		

	concordança de les dades amb la sessió actual		
	15. Dona accés a l'actor al recurs demanat		

Alternatives i excepcions:

4. Error al connectar amb l'autoritat de segell de temps per modificar la signatura.

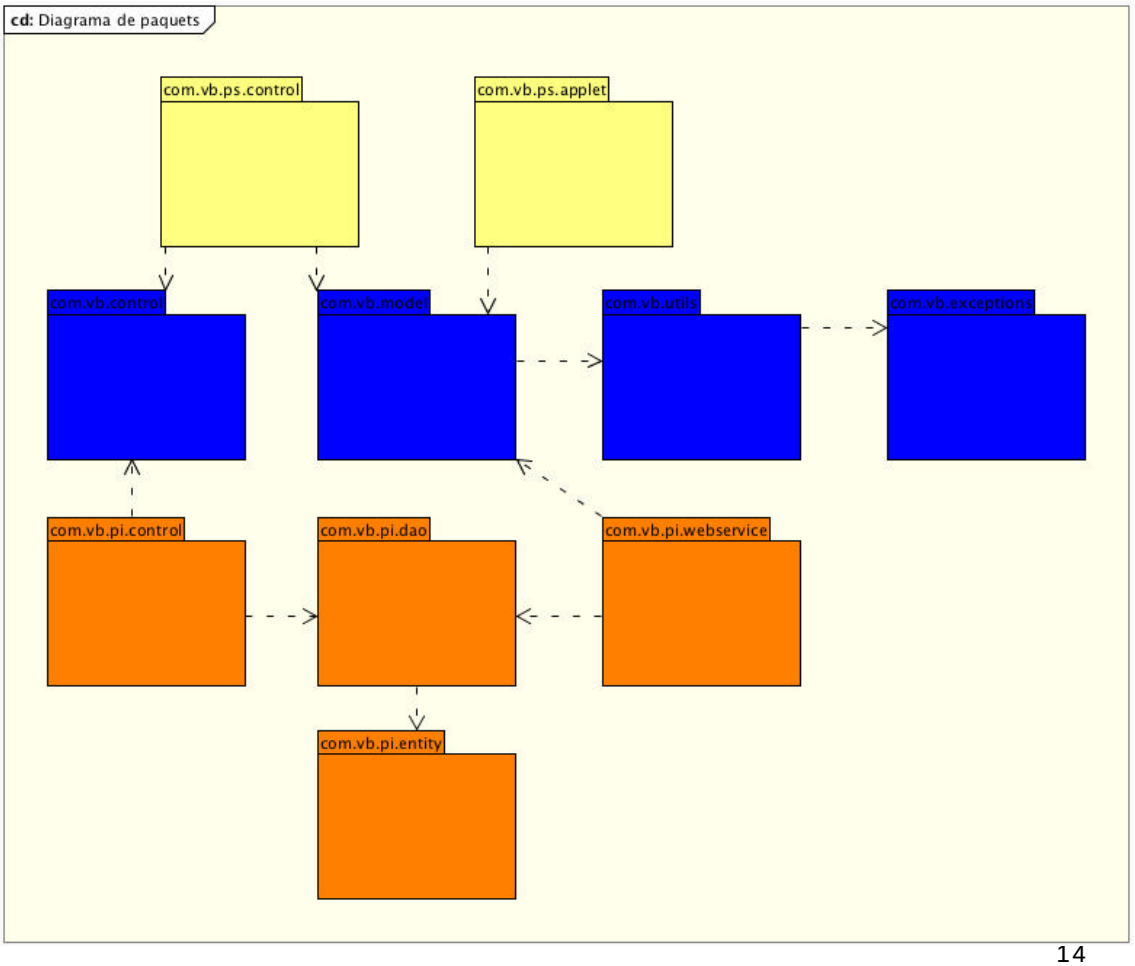
7. No pot accedir cap suport físic amb el certificat de l'actor.

10. Error al accedir al repositori.

14. Verificació del PI incorrecta.

Diagrama de paquets

El diagrama de paquets representa la relació entre els diferents paquets dels mòduls de l'aplicació.



A la primera filera trobem els paquets del proveïdor de serveis, on es troben les actions (com.vb.ps.control) i la classe que instància l'applet.

Al grup central, a la filera de color blau, tenim els paquets comuns. Aquest paquets es troben a 1 dels components, i es comparteixen amb els altres, mitjançant comprimits de les classes (.jar). Aquesta metodologia facilitaria un posterior manteniment, ja que no s'n-pliquen les classes amb funcionalitats molt semblant a diferents aplicatius. Aquí es situen el grup s'excepcions, les classes amb les utilitats genèriques, així com els molt importants paquets, de la part comuna del control, i el model, que seran pressents a tota la execució de l'aplicació.

Finalment tenim els paquets del Proveïdor de Identitat, que seran compartits tant per l'applicació web, com pel servidor de webServices. Aquí cal destacar els paquets com.vb.pi.dao i com.vb.pi.entity, encarregats de l'accés a BBDD.

Diagrama de classes

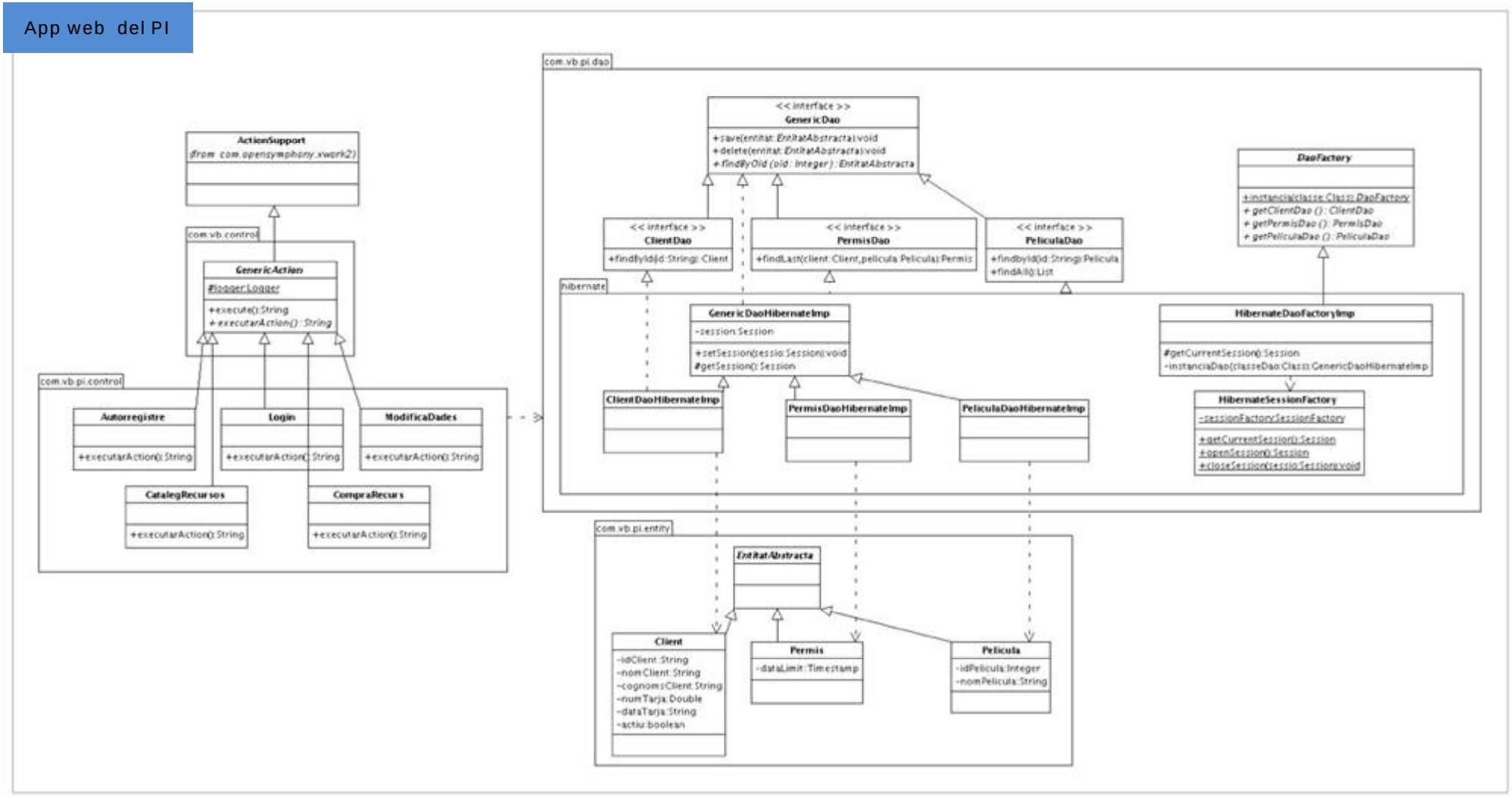
El diagrama de classes és l'eina que s'utilitza per descriure les diferents classes que defineixen els objectes que s'han d'instanciar a una aplicació. També s'expressa com es relacionen les diferentes classes, i la interacció amb components externs a l'aplicatiu.

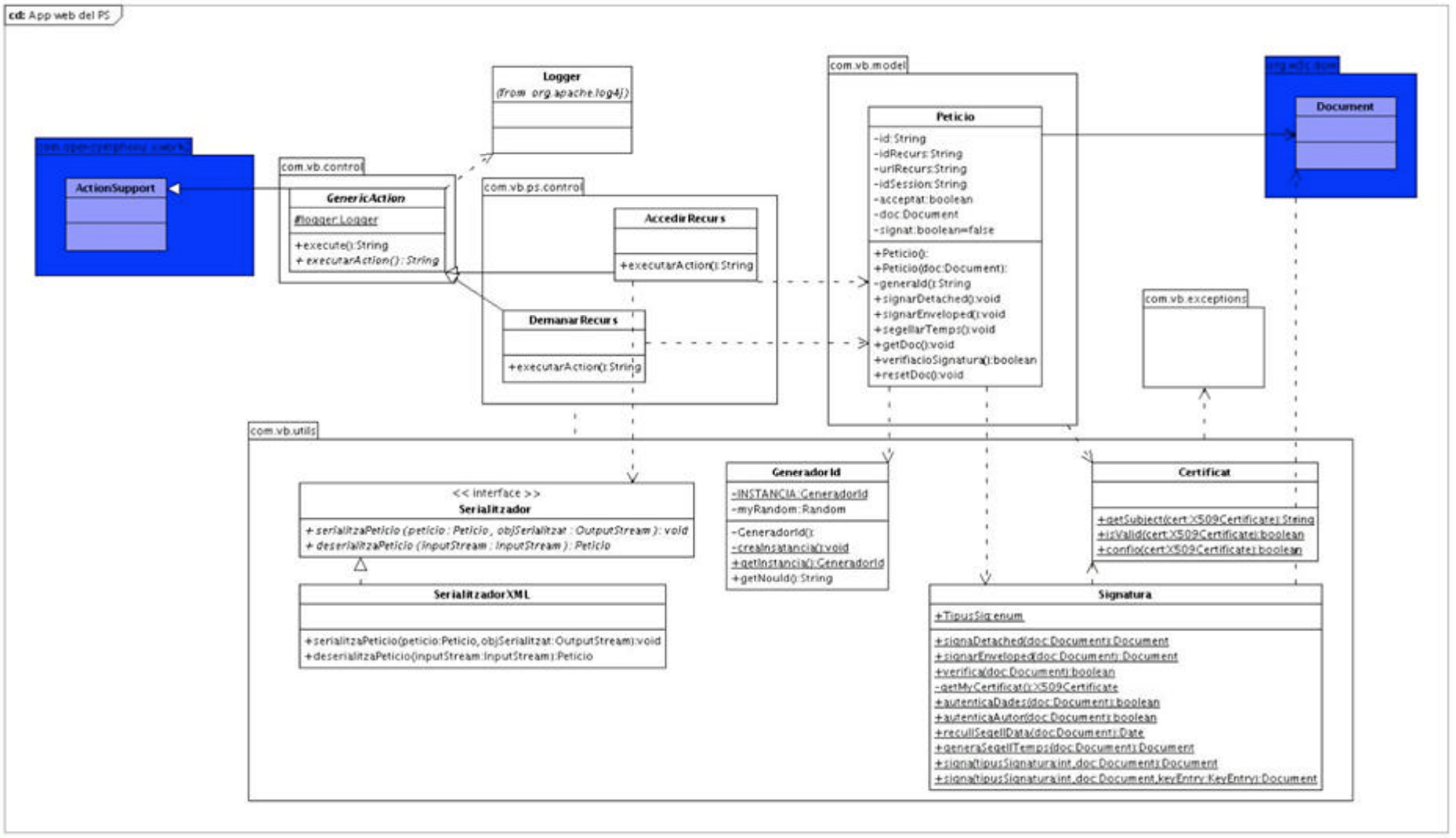
Al diagrama de classes d'aquest treball es podem trobar 4 grups diferenciats, corresponent als 4 subcomponents en que dividim el projecte.

Els 2 primers, *aplicació web del PS* i el *Applet del client*, resideixen al servidor del PS, que enviarà al client l'applet, perquè sigui executat a la màquina del client.

Per una altra banda, els subcomponents *aplicació web del PI* i el *webService*, es troben a les maquines del PI, però es pot donar el cas d'estar en servidors diferents. A aquest treball s'implementarà d'aquesta manera, i per aquesta raó s'ha provocat la separació en 2 subcomponents.

App web del PI




cd: App web del PS
Logger
(from org.apache.log4j)
com.vb.model
Peticio
-id:String
-idRecurs:String
-urlRecurs:String
-idSession:String
-acceptat:boolean
-doc:Document
-signat:boolean=false
+Peticio():
+Peticio(doc:Document):
-generaId():String
+signarDetached():void
+signarEnveloped():void
+segellarTemps():void
+getDoc():void
+verificacioSignatura():boolean
+resetDoc():void
Document
com.vb.control
GenericAction
#logger:Logger
+execute():String
+executarAction():String
ActionSupport
com.vb.ps.control
AccedirRecurs
+executarAction():String
DemanarRecurs
+executarAction():String
com.vb.exceptions
com.vb.utils
<< interface >>
Serialitzador
+serialitzaPeticio (peticio : Peticio , objSerialitzat : OutputStream): void
+deserialitzaPeticio (inputStream : InputStream): Peticio
SerialitzadorXML
+serialitzaPeticio(peticio:Peticio,objSerialitzat:OutputStream):void
+deserialitzaPeticio(inputStream:InputStream):Peticio
GeneradorId
-INSTANCIA:GeneradorId
-myRandom:Random
-GeneradorId():
-creaInstancia():void
+getInstancia():GeneradorId
+getNouId():String
Certificat
+getSubject(cert:X509Certificate):String
+isValid(cert:X509Certificate):boolean
+confia(cert:X509Certificate):boolean
Signatura
+TipusSig:enum
+signaDetached(doc:Document):Document
+signarEnveloped(doc:Document):Document
+verifica(doc:Document):boolean
-getMyCertificat():X509Certificate
+autenticaDades(doc:Document):boolean
+autenticaAutor(doc:Document):boolean
+reculiSegellData(doc:Document):Date
+generaSegellTemps(doc:Document):Document
+signa(tipusSignatura:int,doc:Document):Document
+signa(tipusSignatura:int,doc:Document,keyEntry:KeyEntry):Document

Interfície gràfica

La interfície gràfica és la part de l'aplicatiu on l'usuari te capacitat d'interactuació amb els sistema. Dividim aques apartat en els 2 diferents servidors web. L'applet encarregat de recollir el certificat del client i signar la seva petició, esta inclós dintre l'apartat del proveïdor de servei.

Pel seu desenvolupament he utilitzat la tecnologia jsp (java servlet pages) conjuntament amb les llibreries de tags (taglibs) que incorpora el framework Struts2

Aplicació web del preveïdor de servei

home.jsp
: Mostra una llista de recursos a accedir des d'aquest proveïdor de serveis.

demanarRecurs.jsp
: Pantalla amb applet inclòs que demanarà a l'usuari quin certificat utilitzar per la signa de la petició, així com el PIN del seu DNIe.

accedirRecurs.jsp
: Redirecciona al recurs demanat

accedirRecursError.jsp
: Indica que l'accés al recurs demanat no està autoritzat.

Entitats de persistència

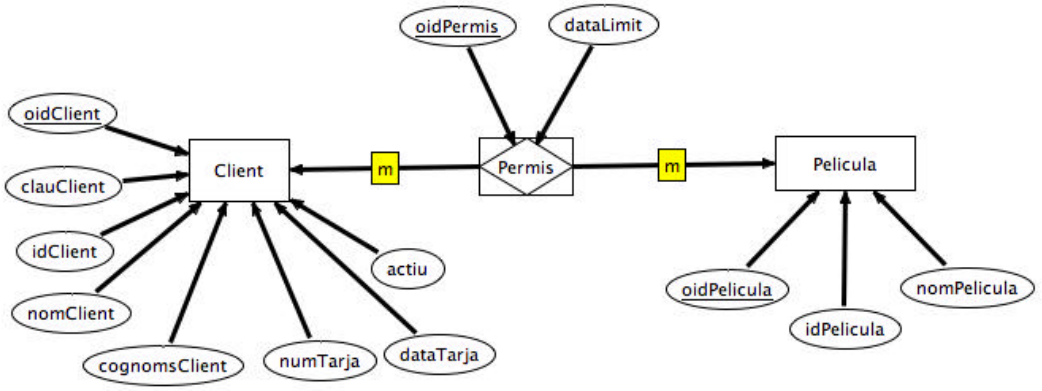
A l'aplicatiu es defineixen 3 entitats de peristència:

Client. Dades del client registrat a la plataforma.

Pelicula. Dades del recurs a accedir.

Permis. Rel·lació Client-Pelicula amb l'atribut dataLimit, que indica quan finalitza el permís d'accés del Client al recurs Pelicula.

La relació entre Client i Pelicula cap a Permis és de *1* a *n*.



Desenvolupament

En el procés de desenvolupament he realitzat diferents modificacions sobre el plantejament inicial.

Un exemple és el canvi de framework utilitzat als servidors web. Degut a l'enderreriment a causa de les tasques de investigació per la implantació de les diferents tecnologies, vaig descartar la opció de utilitzat Spring, i utilitzar Struts2, amb el que ja estava familiaritzat per haver treballat durant anys amb la primera versió. Malauradament, no va ser una gran decisió, ja que el seu funcionament dista més del que em pensava del de la primera versió. En quant a tecnologies, també vaig optat per la versió Axis 1.4, envers l'Axis2, ja que els exemples de connexió amb el servei de signatura de temps funcionaven amb la versió 1.2, i així vaig intentar evitar problemes de incompatibilitat.

Envers al desenvolupament, he realitzat algun canvi funcional com la utilització del DNIe per a la validació dels clients a l'aplicació de lloguer de pel·lícules. El que inicialment era una pantalla de loggin amb usuari i password, va passar a ser un applet que demanava una petició signada, pel correcte registre a la plataforma. Aquest fet em va portar a crear una classe applet genèrica d'on extenen els 2 applets, el del proveïdor de servei i el del proveïdor d'identitat.

La serialització de l'objecte Peticio, el contenidor de les dades i les signatures, inicialment estava plantejat que viatgés serializat en un XML. Degut a les dificultats per verificar la signatura de segell de temps enviant un únic document XML, vaig prendre la decisió de separar les tres signatures a 3 propietats de Peticio, i serialitzar l'objecte en format Base64 per evitar les modificacions que pogués patir al transport per http.

Una altra modificació va ser la migració dels paquets comuns al servidor del proveïdor de identitat, ja que en una suposada implementació real, he trobat més recomanable centralitzar el codi en aquest servidor, que seria el corresponent a la empresa que s'encarregaria del desenvolupament, la empresa vídeo-club.

Pel que fa al model de BBDD, degut a un error a l'anàlisi, vaig incorporar a posteriori l'entitat Proveidor, i obviant la relació amb els recursos per simplificar el desenvolupament. Així he pogut dedicar més temps a tasques més relacionades amb l'àrea que ens ocupa.

Producte

Instal·lació del producte

Per la instal·lació del producte és necessari dispossar del següent software:

- 2 Servidors d'aplicacions Apache Tomcat 6.0

- Sistema Gestor de BBDD MySQL 4.0

- Apache Ant 1.7.1

Configuració de la connexió SSL als servidors Tomcat

L'accés als servidors d'aplicacions es realitzen mitjançant un canal segur. Per aquesta raó s'han d'instal·lar als servidors el seus certificats:

1. Creació del directori *service-provider-server/conf/cert* i copia del certificat en format PKCS#12 (*youstream.p12*)
2. Modificació del fitxer *service-provider-server/conf/server.xml*:
   Descomentar la definició del connector HTTP/1.1 amb SSL

   Agregar als atributs del connector:

   keystoreFile="conf/cert/youstream.p12" keystorePass="tfc2008" keystoreType="PKCS12"

   Exemple:

```
<Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"
           maxThreads="150" scheme="https" secure="true"
           clientAuth="false" sslProtocol="TLS"
           keystoreFile="conf/cert/youstream.p12"
           keystorePass="tfc2008" keystoreType="PKCS12"/>
```

De forma anàloga s'instal·la a l'identity-provider-server el certificat *vodbusters.p12*

Instal·lació al proveïdor d'identitat

Base de dades

Execució de l'script per a la creació de la BBDD, les taules de l'aplicació i càrrega inicial. (veure annexe 1)

Aplicació web

1. Desplegament del fitxer identityProviderWeb.war al servidor *identity-provider-server*
2. Configuració de la BBDD al fitxer *WEB-INF/classes/hibernate.cfg.xml*
3. Configuració del sistema de logs al fitxer *WEB-INF/classes/log4j.properties* (Indicar el path absolut dels fitxers de log: *vb.log* i *warnings_vb.log*)
4. Reiniciar l'applicació.

Aplicació de webservices

1. Desplegament del fitxer identityProviderWebService.war al servidor identity-provider-server
2. Configuració al fitxer d'ant build.xml les propietats del servidor webservice (servidor, port de connexió i nom del contexte)
3. Desplegament del servei web utilitzant la eina Ant. (*axis_build.xml* al directori arrel del context).
   Es complilarà el projecte i es generan els fitxers de configuració Axis .wsdl i .wsdd, així com les classes de connexió.
5. Configuració de la BBDD al fitxer *WEB-INF/classes/vb.properties*
6. Reiniciar l'applicació.

Instal·lació al proveïdor de servei

Aplicació web

1. Desplegament del fitxer serviceProviderWeb.war al servidor *service-provider-server*
2. Configuració del sistema de logs al fitxer *WEB-INF/classes/log4j.properties* (Indicar el path absolut dels fitxers de log: *vb.log* i *warnings_vb.log*)
3. Configuració de l'accés al certificat i al clauer, al fitxer *WEB-INF/classes/*vb.*properties*. El certificat a configurar és l'utilitzat per la signatura de missatges (no pel canal segur SSL), reconegut pel proveïdor de suministra la funcionalitat de segell de temps. Per defecte està al directori *kstore* del contexte
4. Reiniciar el servidor.

Instal·lació al client

Pel funcionament de la connexió SSL, i la correcta validació dels certificats al navegador del client serà necessari:

1. Modificar el fitxer *host* per simular una connexió remota als dominis que registren els certificats.
   Incloure al fitxer:

   127.0.0.1 www.vodbusters.es

   127.0.0.1 www.youstream.es

   (o la IP dels servidors on estiguin instal·lades les aplicacions)

2. Instal·lació del certificat de la Autoritat Certificadora *tfCAuthorityCert.pem* a cada navegador (Depenent del sistema operatiu)

Funcionament del producte

Tal com he plantejat el TFC, l'aplicació serviria per l'accés restringit a pel·lícules de lloguer amb el marc de Video On Demand (VOD). Per tant el proveïdor de identitat passa a ser el video-club que te tota la informació dels seus clients, i és qui gestiona els pagaments dels lloguers de pel·lícules. A aquest video club l'he anomenat VodBusters. Per altra banda, tenim un proveïdor de serveis que seria el proveïdor de video-streaming, qui s'encarrega de subministrar als clients, les pel·lícules que han llogat. A aquest proveïdor li dierem YouStream. I finalment, dir que amb aquest sistema, un client que lloga una pel·lícula, podria tenir accés des de qualsevol ordinador amb connexió a internet de banda ample, portant únicament el seu DNIe y un lector de targes intel·ligents.

Dividirem l'execució en l'accés a les 2 aplicacions web.

YouStream. Visualització de pel·lícules

La execució del producte comença al accedint a la web de YouStream (http://www.youstream.es:8080/serviceProviderWeb), on es llisten les pel·lícules a accedir. Seleccionant la primera, que és la que permet la execució complerta de la demo, s'està demanant a YouStream permís per accedir a la visualització de la pel·lícula. El servidor genera un identificador únic (classe GeneradorId) al crear l'objecte Peticio. Aquest objecte serà l'encarregat a tot el procés de encapsular les diferents dades de la petició i les signatures adients a cada moment del procés. Amb les dades generades (id de petició, url del recurs a accedir, identificador de la sessió de java i nom del fitxer on és la pel·lícula, que ens servirà com a identificador de recurs), l'objecte es signa les seves dades utilitzant l'estándar XMLDsig. Aquesta signatura s'envia mitjançant un servei web Axis, al proveïdor que actualitzarà la signatura amb un segell de temps, per deixar constància del moment que s'ha realitzat la petició.

L'objecte petició viatja serialitzat com a paràmetre de l'applet que s'executarà al navegador del client. Aquest applet accedeix al keystore del DNIe per signar les dades de la Petició. Per la correcta execució de l'applet, s'ha de disposar d'un navegador en entorn Windows, amb la JRE 6.0 de Java instal·lada. Això és degut a que la API de java utilitzada per singar amb el certificat del client, només està inclosa a aquesta versió. Aquest accés al certificat del DNIe és una mica lent i ferragós, ja que demana 2 vegades el PIN a l'usuari. El tema de la lentitud també pot ser causat perquè l'entorn on realitzo les proves es un host virtual de Windows dintre la meva màquina que treballa amb OS X. El cert es que no he realitzat les proves a un Windows nadiu.

Un cop signades les dades, es connecta mitjançant un webservice al servidor de VodBusters, per validar si aquell client té la pel·lícula llogada en aquell moment.

Aquesta validació consisteix en comprovar que les dades no han sigut alterades, mitjançant la signatura, que el proveïdor ha signat les dades amb una autoritat certificadora TS, que la url de la petició correspon a un dels proveïdors que te registrats a la BBDD, que el certificat del DNIe no ha sigut revocat, mitjançant una validació OCSP, i que el client té llogada la pel·lícula a la que vol accedir en el moment de realitzar la petició. Per evitar problemes de canvi horari a diferents parts del món, com per exemple si el client està a Austràlia i el proveïdor de video-streaming està a EEUU, he utilitzat l'estandar de la franja horària central europea (CET). De manera que les hores límit per accedir els clients a visualitzar les pel·lícules, es passen a format CET abans de ser persistides. I al recuperar-les també es té en conte per comparar-les amb l'hora en que s'ha realitzat la petició.

L'accés a la BBDD l'he realitzat mitjançant les llibreries ODBC, malgrat al disseny estava plantejat fer-ho amb hibernate. Això és degut a un problema de compatibilitat entre Axis 1.4 i l'Hibernate. Una opció que es podria implementar en l'inici d'una etapa de manteniment, en un cas real, seria realitzar un servlet que recollís la petició SOAP i la gestionés sense necessitat d'Axis.

Si l'usuari ha realitzat la petició durant les 48 hores següents al lloguer de la pel·lícula, i totes les validació han sigut possitives, VodBuster envia una petició amb la conformitat de l'accés, signada amb el seu certificat emès per tfCAutority (el nostre CA), i envia la petició al Client, ara sense les signatura de YouStream ni de l'usuari.

L'applet del client, revisa la Peticio rebuda i informa a l'usuari si ha sigut acceptat o denegat l'accés, i la reenvia a YouStream.

Quan es rebuda al servidor, es verifica que les dades són íntegres, que l'id de la sessió de java correspon al de la sessió en curs, que VodBuster ha acceptat l'accés, i finalment, que confia en el certificat del l'emissor. Per aquesta última comprovació es compara el DN de la autoritat certificadora del certificat que signa el missatge, amb el que certifica el seu propi. Degut a que tots dos han sigut emesos per tfCAutority, YouStream confia en la signatura.

En aquest punt es dóna o es denega l'accés a la pel·lícula segons hagi decidit l'entitat VodBuster.

Per aquesta última part, he observat una millora que m'ha sigut impossible d'implementar degut a l'ús de Struts2. El client faria directament la petició al recurs que desitja, i seria un filtre al servidor de YouStream qui comprovaria si ha sigut realitzada tota la validació, i es pot donar accés al video, o si ha de redireccionar cap a l'acción de petició de recurs per iniciar tot el procés. El cert es que sembla ser que no es pot realitzar amb la meva configuració, degut a que Strut2 dispatxa el seu servlet abans que qualsevol altre, encara que es canviï l'ordre al web.xml, i un cop executat el nostre filtre, no hi ha manera que Struts cachegi el nostre redireccionament des de el filtre. He deixat l'exemple del filtre al paquet com.vb.ps.filtres a l'aplicació del proveïdor de servei.

Vodbuster. Aplicació de lloguer de pel·lícules.

En aquesta aplicació s'integra el lloguer de les pel·lícules, així com l'autoregistre i la modificació de les dades del Client. Pel seu desenvolupament s'ha implementat el model Data Acces Object (DAO), per aïllar totalment el negoci de la BBDD utilitzada. En cas que fos necessari una migració cap a un repositori de dades diferent, només s'hauria de implementa les classes DAO amb els accessos a la BBDD segons dicten les inteficies d'accés (paquet com.vb.pi.dao).

El registre del usuari a la plataforma, utilitza el DNIe. El client reb un applet, que crearà i signarà un objecte Petició amb la única dada del identificador únic de la petició. Un cop rebuda la petició, el valida i es valida l'usuari utilitzant el DN del certificat X509 enviat a la petició.

Un cop l'usuari registrat, pot operar lliurement, ja que la seva validació de registre resta a la sessió de la httprequest.

Conclusions

Un cop finalitzat el desenvolupament d'aquest TFC, puc dir que em porto 2 grans aprenentatges. Per una banda que a la valoració del treball necessari per una implementació informàtica, és molt i molt important tenir en compte els coneixements que es tenen de les eines a utilitzar. Ja siguin les tecnologies que intervenen, com les eines de gestió.

El segon gran aprenentatge ha sigut totes les eines i tecnologies que he conegut o he consolidat els meu coneixements, així com un guia que anat fent per no deixar a un calaix les hores de treball i de investigació.

Com a crítica podria dir que penso que és ha sigut un projecte molt gran per ser un TFC. Inicialment pensava que en unes 300 hores el podria enllestir, i val a dir que han sigut unes quantes més. Possiblement m'hagi complicat més del compte, però així és com he entès el que es demanava.

Però en general he gaudit motl, i ha sigut molt enriquidor el fet d'enfrontar-me a un desenvolupament de 0, escollint les tecnologies i investigant per la seva implantació. Potser un tercer gran aprenentatge seria la capacitat per cercar informació, i la importància de ser metòdic a les tasques per evitar problemes durant el desenvolupament.

Bibliografia

Webs

JavaJava™ Platform, Standard Edition 6. API Specification

http://java.sun.com/javase/6/docs/api/

Struts

http://struts.apache.org/2.x/docs/guides.html

http://struts.apache.org/2.x/docs/tag-reference.html

Safelayer. Signatura de temps

http://labs.safelayer.com

Tutorial creació d'Applets

http://pisuerga.inf.ubu.es/lsi/Invest/Java/Tuto/VI_2.htm

Applets. How Java to Javascript Communication Works in Java Plug-in.

http://java.sun.com/products/plugin/1.3/docs/jsobject.html

Signar applets.

http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=app_keyt_jars

SSL. Instal·lació a Tomcat

http://tomcat.apache.org/tomcat-4.0-doc/ssl-howto.html

http://cosasdelinux.wordpress.com/2008/05/27/instalar-ssl-para-tomcat/

Tomcat. Configuració d'un Virtual Host

http://publib.boulder.ibm.com/iseries/v5r2/ic2924/info/rzaie/rzaie9setvh.htm

Tomcat wiki

http://wiki.apache.org/tomcat/HowTo

Sun. Java ™ Cryptography Architecture (JCA) Reference Guide

http://java.sun.com/javase/6/docs/technotes/guides/security/crypto/CryptoSpec.html

http://java.sun.com/javase/6/docs/technotes/guides/security/StandardNames.html

Sun. Java XML Digital Signatures

http://java.sun.com/developer/technicalArticles/xml/dig_signatures/

W3C. XML Signature Syntax and Processing

http://www.w3.org/TR/xmldsig-core/

Document Object Model (DOM). Tutorial d'introducció

http://www.latascadexela.es/2008/07/java-y-xml-dom-i.html

ANT

http://ant.apache.org/manual/index.html

http://ant.apache.org/manual/CoreTasks/

Patrón Singleton

http://es.wikipedia.org/wiki/Singleton

Relaciones en JPA

http://debugmodeon.com/item/627/relaciones-en-jpa

Hibernate. Tutorial d'introducció.

http://www.ada.com.co/index.php?option=com_content&task=view&id=67&Itemid=19

Hibernate

http://www.hibernate.org/

MYSQL. Referencia de SQL

http://dev.mysql.com/doc/refman/5.0/en/sql-syntax.html

Webservices. Introducció.

http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=soap

Axis. Guia d'usuari.

http://ws.apache.org/axis/java/user-guide.htm

Especificació RFC 2560 (protocol OCSP)

http://www.ietf.org/rfc/rfc2560.txt

Tutorial d'accès al DNIe

http://thinkincode.net/2007/2/14/java-6-y-tu-dni-electronico

Criptografia

http://www.kriptopolis.org

Llibres

David Hook, *Beginning Cryptography with Java*

Kathy Sierra, BertBates, *Sun Certified Programmer for Java 6 Study Guide.* Mc Graw Hill

Annexos

Annex 1. Creació i càrrega inicial de BBDD.

S'haurà de tenir en compte la url del instal·lació del proveïdor de serveis per l'insert a la taula de proveidors

```
DROP DATABASE IF EXISTS VODBUSTERS;
CREATE DATABASE IF NOT EXISTS VODBUSTERS;

USE VODBUSTERS;

-- CLIENTS

DROP TABLE IF EXISTS CLIENTS;
CREATE TABLE CLIENTS
(
    OIDCLIENT INTEGER ( 5 ) NOT NULL AUTO_INCREMENT,
    IDCLIENT VARCHAR ( 200 ) UNIQUE,
    NOMCLIENT VARCHAR ( 20 ),
    COGNOMSCLIENT VARCHAR ( 60 ),
    NUMTARJA DOUBLE,
    DATATARJA VARCHAR ( 5 ),
    ACTIU BOOLEAN ,

    PRIMARY KEY ( OIDCLIENT )
);

-- PERMISOS

DROP TABLE IF EXISTS PERMISSOS;
CREATE TABLE PERMISSOS
(
    OIDPERMIS INTEGER ( 5 )  NOT NULL UNIQUE AUTO_INCREMENT,
    DATALIMIT TIMESTAMP NOT NULL,
    FKOIDPELICULA INTEGER ( 5 ) NOT NULL,
    FKOIDCLIENT INTEGER ( 5 ) NOT NULL,

    FOREIGN KEY (FKOIDPELICULA) REFERENCES PEL_LICULES,

    FOREIGN KEY (FKOIDCLIENT) REFERENCES CLIENTS,

    PRIMARY KEY ( OIDPERMIS,FKOIDPELICULA,FKOIDCLIENT )
);

-- PROVEIDORS
DROP TABLE IF EXISTS PROVEIDORS;
CREATE TABLE PROVEIDORS
(
    SUBJECT VARCHAR ( 200 ) NOT NULL ,
    URL VARCHAR ( 60 ) NOT NULL,
    NOM VARCHAR ( 60 ),

    PRIMARY KEY ( SUBJECT )
);


-- PEL_LICULES
DROP TABLE IF EXISTS PEL_LICULES;
CREATE TABLE PEL_LICULES
(
    OIDPELICULA INTEGER ( 5 )  NOT NULL AUTO_INCREMENT,
    IDPELICULA VARCHAR ( 45 ) NOT NULL UNIQUE,
    NOMPELICULA VARCHAR ( 80 ) NOT NULL,

    PRIMARY KEY ( OIDPELICULA )
);
```

```
----- CARREGA INICIAL -----

-- TAULA `PEL_LICULES`
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (1,
'requiem_for_a_dream.m4v', 'Requiem for a dream');
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (2,
'lethal_weapon_3.m4v', 'Lethal Weapon 3');
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (3,
'batman_returns.m4v', 'Batman Returns');
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (4,
'unforgiven.m4v', 'Unforgiven');
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (5,
'the_bodyguard.m4v', 'The Bodyguard');
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (6,
'under_siege.m4v', 'Under Siege');
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (7,
'free_willy.m4v', 'Free Willy');
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (8,
'dennis_the_menace.m4v', 'Dennis the Menace');
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (9,
'body_snatchers.m4v', 'Body Snatchers');
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (10,
'batman_mask_of_the_phantasm.m4v', 'Batman: Mask of the Phantasm');
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (11,
'the_secret_garden.m4v', 'The Secret Garden');
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (12,
'the_fugitive.m4v', 'The Fugitive');
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (13,
'the_hudsucker_proxy.m4v', 'The Hudsucker Proxy');
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (14,
'the_client.m4v', 'The Client');
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (15,
'thumbelina.m4v', 'Thumbelina');
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (16,
'a_troll_in_central_park.m4v', 'A Troll in Central Park');
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (17,
'little_giants.m4v', 'Little Giants');
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (18,
'interview_with_the_vampire.m4v', 'Interview with the Vampire');
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (19,
'police_academy_mission_to_moscow.m4v', 'Police Academy: Mission to Moscow');
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (20,
'natural_born_killers.m4v', 'Natural Born Killers');
INSERT INTO `PEL_LICULES` (`OIDPELICULA`, `IDPELICULA`, `NOMPELICULA`) VALUES (21,
'the_pink_panther_2.m4v', 'The Pink Panther 2');

-- TAULA `PROVEIDORS`
INSERT INTO `PROVEIDORS` (`SUBJECT`, `URL`, `NOM`) VALUES ('EA=xaviseg@gmail.com,
CN=Xavier Segura, OU=IT, O=YouStream, C=ES', 'https://192.168.1.3:8443', 'You Stream');
```