Disseny i desenvolupament d'un esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions

Josué Rodríguez Garduño
Enginyeria en Informàtica

Jordi Castellà Roca

11 de juny de 2008

A l'Izan que, tot i que ell no ho sap encara, m'ha donat la força i empenta necessària

A la Kesía pel seu suport incondicional i comprensió

A la família pel seu recolzament

Gràcies a en Jordi per la seva ajuda al projecte, la fita tant desitjada

RESUM
Les dades clíniques tenen una importància màxima tant per el personal sanitari com pels propietaris d'aquesta informació, els pacients. Es contraposen aquí el fet de la sensibilitat i privadesa de les dades tractades versus la necessitat d'un accés ràpid al màxim volum de dades d'un pacient per a determinar un diagnòstic i tractament adequat en el mínim temps.

La tecnologia actual permet disposar de repositoris electrònics amb aquesta tipologia de dades accessibles local i remotament. Però aquestes facilitats no han vulnerar els drets bàsics dels pacients en el referent a les seves dades mèdiques privades.

El PFC té com objectiu dissenyar i implementar un esquema criptogràfic per a garantir un accés segur a les dades proporcionant mecanismes per salvaguardar la confidencialitat, autenticitat i integritat de les dades i no repudi de les accions portades a terme pels usuaris.

El producte té dos principals rols d'usuaris:

- *Metges*
  Disposaran d'accés de lectura i escriptura al historial clínic dels pacients assignats i podran actualitzar les dades deguts als esdeveniments produïts durant el temps (visites)

- *Pacients*
  Podran consultar de les seves dades personals i mèdiques

La seguretat del sistema serà responsabilitat d'un component de programari nucli de l'aplicació servidor: el Gestor del Sistema. Aquesta entitat serà l'encarregada, a més de la gestió pròpia, de donar accés segur i controlat a les dades a ambdós grups d'usuaris.

La solució tecnològica estarà basada en un entorn Java per la capa presentació i lògica de negoci i un SGBD relacional per la capa de dades.

PARAULES CLAU
historial, pacient, metge, esquema criptogràfic, criptografia, seguretat, confidencialitat, autenticitat, integritat, no repudi, protocol, PKI, certificat, RMI, IAIK, XML, MySQL, SWT.

ÀREA
Seguretat Informàtica

Índex

Índex de les figures

1. INTRODUCCIÓ

1.1 Justificació del PFC i context en el qual es desenvolupa: punt de partida i aportació del PFC

Uns dels actius més valuosos de les persones són les seves dades. És, per tant, prioritat primordial salvaguardar la informació personal sigui quin sigui el medi en el que s'emmagatzema o el mètode d'accés. Per altra banda, l'èxit en la presa de decisions depèn en un alt percentatge a tenir disponible tota la informació possible, és a dir, les dades, respecte a l'assumpte a tractar. Les xarxes de comunicacions ens permeten accedir a un gran volum d'informació molt ràpidament sense la necessitat de desplaçar-nos, i amb independència de l'instant de temps.

Aquestes afirmacions són especialment crítiques en un entorn hospitalari on el personal facultatiu pot disposar d'avantatges que reportaran en beneficis al seu pacient si es proporciona d'un accés a un complet i detallat historial mèdic. L'èxit del diagnòstic i tractament dependrà en bona part de les dades disponibles.

No obstant, les dades clíniques d'un pacient són altament confidencials i, per tant, s'ha de protegir la confidencialitat al ser accedides exclusivament pel propi pacient o pel personal mèdic que tracti el cas. Degut al tipus d'informació i les repercussions de les accions a prendre pels facultatius, les dades mèdiques tenen un gran valor. Això implica un especial interès per protegir la integritat i la autenticitat de les mateixes.

La legislació obliga a garantir la confidencialitat, integritat i autenticitat de les dades personals emmagatzemades en medis electrònics. Son referència clau aquí la Llei Orgànica 15/1999, de 13 de desembre, de Protecció de Dades de Caràcter Personal [LOPD99] on es garanteix i protegeix els drets d'honor e intimitat personal i familiar en el tractament de dades personals, i el Reial Decret 994/1999, de 11 de juny, de Mesures de Seguretat dels fitxers automatitzats que continguin dades de caràcter personal [RDMS99] el qual té per objecte establir les mesures tècniques i organitzatives per garantir la seguretat dels sistemes que tracten dades personals. I, com a última referència, el 21 de desembre de 2007 s'aprova el Reial Decret 1720/2007 de Desenvolupament de la Llei Orgànica 15/1999, de 13 de desembre, de protecció de dades de caràcter personal [RDDL07] que estarà en vigor 3 mesos després de la publicació al BOE.

Aquesta legislació estatal gira entorn a un dret fonamental del ciutadà delimitat als articles 18.1 i 18.4 de la Constitució Espanyola de 1978 [CONS78] on es declara: "Es garanteix el dret a l'honor, a la intimitat personal i familiar i a la pròpia imatge" i també: "La llei limitarà l'ús de la informàtica per a garantir l'honor i la intimitat personal i familiar dels ciutadans i el ple exercici dels seus drets".

Queda clar, doncs, que tot sistema informàtic que tracti dades mèdiques, fortament sensibles i privades, haurà d'implementar les mesures necessàries per tal de garantir els drets indicats anteriorment. No obstant això, és ben cert que la actual tecnologia i comunicacions ens ofereix les facilitats per accedir a una gran quantitat d'informació en format electrònic des de quasi qualsevol punt del planeta. És aquí on es planteja aplicar aquests avenços al camp mèdic per tal de donar accés a historials mèdics en format electrònic d'una forma ràpida i segura, però sense faltar als drets abans esmentats.

1.2 Objectius del PFC

L'objectiu d'aquest PFC és dissenyar i implementar un esquema criptogràfic que garanteixi les necessitats de seguretat d'un historial mèdic accedit i gestionat a través d'una xarxa de comunicacions.

Les propietats de seguretat a assegurar en tots els serveis que el sistema produït proporcioni són:

- **Confidencialitat**
  S'ha de preservar la privadesa de les dades dels historials mèdics

- **Autenticitat**
  Es requereix una prova d'autenticitat de les dades emmagatzemades al sistema

- **Integritat**
  S'ha de garantir que la informació dels historials no serà alterada un cop guardada.

- **No Repudi**
  Es necessari disposar de mecanismes per evitar la negació d'accions realitzades pels usuaris del sistema

Els requisits funcionals principals que ofereix el sistema són:

a) Autenticació de usuaris
Tot usuari registrat s'autentica de forma segura per tal de confirmar la seva identitat davant el sistema i s'identifica dintre del conjunt de rols disponibles: pacient o metge.

b) Registre d'usuaris
El sistema permet l'alta d'usuaris amb rol pacient i/o metge.

c) Consultes
Els pacients poden consultar:
- Les dades del seu historial.

Els metges poden consultar:
- Les visites del historial d'un pacient assignat o, temporalment, a les visites d'un pacient derivat per altre metge.
- Les dades generals d'un pacient registrat.
- La llista de pacients assignats.

d) Modificacions
Els metges poden afegir:
- Visites noves als historials del seus pacients. En cap cas, pot modificar dades de visites, diagnosi o tractament realitzades anteriorment.

El gestor central pot modificar:
- Les dades generals dels pacients.

e) Eliminacions
Els metges no poden eliminar les dades de l'historial d'un pacient.
El gestor central pot donar de baixa usuaris amb rol pacient i/o metge.

1.3 Enfocament i mètode seguit

El PFC ha estat dividit en vuit fases a executar de forma seqüencial de forma que el desenvolupament del producte final ha seguit un mètode incremental. Després de la construcció i finalització d'una fase s'han efectuat les corresponents proves unitàries i d'integració per tal de verificar el bon funcionament de forma independent i en dependència dels components ja construïts.

Les fases assolides han estat:

1. Preparació de l'entorn de treball
2. Esquema criptogràfic
3. Representació de les dades
4. Comunicació dels components
5. Gestió de la informació
6. Interfície client
7. Interfície del gestor del sistema
8. Documentació

1.4 Planificació del projecte

El projecte ha seguit la planificació marcada en el llançament pel Consultor de Projecte en el que es fixava la data 11 de juny de 2008 com a fita per a fer el lliurament del PFC. La data d'inici va ser 28 de febrer de 2008.

La planificació seguida ha estat:

- **Fase 1:** Setmana 1 i 2 (del 28 de febrer a 5 de març)
  - Configuració de l'entorn de treball
  - Instal·lació de PKI
  - Generació dels certificats pels usuaris del sistema

- **Fase 2:** Setmana 2 a 5 (del 6 al 30 de març)
  - Disseny, Implementació, Proves i Documentació de l'esquema criptogràfic

- **Fase 3:** Setmana 6 i 7 (del 31 de març al 13 d'abril)
  - Disseny, Implementació, Proves i Documentació de la Representació de dades

- **Fase 4:** Setmana 8 i 9 (del 14 al 27 d'abril)
  - Disseny, Implementació, Proves i Documentació de la Comunicació entre components

- **Fase 5:** Setmana 10 i 11 (del 28 d'abril al 11 de maig)
  - Disseny, Implementació, Proves i Documentació de la Gestió de la Informació (base de dades)

- **Fase 6:** Setmana 12 i 13 (del 12 al 25 de maig)
  - Disseny, Implementació, Proves i Documentació de la Interfície client (pacient i metge)

- **Fase 7:** Setmana 14 (del 26 de maig al 1 de juny)
  - Disseny, Implementació, Proves i Documentació de la Interfície del Gestor del Sistema

- **Fase 8:** Setmana 15 i 16 (del 2 al 10 de juny)
  - Confecció i revisió de la Memòria
  - Confecció de la presentació per la defensa del PFC

- **Lliurament** del PFC (11 de juny)

A la següent figura es mostra la planificació de forma gràfica:

Id	Nombre de tarea	Duración	Comienzo
1	**PFC Seguretat**	**75 días**	**jue 28/02/08 9:00**
2	Publicació enunciat	0 días	jue 28/02/08 9:00
3	**Fase 1: Instal·lació IAIK i PKI**	**5 días**	**jue 28/02/08 9:00**
4	Configuració entorn de treball	2 días	jue 28/02/08 9:00
5	Instal·lació Pki	1 día	lun 03/03/08 9:00
6	Generació certificats usuaris	2 días	mar 04/03/08 9:00
7	Entorn de treball preparat	0 días	mié 05/03/08 19:00
8	**Fase 2: Esquema criptogràfic**	**17 días**	**jue 06/03/08 9:00**
9	Disseny, Implementació i Proves	17 días	jue 06/03/08 9:00
10	Esquema criptogràfic enllestit	0 días	vie 28/03/08 19:00
11	**Fase 3: Representació de les dades**	**10 días**	**lun 31/03/08 9:00**
12	Disseny, Implementació i Proves	10 días	lun 31/03/08 9:00
13	Representació dades XML enllestit	0 días	vie 11/04/08 19:00
14	**Fase 4: Comunicació de components**	**10 días**	**lun 14/04/08 9:00**
15	Disseny, Implementació i Proves	10 días	lun 14/04/08 9:00
16	Comunicació RMI enllestida	0 días	vie 25/04/08 19:00
17	**Fase 5: Gestió de la informació**	**10 días**	**lun 28/04/08 9:00**
18	Instal·lació i configuració SGBD MySQL	1 día	lun 28/04/08 9:00
19	Disseny, Implementació i Proves	9 días	mar 29/04/08 9:00
20	Gestió de la informació enllestida	0 días	vie 09/05/08 19:00
21	**Fase 6: Interfície client**	**10 días**	**lun 12/05/08 9:00**
22	Disseny, Implementació i Proves	10 días	lun 12/05/08 9:00
23	Interfície pacient i metge enllestida	0 días	vie 23/05/08 19:00
24	**Fase 7: Interfície gestor**	**5 días**	**lun 26/05/08 9:00**
25	Disseny, Implementació i Proves	5 días	lun 26/05/08 9:00
26	Interfície del gestor del sistema enllestida	0 días	vie 30/05/08 19:00
27	**Fase 8: Documentació**	**8 días**	**lun 02/06/08 9:00**
28	Confecció Memòria de Projecte	8 días	lun 02/06/08 9:00
29	Documentació enllestida	0 días	mié 11/06/08 19:00
30	**Lliurament PFC**	0 días	mié 11/06/08 19:00

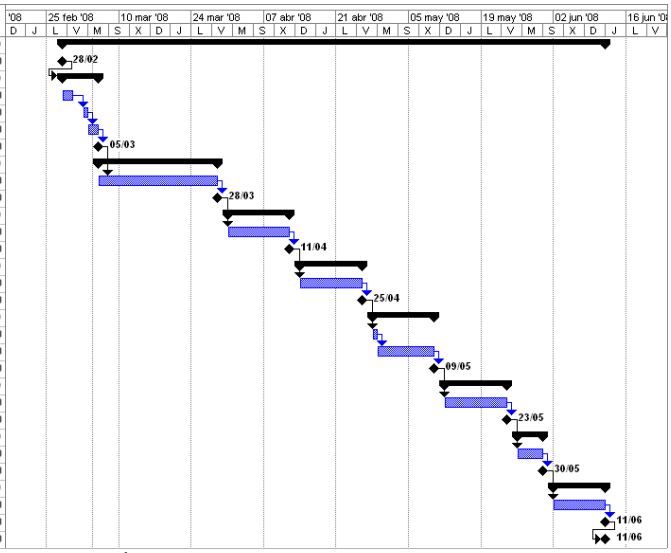

Figura 1. Planificació del PFC

Només s'ha sofert un petit desviament de dos dies en l'entrega de la fase 5 que es va produir el 13 de maig.

1.5 Productes obtinguts

Els sistema es compon de tres components principals:

- **Aplicació Metge**
  Programari utilitzat per un usuari amb rol metge per a accedir de forma segura al gestor del sistema. Permet la consulta i inserció de informació d'un usuari pacient.

- **Aplicació Pacient**
  És el programari que utilitzarà un pacient per a accedir de forma segura al gestor del sistema amb els requeriments funcionals definits en el projecte. Permet la consulta de les dades del historial mèdic del pacient.

- **Gestor Central**
  Programari encarregat de la gestió del repositori d'historials mèdics de forma centralitzada. És responsable també de l'autenticació, identificació i gestió dels accessos a les dades especialment protegides.

1.6 Descripció dels següents capítols de la memòria

Als següents capítols es descriuen les decisions de disseny, arquitectura i implementació dels components necessaris del projecte, estructurats de la següent forma:

Capítol 2. Arquitectura del sistema
S'assenyalen el disseny de tots els components que formen el sistema així com les justificacions de les decisions presses a cada fase.

Capítol 3. PKI
Per el que es veurà al capítol 4 es necessitarà l'ús de tècniques de criptografia de clau pública. Això provocarà la creació de la infraestructura de clau pública (PKI) per a l'emissió i gestió dels certificats utilitzats.

Capítol 4. Esquema criptogràfic
La necessitat d'un conjunt de protocols criptogràfics per a garantir les propietats de seguretat requerides a les funcionalitats demanades configuren l'esquema criptogràfic pel sistema desenvolupat. Aquest apartat explica detalladament què i com s'ha implementat per a aconseguir aquest objectiu.

Capítol 5. Representació de les dades
En el PFC es defineix la utilització de XML (*eXtensible Markup Language*) per l'intercanvi de la informació en l'execució dels protocols criptogràfics. Aquest capítol es dedica en la explicació del disseny i la implementació relatiu a aquest component del sistema.

Capítol 6. Comunicació dels components
La comunicació dels diferents components és una part important del projecte. Aquest apartat explicarà per què l'ús del protocol *Remote Method Invocation* (RMI) de Java és un avantatge en la reducció del temps de desenvolupament i com s'ha aplicat a la comunicació entre la part client i la part servidora.

Capítol 7. Gestió de la informació
La base de dades és necessària per fer persistent les dades dels historials mèdics. El capítol explicarà tots els detalls respecte al SGBD escollit: MySQL.

Capítol 8. Interfície del pacient
S'explicarà el disseny i la implementació de les funcionalitats requerides per la aplicació client pel usuari de tipus pacient.

Capítol 9. Interfície del metge
Igual que capítol anterior, es farà focus al disseny i implementació de l'aplicació client per als usuaris amb rol de metge.

Capítol 10. Interfície del gestor del sistema
Aquest capítol descriurà les decisions de disseny i tasques d'implementació de la part servidora: el gestor del sistema. Aquest component principal serà el màxim responsable per a garantir les propietats de seguretat del sistema i l'accés a les dades confidencials dels pacients per part de la comunitat d'usuaris.

Capítol 11. Joc de Proves
Es descriurà el Pla de Proves realitzat en cadascuna de les fases del projecte i una comparació entre els resultats esperats i els obtinguts al realitzar les proves unitàries i d'integració.

Capítol 12. Conclusions
S'exposarà com s'han assolit els objectius marcats del PFC, quin treball futur es pot plantejar per a ampliar la solució i les consideracions i experiència del projectista en el treball realitzat durant el projecte.

2. ARQUITECTURA DEL SISTEMA

El sistema segueix una estructura Client/Servidor on les aplicacions Pacient i Metge conformen la part client i el Gestor Central del sistema és l'encarregat de fer el rol de servidor.

El sistema està dividit en tres capes o nivells ben diferenciades: presentació, lògica de negoci i dades:

1. **Capa Presentació:** Es compon de les aplicacions Pacient i Metge on disposen de les funcionalitats principals del sistema. És la interfície dels usuaris

2. **Capa Lògica de Negoci:** Es implementada pel Gestor Central del sistema. S'encarrega de servir les peticions de la capa de presentació, aplicant les regles definides, i servir-les a partir de les dades obtingudes a la capa inferior (capa de dades).

3. **Capa de Dades:** Es on resideix la representació de les dades del sistema. En aquest cas es guardarà únicament les dades referent a les visites realitzades dissociades de qualsevol dades de pacient[1].

Amb aquesta arquitectura es garanteix que des de la capa de presentació no es tingui accés directe a les dades i sempre es requereixi el control de la capa de negoci per tal de gestionar les peticions dels usuaris del sistema.

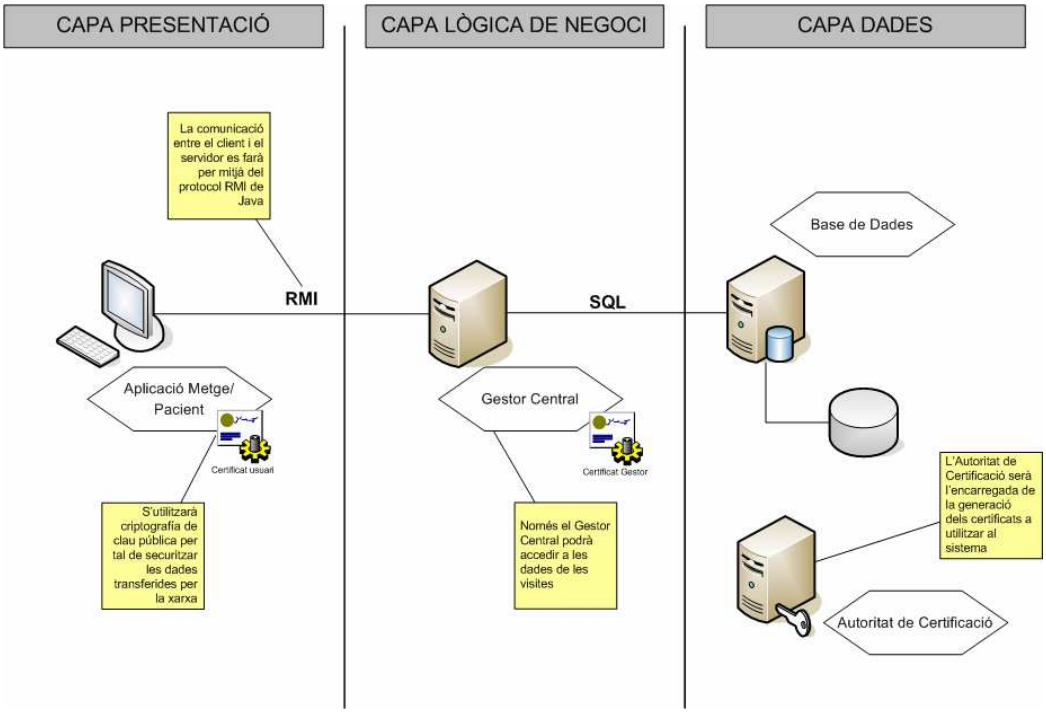

Figura 2. Arquitectura de tres nivells

Encara que no es desenvolupa en aquest projecte, en un sistema en producció, cada capa estaria protegida per un tallafocs que separaria i controlaria els accessos entre els diferents nivells.

[1] Per detall sobre aquest punt veure l'apartat "Model de Seguretat" i el capítol 4 "Esquema criptogràfic".

L'aplicació, doncs, aplica el patró d'arquitectura del programari Model Vista Controlador (MVC).

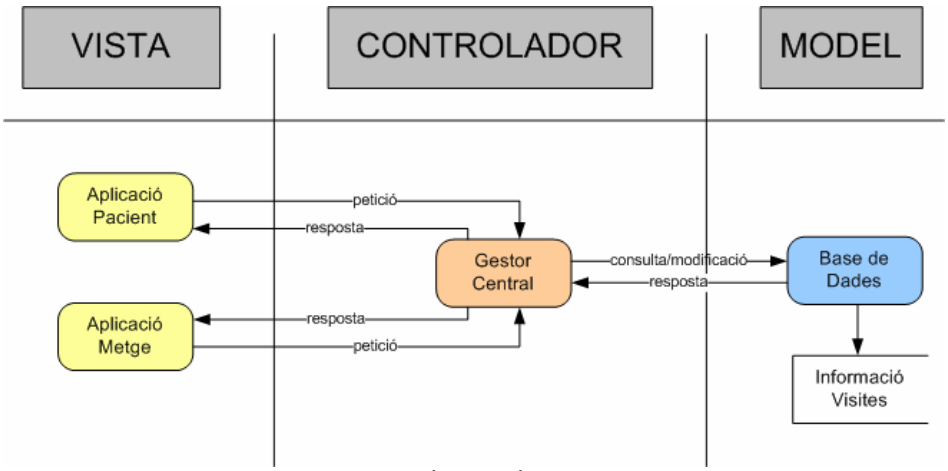

Figura 3. Aplicació del patró de disseny MVC

El flux seguit en aquest model és:

1) L'usuari interacciona amb la interfície del sistema (la Vista, implementada en la Capa de Presentació)

2) El Controlador (implementada a la Capa de Negoci) rep les peticions i les gestiona.

3) El Controlador accedeix al Model (Capa de Dades) per tal de consultar o actualitzar-lo adequadament en funció del que ha demanat l'usuari i retorna les dades a la Vista.

4) La Vista mostra les dades o el resultats de les operacions a l'usuari.

2.1 Organització de la Informació

La informació dels historials s'ha estructurat com es descriu a continuació.[2]

L'historial conté les dades generals del pacient, una llista de visites protegida i una llista de metges al que està assignat protegida.

Les dades generals tindran la següent informació: nom i cognoms, CIP (Codi Identificació del Pacient), DNI, grup sanguini, al·lèrgies i el certificat X.509 del pacient.

La llista de visites es compon de dues parts: una llista de descriptors de visita protegida (signada pel gestor i xifrada amb una clau de sessió (criptosistema simètric) i una llista d'accés formada per una llista de criptogrames corresponents al xifratge amb la clau pública de la clau de sessió amb la clau pública del gestor, del pacient i dels metges autoritzats a accedir a l'historial. S'empra la tècnica del sobre digital [CRIPTO] per aconseguir aquest accés de forma eficient.

La llista de metges protegida és una llista d'identificadors de metges i rangs de dates d'accés permès xifrada amb una clau de sessió i signada amb la clau privada del gestor. També incorporarà la clau de sessió xifrada amb la clau pública del gestor.

[2] Es descriurà exhaustivament la representació i el model de dades en els capítols 5 i 7, respectivament.

El sistema contindrà totes les visites que han realitzat els metges. Cada visita està formada per un descriptor de visita, les dades de la visita i la signatura digital del metge (tant del descriptor com de les dades de la visita).

Les dades de la visita incorporen la següent informació: anamnesi, diagnosi i tractament. Un descriptor de la visita conté la informació bàsica i està identificat de forma única al sistema per un identificador. Les dades incloses són: identificador de la visita (únic i aleatori), data, hora, tema i metge (identificador del metge que fa la visita).

El sistema també disposarà informació bàsica dels metges i està formada per: nom i cognoms, número de col·legiat, DNI, especialitat, certificat X.509 del metge i llista de pacients protegida.

La llista de pacients protegida es signarà amb la clau privada del gestor i es xifrarà amb una clau de sessió (simètrica). Aquesta clau estarà guardada de forma xifrada pel la clau pública del gestor i per la del metge (de nou s'utilitza sobre digital).

La base de dades també contindrà una relació dels usuaris amb accés al sistema i el seu certificat (amb la clau pública). Es tindrà, doncs, l'associació entre identificador d'usuari i clau pública.

2.2 Model de Seguretat

A partir dels requeriments de seguretat plantejats inicialment (Confidencialitat, Autenticitat, Integritat i No repudi) s'estableix un model per tal de garantir-los en totes les funcionalitats demanades.

Al sistema es disposa d'un Gestor Central que vetllarà per la seguretat de les dades que conté i implementarà els mecanismes criptogràfics per protegir l'associació entre un pacient i les dades mèdiques sobre les visites realitzades pels facultatius.

2.2.1 Autenticació

Els usuaris han d'autenticar la seva identitat a cada acció. En l'operació s'indicarà el tipus de petició a realitzar.

Tot usuari haurà de disposar una parella de claus (criptosistema de clau pública) i del certificat X.509 [X509] en format DER [DERFMT] generat per la Autoritat de Certificació[3] del sistema.

La parella de claus estarà emmagatzemada en un contenidor PKCS#12 [PKCS12] i protegit amb una paraula de pas o *passphrase* només coneguda per l'usuari.

El procés d'autenticació contra el servidor es basa en el protocol Needham-Schroeder [NESC78] amb clau pública. L'aplicació client demanarà la identificació de l'usuari i la *passphrase* per a accedir al parell de claus del PKCS#12. Amb aquestes credencials iniciarà el protocol per tal d'identificar-se de manera segura davant el Gestor Central que comprovarà la identitat de l'usuari a partir de la clau pública de l'usuari.

El Gestor Central haurà de disposar dels certificats (que inclouen la clau pública) de tots els usuaris del sistema. Aquesta informació estarà guardada a la base de dades i relacionada amb l'identificador de l'usuari.

2.2.2 Polítiques d'accés

Es disposen de dos rols per l'accés al sistema: Pacient i Metge.

[3] Al Capítol 3 "PKI" es veurà detalladament la necessitat i funcions d'aquesta entitat.

El sistema gestor permetrà que un usuari pugui realitzar una acció o una altra segons quina sigui la seva identitat. A les següents figures es poden veure els casos d'usos definits per aquest rols on s'especifiquen les operacions disponibles.

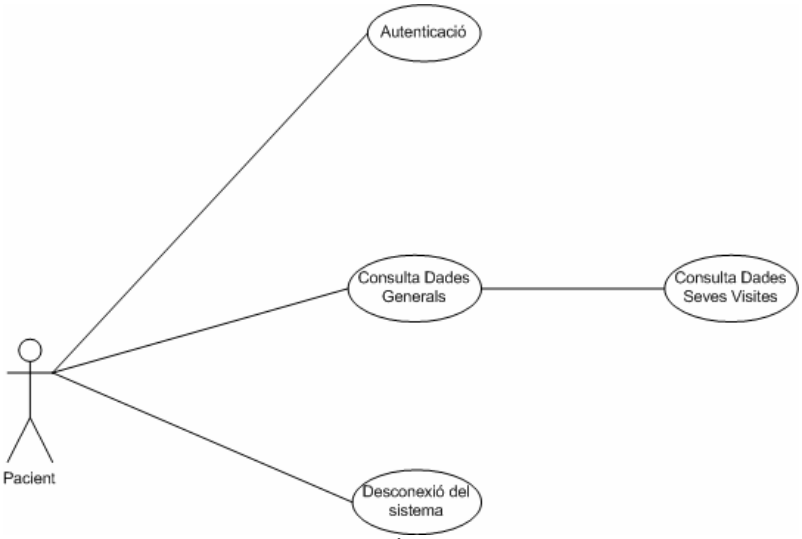

Figura 4. Cas d'ús d'actor Pacient

Ambdós tipus d'usuaris utilitzaran el N.I.F. (Número d'Identificació Fiscal en format XXXXXXXX-X) com a identificador únic del seu historial mèdic.

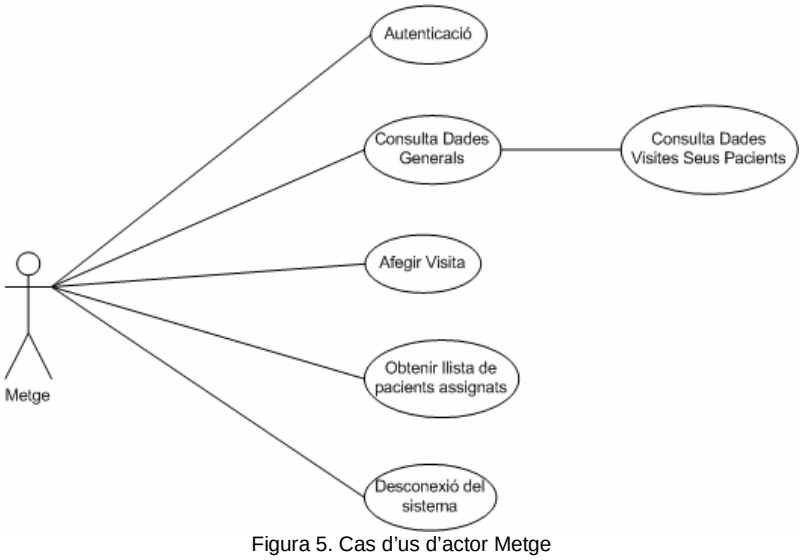

Figura 5. Cas d'us d'actor Metge

Quan un metge o un pacient facin una consulta cal que indiquin el seu identificador d'usuari, de manera que el gestor pugui trobar el seu certificat a la Base de Dades. Aquest certificat s'utilitzarà per fer l'autenticació i també per saber a quin rol pertany degut a que el camp `Organizational Unit Name (eg, section)` ens indicarà si l'usuari té rol de Metge o Pacient, i el camp `dnQualifier` contindrà el seu identificador (N.I.F.).

El Gestor Central s'identificarà per mitjà del resum SHA del *fingerprint* del seu certificat.

2.2.3 Protecció de les dades

S'utilitzarà mètodes de criptografia simètrica i asimètrica per tal de xifrar les dades.

Com es descriurà més detalladament al capítol 4 s'han escollit els següents algorismes criptogràfics:

4. **Criptografia simètrica** (clau secreta)
    a. Xifra de bloc: AES256 en mode CBC
    b. Mètode de Farciment (*padding*): PKCS#5.

5. **Criptografia asimètrica** (clau pública)
    a. Algorisme de xifratge/signatura: RSA [RSA77] amb claus de 2048 bits.
    b. Funció de resum (*hash*): SHA-1 [SHA101]

2.2.4 Dissociació de dades

A la capa de dades es farà persistent totes les dades referent als historials mèdics i la seva gestió. En aquest nivell es protegirà l'associació d'una visita amb un pacient. Aquest fet implica que encara que un atacant malintencionat tingui accés a la base de dades no podrà associar les dades mèdiques amb cap pacient.

Serà el Gestor Central l'encarregat de disposar dels mecanismes necessaris per poder fer l'associació entre un pacient i la llista de les seves visites.

Com es descriu detalladament a l'apartat "Organització de la Informació" el Gestor tindrà al seu abast els procediments criptogràfics per a accedir a la informació per esbrinar els metges assignats a un pacient i, paral·lelament, la llista de pacients d'un metge.

Per tal de garantir la integritat de la informació es signaran les llistes amb la clau privada del Gestor de forma que es podrà verificar a posteriori amb la clau pública del Gestor Central. També es guardaran signades pel metge la informació de les visites.

2.2.5 Comunicació segura

Degut a la implementació dels protocols criptogràfics descrits al capítol 4 la comunicació de les dades anirà xifrada amb la clau pública del destinatari una vegada es passi la fase d'autenticació. D'aquesta manera es garantirà la confidencialitat dels objectes que viatgen per la xarxa.

Un cop el destinatari rep el objecte, el desxifrarà utilitzant la seva clau privada i, per fi, accedirà al contingut del objecte. En algunes ocasions caldrà revisar la integritat i autenticitat per mitjà de la verificació de la signatura de les dades rebudes.

Aquest esquema de comunicació s'aplicarà tant en el cas dels diàlegs entre pacient i gestor com entre metge i gestor. No es permetrà en cap cas la comunicació entre pacient i metge.

2.3 Funcionalitats addicionals

A més dels requeriments funcionals descrits anteriorment serà necessari afegir les següents funcionalitats referents al manteniment del sistema:

- **Registre d'usuaris:**
  Inclourà l'alta, modificació i baixa d'usuaris al sistema.

- **Registre de pacients a l'historial:**
  Les dades dels historials dels pacients del sistema hauran de ser carregades de forma inicial i es disposaran de les operacions necessàries per tal de afegir o modificar les dades generals del pacient.

- **Registre de metges:**
  Per tal de tenir disponible les dades inicials del metges es necessari la implementació de les utilitats per donar de alta els facultatius al sistema.

Aquestes operacions addicionals no podran ser utilitzades per usuaris amb rol Pacient o Metge.

2.4 Especificacions tècniques

En aquest apartat s'especifica les tecnologies utilitzades a la construcció del productes lliurats en aquest projecte.

Llenguatge de programació

Tant l'aplicació client com la part servidor s'han implementat en Java, utilitzant la versió de desenvolupament JDK 1.6.0 *update* 4 [SDK16] de Sun. Per la facilitat d'ús, potència, funcionalitats, fàcil portabilitat a diferents plataformes, amplia disponibilitat de llibreries i utilitats i la gran difusió s'ha escollit aquest llenguatge de programació.

La llibreria utilitzada per a implementar els procediments i protocols criptogràfics ha estat IAIK versió 3.16 [IAIK316]. Ha calgut actualitzar les polítiques de JCE (*Java Cryptography Extension*) per tal de evitar les restriccions sobre la mida de les claus.

Com a suport pel desenvolupament s'ha utilitzat la eina IDE Eclipse 3.3.2 *Classic* [ECL332].

Representació de les dades

S'utilitza XML (*eXtensible Markup Language*) per a fer la representació de les dades que s'envien durant l'execució dels protocols criptogràfics. Per tal d'implementar aquest component ha calgut emprar les llibreries Java JDOM 1.1 [JDOM11].

Comunicacions

La comunicació entre les aplicacions metge, pacient i el gestor central es realitza utilitzant el protocol RMI (*Remote Method Invocation*) natiu de Java. El gestor es comunica amb la base de dades utilitzant JDBC (*Java DataBase Connectivity*).

Sistema gestor de base de dades

S'ha fet servir com a gestor MySQL Community Server versió 5.0 [MYSQL5].

3. PKI

3.1 Justificació de la necessitat d'una PKI

Cada protocol criptogràfic implementat en aquest projecte necessita que els usuaris i el gestor del sistema disposin d'una parella de claus i el seu corresponent certificat.

Per tal de gestionar els certificats (emissió, renovació, revocació, etc.) d'un grup d'usuaris s'empra una infraestructura de clau pública. Típicament per fer referència a una infraestructura d'aquest tipus s'utilitzen les sigles PKI, que corresponen al terme en anglès *Public Key Infrastructure*.

La PKI serà l'entitat que garantirà l'autenticitat de les claus públiques que utilitzarem al desxifrar, dient que les claus són realment de qui diu que són. D'aquesta manera s'evita l'atac del 'home a mig camí'.

Si no fos així, no es podria garantir l'autenticitat dels participants en el moment de la utilització d'operacions criptogràfiques que emprin claus públiques. En general, no es podria garantir que les dades signades han estat signades per qui es creu si no es disposa d'una clau pública de confiança. Tampoc es tindria la seguretat de que la clau pública és del destinatari i, per tant, les dades xifrades a enviar podrien estar compromeses.

3.2 Disseny de la PKI

Una PKI consta d'una autoritat de certificació notada amb CA, aquestes sigles corresponen al terme en anglès *Certification Authority*. Un altre component de la PKI són les autoritats de registre, notades amb les sigles RA[4] (*Registry Authority*).

Quan un usuari vol obtenir un certificat es segueixen els següents passos:

1) En un primer pas l'usuari crea una parella de claus i genera una petició de certificat (*Certificate Signing Request* o CSR), que conté les dades de l'usuari i la seva clau pública.

2) L'usuari envia la CSR a la CA.

3) La CA valida la identitat del peticionari, emet el certificat demanat signant la clau pública rebuda i incorpora la clau pública de la CA.

4) La CA retorna el certificat a l'usuari.

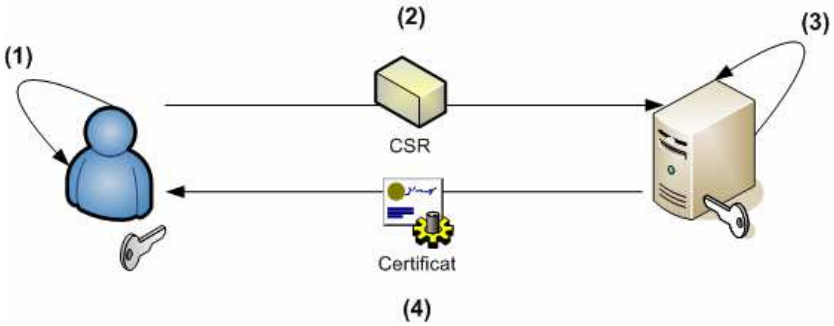

Figura 6. Procés d'emissió d'un certificat

[4] Aquest component és opcional i al projecte no s'ha desplegat

La clau privada de la CA és una peça d'informació molt sensible, i per això està en un entorn amb un alt nivell de seguretat. Normalment s'emmagatzema de forma segura en un dispositiu físic criptogràfic anomenat HSM (*Hardware Security Module*) o en una targeta intel·ligent (*Smartcard*).

3.3 Implementació

En el nostre cas construirem una PKI mínima amb la eina OpenSSL [OPSSL] versió 0.9.8g.

L'estructura de directoris i explicació de continguts és el que es mostra a continuació:

Arxiu o Directori	Descripció
`aleatori`	És un valor aleatori.
`openssl.cnf`	És el fitxer de configuració[5] de la CA.
`bin\`	Conté els *scritps* per crear i gestionar la CA.
`CAPFC\`	Conté l'estructura de la CA.
`CAPFC\CA.crt`	Certificat de la CA
`CAPFC\newcerts`	Certificats generats per la CA
`CAPFC\crl\`	Directori on es guarda la llista de revocació de certificats (CRL)
`CAPFC\private\CA.key`	Clau pública i privada de la CA

Per tal d'afegir el camp `dnQualifier` s'ha hagut de modificar el fitxer de configuració indicant com a valor necessari per la emissió dels certificats. Recordem que serà en aquest camp on es registrarà l'identificador únic de l'usuari del sistema que serà el DNI.

[5] Al Annex B d'aquesta memòria s'inclou la configuració específica per la PKI creada al projecte.

4. ESQUEMA CRIPTOGRÀFIC

4.1 Descripció dels protocols i procediments

Cada tipus d'acció es realitza d'acord a un protocol criptogràfic. El conjunt de protocols s'anomena esquema criptogràfic.

A la figura es mostra gràficament un mapa de la utilització de les diferents parts de l'esquema criptogràfic. Tant els usuaris com els gestor participen en els protocols.

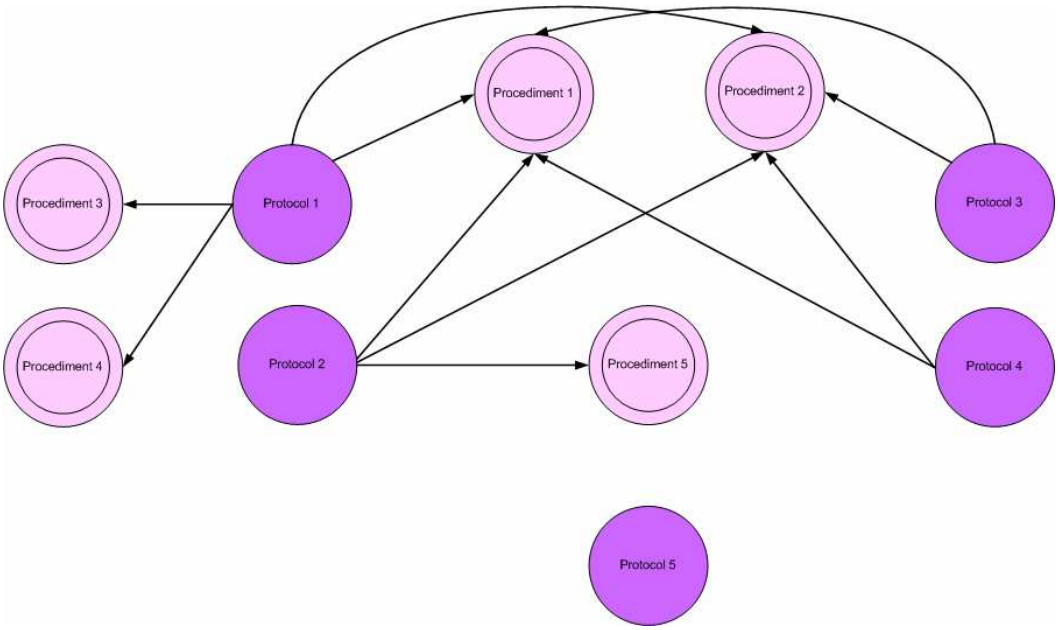

Figura 7. Mapa de protocols i procediments criptogràfics

A continuació es detallen els protocols i procediments criptogràfics dissenyats[6] i la implementació realitzada al projecte.

4.1.1 Protocol 1: Consulta de les dades generals d'un pacient

El Protocol 1 pot ser utilitzat per un metge o per un pacient. El gestor verifica en cada cas el tipus d'usuari i només facilita l'historial si l'usuari hi té accés. Verificarà que:

- L'usuari demana les seves dades generals.
- L'usuari és un metge.

Al Protocol 1 cada usuari *U* s'identifica amb Id_usuari_U i disposa d'una parella de claus (P_U, S_U) amb el corresponent certificat $Cert_U$. En el cas del gestor *G* el seu identificador d'usuari Id_usuari_G és el *hash* del certificat.

[6] Veure Annex A "Notació" per a observar la notació emprada

Descripció del protocol

1. U realitza les operacions següents:
   (a) Executar el Procedure 1 amb la clau pública P_G, i obtenir $P_G[N_i, Id\_usuari_U]$. U ha de guardar N_i de forma local perquè el necessitarà després per a certes verificacions;
   (b) Enviar $P_G[N_i, Id\_usuari_U]$ a G;

2. G realitza les operacions següents:
   (a) Executar el Procedure 2 amb $P_G[N_i, Id\_usuari_U]$, i obtenir $P_U[N_i, N_G, Id\_usuari_G]$;
   (b) Enviar $P_U[N_i, N_G, Id\_usuari_G]$ a U;

3. U realitza les operacions següents:
   (a) Desxifrar $P_U[N_i, N_G, Id\_usuari_G]$ amb la clau privada S_U, i obtenir N_G, N_i' i Id_usuari_G;
   (b) Si $N_i' = N_i$ fer:
      i. Xifrar N_G, Consulta_dades_generals, Id_usuari amb la clau pública P_G de G, i obtenir $P_G[N_G, Consulta\_dades\_generals, Id\_usuari]$.
         Consulta_dades_generals indica que es volen consultar les dades generals de l'usuari identificat amb Id_usuari;
      ii. Enviar $P_G [N_G, Consulta, Id\_usuari, Id\_usuari_U]$ a G;
   (c) Sino retornar error;

4. G realitza les operacions següents:
   (a) Desxifrar $P_G [N_G, Consulta\_dades\_generals, Id\_usuari]$ amb la clau privada S_G, i obtenir N_G', Consulta_dades_generals, Id usuari;
   (b) Recuperar N_G de la BD. En el pas 4 del Procedure 2 N_G i N_i han estat guardats a la BD;
   (c) Si $N_G' = N_G$ fer:
      i. Si (Id_usuari_U = Id_usuari) o (Id_usuari_U és metge 1) fer:
         A. Executar el Procedure 3 amb Id_usuari i P_U, i obtenir $P_U[H]$;
         B. Enviar $P_U[H]$ a U;
      ii. Sino retornar error;
   (d) Sino retornar error;
   (e) Borrar N_G i N_i de la BD;

5. U realitza les operacions següents:
   (a) Executar el Procedure 4 amb $P_U[H]$, i obtenir H;
   (b) Mostrar H.

Decisions de disseny particulars

Per a verificacions posteriors, caldrà que l'usuari *U* guardi el valor aleatori generat en el pas 1 (a). Aquesta decisió és vàlida para la resta de protocols.

4.1.2 Protocol 2: Consulta d'una visita d'un pacient

En el Protocol 2 l'usuari *U* s'identifica amb *Id_usuariU*, on l'usuari pot ser un metge o un pacient. *G* verifica en cada cas el tipus d'usuari i només facilita l'historial si l'usuari hi té accés. Per tant, *G* ha de verificar que:

- L'usuari demana una de les seves visites.
- L'usuari és un metge que té accés a les visites del pacient.

El protocol 2 s'haurà de cridar després del protocol 1 per a obtenir la llista de descriptors del pacient i demanar la consulta de les dades corresponents a la visita d'aquell identificador.

Descripció del protocol

1. U realitza les operacions següents:
   (a) Executar el Procedure 1 amb la clau pública P_G, i obtenir $P_G[N_i, Id\_usuari_U]$. U ha de guardar N_i de forma local perquè el necessitarà després per a certes verificacions;
   (b) Enviar $P_G[N_i, Id\_usuari_U]$ a G;

2. G realitza les operacions següents:
   (a) Executar el Procedure 2 amb $P_G[N_i, Id\_usuari_U]$, i obtenir $P_U[N_i, N_G, Id\_usuari_G]$;
   (b) Enviar $P_U[N_i, N_G, Id\_usuari_G]$ a U;

3. U realitza les operacions següents:
   (a) Desxifrar $P_U[N_i, N_G, Id\_usuari_G]$ amb la clau privada S_U, i obtenir N_G, N_i' i Id_usuari_G;
   (b) Si $N_i' = N_i$ fer:
      i. Xifrar N_G, Consulta_visita, Id_usuari, descriptor_de_visita amb la clau pública P_G de G, i obtenir $P_G[N_G$, Consulta_visita, Id_usuari, descriptor_de_visita];
         Consulta_visita indica que es vol consultar la visita identificada per descriptor_de_visita de l'usuari identificat amb Id_usuari;
      ii. Enviar $P_G[N_G$, Consulta_visita, Id_usuari, descriptor_de_visita, $Id\_usuari_U]$ a G;
   (c) Sino retornar error;

4. G realitza les operacions següents:
   (a) Desxifrar $P_G[N_G$, Consulta_visita, Id_usuari, descriptor_de_visita] amb la clau privada S_G, i obtenir N_G', Consulta_visita, Id_usuari, i descriptor_de_visita;
   (b) Recuperar N_G de la BD. En el pas 4 del Procedure 2 N_G i N_i han estat guardats a la BD;
   (c) Si $N_G' = N_G$ fer:
      i. Si Procedure 5[Id_usuariU,Id_usuari, descriptor_de_visita] retorna que totes les verificacions són correctes fer:
         A. Obtenir la visita (V) identificada per descriptor_de_visita
            i calcular $P_U[V]$;
         B. Enviar $P_U[V]$a U.
      ii. Sino retornar error;
   (d) Sino retornar error;
   (e) Borrar N_G i N_i de la BD;

5. U realitza les operacions següents:
   (a) Desxifrar $P_U[V]$, i obtenir V;
   (b) Mostrar V.

4.1.3 Protocol 3: Consulta dels pacients assignats a un metge

En el protocol 3 el metge consulta la llista dels pacients que hi te assignat. Posteriorment podrà visualitzar l'historial dels seus pacients. Aquest protocol és utilitzat exclusivament pel rol Metge. El gestor comprovarà que l'usuari (identificat amb Id_usuari_U) és metge[7].

Descripció del protocol

1. U realitza les operacions següents:
   (a) Executar el Procedure 1 amb la clau pública P_G, i obtenir $P_G[N_i, Id\_usuari_U]$. U ha de guardar N_i de forma local perquè el necessitarà després per a certes verificacions;
   (b) Enviar $P_G[N_i, Id\_usuari_U]$ a G;

[7] El camp `Organizational Unit Name (eg, section)` del certificat de $Id\ usuari_U$ indica si l'usuari és Metge o Pacient, i amb la base de dades podem saber els pacients assignats a un metge

2. G realitza les operacions següents:
   (a) Executar el Procedure 2 amb $P_G[N_i, Id\_usuari_U]$, i obtenir $P_U[N_i, N_G, Id\_usuari_G]$;
   (b) Enviar $P_U[N_i, N_G, Id\_usuari_G]$ a U;

3. U realitza les operacions següents:
   (a) Desxifrar $P_U[N_i, N_G, Id\_usuari_G]$ amb la clau privada S_U, i obtenir N_G, N_i' i Id_usuari_G;
   (b) Si $N_i' = N_i$ fer:
      i. Xifrar N_G i Llista_pacients amb la clau pública P_G de G, i obtenir $P_G[N_G, Llista\_pacients]$;
         Llista_pacients indica que es vol un llistat dels pacients del metge identificat amb Id_usuari_U;
      ii. Enviar $P_G[N_G, Llista\_pacients, Id\_usuari_U]$ a G;
   (c) Sino retornar error;

4. G realitza les operacions següents:
   (a) Desxifrar $P_G[N_G, Llista\_pacients]$ amb la clau privada S_G, i obtenir N_G' i Llista_pacients;
   (b) Recuperar N_G de la BD. En el pas 4 del Procedure 2 N_G i N_i han estat guardats a la BD;
   (c) Si $N_G' = N_G$ fer:
      i. Si Id $usuari_U$ és metge fer:
         A. Calcular $P_U[llista\_pacients\_protegida]$;
         B. Enviar a U $P_U[llista\_pacients\_protegida]$;
      ii. Sino retornar error;
   (d) Sino retornar error;
   (e) Borrar N_G i N_i de la BD;

5. U realitza les operacions següents:
   (a) Desxifrar $P_U[llista\_pacients\_protegida]$ i obtenir llista_pacients_protegida;
   (b) Tractar la llista_pacients_protegida i mostrar la llista de pacients a l'usuari.

Decisions de disseny particulars
El tractament de la llista de pacients inclourà la verificació de la signatura del gestor (amb la seva clau privada) i el desxifratge de la clau de sessió amb la que està protegida.

Caldrà que l'usuari (el metge) tingui accés a aquesta llista ja que només podrà realitzar aquesta comprovació un metge sobre la seva relació de pacients.

Recordem que degut a les limitacions de la criptografia asimètrica per xifratge de gran volum de dades s'opta per utilitzar sobre digital[8].

Es seguirà, per tant, una estructura similar a la llista d'accés de la llista de descriptors d'un historial.

4.1.4 Protocol 4: Afegir una visita a l'historial mèdic

En aquest protocol suposem que prèviament a la inserció de les dades el metge *M* ha consultat l'historial del pacient *P* i per tant coneix Id_usuari_P. Només un usuari metge pot fer servir aquest protocol.

El Protocol 4 està pensat únicament per afegir una nova visita *V* a l'historial. El gestor un cop rep un visita *V* d'un pacient *P* verifica que ha estat signada pel metge *M* assignat al pacient. A continuació afegeix el descriptor de la visita la llista de descriptors protegida i la visita a la base de dades.

[8] Veure apartat "Decisions Generals de Disseny" per més detall

Descripció del protocol

1. M realitza les operacions següents:
   (a) Executar el Procedure 1 amb la clau pública P_G, i obtenir $P_G[N_i, Id\_usuari_U]$. M ha de guardar N_i de forma local perquè el necessitarà després per a certes verificacions;
   (b) Enviar $P_G[N_i, Id\_usuari_U]$ a G;

2. G realitza les operacions següents:
   (a) Executar el Procedure 2 amb $P_G[N_i, Id\_usuari_U]$, i obtenir $P_M[N_i, N_G, Id\_usuari_G]$;
   (b) Enviar $P_M [N_i, N_G, Id\_usuari_G]$ a U;

3. M realitza les operacions següents:
   (a) Desxifrar $P_M [N_i, N_G, Id\_usuari_G]$ amb la clau privada S_U, i obtenir N_G, N_i' i Id_usuari_G;
   (b) Si $N_i' = N_i$ fer:
      i. Obtenir les dades de la visita V;
      ii. Signar V amb la clau privada S_M de M, SM[V];
      iii. Xifrar N_G, V, Id_usuari i S_M [V] amb la clau pública P_G de G, i obtenir $P_G [N_G$, Afegir_visita, V, Id_usuari, S_M [V]];
         Afegir_visita indica que es vol afegir V a l'historial del pacient P identificat per Id_usuari;
      iv. Enviar $P_G [N_G$, Afegir_visita, V, Id_usuari, S_M [V] , Id_usuari_U] a G;
   (c) Sino retornar error;

4. G realitza les operacions següents:
   (a) Desxifrar $P_G [N_G$, Afegir_visita, V, Id_usuari, S_M [V]] amb la clau privada S_G, i obtenir N_G', Afegir_visita,V, Id_usuari i S_M [V];
   (b) Recuperar N_G de la BD. En el pas 4 del Procedure 2 N_G i N_i han estat guardats a la BD;
   (c) Si $N_G' = N_G$ fer:
      i. Verificar que Id_usuari_M és metge;
      ii. Verificar que Id_usuari és un pacient [9]assignat a Id_usuari_M;
      iii. Si les verificacions anteriors són correctes fer:
         A. Verificar la signatura digital S_M [V]amb la clau pública P_M;
         B. Obtenir el descriptor de la visita de V;
         C. Afegir el descriptor de la visita a la llista_descriptors_de_visita;
         D. Signar amb la clau privada del Gestor S_G la llista_descriptors_de_visita;
         E. Xifrar la llista_descriptors_de_visita amb una clau de sessió K, E_K(llista_descriptors_de_visita);
         F. Xifrar la clau de sessió K amb les claus públiques dels metges que estan a la llista de metges de l'historial de l'usuari identificat amb Id_usuari, obtenint una nova llista_descriptors_de_visita_protegida;
         G. Afegir V a la Base de Dades.
      iv. Sino retornar error.
   (d) Sino retornar error.
   (e) Borrar N_G i N_i de la BD;

Decisions de disseny particulars

En aquest protocol he decidit recalcular la clau de sessió cada vegada que es modifiqui la llista[10].

[9] Amb la llista_de_pacients_protegida del metge i la llista_de_metges del pacient es poden fer aquestes verificacions
[10] Veure apartat "Decisions Generals de Disseny" per més detall

També ha estat necessari decidir com es calcularia l'identificador del descriptor de la visita. Qui faci servir aquest protocol haurà de proporcionar un identificador del descriptor de visita únic al sistema i no relacionat amb cap dada de l'historial del pacient.

Per fer això, es defineix el mètode per a generar un identificador que correspondrà a una marca de temps (*timestamp*) concatenat amb un nombre aleatori gran.

En concret, el format d'un identificador serà:

`TIMESTAMP.RANDOMN`

On:
- "`.`" Indica la operació de concatenació.
- `TIMESTAMP` és la marca de temps en format "AAAAMMDDHHmmSSllll" (A = any, M = mes, D = dia, H = hora, m = minut, S = segon, l = mil·lisegon).
- `RANDOMN` és un nombre pseudo-aleatori de 128 bits.

S'ha escollit aquesta solució perquè per a un instant de temps la probabilitat de repetir un identificador és molt baixa degut a dos factors principals: la precisió de la marca de temps és fins al mil·lisegon i el nombre aleatori és gran (128 bits).

4.1.5 Protocol 5: Autenticació

S'implementa el protocol de Needham-Schroeder [NESC78] de clau pública pel cas d'un usuari P_i i el gestor del sistema *G*. L'objectiu del protocol es autenticar de forma bilateral les dues parts.

Descripció del protocol

1. Pi realitza les operacions següents:
   (a) obtenir un valor de forma aleatòria, N_i;
   (b) xifrar N_i i IP_i amb la clau pública de G, $P_G(N_i, IP_i)$;
   IP_i és l'identificador de P_i;
   (c) enviar $E_G(N_i, IP_i)$ a G;

2. G realitza les operacions següents:
   (a) desxifrar $E_G(N_i, IP_i)$amb S_G, i obtenir N_i i IP_i;
   (b) obtenir el certificat de P_i amb IP_i. A partir del certificat obtindrà PP_i;
   (c) obtenir un valor de forma aleatòria, N_G;
   (d) xifrar N_i,N_G, I_G amb la clau pública PP_i de P_i, $PP_i(N_i,N_G, I_G)$;
   (e) enviar $EP_i(N_i,N_G, I_G)$ a P_i;

3. P_i realitza les operacions següents:
   (a) desxifrar $EP_i(N_i,N_G, I_G)$ amb la clau privada SP_i , i obtenir N_G, N_i i I_G;
   (b) xifrar N_G amb la clau pública P_G de G, $P_G(N_G)$;
   (c) enviar $P_G(N_G,IP_i)$ a G;

4. G realitza les operacions següents:
   (a) desxifrar $P_G(N_G)$ amb la clau privada S_G, i obtenir N_G';
   (b) si N_G' = N_G, G i P_i estan autenticats bilateralment.
   (c) Borrar N_G i N_i de la BD;

4.1.6 Procedure 1

El Procedure 1 conté una part de l'autenticació del protocol de Needham-Schroeder. Aquests passos els utilitzarem en altres protocols. Aquest procediment serà utilitzat pels metges i pacients.

Descripció del procediment
1. Obtenir un valor de forma aleatòria, N_i;

2. Xifrar N_i i Id_usuari_U amb la clau pública de G, $P_G(N_i, Id\_usuari_U]$;

3. Enviar $P_G [N_i, Id\_usuari_U]$ a G.

Decisions de disseny particulars
El nombre generat ha de ser pseudo-aleatori. Per tal d'aconseguir-ho s'implementarà el mètode generador utilitzant la classe Java `java.security.SecureRandom()` on es proveeix d'un generador criptogràfic de nombres aleatoris fort. Aquest generador compleix amb els test estadístics especificats a l'estàndard "*FIPS 140-2*, *Security Requirements for Cryptographic Modules*" [FIPS1402], secció 4.9.1.

L'objectiu és obtenir una sortida no determinista i no predicible tal i com es descriu a la RFC 1750: "*Randomness Recommendations for Security*" [RFC1750].

4.1.7 Procedure 2

El Procedure 2 conté una altra part de l'autenticació del protocol de Needham-Schroeder. Aquesta part serà executada pel gestor *G*.

Descripció del procediment
1. Desxifrar $P_G [N_i, Id\_usuari_U]$ amb S_G, i obtenir N_i i Id_usuari_U;

2. Obtenir el certificat de U a partir de Id_usuari_U;
   Suposem que el sistema disposa d'una Base de Dades (BD) on per cada Id_usuari trobem el seu certificat corresponent. A partir del certificat es pot obtenir la clau pública P_U;

3. Obtenir un valor de forma aleatòria, N_G;

4. Guardar a la BD els valors N_i i N_G associats amb U;

5. Xifrar N_i, N_G, Id_usuari_G, amb la clau pública P_U de U, $P_U[N_i, N_G, Id\_usuari_G]$;

6. Retornar $P_U[N_i, N_G, Id\_usuari_G]$.

4.1.8 Procedure 3

El gestor *G* utilitza el Procedure 3 per trobar l'historial que se li ha demanat i xifrar-lo amb la clau de l'usuari que el vol consultar.

Descripció del procediment
1. Buscar l'historial H corresponent a Id_usuari;

2. Xifrar H amb la clau pública P_U, $P_U[H]$;

3. Retornar $P_U[H]$.

4.1.9 Procedure 4

Un usuari utilitza el Procedure 4 per tal de desxifrar un historial enviat pel gestor *G* i verificar que l'historial és correcte.

Descripció del procediment

1. Desxifrar $P_U[H]$ amb la clau privada S_U de U, $S_U[P_U[H]]$;

2. Desxifrar una de les entrades de la llista d'accés i obtenir la clau de sessió;

3. Desxifrar la llista de descriptors de visites xifrada;

4. Verificar la signatura digital de G sobre la llista dels descriptors de visites xifrada;

5. Retornar H.

Decisions de disseny particulars

Aquest procediment també mostrarà la llista de descriptors de visita una vegada verificada. La raó d'aquesta decisió es per què l'usuari que ha provocat l'execució d'aquest procediment necessita la informació referent als identificadors de visita del pacient.

4.1.10 Procedure 5

El procedure 5 verifica:

- Si l'usuari vol consultar una dada seva, que el descriptor de visita que es vol consultar està a la seva llista de descriptors de visita protegida.

- Si l'usuari és metge, verifica:

    1. que el pacient (identificat per *Id_usuari*) està la llista de pacients del metge (identificat per Id_usuari_U).

    2. que el metge està assignat a la llista de metges de *Id_usuari*.

    3. que el descriptor de visita que es vol consultar està a la a llista de descriptors de visita protegida de *Id_usuari*.

Aquest procediment es utilitzat pel gestor *G* en el protocol de consulta de dades de visita.

Descripció del procediment

1. Si (Id_usuari_U = Id usuari) fer:
    (a) Verificar si descriptor de visita està dins de la llista de descriptors de visita xifrada de l'usuari identificat per Id_usuari_U;
    (b) Retornar el resultat de la verificació.

2. Si (Id_usuari_U) és metge fer:
    (a) Verificar si Id_usuari està dins de la llista de pacients protegida del metge identificat per Id_usuari_U;
    (b) Verificar si el metge identificat per Id_usuari_U està a la llista de metges de l'usuari identificat per Id _usuari;
    (c) Verificar si el descriptor de visita pertany a l'usuari Id_usuari emprant la llista de visites protegida de l'usuari;
    (d) Retornar el resultat de la verificació

Decisions de disseny particulars
La verificació de l'assignació d'un pacient a un metge es realitzarà de forma doble: el metge ha de estar en la llista de metges del pacient i també el pacient ha de estar en la llista de pacients del metge.

Tant l'estructura de dades com l'esquema criptogràfic es basen en el fet que les dues llistes siguin congruents. Si hi ha alguna diferència es pot deure a que un usuari malintencionat hagi modificar la llista. No obstant, per a més seguretat sobre la integritat, i com ja s'ha comentat, aquestes llistes hauran d'estar signades pel gestor *G*, signatura que es verificarà en el moment de llegir el contingut.

4.2 Decisions Generals de Disseny

En aquest apartat s'expliquen les decisions presses respecte al disseny d'aspectes generals a tot l'esquema criptogràfic implementat.

a) Concatenació de camps en tires de bytes
Prèviament a la signatura o xifratge de camps amb informació es realitza una concatenació. Per tal de reconèixer els diferents camps en el moment de desxifratge hem aplicat dues solucions possibles: tractar els camps com longitud fixa allà on es podia (com per exemple els tipus d'operacions o els identificadors d'usuari) o generar una llista que pugui ser serialitzada com a objecte.

A l'hora de fer persistents aquest camps s'hauran de codificar en format base64 per tal de poder ser tractats convenientment[11].

b) Utilització de clau publica de confiança
En la definició i implementació d'aquest esquema criptogràfic s'ha decidit sempre utilitzar una clau pública de confiança.

Per una banda, només s'acceptarà aquella clau certificada per una CA reconeguda (la PKI creada expressament pel projecte) i, per altra banda, l'usuari confiarà en la clau pública emmagatzemada localment i que ha estat enviada prèviament per un canal segur.

En el cas del gestor, es disposarà d'una taula a la base de dades on podrà consultar a partir de l'identificador de l'usuari el certificat amb la clau pública de confiança. La resta d'usuaris hauran de rebre per un mètode segur el certificat del gestor per a poder comunicar-se amb ell de forma segura.

c) PKCS#7 per Criptografia asimètrica
En la implementació dels mètodes criptogràfics de clau pública s'ha decidit utilitzar les funcionalitats de la llibreria IAIK pel tractament de contenidors PKCS#7 (que contindrà dades criptogràfiques).

D'aquesta manera s'ha simplificat molt el desenvolupament dels mètodes de xifratge, desxifratge, signatura i verificació.

d) Recàlcul de clau de sessió
Es recalcularà la clau de sessió cada vegada que es modifiqui la llista.

És en aquest punt on s'havia de decidir entre les següents opcions:

1) Recalcular-la cada vegada que és consultava

[11] Al capítol 5 "Representació de Dades" es tractarà aquesta qüestió més en profunditat

2) Recalcular-la en el moment d'una modificació de dades

3) No recalcular-la mai

La primera opció és la més segura però també és la més costosa. Si suposem que el sistema haurà d'albergar una gran quantitat d'historials, visites i metges, el cost temporal de refer la llista d'accés és molt alt. Per aquesta raó, descartem la primera opció.

La última opció la rebutgem perquè no és segura. Si mai canviem la clau de sessió estem davant dos perills: que un metge malintencionat divulgui la clau o que algú aconsegueixi accés al sistema i que, amb temps, aconsegueixi la clau (encara que al nostre sistema a dia d'avui seria difícil d'aconseguir ja que utilitzem AES amb clau de 256 bits).

Amb aquesta reflexió, escollim la segona opció que és la més adient. Aquesta decisió també aplica a totes les llistes protegides amb sobre digital.

e) Sobre digital

Les llistes protegides han estat tractades amb la tècnica de sobre digital. Aquesta tècnica combina criptografia simètrica i asimètrica per tal de aconseguir un bon rendiment en el moment d'aplicar mètodes criptogràfics.

De manera resumida, un sobre digital es compon de la dada signada amb la clau privada del generador de la informació, xifrada amb una clau de sessió (simètrica) i s'adjunta la clau de sessió emprada xifrada amb la clau pública del destinatari.

És a dir, si volem l'usuari A vol protegir la dada D i enviar-la a B enviarà[12]:

$$< E_k[S_A[D]] , P_B[k] >$$

Quan B rep el criptograma ha de seguir els següents passos per obtenir D:

6. Desxifrar $P_B[k]$ amb la seva clau privada i obtenir k
7. Desxifrar $E_k[S_A[D]]$ amb k i obtenir $S_A[D]$
8. Amb la clau pública de A, verificar $S_A[D]$ i obtenir D

4.3 Implementació

La implementació de l'esquema criptogràfic ha estat desenvolupat exclusivament per mitja de classes, objectes i mètodes Java.

La llibreria criptogràfica utilitzada com a base ha estat IAIK [IAIK316]. Es tracta d'una llibreria que implementa múltiples algoritmes criptogràfics per a ser usats dins d'aplicacions Java.

Entre els múltiples algoritmes que s'inclouen a IAIK estan els orientats a la generació i gestió de claus (tant simètriques com asimètriques), xifrat i desxifrat de dades, funcions de resum i creació i gestió de contenidors criptogràfics i certificats.

Si extraiem la resta de components del sistema desenvolupades a la resta de fases del projecte, podem disposar d'un diagrama de classes senzill que defineix aquest component del sistema construït.

[12] k és la clau de sessió

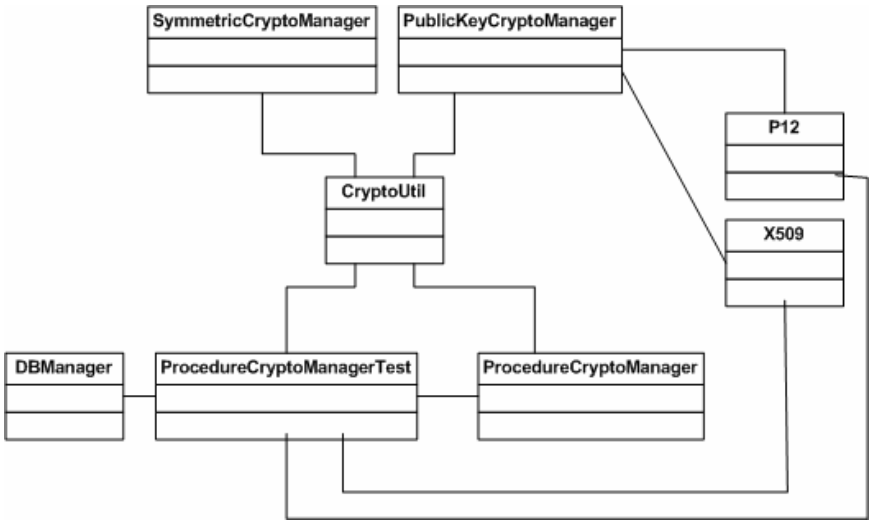

Figura 8. Diagrama de classes reduït per l'esquema criptogràfic

Com es pot observar es disposa d'una classe dedicada a la criptografia simètrica i una classe destinada a la criptografia de clau pública. Només la classe `CryptoUtil` té accés als mètodes de criptografia (tant simètrica com asimètrica) i fa l'abstracció per la resta de components del criptosistema proposat al projecte.

La classe `ProcedureCryptoManager` implementa els procediments descrits en els apartats anteriors, on cada mètode és un procediment. Els mètodes d'aquesta classe són cridats per `ProcedureCryptoManagerTest` on s'ha implementat els protocols[13].

Per aquesta fase s'ha necessitat implementar les classes `P12` i `X509` que gestionaran l'accés i tractament de contenidors de claus en format PKCS#12 i certificats en format X.509, respectivament.

La funció de cadascuna de les classes implementades en aquesta fase queda especificada de la següent forma:

- `P12`: Implementa la gestió d'objectes de tipus contenidor PKCS#12.
- `X509`: Implementa la gestió de certificat de tipus X.509.
- `SymmetricCryptoManager`: Classe que implementa les funcions de xifratge i desxifratge amb algoritmes simètrics.
- `PublicKeyCryptoManager`: Classe que implementa les funcions de xifratge, desxifratge, signatura digital i verificació amb algoritmes asimètrics o de clau pública.
- `CryptoUtil`: Classe que fa de façana i publica un conjunt de mètodes criptogràfics a utilitzar per la resta de capes de l'aplicació.
- `ProcedureCryptoManager`: Implementa els procediments criptogràfics explicats en aquest capítol.

[13] A la fase de RMI es discutirà com la funcionalitat dels protocols implementada s'haurà de dividir en les diferents classes que implementin les aplicacions client (pacient i metge) i servidor (gestor)

- `ProcedureCryptoManagerTest`: Incorpora la implementació dels protocols criptogràfics especificats per l'esquema. Com es veurà més endavant, aquesta classe serà dividida per les implicacions en la comunicació entre els components principals: client i gestor.

5. REPRESENTACIÓ DE LES DADES

5.1 XML: Definició i Justificació

XML és l'acrònim de *eXtensible Markup Language*. Des de que va aparèixer aquesta forma de presentar les dades s'ha imposat com una de les formes més eficients per intercanviar i emmagatzemar les dades entre aplicacions i/o protocols.

Un requisit en l'arquitectura del PFC és la utilització d'estructures XML per l'intercanvi de les dades durant l'execució dels protocols definits anteriorment. Aquestes estructures, que representen les dades, es faran persistents a la capa de dades[14].

Els documents XML són documents en text pla delimitats per etiquetes per a definir elements. L'estàndard XML utilitza aquest mecanisme com a representació del contingut i tipus de dades que està tractant i no de l'aparença final que tindran. XML és flexible ja que es permet la definició de noves etiquetes i/o ampliar les existents.

Es justifica la utilització d'estructura de dades XML enlloc d'objectes serialitzables per a eliminar la vinculació de la representació de dades amb el llenguatge de programació usat. Com ens trobem en un entorn de desenvolupament on tots els components són Java es podria utilitzar sense problemes Objectes per a la representació i comunicació (amb serialització RMI) entre els diferents actors. No obstant, si per algun motiu es necessita modificar la plataforma, es vol que la representació de dades quedi inalterada i es requereixi un esforç menor a la nova codificació.

A més, recordem que estem utilitzant un patró MVC. D'aquesta forma tenim un capa *middleware* que es l'encarregada de carregar i guardar les dades a base de dades. Aquest component pot lliurar les dades al Controlador com documents XML o directament, com objectes Java. Si utilitzen XML i els objectes o classes canvien no caldria fer cap canvi al *middleware*.

5.2 Disseny de la Representació de les dades

En aquest apartat es documenta les decisions de disseny respecte a la implementació de la representació de les dades.

5.2.1 Tipus de XML usats

A continuació es detalla l'estructura de cada document XML tipus emprat als protocols.

XML HISTORIAL

Aquest seria la representació d'un historial mèdic en XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Historial SYSTEM "dtd/historial.dtd">

<Historial>
  <DadesGenerals>
    <nom></nom>
    <cognom1></cognom1>
    <cognom2></cognom2>
    <cip></cip>
    <dni></dni>
    <grupSanguini></grupSanguini>
    <alergies></alergies>
    <certificat></certificat>
  </DadesGenerals>
  <LlistaVisitesProtegida>
    <LlistaDescriptors></LlistaDescriptors>
```

[14] Les decisions sobre la persistència de les dades dels historials, visites, pacients, metges, etc. es detallaran al capítol 7.

```
    <LlistaAccess></LlistaAccess>
  </LlistaVisitesProtegida>
  <LlistaMetgesProtegida></LlistaMetgesProtegida>
</Historial>
```

Figura 9. Representació XML de Historial

Aquest document es lliuraria quan un usuari volgués consultar les seves dades mèdiques.

XML VISITA

Representació d'una Visita. Es lliurà en el moment de consultar-la per un usuari o quan es vol afegir al sistema.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Visita SYSTEM "dtd/visita.dtd">

<Visita>
  <DescriptorVisita>
    <idVisita></idVisita>
    <any></any>
    <mes></mes>
    <dia></dia>
    <hora></hora>
    <minut></minut>
    <tema></tema>
    <idMetge></idMetge>
  </DescriptorVisita>
  <DadesVisita>
    <anamnesi></anamnesi>
    <diagnosi></diagnosi>
    <tractament></tractament>
  </DadesVisita>
  <SignaturaVisita></SignaturaVisita>
</Visita>
```

Figura 10. Representació XML de Visita

XML METGE

En la figura següent es mostra la representació de les dades d'un Metge. Aquesta estructura serà utilitzada internament pel Gestor per a la representació d'aquest objecte.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Metge SYSTEM "dtd/metge.dtd">

<Metge>
  <nom></nom>
  <cognom1></cognom1>
  <cognom2></cognom2>
  <colegiat>/colegiat>
  <dni></dni>
  <especialitat></especialitat>
  <certificat></certificat>
  <LlistaPacients></LlistaPacients>
</Metge>
```

Figura 11. Representació XML de Metge

XML SERIALIZEDLIST

I, finalment, la representació d'una llista d'elements.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE SerializedList SYSTEM "dtd/serializedlist.dtd">

<SerializedList>
  <elementList></elementList>
  <elementList></elementList>
  <elementList></elementList>
  <elementList></elementList>
  <elementList></elementList>
</SerializedList>
```

Figura 12. Representació XML de Missatge

Aquest tipus de document XML s'utilitza per la representació de les conversacions i missatges enviats i rebuts entre usuaris i gestor en la implementació dels protocols criptogràfics. En lloc de concatenar els diferents camps s'utilitzarà un element del document XML del tipus `elementList`.

5.2.2 Codificació Base64

En els protocols estem treballant amb dades de tipus tira de bytes (*byte array*, en anglès). La representació en un text pla com és un document XML no es pot fer directament ja que amb molta probabilitat aquests camps contenen caràcters no imprimibles.

Com a pas previ per a guardar un document XML serà la codificació en Base64 [BASE64] d'aquells camps que puguin contenir caràcters no representables. El mateix passarà quan es carregui un document XML, ja que serà necessària la descodificació de camps en base64 a tira de bytes.

Aquesta codificació estàndard, proposada inicialment en la definició del format PEM [PEMFMT], codifica dades binàries al tractar-les numèricament i traduir-les en una representació en base 64. La utilització d'aquesta base està relacionada amb a una decisió històrica respecte a la representació dels jocs de caràcters imprimibles.

5.2.3 Comprovacions de documents XML

Un document està ben format (*well-formed*) si s'ajusta a la sintaxi del llenguatge XML. Cap document que no estigui ben format és un document XML.

Un document XML és vàlid si el seu contingut respecta les regles del seu document de definició associat. El conjunt de regles gramaticals que es defineix en la declaració de marcatge associada permet indicar:

- Els elements i atributs vàlids d'un document XML
- Els elements que es poden utilitzar dins d'altres elements
- Els elements i atributs opcionals

Els *parsers* XML emprats al projecte sempre comproven si un document XML està ben format i opcionalment poden verificar la validesa si s'indica el document de definició. A la implementació s'utilitzarà DTD's[15] per a la definició de les regles gramaticals.

```
SAXBuilder parser = new SAXBuilder(true);
Reader stringReader=new StringReader(new String(docXMLtoValidate));
Document xmldoc = null;
```

[15] A l'Annexa C es poden consultar els DTD's utilitzats per a la validació XML

```
try {
        xmldoc = parser.build(stringReader);
} catch (Exception e) {
        e.printStackTrace();
}
```

Figura 13. Exemple de codi on es valida un document XML

S'ha decidit que és requisit necessari que els documents XML utilitzats per la transferència de dades als protocols siguin ben formats i vàlids.

5.2.4 Model del Comunicació dels documents XML

Una vegada es disposa del document XML que representa un conjunt de dades cal pensar en com es transferirà per la xarxa [JAVAW2000].

Una dada important a considerar es la utilització de RMI[16]. Això ens permet enviar objectes Java serialitzats. Per motius d'eficiència es convertirà prèviament el document XML (classe Document de JDOM) en una tira de bytes i després es transferirà.

A la següent figura es mostra el model de comunicació de dades emprat des de que un document es genera fins que s'envia i es rep al destí.

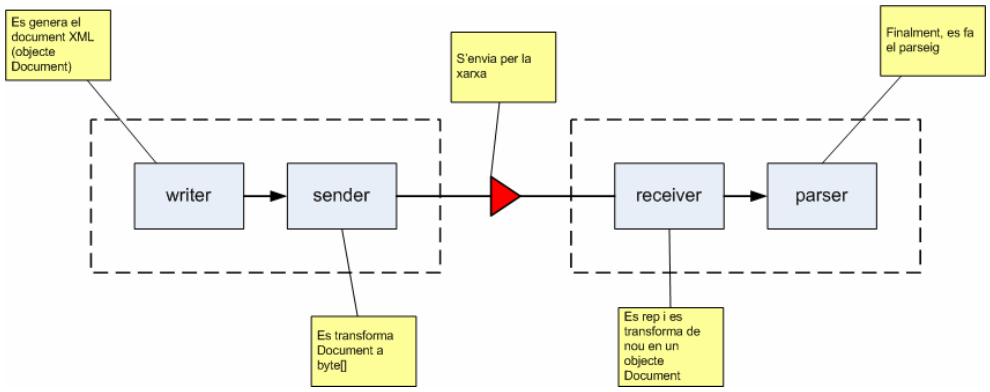

Figura 14. Model de comunicació d'un document XML

5.3 Implementació

Per tal d'implementar la funcionalitat XML per a la representació de les dades de l'aplicació de gestió d'historials mèdics segurs s'ha utilitzat la llibreria JDOM (*Java Document Object Model*) [JDOM11].

JDOM és un API pensada específicament per al processament de documents XML amb Java que permet l'obtenció d'una representació d'objectes en forma d'arbre dels elements, atributs, comentaris, instruccions de processament, etc. d'un XML. Una vegada construït aquesta estructura a partir del document XML parsejat es pot accedir de forma directa a qualsevol del seus components.

A la solució proposada, la capa de representació de les dades està implementada a la classe dedicada `XMLManager`.

[16] Aquesta decisió es veurà en detall en el capítol 6.

La funcionalitat d'aquest component és exclusiu per a la gestió de objectes XML i és l'únic component del sistema especialitzat en la gestió i transformació d'objectes a documents XML. Aquesta classe permet fer una abstracció sobre la representació i es utilitzada per la resta de classes que implementen els protocols per a la transformació d'un objecte (historial, visita, missatge, etc.) en una estructura estàndard.

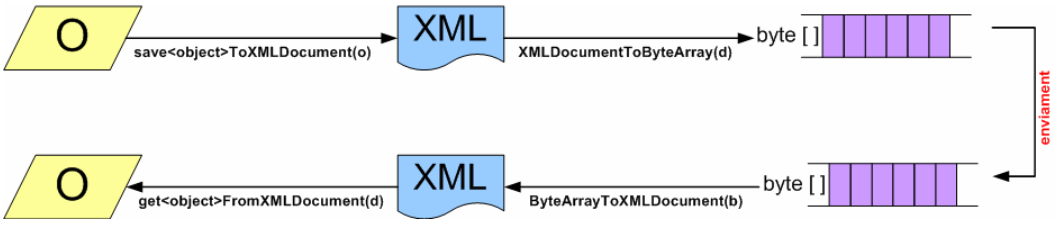

Figura 15. Procés de conversió de dades i crides a mètodes adients

Cal destacar dos tipus de mètodes a la implementació:

1) Mètodes que transformen un objecte a un document XML i viceversa.
   Pertanyen a aquest grup:
   - `saveHistorialToXMLDocument()`: Transforma un objecte Historial en un document XML segons la representació escollida.
   - `saveVisitaToXMLDocument()`: Transforma un objecte Visita en un document XML segons la representació escollida.
   - `saveMetgeToXMLDocument()`: Transforma un objecte Metge en un document XML segons la representació escollida.
   - `saveSerializedListToXMLDocument()`: Transforma un objecte llista serialitzada en un document XML segons la representació escollida.
   - `getHistorialFromXMLDocument()`: Genera un objecte Historial a partir d'un document XML que el representa.
   - `getVisitaFromXMLDocument()`: Crea un objecte Visita a partir d'un document XML que el representa.
   - `getMetgeFromXMLDocument()`: Retorna un objecte Metge a partir d'un document XML que el representa.
   - `getSerializedListFromXMLDocument()`: Construeix un objecte llista serialitzada a partir d'un document XML que el representa.

2) Mètodes que converteixen un document XML a tira de bytes i viceversa.
   Aquest grup està format per:
   - `XMLDocumentToByteArray()`: Serialitza un objecte Document (XML) en una tira de bytes.
   - `ByteArrayToXMLDocument()`: Construeix un document XML a partir d'una tira de bytes.

Al abordar aquesta fase només ha calgut incloure els canvis necessaris als protocols i procediments criptogràfics per a realitzar l'enviament o la recepció de dades en format XML en lloc d'una tira de bytes instanciant la classe i cridant als mètodes apropiats.

6. COMUNICACIÓ DELS COMPONENTS

6.1 RMI: Definició i justificació

La comunicació dels diferents components és una part essencial del PFC. Els usuaris (metges o pacients) es comuniquen amb el gestor del sistema per tal de sol·licitar un servei.

La comunicació dels diferents components del sistema tradicionalment suposaria el disseny d'un protocol o mecanisme de comunicació propi. No obstant, aprofitant l'existència de protocols àmpliament utilitzats i degut a l'homogeneïtat de la plataforma de desenvolupament (Java), s'ha escollit RMI (*Remote Method Invocation*) [RMI16] com a tecnologia per a la comunicació entre els components del sistema. Java incorpora aquesta tecnologia a l'API estàndard. En el cas d'un entorn heterogeni s'haurien haver avaluat solucions basades en, per exemple, *Web Services*.

RMI permet al desenvolupador crear aplicacions distribuïdes en Java (aquest és un requeriment obligatori), en els quals els mètodes dels objectes remots de Java poder ser invocats des de màquines virtual de Java remotes.

Les aplicacions RMI es composen de dos components separats: un servidor i un client. A la part servidor s'executen diferents instàncies de les classes servidores que es necessiten. Les aplicacions que volen emprar els mètodes remotes només necessiten saber la interfície del servidor, és a dir, els mètodes que ofereix la classe servidora. D'aquesta forma es fa independent la implementació de la interfície que veu el client. Un canvi de la implementació no té perquè suposar un canvi a la interfície ni al programari que el client faci servir.

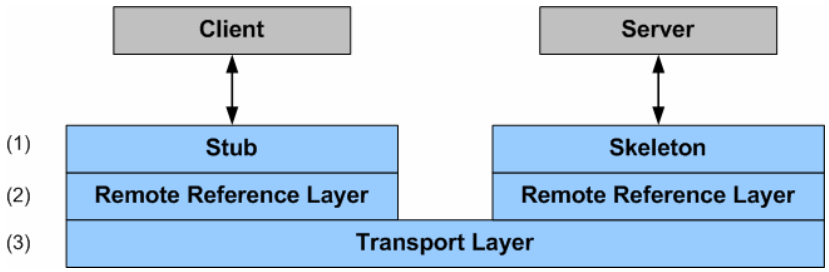

Figura 16. Arquitectura de comunicacions amb tecnologia RMI

RMI proporciona els mecanismes necessaris per que la part client i la part servidora es comuniquin i es passin informació:

(1) Capa d'Enllaç:
Interfície entre client i servidor. L'objectiu d'aquest nivell és assegurar les crides remotes. Per la part client es coneix com *Stub* i en la part servidora s'anomena *Skeleton*. La generació d'aquest elements es realitza amb l'eina `rmic`.

(2) Capa de Referència:
Es l'encarregada de la gestió de les crides de la capa d'enllaç per a que s'executin realitzant els objectes i executant els mètodes dels mateixos.

(3) Capa de Transport:
S'encarrega de mantenir la connexió client/servidor. Es manté una taula d'objectes remots disponibles per a ser servits a la capa superior.

6.1.1 Publicació del servei

El servidor RMI que crida a la implementació dels mètodes oferts a la seva interfície haurà de publicar-se (amb un nom de servei i un port) per a que els clients puguin accedir-ne. La aplicació registrarà els seus objectes remots amb una eina inclosa al JDK de Java: `rmiregistry`.

Per tal de fer això, la interfície de la classe servidora haurà d'estendre la interfície `java.rmi.Remote`. Aquesta acció suposa la creació d'una interfície remota.

6.1.2 Localització d'objectes remots

Com s'ha apuntat anteriorment, la tecnologia RMI permet executar mètodes d'objectes remots com si estiguessin ubicats localment de forma que sigui totalment transparent al programador. Per a aconseguir això, RMI utilitza un servei de resolució de noms que possibilita fer una invocació al objecte pel seu nom, independentment de la màquina on resideixi.

Les referències a objectes remots es guarden utilitzant mètodes basats en la classe `java.rmi.Naming`. Abans de la invocació, el client ha d'obtenir la referència de l'objecte remot a partir de la utilització dels mètodes que proveeix la classe `java.rmi.Naming` per cercar i enllaçar (`bind`), reenllaçar (`rebind`), desenllaçar (`unbind`) o llistar el nom de l'objecte.

6.2 Disseny de la capa de comunicacions dels components

En aquest apartat s'incorporen les decisions de disseny relatives a la capa de comunicacions així com les afectacions a altres fases del desenvolupament d'aquest PFC.

6.2.1 Millora de protocols criptogràfics

Com s'explica a l'apartat d'Implementació ha calgut separar els protocols criptogràfics per a que tant l'usuari com el gestor executi la seva part de forma remota.

El protocol d'autenticació proposat per cadascun dels protocols inclou la generació de nombres per a aplicar un mecanisme de desafiament/resposta (N_i i NG). En una execució local i lineal no hi ha problema en la identificació de quin identificador d'usuari està cridant al protocol. No obstant, en un entorn distribuït cal que quedi clar el remitent de tot missatge de qualsevol protocol.

Per tal de poder fer això, s'ha modificat el pas 3 de tots els protocols i s'ha afegit al missatge on s'indica el tipus de protocol (o servei) un camp amb l'identificador de l'usuari (a vegades serà un metge i d'altres un pacient) qui demana al gestor una acció.

Per exemple, suposem que un usuari (U) vol executar el protocol 1 (consulta de les dades generals d'un pacient) i ho demana al gestor (G). Els esdeveniments seguirien, de forma resumida, aquesta línea d'execució:

1) U executarà procedure1 i enviarà el resultat a G

2) G respondrà a U amb el resultat del procedure2. Aquí es guardaria, Id_usuariU, N_i i NG a la base de dades

3) U enviarà petició amb [NG, Consulta_dades_generals, Id_usuari]. Aquest últim camp és l'identificador del pacient del qui es vol consultar l'expedient mèdic i no té perquè coincidir amb l'identificador de l'usuari, és a dir, Id_usuariU

4) G comprovarà que NG = NG' i Id_usuariU == Id_usuariU'. Aquí es on G hauria de recuperar NG de la BBDD d'acord amb el Id_usuariU que ha requerit el servei.

Com que els diferents passos es poden realitzar de forma remota i concurrent amb altres usuaris, en el pas 3 caldrà que U indiqui el seu identificador. Per tant, afegir el camp Id_usuari al missatge quedarà com: [NG, Consulta_dades_generals, Id_usuari, Id_usuariU].

G podrà ara verificar si a la seva base de dades existeix un registre NG corresponent al Id_usuariU rebut al missatge.

Al capítol 4 s'ha indicat aquest canvi a la documentació dels protocols marcant els camps inserits als missatges en color blau.

6.2.2 Gestor de serveis

La filosofia seguida per G serà la d'un productor de serveis (el client serà el consumidor). Per tant, una vegada passada la fase d'autenticació entre l'usuari i el gestor, es demanarà l'execució d'un servei per part de la part servidora.

Amb aquest model de funcionament girarà el disseny de la capa de comunicacions per a que G pugui rebre la petició, analitzar el tipus i correctesa, realitzar les tasques relatives a la sol·licitud, respondre amb els resultats al client.

Un benefici clar d'aquest disseny és que afegir un nou servei serà tan fàcil com incloure un nou servei gestionat pel gestor.

6.2.3 Protocols criptogràfics en l'entorn distribuït

Amb les anteriors premisses clarificades s'adjunta a continuació els diagrames de seqüència reduïts pels protocols criptogràfics descrits al capítol 4 d'aquesta memòria.

Protocol 1

L'usuari comença el protocol iniciant l'autenticació amb el gestor. Una vegada es supera aquesta fase (amb la crida del procedure 1 i 2) el client envia la petició al servidor. Dintre del missatge indicarà el tipus de petició i els paràmetres necessaris per a l'execució.

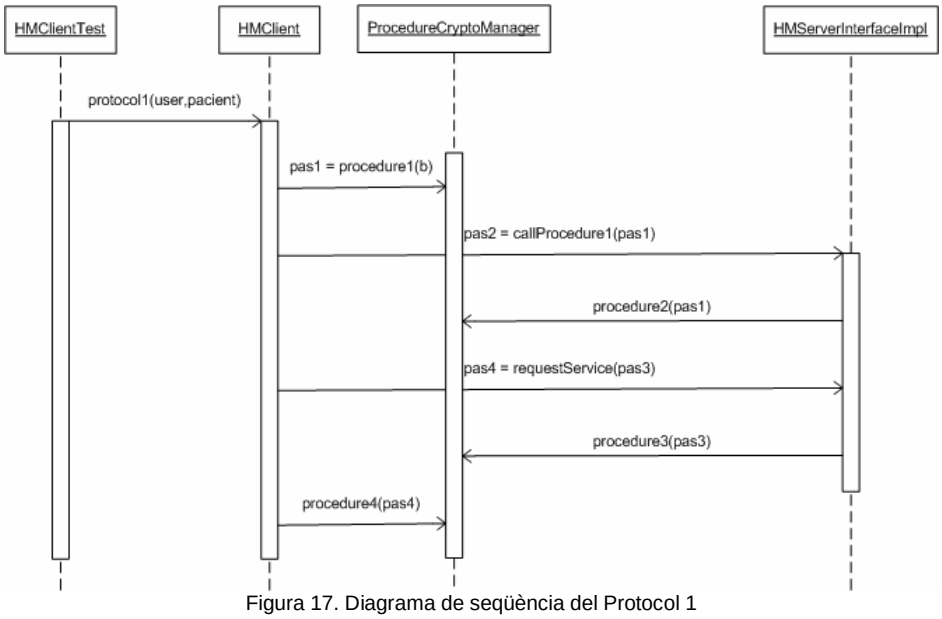

Figura 17. Diagrama de seqüència del Protocol 1

En aquest protocol el client demana la consulta d'un historial. El gestor després de fer algunes comprovacions cridant al procedure 3, consultarà el historial i el farà arribar al sol·licitant.

Protocol 2

L'usuari vol consultar les dades d'una visita. Igual que en la resta de protocols, primer s'inicia l'autenticació. Si és positiva, s'envia la petició al gestor. Aquest últim la serveix, consultant l'historial del pacient i retorna el resultat al client. El receptor tracta la informació rebuda per a poder mostrar-la.

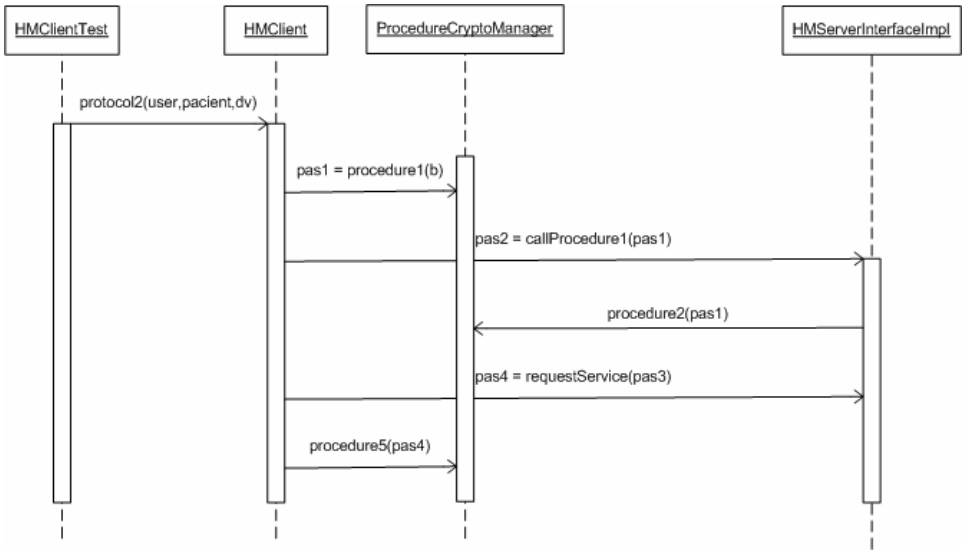

Figura 18. Diagrama de seqüència del Protocol 2

Protocol 3

L'objectiu d'aquest protocol és obtenir la llista de pacients assignats a un metge. Després de l'autenticació, el gestor serveix a aquesta informació al client, que haurà de tenir el rol de metge.

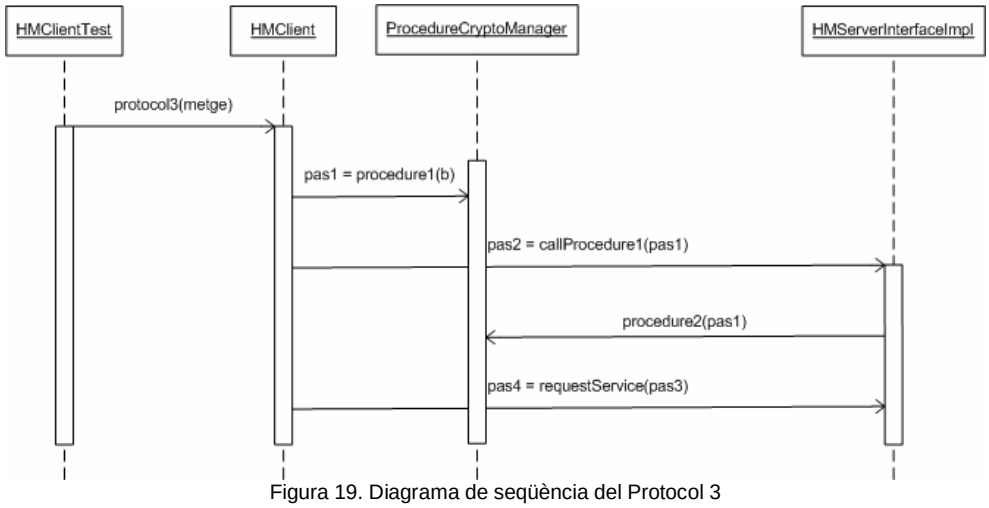

Figura 19. Diagrama de seqüència del Protocol 3

Protocol 4
En aquest protocol es vol afegir una visita a l'historial d'un pacient. S'aconsegueix quan, després de l'autenticació prèvia, el client indica al gestor o servidor central, les dades de la visita. El gestor fa persistent aquesta dada a l'historial corresponent.

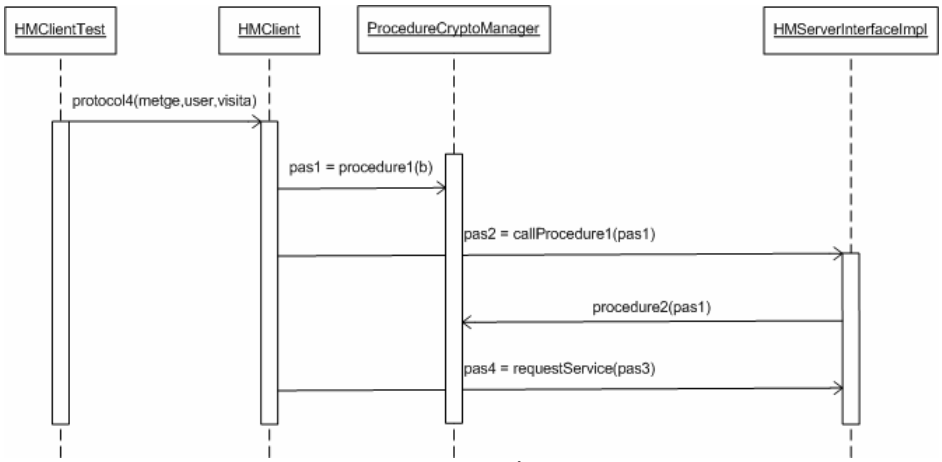

Figura 20. Diagrama de seqüència del Protocol 4

Protocol 5
Aquest protocol és la implementació protocol d'autenticació. És realitza en dues fases degut a que la validació de la identitat és bilateral (client i servidor).

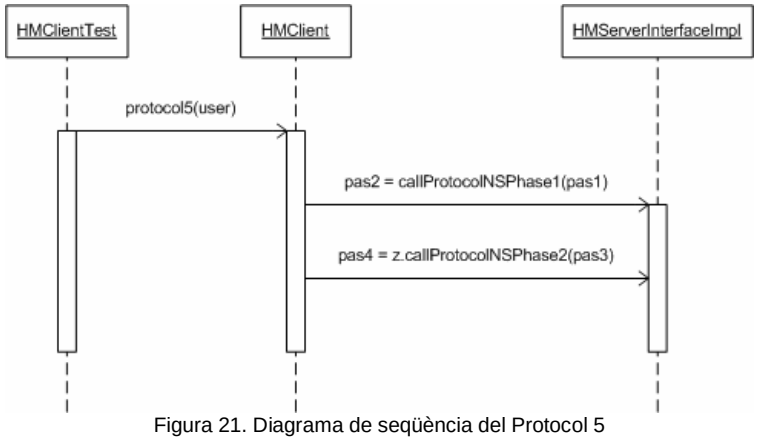

Figura 21. Diagrama de seqüència del Protocol 5

6.3 Implementació
Com s'ha explicat en l'apartat de disseny, s'ha decidit utilitzar la tecnologia RMI per a implementar el protocol de comunicacions entre els dos components principals del sistema: el client (tant pugui ser un usuari amb rol pacient com metge) i el gestor. A continuació s'explica amb detall les consideracions relatives a la implementació portada a terme.

6.3.1 Tecnologia RMI i la implementació
Per a la implementació d'aquesta fase s'ha requerit separar la funcionalitat entre la part client i la part servidora. El que això implica és implementar de forma separada els protocols criptogràfics

definits anteriorment dotant als components que es comuniquen de la seva part del protocol. La implementació dels procediments no ha estat modificada.

En la següent figura il·lustra com s'ha implementat a nivell de classes la tecnologia RMI escollida en el de disseny d'aquesta fase de desenvolupament del PFC.

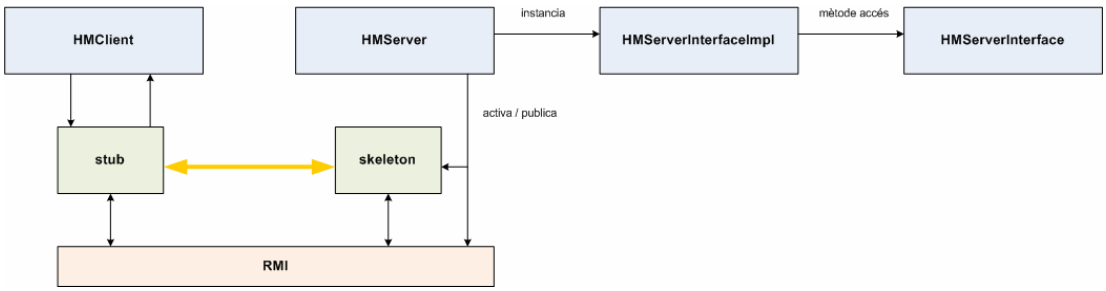

Figura 22. Model de comunicació RMI al projecte

6.3.2 Funció de les classes desenvolupades

A continuació una descripció més detallada de la funció de cadascuna de les classes desenvolupades:

- `HMClient`: Implementa la part client dels protocols definits a l'esquema criptogràfic. Serà cridada pel llançament dels protocols per la interfície d'usuari (Pacient o Metge). Aquesta interfície visual es desenvoluparà en fases posteriors[17].

- `HMServer`: Classe que publicarà per RMI el servidor instanciant la classe que implementa la interfície remota.

- `HMServerInterface`: Classe que defineix la interfície remota que ha de conèixer el client per a poder fer les crides a mètodes remots del servidor RMI.

- `HMServerInterfaceImpl`: Classe que implementa tots els mètodes de la interfície remota. Aquesta classe incorporarà també la implementació de la part servidora dels protocols criptogràfics.

  La classe `HMServerInterfaceImpl` incorpora el mètode `requestService()` que implementarà la gestió de serveis disponibles als usuaris.

  Una vegada es genera aquesta classe, el servidor utilitzarà la eina `rmic` incorporada al JDK de Java per a generar els corresponents *stubs* necessaris per a la comunicació per RMI entre els components.

És interessant notar que afegir una nova funcionalitat serà tan fàcil com:

1. Definir un protocol criptogràfic que garanteixi les propietats de seguretat predefinides.
2. Incorporar la crida del client amb un nou tipus de petició i els paràmetres necessaris.
3. Afegir al mètode `requestService()` del gestor la implementació d'aquest nou tipus de petició.

[17] Veure capítol 8 i 9

A la següent figura es pot veure el diagrama de classes UML de la comunicació RMI:

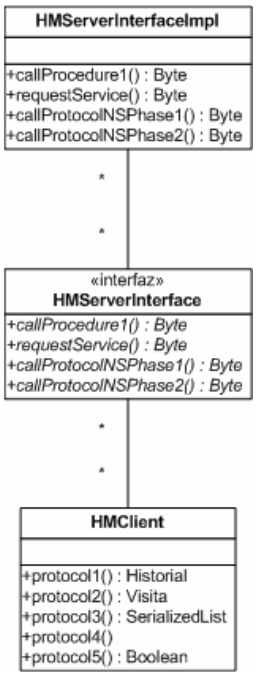

Figura 23. Diagrama de Classes per la comunicació RMI

7. GESTIÓ DE LA INFORMACIÓ

7.1 Definició

7.1.1 Capa de persistència

La capa de persistència ha estat dissenyada seguint un model Entitat-Relació. Ha estat necessari escollir una base de dades relacional i un SGBD adequat.

S'ha seleccionat MySQL Community Edition Databaser Server versió 5.0.51a [MYSQL5] . Aquest gestor és idoni per les necessitats del projecte. Es tracta d'un sistema de dades relacional SQL de codi obert (llicència GPL). Està desenvolupat, distribuït i suportat per Sun Microsystems (abans MySQL AB).

La versió escollida és lliure i gratuïta sempre i quan es destini a fins acadèmics, com és el cas.

7.1.2 Instal·lació i configuració del SGBD

Per a la instal·lació s'ha utilitzat el assistent de la distribució del programari. Addicionalment s'ha usat aquest programa d'ajuda per a la configuració dels paràmetres del gestor de base de dades relacional.

Els paràmetres més significatius són:

Paràmetre	Valor
Port de connexió:	tcp/3306
Tipus de base de dades:	OLTP (*OnLine Transaction Processing*)
Número de connexions simultànies:	20
Tipus d'emmagatzemament per defecte:	InnoDB
Mida del fitxer de log:	10 MBytes
Usuari administrador del SGBD:	root

Figura 24. Paràmetres del SGBD

7.1.3 Llibreries per a la connexió a base de dades

L'accés a la base de dades es realitza mitjançant JDBC (*Java DabaBase Connectivity*). El SDK de Java incorpora les llibreries pròpies per a l'accés per JBDC a una base de dades relacional.

No obstant, per a millorar l'eficiència s'ha optat instal·lar les llibreries pròpies de MySQL per a la connexió entre aplicacions desenvolupades en Java i la base de dades. MySQL anomena aquest component de programari `MySQL Connector/J`.

En el apartat d'annexos s'inclou una descripció breu de com s'instal·la aquesta llibreria específica, juntament amb la resta de programari necessari per a l'execució del producte construït.

7.2 Disseny de la capa de dades

Tenint en compte la tecnologia per la capa de persistència i amb els requisits sobre el tipus de dades a emmagatzemar es prenen les decisions sobre el model de dades i que després incidiran en el desenvolupament.

7.2.1 Model Conceptual

Com ja s'ha introduït en capítols anteriors, a la base de dades es guardaran les dades referents a els historials dels pacients, les visites, els usuaris de l'aplicació (pacients i metges) i els certificats dels usuaris.

La informació referents als historials, metges i visites es guardaran en XML directament per fer més fàcil la consulta i/o modificació. Les dades dels historials i visites estaran protegides tal i com s'ha definit en l'esquema criptogràfic. Un requisit a destacar és que no hi ha d'haver cap relació entre les visites i els pacients.

A la següent figura es mostra el model conceptual dissenyat per la capa de persistència de dades que s'implementarà.

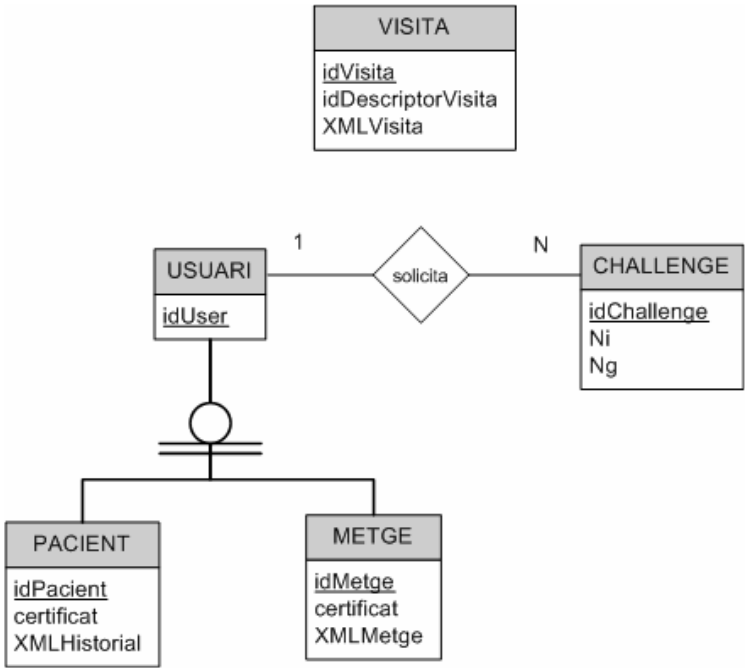

Figura 25. Diagrama relacional de la base de dades. Model conceptual

Entitat VISITA
Contindrà les visites registrades al sistema. El camp `idVisita` serà la clau primària. El camp `idDescriptorVisita` contindrà l'identificador del descriptor de visita. Les dades de la visita es guardaran en format XML directament al camp `XMLVisita`.

Entitat USUARI
Es guardaran els usuaris amb accés al sistema. El camp `idUser` serà l'identificador (N.I.F.) de l'usuari que servirà com a clau primària de l'entitat.

Entitat PACIENT
Representarà a un usuari amb rol pacient. Serà part d'un subconjunt de l'entitat usuari. El camp `idPacient` serà l'identificador del pacient, clau primària i forana amb referència a l'entitat USUARI.

Es desaran les dades del seu historial al camp `XMLHistorial` en format XML. El camp `certificat` conté el certificat X.509 de l'usuari.

Al tractar-se d'un subconjunt es permetrà que un usuari pugui tenir un rol de metge i a la vegada ser un pacient.

Entitat METGE
Representarà a un metge. S'identificarà amb el camp `idMetge` i contindrà les dades relatives al metge en forma XML al camp `XMLMetge`.

Igual que passava amb l'entitat PACIENT serà part d'un subconjunt de USUARI. El camp `certificat` contindrà el certificat X.509 del metge.

Entitat CHALLENGE
Es guardaran els desafiaments actius (derivats de la implementació dels protocols criptogràfics proposats) entre els usuaris i el gestor.

El camp `idChallenge` servirà com a clau primària de l'entitat. `Ni` i `Ng` seran els nombres aleatoris generats pel usuari i gestor, respectivament, en l'execució d'un protocol.

Relació SOLICITA (USUARI → CHALLENGE)
Relaciona un usuari amb els desafiaments actius. És una relació 1:N, ja que un mateix usuari pot tenir un o varis desafiaments en curs, però un desafiament (parella Ni i Ng) només pertany a un usuari.

7.2.2 Usuari de connexió

A la base de dades només accedirà el Gestor que serà l'encarregat de fer arribar les dades a les aplicacions externes (tant dels pacients com dels metges).

No obstant, per a minimitzar els permisos d'accés al SGBD des del Gestor es crea un usuari específic de base de dades amb el mínim de privilegis sobre la base de dades de l'aplicació. Aquest usuari de connexió només podrà connectar-se[18] localment a la base de dades i tindrà permisos de inserció, selecció, modificació i esborrament dels registres.

D'aquesta manera es redueix el perill, ja sigui per error o de forma malintencionada, de realitzar accions sobre el SGBD que afectin a la seva integritat.

7.2.3 Gestió dels desafiaments

Una vegada el protocol s'executi caldrà que s'elimini de la base de dades el Ni i Ng utilitzat. Per això, com a últim pas en la part servidora, el gestor esborrarà de l'entitat CHALLENGE el registre corresponent.

Aquesta modificació implica l'addició d'un pas més en alguns dels protocols criptogràfics[19], en concret, al pas 4 que executa G.

[18] Veure scripts de creació de la base de dades a l'apartat d'Annexos per a més informació sobre aquesta configuració
[19] Veure capítol 4 "Esquema Criptogràfic" per més detall

7.2.4 Gestor del rol d'usuari en base al seu certificat

Un usuari podrà disposar del rol de pacient i metge de forma simultània. Per exemple, als matins un usuari podria voler accedir al sistema com a metge degut a la seva professió i per la tarda, podria ser usuari del sistema com a pacient al rebre assistència mèdica per altre facultatiu.

Aquest fet serà controlat pel Gestor per mitjà de verificar el rol indicat al certificat que l'usuari presenta en la fase d'autenticació (camp `Organizational Unit Name`).

Per l'exemple esmentat, l'usuari haurà de tenir dos certificats: un com a metge que presentarà al matí i altre com a pacient que utilitzarà a la tarda.

7.2.5 Validació del certificat d'usuari

Una mesura de seguretat addicional que realitzarà el gestor serà la de validar la caducitat del certificat emmagatzemat quan s'utilitzi als protocols. Aquesta verificació es farà sempre que es recuperi el certificat de la base de dades.

Si el certificat està caducat o encara no és vàlid (perquè la seva data d'inici es posterior a la data actual) no es continuarà amb l'execució del protocol i es retornarà un error a l'usuari indicant aquest fet.

7.3 Implementació

La implementació d'aquesta fase ha tingut dos tasques principals:

1) Generació d'un model lògic a partir del model conceptual proposat en el disseny

2) Implementar una classe que realitzi la gestió i accés al model de dades

7.3.1 Model Lògic de Dades

A partir del model conceptual tenim el següent model de dades:

```
USUARI (idUser)

PACIENT (idPacient, certificat, XMLHistorial)
      on {idPacient} referencia USUARI(idUser)

METGE (idMetge, certificat, XMLMetge)
      on {idMetge} referencia USUARI(idUser)

CHALLENGE (idChallenge, Ni, Ng, idUser)
      on {idUser} referencia USUARI(idUser)

VISITA (idVisita, idDescriptorVisita, XMLVisita)
```

Es tradueix el subconjunt de l'entitat USUARI en tres taules: USUARI que és la taula principal on l'identificador d'usuari és la clau primària; PACIENT i METGE on tindran una clau forana que apuntarà a la clau primària de la que són subconjunt. Aquestes dues taules incorporaran les dades específiques al tipus d'usuari.

Per l'entitat VISITA només necessitarem una taula, amb la seva corresponent clau primària que serà l'identificador de la visita.

Per la relació SOLICITA es crearà una taula (CHALLENGE) per a emmagatzemar els diferents desafiaments. Al ser una relació 1:N serà necessària una clau forana que faci referència a l'identificador de l'usuari el qual ha llançat el desafiament.

A l'annex D d'aquest document s'inclou els *script* de comandes SQL per a la creació d'aquest model al SGBD escollit.

7.3.2 Gestor de la capa de dades

La classe `DBManager.java` serà l'encarregada de l'accés a les dades emmagatzemades a la base de dades relacional. Els seus mètodes permeten consultar, inserir, modificar i esborrar els registres de les taules del model lògic segons sigui necessari per l'execució de l'aplicació que es desenvolupa.

`DBManager` serà l'única classe que coneix el model de dades i el SGBD emprat. Un canvi de SGBD o de model serà transparent per la resta de classes i només caldrà fer els canvis necessaris en la classe que fa la interfície amb la capa de dades.

Només serà invocada pel servidor, el gestor, ja que serà ell qui té accés exclusiu a la capa de dades de l'aplicació.

8. INTERFÍCIE DEL PACIENT

8.1 Definició

Els usuaris interaccionen amb el sistema mitjançant la interfície gràfica, per tant és una part molt important de qualsevol sistema. L'objectiu principal és aconseguir una interfície simple, intuïtiva, i fàcil d'utilitzar.

Encara que la temàtica del projecte es centra en la robustesa de l'aplicació des d'un punt de vista de la seguretat informàtica, s'ha dedicat una fase per al disseny i implementació de la part gràfica, visible als dos rols principals de l'aplicació desenvolupada: els pacients i els metges.

En aquest capítol es mostrarà quines decisions de disseny s'han pres per aconseguir els objectius indicats anteriorment, quines eines i quins productes s'han obtingut en aquesta fase.

8.2 Disseny de la Interfície del Pacient

La interfície gràfica ha de proporcionar les funcionalitats demanades als requeriments, que de forma resumida són:

Pel rol Pacient:

- Autenticació
- Consulta de les dades del seu historial
- Consulta de les dades de les seves visites

Pel rol Metge:

- Autenticació
- Consulta de les dades del seu historial i els seus pacients
- Consulta de les dades de les seves visites i els seus pacients
- Llistar els pacients assignats
- Afegir una visita a un pacient assignat

A continuació s'expliquen les decisions de disseny per a assolir els objectius marcats per la interfície client del pacient[20]. Com es podrà veure a l'apartat 8.3 la interfície gràfica és comuna excepte en la implementació de les operacions disponibles només per a usuaris amb rol metge.

8.2.1 Interfície principal

La interfície principal serà l'aplicació cridada per l'usuari. Aquesta finestra principal està dissenyada de forma que:

- A la part superior es troba un menú amb les opcions disponibles: Inici, Opcions, Ajuda.

[20] Al següent capítol es descriurà més detalladament les particularitats de la interfície client del metge

- A la part principal, sota el menú, es troba la zona principal on es mostrarà el resultat de les diferents consultes.

8.2.2 Opcions disponibles per a un pacient

Una vegada passada la fase d'autenticació de forma satisfactòria ("Inici de sessió") es faran visibles al menú "Opcions" les possibles operacions per a un pacient, que seran:

1) Consulta d'Historial (només el seu)

2) Consulta de visites (del seu historial)

8.2.3 Ús de pestanyes

Per a mostrar la informació de forma més ordenada s'ha decidit dividir la zona central en pestanyes que podran ser de dos tipus: fixes, no fixes.

Les pestanyes fixes correspondran a 1) les dades generals de l'historial del pacient i 2) el llistat de les visites (data i hora, identificador, tema i metge) de l'historial del pacient.

Les pestanyes no fixes i, per tant, no apareixeran per defecte al arrencar el programa i podran ser tancades en qualsevol moment, podran contenir la següent informació: dades d'una visita o llistat de pacients. Aquest tipus de pestanya només podrà ser utilitzada per un usuari amb rol metge.

Les pestanyes amb les dades d'una visita s'aniran numerant de forma seqüencial per a poder-les identificar de forma fàcil.

8.2.4 Ús de taules per al llistat de visites i relació amb la consulta de visita

Per a fer fàcil l'accés a les visites de l'historial es disposarà el llistat amb les dades principals d'un descriptor de visita en forma de taula.

Degut a la llargària i complexitat de l'identificador de visita l'usuari seleccionarà el element de la taula corresponent la visita i podrà optar per consultar les dades utilitzant la operació adient del menú d'opcions[21].

8.2.5 Inici de sessió

Per tal d'iniciar la sessió de forma correcta, un usuari haurà de identificar-se amb el seu N.I.F. (en format XXXXXXXX-X) i una contrasenya.

La contrasenya haurà de correspondre amb la paraula de pas que protegeix el certificat d'usuari (en un contenidor PKCS#12 [PKCS12]). També es comprovarà que l'identificador proporcionat és igual que el camp `dnQualifier` del certificat expedit per l'Autoritat de Certificació.

Una vegada obert el certificat i carregades les credencials es passarà a executar el protocol criptogràfic d'autenticació (número 5). Si el resultat és positiu l'usuari es considerarà autenticat i, per tant, haurà iniciat sessió.

Es controlarà el nombre de reintents, que inicialment seran 3, per al protocol criptogràfic d'autenticació. Si l'usuari supera aquest nombre màxim permès el programa finalitzarà. No obstant, no s'ha considerat la opció de bloquejar l'accés de forma definitiva.

[21] Al manual d'usuari incorporat als annexes es dona més informació al respecte

Cal dir, que degut a que totes les operacions criptogràfiques executen un protocol d'autenticació bilateral previ, el servidor no haurà de guardar i gestionar les sessions.

8.3 Implementació

La implementació es basa en la utilització del conjunt d'eines gràfiques *Standard Widget Toolkit* (SWT) [SWT332] creat inicialment per l'empresa IBM i actualment desenvolupat per Eclipse Foundation.

Aquestes eines permeten dissenyar d'una manera senzilla la interfície gràfica d'una aplicació Java. SWT es compon bàsicament per tres elements:

1. Llibreria nativa per interacció amb el sistema operatiu. És depenent de la plataforma.
2. Classe `Display`, interfície utilitzada per comunicar-se amb la interfície gràfica.
3. Classe `Shell`, que representa la finestra a alt nivell i on es poden afegir els *widgets*, controls i components.

8.3.1 Classes desenvolupades per la interfície gràfica del pacient

Les classes dedicades al nivell de presentació pel rol de pacient són:

- `UIMain`: Encarregada de mostrar a l'usuari la finestra principal, el menú amb les opcions disponibles i els resultats de les consultes.

- `UILogin`: Implementació de la finestra d'autenticació. S'encarrega de llançar el protocol criptogràfic número 5.

- `UISearchHistorial`: Finestra de captura de dades per a la cerca de l'historial (donat un identificador de pacient). Crida al protocol criptogràfic 1.

- `UISearchVisita`: Implementa la funcionalitat de cerca de visita. Donat un identificador de visita seleccionat crida al protocol 2 per a la cerca de les dades de la visita.

- `UIAuthors`: Mostra la finestra de crèdits informant de la versió i autors de l'aplicació.

8.3.2 Finestra principal

Formada pel menú i la zona principal de dades dividida en pestanyes. A la figura es veu la pestanya de Dades Generals que conté part de les dades del historial del pacient, és a dir, nom i cognoms, C.I.P., N.I.F., grup sanguini i al·lèrgies.

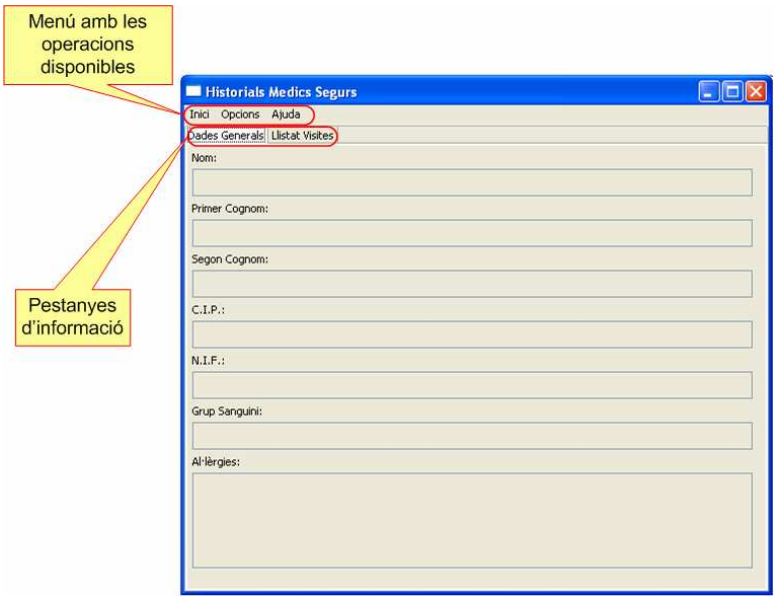

Figura 26. Interfície gràfica principal de pacient i metge

A la figura següent es mostra la pestanya amb el llistat de les visites de l'historial del pacient. Serà una taula amb els camps data, hora, identificador de la visita, tema i identificador del metge.

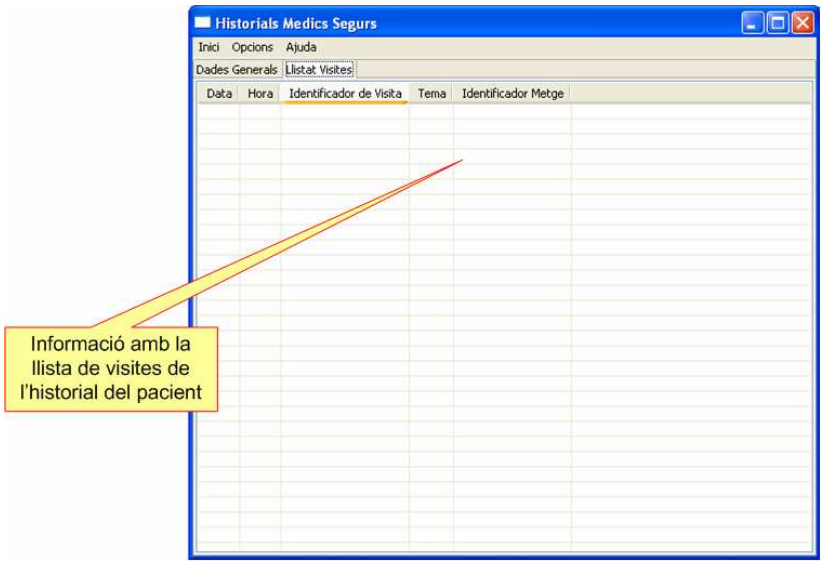

Figura 27. Interfície principal. Pestanya "Llistat Visites"

8.3.3 Finestra d'inici de sessió

En aquesta finestra l'usuari haurà d'introduir el seu identificador i la contrasenya del PKCS#12 utilitzat per a la validació.

Figura 28. Finestra Inici de Sessió

8.3.4 Finestra de consulta d'historial

Per consultar les dades de l'historial, l'usuari prem la opció del menú "Consultar historial..." i indica en la finestra corresponent l'identificador del pacient a realitzar la consulta d'expedient.

Figura 29. Finestra de Consulta d'historial

8.3.5 Finestra de consulta de dades de visita

Una vegada es disposa de les dades de l'historial, l'usuari accedeix a la pestanya amb el llistat de visites i selecciona la que vol obtenir.

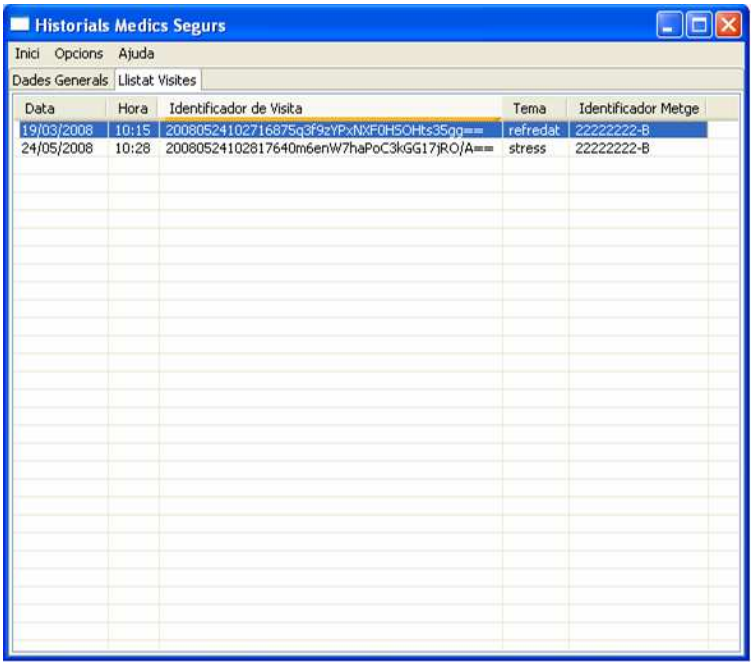

Figura 30. Finestra amb una visita seleccionada

Després escull la opció “Consultar visita seleccionada” del menú.

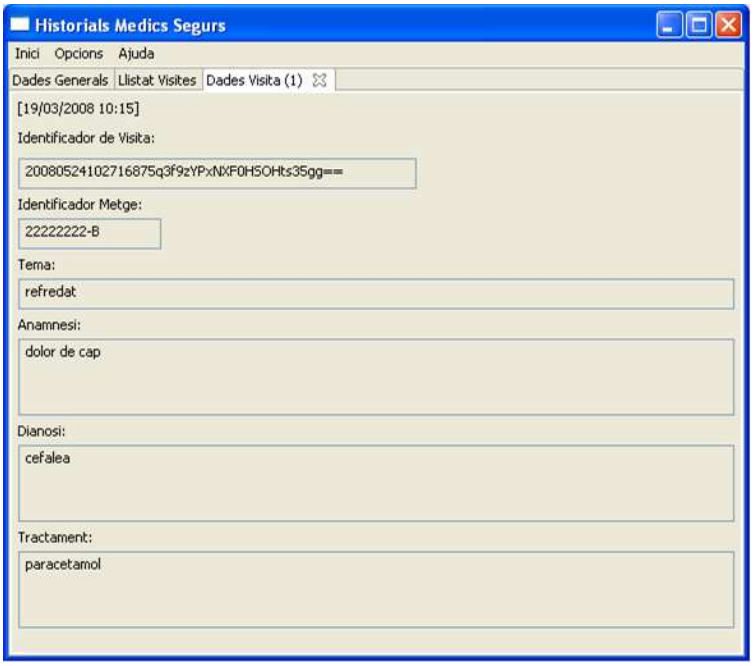

Figura 31. Finestra amb les dades d’una visita

El capítol 11 “Joc de Proves” inclourà més detall sobre exemples i ús de l’aplicació.

9. INTERFÍCIE DEL METGE

9.1 Definició

L'usuari de l'aplicació amb rol metge utilitzarà la mateixa interfície gràfica que un pacient, però degut a la seva funció, li mostrarà més funcionalitats i, per tant, es comportarà de manera diferent.

9.2 Disseny de la Interfície del Metge

Per aquesta interfície client les decisions presses a l'anterior capítol són igual de vàlides. No obstant, a continuació s'expliquen les decisions de disseny específiques per la interfície gràfica del metge.

9.2.1 Canvi de certificat

Per motius pràctics s'ha decidit afegir la funcionalitat de canvi de certificat on l'usuari podrà escollir el certificat a utilitzar a l'aplicació. Principalment aquesta opció està dedicada per aquells usuaris que disposen múltiples rols. Per exemple, un metge vol accedir algunes vegades com a tal, però en altres ocasions vol ser usuari pacient de l'aplicació. En aquest cas, l'usuari haurà d'utilitzar el certificat adient al rol. També seria útil per a equips informàtics compartits o públics on poden l'aplicació pot ser usada per una o varies persones.

Per a facilitar el canvi de rol s'ha inclòs la opció "Canvi de certificat..." al menú "Inici". Per defecte, l'aplicació utilitzarà el PKCS#12 de la localització indicada al fitxer de configuració del client.

Un canvi de certificat provoca un canvi de context de l'usuari i, per tant, es forçarà l'inici de sessió, s'ocultaran les opcions disponibles fins aquell moment i es netejaran les dades de historial i llistat de visites que poguessin haver a la zona central.

9.2.2 Selecció i focus per les consultes

La manera de funcionar per a llançar les consultes és, primer, obtenir un llistat amb les dades bàsiques que es van a consultar, segon, seleccionar la que es vol més detall i, finalment, polsar l'opció adient del menú.

Aquesta manera de treballar és especialment important considerar-la en el cas de les consultes d'historial d'un pacient i dades d'una visita del l'historial.

9.2.3 Opcions disponibles per a un metge

Una vegada autenticat el metge tindrà disponible al menú "Opcions" les possibles operacions que seran:

1) Consulta d'Historial
2) Consulta historial de pacient seleccionat
3) Consultar visita seleccionada
4) Consultar pacients assignats
5) Afegir visita

9.3 Implementació

Igualment que per les opcions i operacions de la interfície client d'un pacient s'ha utilitzat la llibreria SWT per a la implementació.

9.3.1 Classes desenvolupades per la interfície gràfica del metge

Com la interfície principal és comuna, a la implementació de la interfície gràfica pel rol metge s'utilitzen les classes descrites anteriorment: `UIMain`, `UILogin`, `UIAuthors`, `UISearchHistorial` i `UISearchVisita`.

Addicionalment, s'han desenvolupant les següents classes encarregades de les operacions només disponibles per un usuari metge:

- `UISearcHistorialSelected`: Classe que implementa la cerca de l'historial d'un pacient seleccionat de la llista de pacients del metge. Crida al protocol 4.

- `UIGetLlistaPacients`: Classe que crida al protocol criptogràfic número 3 i obté la llista de pacients que mostra en una nova pestanya de la finestra principal.

- `UIAddVisita`: Classe que implementa la finestra d'inserció de visita. També és l'encarregada, una vegada captura les dades, de cridar l'execució del protocol 4 per a que la visita sigui afegida a l'historial del pacient seleccionat.

9.3.2 Finestra Llistat de Visites

Es disposaran en forma de taula les dades principals (N.I.F., nom i cognoms) dels pacients assignats al metge.

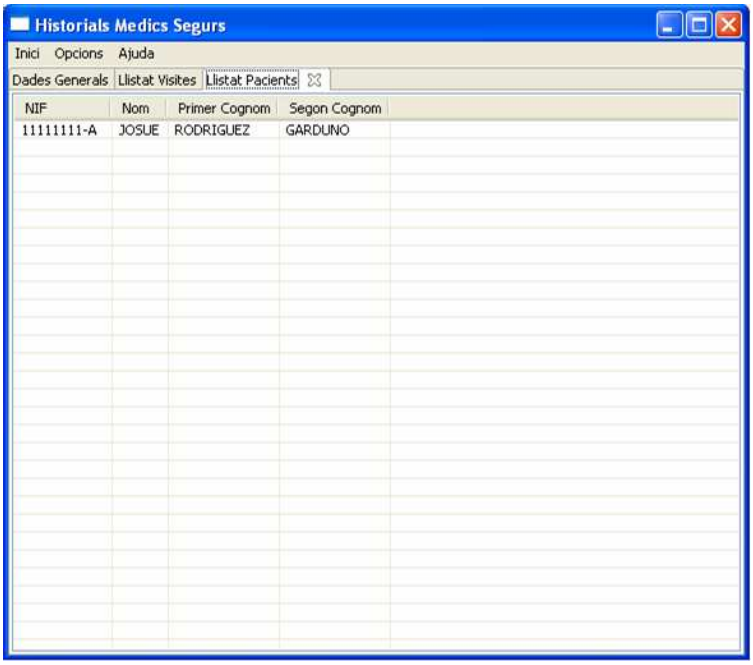

Figura 32. Finestra Llistat de Pacients assignats

A partir d'aquest llistat es podran consultar les dades de l'historial dels pacients.

9.3.3 Finestra de canvi de certificat d'usuari

Amb l'opció "Canvi de certificat" del menú d'Inici l'usuari indicarà la ruta on es troba el nou certificat i claus que vol utilitzar per l'autenticació. Es pot observar com SWT crida al diàleg d'obertura d'un fitxer del sistema operatiu.

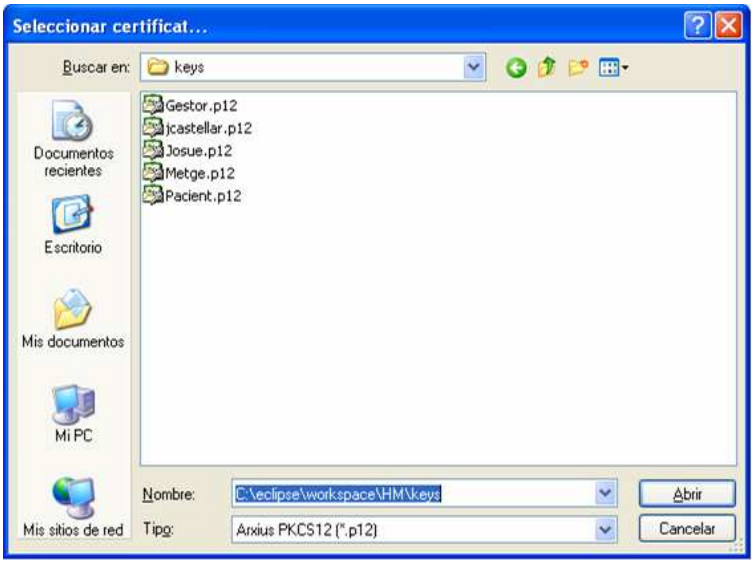

Figura 33. Finestra de selecció de certificat d'usuari

9.3.4 Finestra d'inserció de visita

El metge utilitzarà aquesta finestra per donar de alta una nova visita a l'expedient d'un pacient seleccionat. Tots els camps editables són obligatoris.

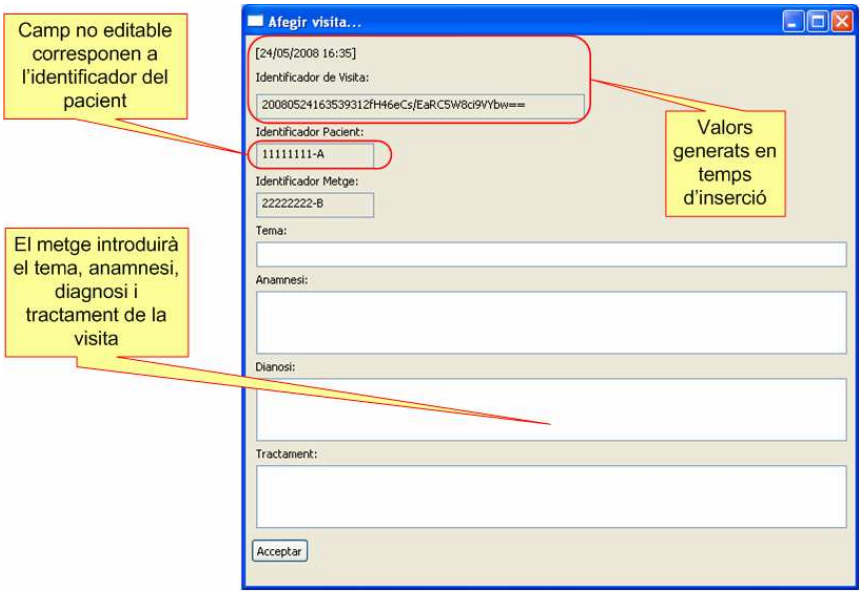

Figura 34. Finestra d'Inserció d'una Visita

10. INTERFÍCIE DEL GESTOR DEL SISTEMA

10.1 Definició

El gestor s'encarrega del repositori d'historials mèdics de forma central amb les funcionalitats bàsiques exposades en el apartat de requeriments d'aquesta memòria.

La seva interfície te per objecte facilitar les tasques administratives per a l'arrencada, parada i configuració d'aquesta aplicació servidor.

S'assoleixen en aquesta fase les següents tasques:

- Automatitzar el procés d'arrencada/parada del servidor
- Parametrització de la configuració del servidor
- Protecció de les contrasenyes del servidor
- Inicialització de dades de proves a la base de dades

Encara que l'últim punt no és part essencial de la interfície s'ha abordat en aquesta fase degut a que es necessària per a l'execució del joc de proves documentat al següent capítol.

10.2 Disseny de la Interfície del Gestor del Sistema

Amb els requisits que s'han comentat, es passa a dissenyar la interfície del gestor del sistema com s'explica a continuació. Aquesta interfície només haurà d'interaccionar un usuari administrador.

10.2.1 Procés de d'arrencada i parada del servidor

Per a dissenyar aquest procés es proporcionarà el *scripts* necessaris per tal de realitzar les següents funcions:

Script d'inici:

1) Generació de la interfície pública del servidor (*Stubs*)
2) Arrencada del servei de publicació d'objectes remots (*rmiregistry*)
3) Arrencada del gestor del sistema amb les opcions disponibles

Script de parada:

1) Parada del gestor del sistema
2) Parada del servei de publicació d'objectes remots

10.2.2 Parametrització de la configuració del servidor

Per tal de facilitar la configuració el servidor utilitzarà un fitxer de configuració on es determinarà la següent informació:

- Ubicació del fitxer amb el certificat del servidor.

- Ubicació del fitxer amb les claus (pública i privada) en format PKCS#12 del servidor.

- Dades per a la connexió a la base de dades: nom del servidor de base de dades, port TCP per a la connexió, nom de la base de dades, usuari de connexió de base de dades i contrasenya d'accés xifrada.

Com es veurà a la sintaxi del procés d'inici, el port per on el servidor escoltarà les peticions dels clients es determina en temps d'arrencada.

10.2.3 Protecció de les contrasenyes del servidor

Les contrasenyes que el servidor ha d'utilitzar per a operar són la *passphrase* que protegeix el contenidor PKCS#12 amb les seves claus i la contrasenya de l'usuari d'accés a la base de dades.

Des del punt de vista de la seguretat és inadmissible que aquestes dades estiguin en un fitxer de configuració degut a que qualsevol usuari amb accés al directori on estigui emmagatzemat al sistema de fitxers del servidor tindria accés a la clau privada. Tampoc és possible indicar aquestes contrasenyes per paràmetre quan s'inicia el procés gestor degut a que aquesta seria visible en un llistat de processos del sistema operatiu.

Per tal de donar la protecció adient a aquestes dades sensibles es determina que la contrasenya del PKCS#12 es demanarà per consola a l'administrador de forma interactiva en el procés d'arrencada.

Per altre banda, la contrasenya de l'usuari d'accés a la base de dades serà xifrada amb la clau pública del gestor i emmagatzemada com a paràmetre al fitxer de configuració. D'aquesta manera, només el gestor podrà tenir accés a aquesta dada una vegada la desxifri amb la seva clau privada.

10.2.4 Inicialització de les dades de prova

Per tal d'afegir alguna dada necessària en el procés de proves es possibilita la inicialització de dades a l'historial. Aquest procés s'ha d'executar únicament quan s'ha creat la base de dades.

10.3 Implementació

La implementació de la interfície es basa fonamentalment en *scripting* i la generació del codi necessari per a tractar les contrasenyes del PKCS#12 i de l'usuari de connexió utilitzat pel gestor.

10.3.1 *Scripts* d'arrencada i parada

La sintaxi del procés d'arrencada és:

```
startHMServer.bat [init]
```

Si el paràmetre opcional `init` és igual a "1" es força la inicialització de les dades de prova.

El contingut d'aquest procés és:

```
rmic rmi.HMServerInterfaceImpl
start rmiregistry
start "HMServer" java rmi.HMServer 1099 %1
```

Com es pot apreciar, l'últim pas és la crida al gestor seguint la sintaxi definida:

```
HMServer port [init]
```

El paràmetre `port` correspon al port TCP a utilitzar per a servir les peticions des de la interfície client i, de nou, `init` indica al servidor que ha d'executar els processos d'inicialització de dades a l'historial.

Per altre banda, la sintaxi del procés de parada és:

```
stopHMServer.bat
```

El contingut és, en aquest cas:

```
taskkill /F /FI "WINDOWTITLE eq HMServer"
taskkill /F /IM rmiregistry.exe
```

10.3.2 Xifrat de la contrasenya de connexió a la base de dades

En aquesta implementació s'ha generat una utilitat per a xifrar amb la clau pública del gestor. El resultat serà mostrat per pantalla i s'usarà per a indicar-ho en el fitxer de configuració del gestor del sistema.

La classe `HidePassword` té la següent sintaxi:

```
HidePassword –p contrasenyaBD –c ubicacioCert
```

On `contrasenyaBD` és la contrasenya de l'usuari de connexió a la base de dades que volem xifrar i `ubicacioCert` correspon la ruta on es troba el certificat del gestor del sistema que s'utilitzarà.

El resultat és una tira de caràcters, en format base64 per a que siguin imprimibles, que correspondran a la contrasenya xifrada. Aquesta sortida és la que s'indicarà al paràmetre dbpwd del fitxer de configuració.

Com a exemple, annexen una execució del xifratge de la contrasenya "`pfcuoc`":

```
# java tools.HidePassword -p pfcuoc -c keys\Gestor.crt
*****************************************************************************
***                                                                       ***
***                    Welcome to the IAIK JCE Library                    ***
***                                                                       ***
*** This version of IAIK-JCE is licensed for evaluation, education,       ***
*** research, and use in open-source projects only.                       ***
*** Commercial use of this software is prohibited.                        ***
*** For details please see http://jce.iaik.tugraz.at/sales/.              ***
*** This message does not appear in the registered commercial version.    ***
***                                                                       ***
*****************************************************************************

MIIBbQIBADGCATkwggE1AgEAMIGdMIGXMQswCQYDVQQGEwJFUzESMBAGA1UECBMJQmFyY2Vsb25hMRIw
EAYDVQQHEwlCYXJjZWxvbmExDDAKBgNVBAoTA1VPQzEWMBQGA1UECxMNUEZDIFNlZ3VyZXRhdDEZMBcG
A1UEAxQQQ0FfUEZDX1NlZ3VyZXRhdDEfMB0GCSqGSIb3DQEJARYQanJvZHJpZ2FAdW9jLmVkdQIBAjAN
BgkqhkiG9w0BAQEFAASBgJmtRFg/ms+2/km75Hv4ag1wg7Q+4rrkbXXcOGBaQ1jeH17fLxO8IKtO4J6y
vSSMlRJuJxAndM+NbSvwn6fyE5lwxlK2trP+HsxseU4HLp09DjHQEKELKHIWcpi2u8Bd7nlfd5q3AxIj
xwOlLXV9bJkYiSiicmS4PQ7Y9zFUQpArMCsGCSqGSIb3DQEHATAUBggqhkiG9w0DBwQIB6JuQDIdWQyA
CKfJAX6rgldW
```

11. JOC DE PROVES

11.1 Definició

El conjunt de proves es poden estructura en dos blocs:

1) Proves unitàries
Proves realitzades de forma incremental conforme es van construint els mòduls de l'aplicació en les diferents fases del PFC. No es realitza cap acció que involucri la revisió de les funcionalitats d'altres mòduls construïts prèviament.

2) Proves d'integració
Conformen les tasques de verificació extrem a extrem.

Aquest capítol es dedica a les proves descrites al segon bloc. L'objectiu de les proves serà la verificació i test funcional dels productes desenvolupats des del punt de vista del client com del servidor.

Per aconseguir aquesta fita es seguirà el següent procediment per cada funcionalitat esperada i descrita a l'apartat de requeriments d'aquest document:

- Descriure la funcionalitat a comprovar.
- Descriure els resultats esperats.
- Especificar el mètode d'execució de la prova (rol utilitzat, estat, opció a escollir, etc.).
- Executar la prova.
- Recollir els resultats.
- Comentar les diferències respecte als resultats esperats, si n'hi ha.
- Adjuntar les evidències.

11.2 Execució

A continuació s'executa el joc de proves definit i es mostren els resultats.

11.2.1 Inici de sessió

Què es vol comprovar?
Funcionalitat d'inici de sessió en l'aplicació d'un usuari.

Quin són els resultats esperats?
El usuari haurà completat l'autenticació utilitzant el seu parell de claus criptogràfiques.

Mètode d'execució:
Utilitzar el menú "Inici", i la opció "Iniciar sessió..." una vegada arrencada l'aplicació.

Resultats obtinguts:
L'usuari s'ha autenticat i iniciat la sessió de forma satisfactòria.

Hi ha diferències respecte al esperat?
No.

Evidències:
Utilitzem el pacient fictici amb identificador "11111111-A" i el certificat adient per a autenticar-se en el sistema. L'usuari escull la opció indicada i es mostra el diàleg e inici de sessió. L'usuari indica el seu identificador i la contrasenya i pressiona el botó "Acceptar".

Figura 35. Joc de Prova 1. Inici de sessió

Es comença el procés d'autenticació. Es rep el missatge amb el resultat de l'inici de sessió. Com es pot veure, ha estat possible autenticar-se amb el gestor.

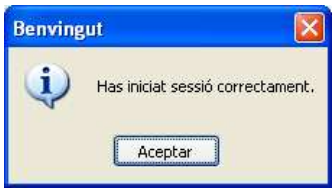

Figura 36. Joc de Prova 1. Missatge de benvinguda

A partir d'ara l'usuari estarà autenticat.

11.2.2 Canvi de certificat d'usuari

Què es vol comprovar?
Funcionalitat de canvi de certificat d'usuari.

Quin són els resultats esperats?
L'usuari podrà indicar la ubicació del seu certificat d'usuari. Es forçarà la autenticació de nou i s'ocultaran les opcions anteriorment disponibles.

Mètode d'execució:
En qualsevol moment, escollir la opció "Canviar certificat..." del menú "Inici".

Resultats obtinguts:
L'usuari canvia la configuració respecte a la ubicació del seu certificat d'usuari.

Hi ha diferències respecte al esperat?
No.

Evidències:
L'usuari escull la opció indicada i el sistema li pregunta per la nova ruta on es troba el seu certificat.

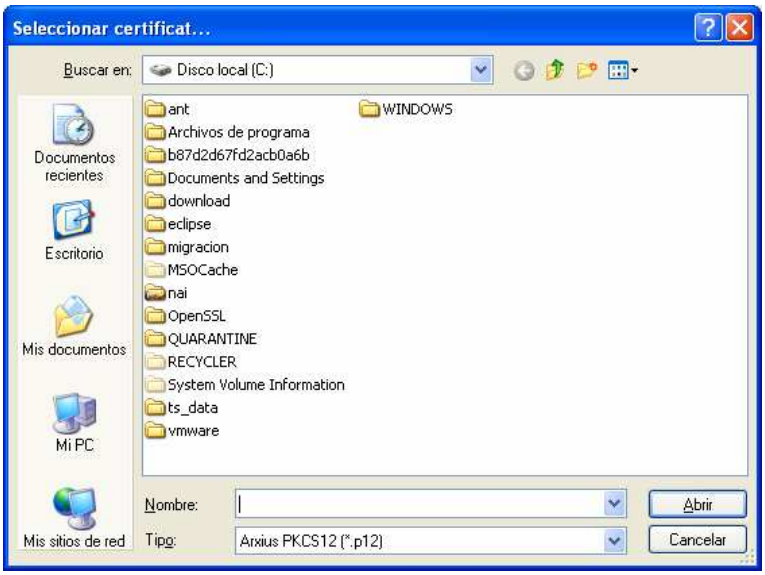

Figura 37. Joc de Prova 2. Canvi de certificat d'usuari

L'usuari indica la ruta completa al diàleg i fa clic a "Abrir". Si no hi ha problemes, es rep un missatge informatiu indicant que s'ha tornar d'iniciar el procés d'autenticació.

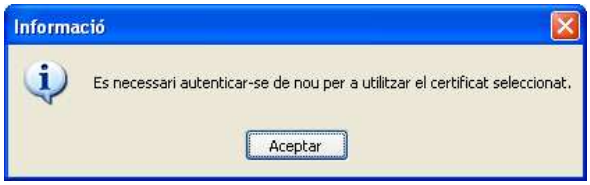

Figura 38. Joc de Prova 2. Missatge de recordatori per a autenticar-se de nou

11.2.3 Sortir de l'aplicació

Què es vol comprovar?
Funció de sortida del programa.

Quin són els resultats esperats?
Es finalitza l'execució de l'aplicació.

Mètode d'execució:
En qualsevol moment, l'usuari escull la opció "Sortir" del menú "Inici".

Resultats obtinguts:
Fi de programa.

Hi ha diferències respecte al esperat?
No.

Evidències:
No aplica.

11.2.4 Pacient: Consulta d'Historial i Llista de Visites

Què es vol comprovar?
Funcionalitat de consulta del historial del propi pacient.

Quin són els resultats esperats?
Te accés a les pestanyes de dades generals de la zona central i contenen les seves dades generals i llistat de visites.

Mètode d'execució:
Després d'iniciar sessió, l'usuari selecciona la opció "Consultar Historial..." del menú "Opcions" i indica el seu identificador.

Resultats obtinguts:
Es mostren les dades generals i llistat de visites de l'usuari pacient.

Hi ha diferències respecte al esperat?
No.

Evidències:
El pacient s'autentica. Després escull la opció adient i indica el seu identificador (usuari de proves: 11111111-A). Fa clic a "Cercar" i s'executa la consulta.

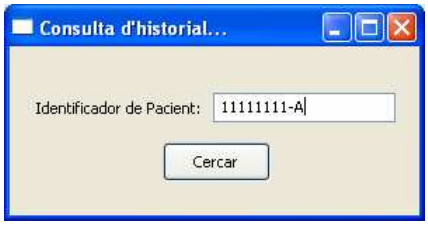

Figura 39. Joc de Prova 4. Consulta d'historial de pacient.

El resultat es troba a les pestanyes de la zona central:

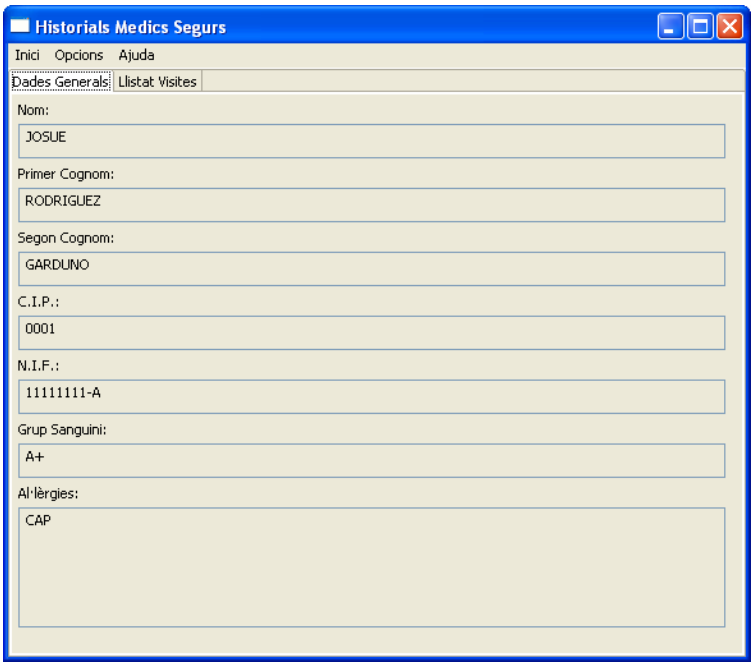

Figura 40. Joc de Prova 4. Dades generals resultat de la consulta

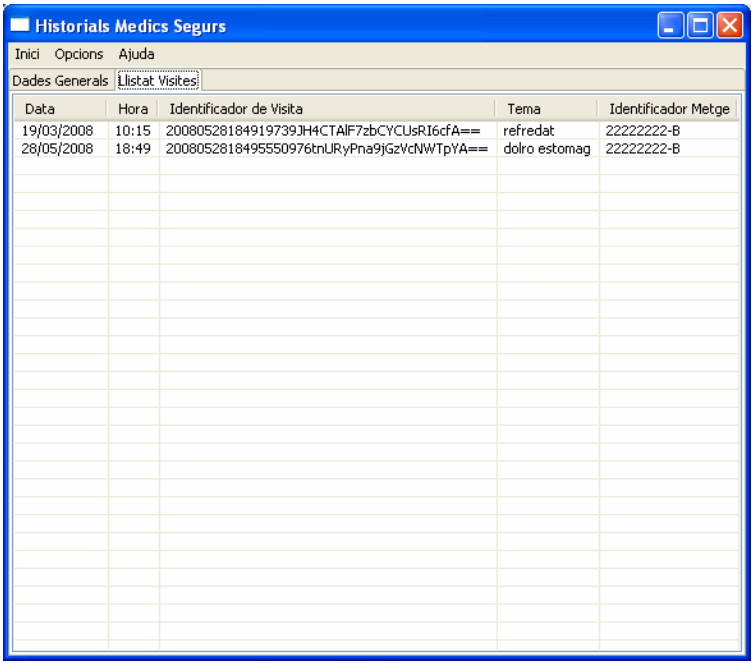

Figura 41. Joc de Prova 4. Llistat de visites resultat de la consulta

Es pot apreciar que es mostra tant les dades generals del pacient com el llistat de la visita.

11.2.5 Pacient: Consulta del detall d'una visita

Què es vol comprovar?
Funcionalitat de visualització del detall de la visita del pacient

Quin són els resultats esperats?
Mostrar les dades de la visita del historial del pacient.

Mètode d'execució:
Una vegada es consulta la llista de visites de l'historial (veure la prova anterior) el pacient selecciona la visita a examinar del llistat i pren la opció "Consultar visita seleccionada" del menú "Opcions".

Resultats obtinguts:
Es crea una nova pestanya a la zona central de la finestra principal amb les dades de la visita.

Hi ha diferències respecte al esperat?
No.

Evidències:
L'usuari de prova selecciona la primera visita del seu historial i escull la opció indicada. El sistema li mostra la següent pantalla:

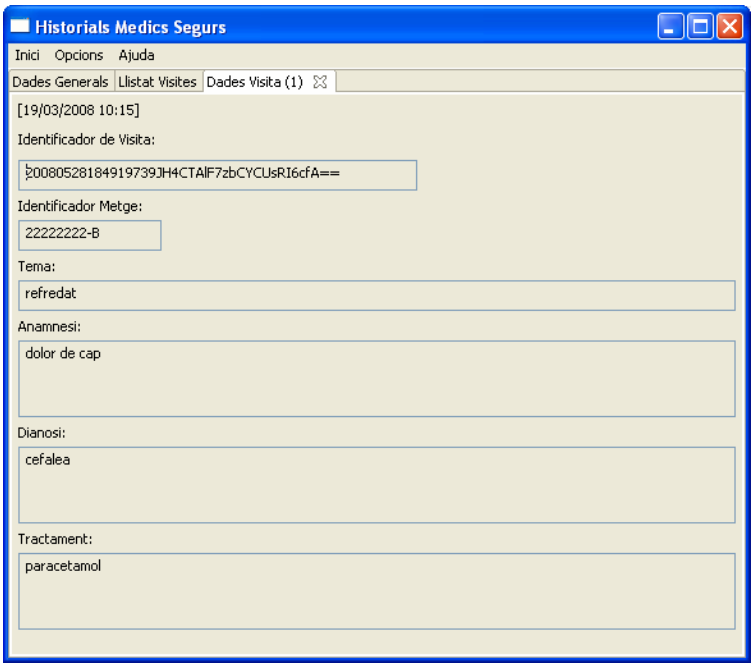

Figura 42. Joc de Prova 5. Consulta de dades de visita

11.2.6 Metge: Consulta del Llistat de Pacients assignats

Què es vol comprovar?
Funcionalitat de llista de pacients assignats a un usuari amb rol metge.

Quin són els resultats esperats?
Obtenir la llista de dades bàsiques (nom complet i NIF) dels pacients que estan assignats a un metge.

Mètode d'execució:
Després de realitzar l'autenticació, l'usuari indica la opció "Consultar pacients assignats" del menú "Opcions".

Resultats obtinguts:
Es visualitza la llista de pacients a una nova pestanya de la zona central.

Hi ha diferències respecte al esperat?
No.

Evidències:
Utilitzant un usuari metge de prova amb identificador 22222222-B es realitza l'autenticació a l'aplicació. Acte seguit es selecciona la opció comentada i s'obté el següent resultat:

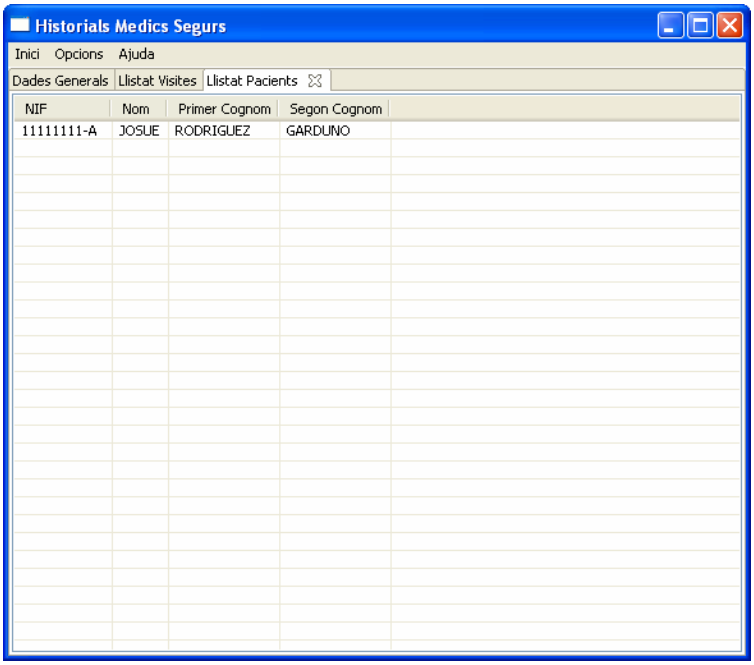

Figura 43. Joc de Prova 6. Consulta de llista de pacients assignats

Com es pot veure, el pacient amb NIF 11111111-A és l'únic pacient assignat al metge.

11.2.7 Metge: Consulta d'Historial i Llista de Visites de pacient assignat

Què es vol comprovar?
Funcionalitat de consulta d'historial d'un pacient assignat a un metge.

Quin són els resultats esperats?
Mostrar les dades de l'expedient i llistat de visites d'un pacient assignat.

Mètode d'execució:
Una vegada autenticat, es pot ver per dos vies:

1) Per mitjà de la opció "Consultar historial..." del menú "Opcions" i indicant el número d'identificador del pacient

2) Realitzar la consulta de la llista de pacients, seleccionar el pacient a visualitzar i escollir la operació “Consultar historial de pacient seleccionat” del menú “Opcions”

Resultats obtinguts:
Es mostra a les pestanyes “Dades Generals” i “Llistat Visites” les dades relatives al pacient consultat.

Hi ha diferències respecte al esperat?
No.

Evidències:
S’adjunta les evidències realitzant aquesta prova amb el segon mètode. Una vegada autenticat el metge de prova (N.I.F. 22222222-B) i seleccionat el seu únic pacient, s’escull la opció indicada.

A la zona central, pestanya “Dades Generals”, es pot visualitzar les dades del pacient.

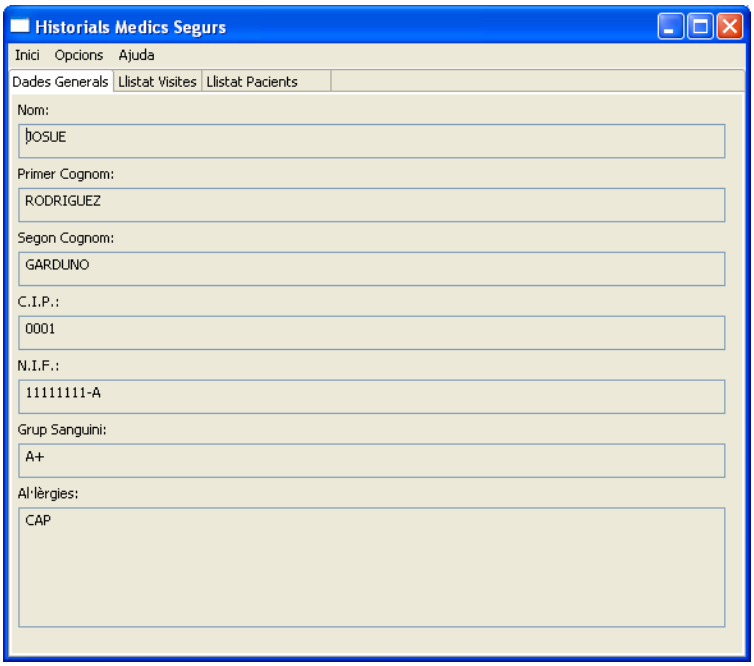

Figura 44. Joc de Prova 7. Consulta de l’historial d’un pacient assignat

11.2.8 Metge: Consulta de detall d’una visita de pacient assignat

Què es vol comprovar?
Funcionalitat de consulta de detall de visita d’un pacient assignat.

Quin són els resultats esperats?
Visualitzar les dades de la visita del pacient assignat a un metge.

Mètode d’execució:
Una vegada obtingut l’historial del pacient (veure joc de prova anterior), es selecciona la visita de la llista de visites i es pren la opció “Consultar visita seleccionada” en el menú “Opcions”.

Resultats obtinguts:
Es mostra el detall de la visita seleccionada.

Hi ha diferències respecte al esperat?
No.

Evidències:
Realitzem les operacions indicades a l'apartat 'Mètode d'execució' d'aquesta prova i obtenim les dades de la visita (en aquesta ocasió escollim la segona):

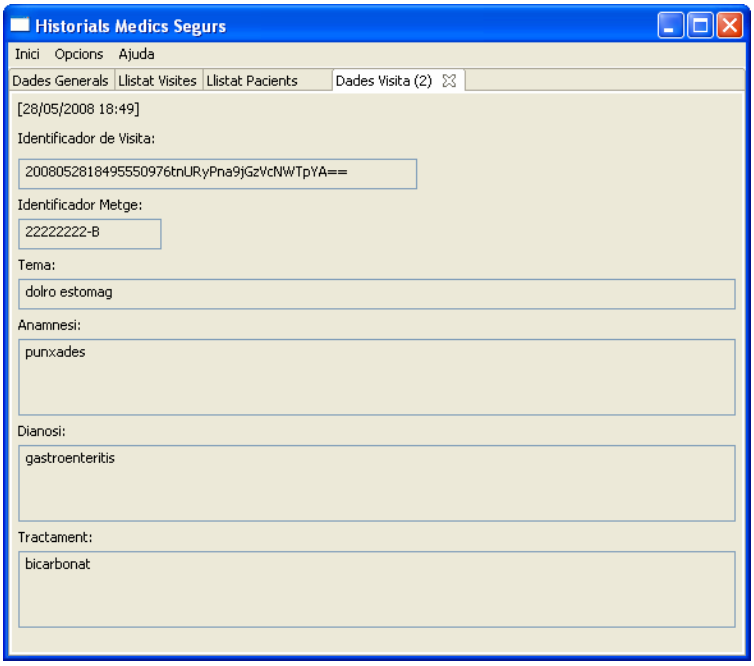

Figura 45. Joc de Prova 8. Consulta de les dades d'una visita d'un pacient assignat

11.2.9 Metge: Inserció d'una visita a l'historial d'un pacient

Què es vol comprovar?
Funcionalitat d'afegir una visita a l'historial d'un pacient assignat.

Quin són els resultats esperats?
S'afegeix les dades d'una visita al historial del pacient.

Mètode d'execució:
Una vegada consultat la llista de pacients, el metge selecciona el pacient que vol afegir-li la visita i escull la operació "Afegir visita..." del menú "Opcions". Emplena les dades adients i fa clic a "Acceptar".

Resultats obtinguts:
S'afegeix a l'historial les dades de la visita realitzada.

Hi ha diferències respecte al esperat?
No.

Evidències:
Utilitzant el metge de proves amb identificador 22222222-B, es fa un llistat dels pacients assignats.

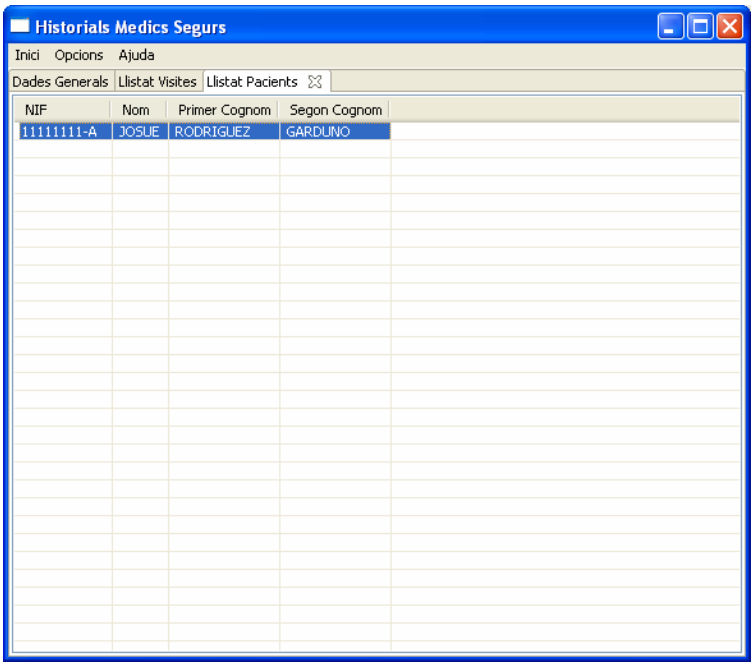

Figura 46. Joc de Prova 9. Selecció de pacient per a inserció de visita a l'historial

Després, es selecciona el pacient desitjat i es pren a l'opció indicada. A continuació el sistema ens mostra un diàleg per a introduir les dades de la visita.

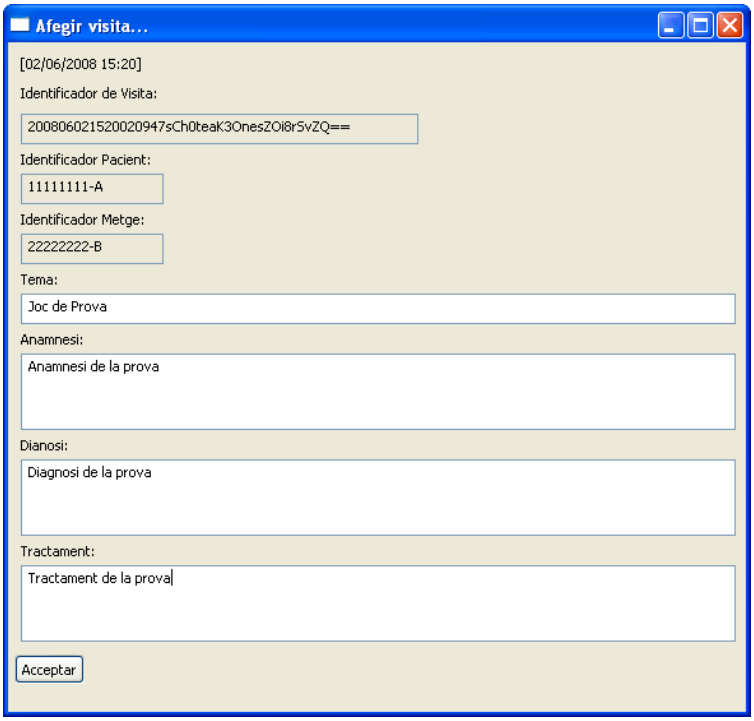

Figura 47. Joc de Prova 9. Inserció de dades d'una nova visita a l'historial

El metge la emplena i prem el botó “Acceptar”. Es rep un missatge conforme la operació ha estat un èxit.

Figura 48. Joc de Prova 9. Missatge de confirmació de la inserció

Si es fa una consulta a l’historial del pacient es pot veure que la visita ha estat afegida.

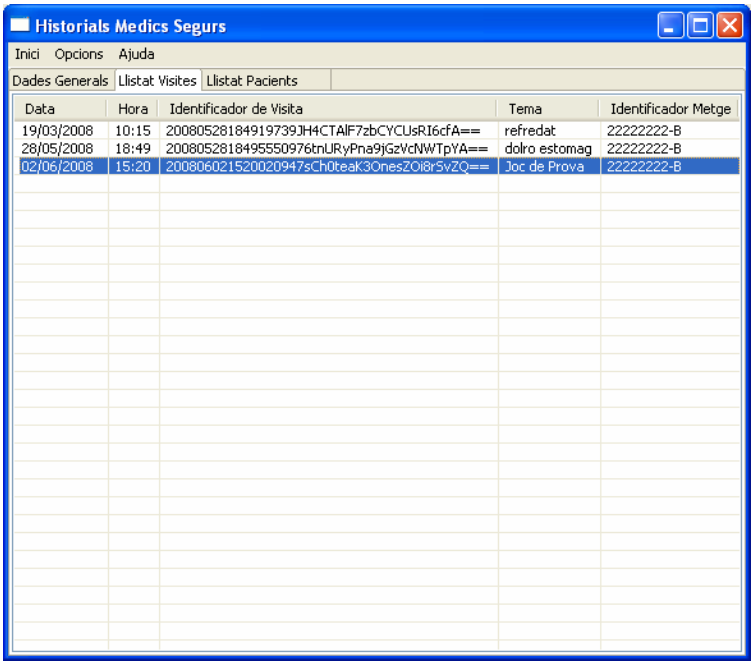

Figura 49. Joc de Prova 9. Verificació de la inserció de la nova visita

12. CONCLUSIONS

Si fem una avaluació dels requisits plantejats en el llançament del PFC i els resultats obtinguts es pot verificat que s'han assolit els objectius de forma satisfactòria en la planificació proposada.

La fita principal ha estat el disseny i implementació d'un esquema criptogràfic per a la gestió i accés remot segur dels historials mèdics de pacients garantint les propietats de seguretat d'autenticació, confidencialitat, integritat i no repudi. Degut a la naturalesa de les dades era imperatiu salvaguardar la confidencialitat.

Des d'un punt de vista funcional, els protocols i procediments que componen l'esquema criptogràfic plantejat permet complir els requeriments:

- Autenticació d'usuaris en base a certificats digitals
- Consultes a l'historial en funció del rol d'usuari
- Modificació de l'historial en funció de l'assignació entre metge i pacient
- Eliminació de dades

Tot aquestes funcionalitats giren en torn a un gestor central que s'encarrega de securitzar totes les operacions.

Altres aspectes que s'han assolit ha estat la implementació d'una capa de representació de dades en XML que ens dona cert grau de flexibilitat i es demostra com aprofitar programari ja desenvolupat com són les llibreries per a gestionar i tractar aquest tipus de document.

El sistema permet la connexió remota entre els components principals i ha estat possible gràcies a la utilització de RMI com a tecnologia incorporada en el llenguatge de programació escollit: Java.

La interfície gràfica, encara que aquest objectiu es podria considerar com a secundària en el conjunt de fites del PFC, ha estat implementat amb llibreries públiques SWT i s'ha dissenyat per fer-la el més fàcil, integrada i amigable possible.

Finalment, la capa de persistència ha estat implementada sobre un model de dades relacional utilitzant un SGBD de llibre distribució i d'àmplia difusió com és MySQL.

Per altre banda, i degut que la planificació ha estat limitada pel calendari acadèmic, es podrien incorporar certes millores que es podrien plantejar com a futures línies de treball:

Llibreries criptogràfiques. Al projecte s'ha emprat les llibreries no comercials per entorns acadèmics IAIK. Si aquest producte es volgués comercialitzar es podria utilitzar les llibreries criptogràfiques incorporades al JDK de Sun.

Xifrat de comunicacions gestor/base de dades. La comunicació entre el gestor del sistema i la base de dades és en clar i, per això, un administrador podria escoltar la conversa capturant els paquets de xarxa. Com a solució caldria implementar el xifratge d'aquestes comunicacions.

Funcionalitat de registre automàtic. L'alta de nous usuaris és manual. Es podria desenvolupar una interfície per al registre automàtic de pacients i metges. Això implicaria la generació del programari necessari per a generar certificats de forma automàtica i incorporar les dades a la base de dades. Caldria estudiar com totes aquestes operacions es fan de forma segura.

Gestió d'assignacions de metges. També seria interessant disposar de la funcionalitat per a la gestió de les llistes de pacients assignats als metges del sistema.

Targeta Intel·ligent com a contenidor del PKCS#12. Actualment els usuaris tenen el fitxer amb el parell de claus criptogràfiques a disc. Una millora que aportaria un nivell elevat de seguretat i donaria cert grau de mobilitat seria la utilització de targetes intel·ligents per emmagatzemar els PKCS#12 utilitzats en l'autenticació davant el gestor. Aquest canvi implicaria modificar la interfície client per a poder accedir al contenidor criptogràfic de la *smartcard*.

Internacionalització de la interfície gràfica. Traduir la interfície gràfica a diferents idiomes i fer configurable el llenguatge per part de l'usuari.

A nivell personal aquest treball ha estat molt enriquidor professionalment. En primer lloc he pogut participar de primera mà en totes les fases d'un projecte informàtic, des de l'especificació formal dels requeriments, el disseny de l'arquitectura, la construcció i implementació i l'execució de les proves. Per altre costat, he consolidat i posat en pràctica els coneixements sobre seguretat informàtica i criptografia que havia adquirit en el decurs dels estudis. I, finalment, he vist directament com integrar moltes tecnologies per separat, algunes desconegudes per mi, per a convertir-les en un producte sòlid.

13. GLOSSARI

AES: *Advanced Encryption Standard*. Esquema de xifrat per bloc adoptat como un estàndard de xifrat pel govern dels Estats Units.

API: *Application Programming Interface*. Conjunt de funcions, mètodes i procediments que s'ofereix com a llibreria per a ser utilitzat per altre programari com a capa d'abstracció.

Base64: Sistema de numeració posicional que utilitza 64 com a base.

CA: *Certification Authority*. Autoritat certificadora encarregada de la gestió i generació de certificats a una PKI.

CBC: *Cipher-block chaining*. Mode d'operació d'una xifra de bloc.

CIP: Codi Identificació del Pacient.

CSR: *Certificate Signing Request*.

DER: *Distinguished Encoding Rules*.

DNI: Document Nacional d'Identitat.

DTD: *Document Type Definition*. Descripció d'estructura i sintaxi d'un document XML.

FIPS: *Federal Information Processing Standards*. Estàndard públic generat pel govern dels Estats Units per a l'ús d'agències governamentals no militars.

GPL: *Gnu Public License*. Llicència de programari lliure.

HSM: *Hardware Security Module*. Dispositiu criptogràfic utilitzat per la salvaguarda de claus i certificats.

IAIK: Llibreria criptogràfica per Java.

IDE: *Integrated Development Environment*. Programari que proveeix d'utilitats als programadors per al desenvolupament.

IBM: *International Business Machines*.

Java: Llenguatge de programació multiplataforma, robust, interpretat, distribuït, orientat a objectes, portable, desenvolupat per Sun Microsystems.

JCE: *Java Cryptography Extension*. API Java que implementa mecanismes criptogràfics.

JDBC: *Java Database Connectivity*. API Java per a la comunicació amb una base de dades relacional.

JDK: *Java Development Kit*. Conjunt d'utilitats creat per Sun Microsystems per al desenvolupament d'aplicacions Java.

JDOM: *Java Document Object Model*. Model d'objectes basat en Java pel tractament de documents XML.

Middleware: Programari que s'utilitza per a connectar diferents components o aplicacions.

MVC: Model Vista Controlador. Patró de programació.

MySQL: Gestor de bases de dades relacionals SQL de llibre distribució.

NIF: Número d'Identificació Fiscal.

OLTP: *Online Transaction Processing*. Tipus de sistema dedicat a la gestió d'aplicacions orientades a transaccions, típicament per a l'entrada o consulta de dades.

OpenSSL: Programari de lliure distribució per gestionar certificats criptogràfics.

Padding: Farciment.

Passphrase: Paraula de pas, contrasenya.

PEM: *Privacy Enhanced Mail*.

PFC: Projecte de Final de Carrera.

PKCS: *Public-Key Cryptography Standards*. Conjunt d'estàndards definits pels laboratoris RSA que especifiquen els estàndards de clau pública.

PKI: *Public Key Infraestructure*. Estructura composta per programari, maquinari i procediments que defineix la gestió de les claus i certificats en un entorn de criptografia pública.

RMI: *Remote Method Invocation*. Mecanisme remot de crida a objectes en Java.

RA: *Registration Authority*. Autoritat de Registre inclosa en una PKI encarregada principalment del registre de peticions de certificació.

RFC: *Request For Comments*. Memoràndums sobre noves investigacions, innovacions i metodologies relacionades amb les tecnologies d'Internet.

RSA: Algorisme de clau pública.

Script: Conjunt d'instruccions procedimentals.

SGBD: Sistema de Gestió de Base de Dades.

SHA: Funció de resum.

Smartcard: Targeta intel·ligent. Normalment utilitzada per l'emmagatzemament segur de les claus criptogràfiques.

SQL: *Structured Query Language*. Llenguatge de programació estàndard per a l'accés d'una base de dades relacional.

SWT: *Standard Widget Toolkit*. Conjunt de programari per a la generació d'aplicacions gràfiques per Java.

TCP: *Transport Connection Protocol*. Protocol de transport orientat a connexió.

Timestamp: Marca de temps.

UML: *Unified Model Language*. Metodologia de disseny d'aplicacions informàtiques utilitzat per a especificar, visualitzar, construir i documentar un sistema orientat a objectes.

X.509: Estàndard per a un certificat generat a una PKI.

XML: *Extensible Markup Language*. Llenguatge de representació de dades.

14. BIBLIOGRAFIA

[ASD]
Universitat Oberta de Catalunya (2007). *Apunts de l'assignatura de Arquitectura de Sistemes Distribuïts*. http://www.uoc.edu

[BASE64]
The Internet Society (2006). *The Base16, Base32, and Base64 Data Encodings*. RFC 4648. http://tools.ietf.org/html/rfc4648

[CONS78]
(1978) *Constitució Espanyola de 27 de desembre de 1978*. https://www.agpd.es/upload/Canal_Documentacion/legislacion/Estatal/Ley%2015_99.pdf

[CRIPTO]
Universitat Oberta de Catalunya (2007). *Apunts de l'assignatura de Criptografia*. http://www.uoc.edu

[DERFMT]
International Telecommunication Union (ITU) (2002) *Information technology – ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)*. http://www.itu.int/ITU-T/studygroups/com17/languages/X.690-0207.pdf

[ECL332]
Eclipse Foundation. *Eclipse IDE*. http://www.eclipse.org/

[FIPS1402]
National Institute of Standards and Technology (NIST) (2007). *FIPS 140-2, Security Requirements for Cryptographic Modules*. http://csrc.nist.gov/groups/STM/index.html

[IAIK316]
Stiftung Secure Information and Communication Technologies SIC. *IAIK Core Crypto Toolkits*. http://jce.iaik.tugraz.at/sic/products/core_crypto_toolkits

[JAVAW2000]
Java World (2000). *Build distributed applications with Java and XML*. Network World Inc. http://www.javaworld.com/javaworld/jw-02-2000/jw-02-ssj-xml.html

[JDOM11]
Jason Hunter. *Java Document Object Model*. http://www.jdom.org/index.html

[LOG4J]
Apache Software Foundation (2008). *Logging Services*. http://logging.apache.org/log4j/1.2/index.html

[LOPD99]
(1999) *Llei Orgànica 15/1999, de 13 de desembre, de Protecció de Dades de Caràcter Personal*. https://www.agpd.es/upload/Canal_Documentacion/legislacion/Estatal/Ley%2015_99.pdf

[MYSQL5]
Sun Microsystems. *MySQL Community Server*. http://dev.mysql.com/downloads/mysql/5.0.html

[NESC78]
Roger Needham i Michael Schroeder (1978). *Needham-Schroeder Public Key*. http://www.lsv.ens-cachan.fr/spore/nspk.html

[OPSSL]
The OpenSSL Project. *OpenSSL toolkit*. http://www.openssl.org/

[PEMFMT]
Internet Engineering Task Force (IETF) (1993). *Privacy Enhancement for Internet Electronic Mail.* RFC 1421, 1422, 1423, 1424. http://tools.ietf.org/html/

[PKCS12]
RSA Laboratories (1999). *Personal Information Exchange Syntax Standard, version 1.0.* http://www.rsa.com/rsalabs/node.asp?id=2138

[RDDL07]
(2007) *Reial Decret 1720/2007 de Desenvolupament de la Llei Orgànica 15/1999, de 13 de desembre, de protecció de dades de caràcter personal*. https://www.agpd.es/upload/Canal_Documentacion/legislacion/Estatal/RD_1720_2007.pdf

[RDMS99]
(1999) *Reial Decret 994/1999, de 11 de juny, de Mesures de Seguretat dels fitxers automatitzats que continguin dades de caràcter personal*. https://www.agpd.es/upload/Canal_Documentacion/legislacion/Estatal/A.8%29%20Real%20Decreto%20994-1999.pdf

[RFC1750]
Internet Engineering Task Force (IETF) (1994). *Randomness Recommendations for Security*. http://www.ietf.org/rfc/rfc1750.txt

[RMI16]
Sun Microsystems. *Java Remote Method Invocation and J2SE 6.0.* http://java.sun.com/javase/6/docs/technotes/guides/rmi/index.html

[RSA77]
Ron Rivest, Adi Shamir i Leonard Adleman (1977). *RSA Cryptography Standard*. Massachusetts Institute of Technology. http://www.rsa.com/rsalabs/node.asp?id=2125

[SDK16]
Sun Microsystems. *Java Platform – Standard Edition Development Kit*. http://java.sun.com/javase/downloads/index.jsp

[SHA101]
National Security Agency (2001). Secure Hash Algorithm 1. RFC 3174. http://tools.ietf.org/html/rfc3174

[SWT332]
Eclipse Foundation (2008). *SWT: The Standard Widget Toolkit*. http://www.eclipse.org/swt/

[X509]
International Telecommunication Union (ITU) (1988). *X.509 : Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks*. http://www.itu.int/rec/T-REC-X.509/en

15. ANNEXOS

A. NOTACIÓ

A la descripció dels protocols de l'esquema criptogràfic s'empra la següent notació:

- *K:* clau d'un criptosistema simètric.
- $E_K(M)$*:* xifratge simètric d'un missatge *M* amb la clau *K*.
- $D_K(C)$*:* desxifratge simètric del criptograma C amb la clau *K*.
- $(P_{Entitat}, S_{Entitat})$*:* parella de claus asimètriques propietat d'Entitat, on *P* correspon a la clau pública, i *S* a la privada.
- $S_{Entitat}[M]$*:* Signatura digital del missatge *M* amb la clau privada *S* d'Entitat.
- $P_{Entitat}[M]$*:* Xifratge del missatge *M* amb la clau asimètrica pública $P_{Entitat}$ d'Entitat.
- *H(M):* sortida d'una funció resum criptogràfica del missatge *M*, aquestes funcions reben el nom de funcions *hash*.

B. FITXER DE CONFIGURACIÓ DE LA PKI

A continuació s'annexa el contingut del fitxer de configuració de la PKI creada per la gestió de certificats del sistema, `openssl.cnf`:

```
#
# OpenSSL example configuration file.
# This is mostly being used for generation of certificate requests.
#

# This definition stops the following lines choking if HOME isn't
# defined.
HOME                    = .
RANDFILE                = $ENV::HOME/.rnd

# Extra OBJECT IDENTIFIER info:
#oid_file               = $ENV::HOME/.oid
oid_section             = new_oids

# To use this configuration file with the "-extfile" option of the
# "openssl x509" utility, name here the section containing the
# X.509v3 extensions to use:
# extensions            =
# (Alternatively, use a configuration file that has only
# X.509v3 extensions in its main [= default] section.)

[ new_oids ]

# We can add new OIDs in here for use by 'ca' and 'req'.
# Add a simple OID like this:
# testoid1=1.2.3.4
# Or use config file substitution like this:
# testoid2=${testoid1}.5.6

####################################################################
[ ca ]
default_ca     = CA_default          # The default ca section

####################################################################
[ CA_default ]

dir            = ./CAPFC             # Where everything is kept
certs          = $dir/certs          # Where the issued certs are kept
crl_dir        = $dir/crl            # Where the issued crl are kept
database       = $dir/index.txt      # database index file.
new_certs_dir  = $dir/newcerts               # default place for new certs.

certificate    = $dir/CA.crt # The CA certificate
serial         = $dir/serial         # The current serial number
crl            = $dir/crl.pem               # The current CRL
private_key    = $dir/private/CA.key # The private key
RANDFILE       = $dir/private/.rand  # private random number file

x509_extensions       = usr_cert            # The extentions to add to the cert

# Extensions to add to a CRL. Note: Netscape communicator chokes on V2 CRLs
# so this is commented out by default to leave a V1 CRL.
# crl_extensions      = crl_ext

default_days   = 365                 # how long to certify for
default_crl_days= 30                 # how long before next CRL
default_md      = sha1                   # which md to use.
preserve       = no                  # keep passed DN ordering

# A few difference way of specifying how similar the request should look
# For type CA, the listed attributes must be the same, and the optional
# and supplied fields are just that :-)
policy         = policy_match

# For the CA policy
[ policy_match ]
```

```
countryName           = match
stateOrProvinceName   = optional
organizationName      = match
organizationalUnitName= optional
commonName            = supplied
emailAddress          = optional
dnQualifier           = optional

# For the 'anything' policy
# At this point in time, you must list all acceptable 'object'
# types.
[ policy_anything ]
countryName           = optional
stateOrProvinceName   = optional
localityName          = optional
organizationName      = optional
organizationalUnitName= optional
commonName            = supplied
emailAddress          = optional
dnQualifier           = optional

####################################################################
[ req ]
default_bits          = 1024
default_keyfile       = privkey.pem
distinguished_name    = req_distinguished_name
attributes            = req_attributes
x509_extensions       = v3_ca # The extentions to add to the self signed cert

# Passwords for private keys if not present they will be prompted for
# input_password = secret
# output_password = secret

# This sets a mask for permitted string types. There are several options.
# default: PrintableString, T61String, BMPString.
# pkix   : PrintableString, BMPString.
# utf8only: only UTF8Strings.
# nombstr : PrintableString, T61String (no BMPStrings or UTF8Strings).
# MASK:XXXX a literal mask value.
# WARNING: current versions of Netscape crash on BMPStrings or UTF8Strings
# so use this option with caution!
string_mask = nombstr

# req_extensions = v3_req # The extensions to add to a certificate request

[ req_distinguished_name ]
countryName                     = Country Name (2 letter code)
countryName_default               = ES
countryName_min                       = 2
countryName_max                       = 2

stateOrProvinceName             = State or Province Name (full name)
stateOrProvinceName_default     = Catalunya

localityName                    = Locality Name (eg, city)
localityName_default            = Barcelona

0.organizationName              = Organization Name (eg, company)
0.organizationName_default      = Universitat Oberta de Catalunya

# we can do this but it is not needed normally :-)
#1.organizationName             = Second Organization Name (eg, company)
#1.organizationName_default     = World Wide Web Pty Ltd

organizationalUnitName          = Organizational Unit Name (eg, section)
organizationalUnitName_default        = Consultors

commonName                      = Common Name (eg, YOUR name)
commonName_max                  = 64

emailAddress                    = Email Address
```

```
emailAddress_max             = 40

dnQualifier                  = D.N.I or N.S.S.
dnQualifier_default          = 00000000-A

# SET-ex3                    = SET extension number 3

[ req_attributes ]
challengePassword            = A challenge password
challengePassword_min        = 4
challengePassword_max        = 20

unstructuredName             = An optional company name

[ usr_cert ]

# These extensions are added when 'ca' signs a request.

# This goes against PKIX guidelines but some CAs do it and some software
# requires this to avoid interpreting an end user certificate as a CA.

basicConstraints=CA:FALSE

# Here are some examples of the usage of nsCertType. If it is omitted
# the certificate can be used for anything *except* object signing.

# This is OK for an SSL server.
# nsCertType                 = server

# For an object signing certificate this would be used.
# nsCertType = objsign

# For normal client use this is typical
nsCertType = client, email

# and for everything including object signing:
# nsCertType = client, email, objsign

# This is typical in keyUsage for a client certificate.
keyUsage = nonRepudiation, digitalSignature, keyEncipherment

# This will be displayed in Netscape's comment listbox.
nsComment                    = "Seguretat en Xarxes de Computadors"

# PKIX recommendations harmless if included in all certificates.
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always

# This stuff is for subjectAltName and issuerAltname.
# Import the email address.
subjectAltName=email:copy

# Copy subject details
issuerAltName=issuer:copy

#nsCaRevocationUrl           = http://www.domain.dom/ca-crl.pem
#nsBaseUrl
#nsRevocationUrl
#nsRenewalUrl
#nsCaPolicyUrl
#nsSslServerName

[ v3_req ]

# Extensions to add to a certificate request

basicConstraints = CA:FALSE
keyUsage = nonRepudiation, digitalSignature, keyEncipherment

[ v3_ca ]
```

```
# Extensions for a typical CA


# PKIX recommendation.

subjectKeyIdentifier=hash

authorityKeyIdentifier=keyid:always,issuer:always

# This is what PKIX recommends but some broken software chokes on critical
# extensions.
#basicConstraints = critical,CA:true
# So we do this instead.
basicConstraints = CA:true

# Key usage: this is typical for a CA certificate. However since it will
# prevent it being used as an test self-signed certificate it is best
# left out by default.
# keyUsage = cRLSign, keyCertSign

# Some might want this also
# nsCertType = sslCA, emailCA

# Include email address in subject alt name: another PKIX recommendation
# subjectAltName=email:copy
# Copy issuer details
# issuerAltName=issuer:copy

# DER hex encoding of an extension: beware experts only!
# obj=DER:02:03
# Where 'obj' is a standard or added object
# You can even override a supported extension:
# basicConstraints= critical, DER:30:03:01:01:FF

[ crl_ext ]

# CRL extensions.
# Only issuerAltName and authorityKeyIdentifier make any sense in a CRL.

# issuerAltName=issuer:copy
authorityKeyIdentifier=keyid:always,issuer:always
```

C. DTD's PER LA VALIDACIÓ DE DOCUMENTS XML

S'inclouen en aquest apartat els documents de definició de tipus per a la validació dels documents XML emprats a la representació de les dades.

DTD HISTORIAL

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT Historial (DadesGenerals,LlistaVisitesProtegida,LlistaMetgesProtegida)>
<!ELEMENT DadesGenerals (nom,cognom1,cognom2,cip,dni,grupSanguini,alergies,certificat)>
<!ELEMENT nom (#PCDATA)>
<!ELEMENT cognom1 (#PCDATA)>
<!ELEMENT cognom2 (#PCDATA)>
<!ELEMENT cip (#PCDATA)>
<!ELEMENT dni (#PCDATA)>
<!ELEMENT grupSanguini (#PCDATA)>
<!ELEMENT alergies (#PCDATA)>
<!ELEMENT certificat (#PCDATA)>
<!ELEMENT LlistaVisitesProtegida (LlistaDescriptors,LlistaAccess)>
<!ELEMENT LlistaDescriptors (#PCDATA)>
<!ELEMENT LlistaAccess (#PCDATA)>
<!ELEMENT LlistaMetgesProtegida (#PCDATA)>
```

DTD VISITA

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT Visita (DescriptorVisita,DadesVisita,SignaturaVisita)>
<!ELEMENT DescriptorVisita (idVisita,any,mes,dia,hora,minut,tema,idMetge)>
<!ELEMENT idVisita (#PCDATA)>
<!ELEMENT any (#PCDATA)>
<!ELEMENT mes (#PCDATA)>
<!ELEMENT dia (#PCDATA)>
<!ELEMENT hora (#PCDATA)>
<!ELEMENT minut (#PCDATA)>
<!ELEMENT tema (#PCDATA)>
<!ELEMENT idMetge (#PCDATA)>
<!ELEMENT DadesVisita (anamnesi,diagnosi,tractament)>
<!ELEMENT anamnesi (#PCDATA)>
<!ELEMENT diagnosi (#PCDATA)>
<!ELEMENT tractament (#PCDATA)>
<!ELEMENT SignaturaVisita (#PCDATA)>
```

DTD METGE

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT Metge (nom,cognom1,cognom2,colegiat,dni,especialitat,certificat,LlistaPacients)>
<!ELEMENT nom (#PCDATA)>
<!ELEMENT cognom1 (#PCDATA)>
<!ELEMENT cognom2 (#PCDATA)>
<!ELEMENT colegiat (#PCDATA)>
<!ELEMENT dni (#PCDATA)>
<!ELEMENT especialitat (#PCDATA)>
<!ELEMENT certificat (#PCDATA)>
<!ELEMENT LlistaPacients (#PCDATA)>
```

DTD SERIALIZEDLIST

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT SerializedList (elementList)+>
<!ELEMENT elementList (#PCDATA)>
```

D. SCRIPT DE CREACIÓ I CONFIGURACIÓ DE LA BASE DE DADES

S'adjunta el *script* de comandes SQL per a la:

- Creació de la base de dades
- Creació del model de dades
- Creació de l'usuari de connexió
- Assignació de permisos adients a l'usuari de connexió

```
create database hm;

use hm;

create table usuari (
  u_idUser VARCHAR(10) NOT NULL
)type=InnoDB;
alter table usuari add constraint pk_usuari primary key (u_idUser);

create table pacient (
  p_idPacient VARCHAR(10) NOT NULL,
  p_certificat mediumblob,
  p_XMLHistorial LONGTEXT
)type=InnoDB;
alter table pacient add constraint pk_pacient primary key (p_idPacient);
alter table pacient add constraint fk_sub_usuari_pacient
    foreign key(p_idPacient) references usuari(u_idUser)
    on delete no action;

create table metge (
  m_idMetge VARCHAR(10) NOT NULL,
  m_certificat mediumblob,
  m_XMLMetge LONGTEXT
)type=InnoDB;
alter table metge add constraint pk_metge primary key (m_idMetge);
alter table metge add constraint fk_sub_usuari_metge
    foreign key(m_idMetge) references usuari(u_idUser)
    on delete no action;

create table visita (
  v_idVisita INTEGER AUTO_INCREMENT NOT NULL,
  v_idDescriptorVisita LONGTEXT NOT NULL,
  v_XMLVisita LONGTEXT,
  PRIMARY KEY(v_idVisita)
)type=InnoDB;

create table challenge (
  c_idChallenge INTEGER AUTO_INCREMENT NOT NULL,
  c_idUser VARCHAR(10) NOT NULL,
  Ni LONGBLOB NOT NULL,
  Ng LONGBLOB NOT NULL,
  PRIMARY KEY(c_idChallenge)
)type=InnoDB;
alter table challenge add constraint fk_challenge
    foreign key(c_idUser) references usuari(u_idUser)
    on delete cascade;

create user 'hmuser'@'localhost' identified by 'orivi65';

grant select, insert, delete, update on hm.* to 'hmuser'@'localhost';

flush privileges;
```

E. FITXERS DE CONFIGURACIÓ

A continuació s'inclouen els fitxers de configuració de la interfície client i servidor, respectivament.

`HMClient.properties`:

```
####################################
#  SERVER CONNECTION CONFIGURATION #
####################################
# Listen port
servername = localhost
port = 1099

####################################
#         KEYS CONFIGURATION       #
####################################
# Server certificate file
servercert = keys/Gestor.crt

# Personal pkcs12
filep12 = keys/Metge.p12
```

- `servername`: Nom del servidor on s'executa el gestor del sistema
- `port`: Port de connexió al gestor
- `servercert`: Ubicació del certificat públic del gestor
- `filep12`: Ubicació del fitxer amb el parell de claus del client (PKCS#12)

`HMServer.properties`:

```
####################################
#         KEYS CONFIGURATION       #
####################################
# Server certificate file
servercert = keys/Gestor.crt

# Server pkcs12
filep12 = keys/Gestor.p12

####################################
#      DATABASE CONFIGURATION      #
####################################
dbhost = localhost
dbport = 3306
dbname = hm
dbuser = hmuser
dbpwd =
MIIBbQIBADGCATkwggE1AgEAMIGdMIGXMQswCQYDVQQGEwJFUzESMBAGA1UECBMJQmFyY2Vsb
25hMRIwEAYDVQQHEwlCYXJjZWxvbmExDDAKBgNVBAoTA1VPQzEWMBQGA1UECxMNUEZDIFNlZ3
VyZXRhdDEZMBcGA1UEAxQQQ0FfUEZDX1NlZ3VyZXRhdDEfMB0GCSqGSIb3DQEJARYQanJvZHJ
pZ2FAdW9jLmVkdQIBAjANBgkqhkiG9w0BAQEFAASBgBkwbGYerayzw7pDpuu+7rt0r6k5Bvey
TnlqbZ0A067G9pnyGxuRE5dvqp/UiztxnuGNzmUbtzRf+19EQWAtvR34YbLc0fOHM7ndTaO4v
eWkpN/i0+iOe1pMrVvRlIIEp0wB+pAR8rFXtzwQP/xfWh87++VYrkNx1jlQDRMbT0jFMCsGCS
qGSIb3DQEHATAUBggqhkiG9w0DBwQIwe1Mmy9Vlh6ACItjRdIakT7u
```

- `servercert`: Ubicació del certificat públic
- `filep12`: Ubicació del fitxer amb el parell de claus (PKCS#12)
- `dbhost`: Nom del servidor de base de dades
- `dbport`: Nom del port TCP del servidor de base de dades
- `dbname`: Nom de la base de dades d'historials
- `dbuser`: Nom de l'usuari de connexió a la base de dades
- `dbpwd`: Contrasenya de l'usuari per a la connexió a la base de dades

F. REGISTRE D'ERRORS DE L'APLICACIÓ

Les aplicacions desenvolupades utilitzen les llibreries log4j [LOG4J] per al registre d'errors. Es defineixen diferents tipus d'errors (amb herència de nivell superior a inferiors) que seran:

ALL:	Tots els nivells
TRACE:	Nivell crític amb informació sobre la traçabilitat de l'error
DEBUG:	Nivell de detall alt
FATAL:	Nivell d'error degut error no recuperable
ERROR:	Nivell d'error recuperable
WARN:	Nivell d'advertència
INFO:	Nivell informatiu

Al fitxer `log.properties` es podrà configurar el nivell d'error desitjat, el format de la marca de temps, la ubicació del fitxer de registre, la mida màxima i el període de rotació.

S'adjunta el fitxer de configuració per la instal·lació del programari:

```
#####################################
#       LOGGER CONFIGURATION        #
#####################################
# Set log levels for rootLogger
log4j.rootLogger=ALL,R

# Print the date in ISO 8601 format
log4j.appender.A1.layout.ConversionPattern=%d [%t] %-5p %c

log4j.appender.R=org.apache.log4j.RollingFileAppender
log4j.appender.R.File=logs/HM.log

log4j.appender.R.MaxFileSize=100KB

# Backup file
log4j.appender.R.MaxBackupIndex=1

log4j.appender.R.layout=org.apache.log4j.PatternLayout
log4j.appender.R.layout.ConversionPattern= %d [%p,%c] %m %n
```

G. MANUAL D'INSTAL·LACIÓ I CONFIGURACIÓ

Aquest manual incorpora les instruccions per a la instal·lació i configuració de l'aplicació client i servidor. S'estructura com un pas a pas que permet a l'usuari i/o administrador servir com a guia bàsica pel desplegament.

1) Revisar el requeriments

Per al correcte funcionament de l'aplicació cal disposar del següent programari instal·lat prèviament:

Pel client:

- JRE 1.6.0 *update* 4 o superior
- Llibreries Java addicionals (veure tercer apartat)

Pel servidor:

- JRE 1.6.0 *update* 4 o superior
- JDK 1.6.0 *update* 4 o superior
- Llibreries Java addicionals (veure tercer apartat)
- MySQL Community Version 5.0.51a o superior

2) Copiar fitxers necessaris pel desplegament de l'aplicació

Copiar el directori `bin\` del paquet de programari proporcionat a la carpeta destí on es vulgui instal·lar l'aplicació. A l'annex H es pot trobar una descripció de l'estructura del paquet lliurat.

3) Instal·lar llibreries necessàries

Es necessari realitzar les següents accions per a preparar l'entorn Java amb les llibreries necessàries:

- [IAIK libs] Copiar `iaik_jce_full.jar` als directoris:

  ```
  %JRE_HOME%\lib\ext
  %JDK_HOME%\jre\lib\ext
  ```

- [JCE policies] Copiar `local_policy.jar` i `US_export_policy.jar` als directoris:

  ```
  %JRE_HOME%\lib\security
  %JDK_HOME%\jre\lib\security
  ```

- [JDOM] Copiar `jdom.jar` als directoris:

  ```
  %JRE_HOME%\lib\ext
  %JDK_HOME%\jre\lib\ext
  ```

- [MYSQL connector] Copiar `mysql-connector-java-5.1.6-bin.jar` als directoris:

  ```
  %JRE_HOME%\lib\ext
  %JDK_HOME%\jre\lib\ext
  ```

- [LOG4J] Copiar `log4j-1.2.15.jar` als directoris:

  ```
  %JRE_HOME%\lib\ext
  %JDK_HOME%\jre\lib\ext
  ```

- [SWT] Copiar `org.eclipse.swt.win32.win32.x86_3.3.3.v3349.jar` als directoris:

  ```
  %JRE_HOME%\lib\ext
  %JDK_HOME%\jre\lib\ext
  ```

- [COMMONS cli] Copiar `commons-cli-1.1.jar` als directoris:

  ```
  %JRE_HOME%\lib\ext
  %JDK_HOME%\jre\lib\ext
  ```

Les variables `%JRE_HOME%` i `%JDK_HOME%` corresponen als directoris d'instal·lació del JRE i JDK de Sun. Només es copiarà al `%JDK_HOME%` si es fa una instal·lació del gestor del sistema. Les llibreries es troben al directori `bin\lib\` de la distribució.

4) Creació de la base de dades i model de dades

Connectar-se a la base de dades (amb la utilitat `mysql` per exemple) i executar l'*script* `createdb.sql` proporcionat al paquet d'instal·lació. Es disposa de més informació a l'annex D.

Aquesta acció crearà la base de dades, l'estructura de taules i l'usuari de connexió.

5) Copiar els certificats necessaris

Es proporcionen els fitxers de claus i certificats necessaris per a la instal·lació. Localitzar-los al paquet que es proporciona i ubicar-los a la ubicació preferida o utilitzar la ruta predefinida al directori `bin\keys\` de la distribució.

Com a mínim es necessitarà un PKCS12 (pel pacient, metge o gestor) i el certificat del gestor.

6) Configuració del client

Editar el fitxer `HMClient.properties` i indicar l'adreça del servidor on s'executa el gestor, port per on escolta les peticions, ubicació del certificat del gestor i ubicació del P12 del client. A l'annex E es descriu les clàusules a modificar.

7) Configuració del servidor

Editar el fitxer `HMServer.properties` i indicar la ubicació del P12 i certificat del gestor, l'adreça del servidor de base de dades, el port d'escolta de la base de dades, el nom de la base de dades, el nom de l'usuari de connexió a base de dades i la contrasenya de la base de dades.

Per tal d'indicar la contrasenya de connexió a la base de dades cal xifrar-la amb la clau pública del gestor. Per fer això, s'ha de fer servir la utilitat proporcionada a la distribució de la següent forma:

```
# java tools.HidePassword –p contrasenyaBD –c ubicacioCert
```

El resultat obtingut s'indica al paràmetre adient del fitxer de configuració.

Configuracions opcionals

Es pot modificar el comportament del registre de l'aplicació obrint i editant convenientment el fitxer `log.properties`. Veure documentació sobre log4j per més detall [LOG4J].

8) Execució del servidor

Al ser la primera vegada que s'executa, s'ha de forçar la inserció de dades de prova i, per tant, executar el fitxer de comandes d'inici de la següent manera:

```
# startHMServer.bat 1
```

En posteriors execucions només caldrà cridar el servidor amb:

```
# startHMServer.bat
```

Nota 1: El servidor escolta per port tcp/1099. Per tant, cal comprovar prèviament que aquest port no estigui ocupat per altre procés. També cal comprovar que les utilitats `rmic` i `rmiregistry` es troben a la variable d'entorn PATH.

Nota 2: Si es força la inicialització de dades caldrà tindre ubicats els següents certificats a les següents ubicacions:
Pacient.p12 a bin\keys\
Metge.p12 a bin\keys\

9) Execució del client

Executar el client de la següent forma:

```
# startHMClient.bat
```

H. RELACIÓ DE FITXERS ADJUNTS A LA MEMÒRIA

A continuació adjunto la relació de fitxers entregats juntament amb aquest document:

Directori	Fitxer/s	Descripció
bin\	README.txt	Explicació breu per iniciat el sistema
	*.class	Classes compilades de l'aplicació (estructurada en paquets)
	sql*.sql	*Scripts* de creació i destrucció de la base de dades
	keys*	PKCS#12 i Certificats d'usuari
	keys\Pacient.p12	Parell de claus d'un usuari amb rol pacient
	keys\Metge.p12	Parell de claus d'un usuari amb rol metge
	logs*	Registre de l'aplicació
	lib*.jar	Llibreries necessàries
	HMServer.properties	Fitxer de configuració del gestor
	HMClient.properties	Fitxer de configuració del client
	log.properties	Fitxer de configuració del registre de l'aplicació
	startHMServer.bat	*Script* per l'arrencada del gestor
	stopHMServer.bat	*Script* per la parada del gestor
	startHMClient.bat	*Script* per l'arrencada del client
doc\	jrodriga_memoria.pdf	Memòria del PFC (format PDF)
	jrodriga_memoria.doc	Memòria del PFC (format Word)
	javadoc*	Documentació del codi font (Javadoc)
pki\	README.txt	Contrasenyes dels PKCS#12 proporcionats
	*	Estructura de PKI (Veure secció 3.3 per més detall sobre el contingut específic)
src\	*.java	Codi font (estructurat en paquets)
project\	*	Exportació del projecte (IDE Eclipse)