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

Juan José Rodríguez Guerra
Enginyeria en Informàtica

Jordi Castellà-Roca
Consultor

11/06/2008

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

Agraïments i dedicatòria.

He de dir que la vida està composta de fites que es van assolint segons la dedicació, ganes i empeny que se li posen per aconseguir-les. Quan ets jove comets errors que després et poden passar factura, i que si reacciones a temps pots arribar a corregir. Un d'aquests errors greus que pots cometre és abandonar els estudis a una edat temprana, ja sigui per falta de motivació externa o interna.

La bona notícia és que sempre estàs a temps de rectificar i reprendre els estudis. Ara bé, l'esforç que hauràs de posar i els obstacles que trobaràs quasi sempre seran superiors que en el temps passat, quan tocava el moment natural d'aquesta apassionant tasca dels estudis. De ben segur que a posteriori no es gaudeixen les comoditats i la falta de responsabilitats que es poden experimentar en l'edat natural dels estudis universitaris.

Però el temp passa i la dedicació fa madurar els seus fruits. Així arribo a aquest punt en el qual he enllestit aquest Projecte de Final de Carrera en l'àrea de la seguretat Informàtica. He acabat una bonica etapa que he disfrutat i he patit amb la satisfació d'haver aconseguit la titulació d'Enginyer Informàtic. Ha estat una estapa de la meva vida que no oblidaré i espero i desitjo poder continuar en endavant.

És per això que vull agrair a la UOC l'oportunitat que ofereix, a estudiants treballadors com jo, d'aconseguir una fita tan important a la vida com és una formació superior i de qualitat.

També he de donar les gràcies a tot l'equip de consultors de la UOC que contribueixen i tenen una part molt important en aquesta formació, i en l'obtenció d'aquesta meta personal tant gratificant. En aquest agraïment no vull oblidar al meu consultor del PFC, en Jordi Castellà-Roca, que molt eficientment ha aclarit dubtes i problemes al llarg del desenvolupament d'aquest treball.

També vull agrair molt especialment a la meva dona Cecília el seu recoltzament que he obtingut per aconseguir aquest somni que em vaig proposar ara ja fa uns quants anys. Gràcies a la seva paciència i tolerància amb la meva dedicació al estudis, que no vol dir que hagi estat un camí sense obstacles, he arribat a aquest punt que tant d'orgull em produeix.

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

Aquest projecte es desenvolupa en aquesta societat de la informació en la qual les Tecnologies de la Comunicació han canviat l'accés a la informació, possibilitant un ventall de prestacions que abans eren impensables.

Aquesta evolució tecnològica, junt amb la de les telecomunicacions, no ha deixat cap sector professional i de la societat sense un impacte considerable. Dins dins d'aquests tenim el de la medicina. Les telecomunicacions obren la porta a la telemedicina, i al tractament de la informació generada pels historials mèdics dels pacients i les visites realitzades d'una forma més eficaç i segura.

El fenòmen Internet ens dóna la capacitat d'interconnectar xarxes locals i corporatives per poder accedir a la informació que aquestes contenen. Però a la vegada creen problemes de seguretat que abans no existien, ja sigui perquè el tractament de la informació es feia amb suport paper, o perquè aquestes xarxes eren d'àmbit local i no estaven interconnectades, o si ho estaven era mitjançant sistemes tancats, com ara línies punt a punt.

La interconnexió de xarxes informàtiques ens permet accedir a gran quantitat d'informació sense necessitat de desplaçar-nos al lloc físic d'ubicació de la mateixa, i amb independència del temps en el que fem l'accés. En el cas del tractament de la informació mèdica i l'accés a historials mèdics, aquestes prestacions aporten un valor afegit molt important per al metge, però també pot ser un avantatge per al pacient. Els historials mèdics donen la informació necessària al metge per a prescriure tractaments, i per a la correcta diagnosi del pacient. A la vegada l'historial mèdic d'un pacient és una informació de gran valor, la qual té un component d'intimitat molt elevada.

És per això que la circulació i l'accés a la informació mèdica generada pels historials dels pacients que contenen els seus expedients i les visites mèdiques realitzades, s'ha de tractar amb unes mesures de seguretat elevades, les quals han de proporcionar integritat, autenticitat al tràfic d'aquesta informació per les xarxes. Però a la vegada també s'ha de proporcionar confidencialitat, ja que la informació dels historials dels pacients és confidencial, i només hauria de ser accessible pel propi pacient, o pel personal mèdic autoritzat.

Això és el que pretén la nostra aplicació, la qual desenvolupa un esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions. L'aplicació proporciona les propietats bàsiques de seguretat que hem esmentat, a més de la propietat de no-repudi que permet assegurar que una informació referent a una visita ha estat introduïda per un metge autoritzat i concret.

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

Índex

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

Índex de figures

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

Índex de taules

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

1. Introducció.

1.1. Justificació del Projecte Final de Carrera i context en el qual es desenvolupa.

Com sabem, les tecnologies de la informació i les telecomunicacions han avançat moltíssim i avui dia possibiliten un ventall de prestacions impensables fa uns anys enrera.

Entre les aplicacions d'aquestes tecnologies tenim la gestió i el control de la informació generada pels historials mèdics dels pacients que visiten aquests professionals. Aquesta informació o historials es compon de l'expedient del pacient i l'historial de visites. La digitalització i el tractament informatitzat d'aquesta informació ha flexibilitzat i possibilitat el seu accés mitjaçant xarxes de computadors.

Partint del fet que la informació que compon els historials mèdics dels pacients de qualsevol ambulatori o hospital és molt sensible i confidencial, trobem una primera justificació del tractament segur d'aquesta informació mitjançant l'esquema criptogràfic de gestió segura dels historials mèdics dels pacients a través d'una xarxa de comunicacions proposat amb aquest Projecte de Final de Carrera.

A més, aquest esquema criptogràfic de gestió segura dels historials mèdics dels pacients a través d'una xarxa de comunicacions que presenta aquest Projecte de Final de Carrera proporciona al tractament dels historials les quatre propietats de la seguretat criptogràfica necessàries per a obtenir el més alt nivell de seguretat. Amb aquest esquema obtenim integritat i autenticitat de les dades, confidencialitat d'aquestes dades a pacients o metges no autoritzats, i la propietat de no-repudi que s'obté mitjançant signatures digitals.

Aquest esquema i tota la seva infraestructura que l'envolta proporcionen les següents avantatges respecte a un tractament diferent del historials mèdics:

- Tots els usuaris que manipulen els historials, metges, pacients i el propi gestor han d'estar autentificats dins el sistema, a més de saber en tot moment quin tipus d'usuari és la persona que manipula els historials.
- Tots els usuaris que manipulen la informació estan registrats i controlats pel sistema.
- El sistema permetrà la consulta del seu historial al pacient d'una forma segura i controlada en tot moment.
- Els metges també estan registrats i tenen assignats la llista dels seu pacients i a més, també poden consultar els historials dels seus pacients prèvia autentificació.
- Els metges també poden donar accés a l'historial d'un dels seus pacients a un altre metge, per exemple, per demanar una segona opinió. El sistema permetrà fer això a metges autorizats.

- Qualsevol metge autoritzat podrà consultar la llista dels seus pacients assignats, així com les dades de cadascun d'ells
- Els metges podran afegir visites a qualsevol dels historials de qualsevol dels seus pacients assignats de forma segura.
- Tan l'eliminació, com la modificació de les dades o visites d'un pacient només serà possible mitjançant el gestor del sistema, fet aquest que dóna un altre nivell de seguretat als historials, els quals no podran ser modificats per altres usuaris.

Quant al context en el qual es desenvolupa el nostre Projecte és clarament tecnològic i dins del que ja s'anomena l'era de la informació, les telecomunicacions i les TIC.

Tan sols uns deu anys enrera la implantació d'aques sistema hauria estat impensable ja que no disposàvem d'una expansió d'Internet com la que tenim avui dia. Tampoc es disposava de connexió a la xarxa d'una forma extensa, ni amb una amplada de banda suficient per a la realització de les operacions que es duen a terme dins de l'àmbit d'aquest projecte.

El tractament de la informació, i en concret dels historials mèdics dels pacients era una tasca farragosa que es prestava a errors, com poden ser pèrdues d'informació, o duplicació de la mateixa.

El nostre projecte es desenvolupa en un temps en el qual Internet s'ha imposat a l'empresa privada, a les institucions, i fins i tot a la llar particular. A més, avui dia la potència de procés del maquinari, la capacitat física d'emmagatzematge, la capacitat de memòria de treball, i l'amplada de banda suficient per a la interconnexió de les xarxes a Internet, proporcionen un entorn idoni per al desenvolupament i la implantació d'un sistema com el que es proposa en aquest projecte. Per tant, totes aquestes circumstàncies fan possible la implantació d'un sistema com aquest amb relativa facilitat.

1.2. Objectius del Projecte Final de Carrera.

L'objectiu principal d'aquest projecte final de carrera és la implementació d'un esquema criptogràfic de tractament d'historials mèdics de pacients de forma segura a través d'una xarxa de comunicacions. Aquest sistema ha de complir les quatre propietats básiques i necessaries per a una gestió segura d'aquesta informació a través de xarxes de comunicació: autenticitat, confidencialitat, integritat i no-repudi. Per a fer complir aquestes propietats utilitzarem criptografia de clau pública [2], i criptografia de clau simètrica [2].

Junt amb aquest objectiu principal del Projecte de la creació d'aquest esquema criptogràfic que s'ha esmentat, s'afegeixen els següents objectius addicionals:

- S'han de crear tres aplicatius per a cadascun dels possibles i inicials usuaris del sistema:

- Aplicació del pacient que aquest farà servir per a accedir de forma segura al gestor del sistema, amb les següents funcionalitats:
  - Autenticació del pacient contra el gestor del sistema.
  - Realització de consultes a les seves dades generals.
  - Obtenir les dades de qualsevol de les seves visites.
  - Abandó segur del sistema.
- Aplcació del metge que aquest usuari utilitzarà per a accedir de forma segura al gestor del sistema, amb les funcions següents:
  - Autenticació del metge contra el gestor del sistema.
  - Consultar les dades generals d'un pacient.
  - Obtenir les dades de qualsevol de les visites d'un dels seus pacients.
  - Afegir una visita a l'historial de qualsevol dels seus pacients.
  - Obtenir una llista dels pacients que té assignats.
  - Sortida segura del sistema.
- Aplcació del gestor, la qual gestionarà el repositori d'historials mèdics de forma centralitzada. Les seves funcions són:
  - Registrar nous usuaris al sistema (pacients o metges).
  - Autenticar a pacients o metges que volen accedir al repositori
  - Acceptar les consultes de pacients i metges.
  - Guardar de forma segura els historials mèdics dels pacients.
  - Verificar l'autenticitat de les dades que s'han inserit o modificat en un historial mèdic i que aquestes provenen d'usuaris autoritzats.
  - Permetre a un usuari que abandoni el sistema de forma segura.

- S'utilitzarà la tecnologia RMI (Remote Method Invocation) [7] per a la comunicació de les aplicacions per a l'obtenció de la informació necessària en cada operació per cadascuna de les parts.

- Es farà servir XML per a les dades que s'intercanviaran entre les diferents aplicacions del sistema, i per a l'emmagatzematge de la informació.

- Es farà servir tecnologia de Base de Dades per a la persistència de les dades dels historials i dels usuaris del sistema.

- També caldrà dissenyar les respectives interfícies per a cadascun els usuaris del sistema, de forma que aquestes siguin funcionals, intuïtives i usables.

1.3. Enfocament i mètode seguit.

El projecte es compon de 9 fases. Aquest es desenvoluparà de forma que cadascuna de les parts sigui funcional i independent en la seva funcionalitat, així s'aniran ampliant les prestacions del sistema fins a arribar a una completa implementació del mateix. De cada mòdul es faran proves unitàries per comprovar el seu correcte funcionament i un cop comprovat s'integrarà amb els mòduls que ja funcionen i es tornaran a fer noves proves, d'aquesta manera s'aconsegueix un desenvolupament esglaonat i fiable de tot el sistema. Les fases del projecte són les següents:

- Esquema criptogràfic.
- Representació de les dades mitjançant XML.
- Comunicacions dels components mitjançant RMI [7].
- Gestió de la informació amb Base de Dades.
- Creació de les interfícies corresponents als actors del sistema.
- Joc de proves, el qual s'anirà desenvolupant a mesura que es van realitzant cadascuna de les fases del Projecte.
- Documentació, que al igual que el Joc de proves es va desenvolupant en paral·lel a la implementació de les fases anteriors.

1.4. Planificació del projecte.

La planificació del projecte comença a l'inci del segon semestre del curs 2007 – 2008. Aquesta planificació s'ajusta a les dates següents:

- **Del 28 de febrer al 2 de març**:
  - Instal·lació d'IAIK [15] i programari necessari, com ara JDK 1.5 i OpenSSL [13].
  - Creació de la Infraestructura de clau pública (PKI), Autoritat de Certificació, parella de claus i certificat del metge, parella de claus i certificat del pacient, i parella de claus i certificat del gestor.
- **Del 3 de març al 30 de març**:
  - Esquema criptogràfic: disseny, implementació, test i documentació dels protocols criptogràfics.
- **Del 31 de març al 13 d'abril**:

  - Representació de les dades mitjançant eXtensible Markup Language (XML) [12]: disseny, implementació, test i documentació de la implementació del tractament i la transferència de dades en la comunicació entre els mòduls de l'aplicatiu mitjançant XML [12].

- **Del 14 d'abril al 27 d'abril**:

  - Comunicacions dels components mitjançant Remote Method Invocation (RMI) [7]. Tecnologia que permetrà comunicar-se als diferents mòduls de l'aplicatiu. Farem el disseny, la implementació, els tests i la documentació de l'ús d'aquesta tecnologia en la comunicació entre els mòduls d'aquest aplicatiu.

- **Del 28 d'abril al 11 de maig**:

  - Gestió de la informació amb Base de Dades, part essencial per a aquest Projecte: disseny, implementació, test i documentació d'aquesta infraestructura din el conjunt del Projecte.

- **Del 12 de maig a l'1 de juny**:

  - Creació de les interfícies corresponents als actors del sistema: usuari, metge i gestor. Consta del disseny, implementació, test i documentació d'aquestes interfícies necessàries per als diferents usuaris del sistema.

- **Del 2 de juny al 8 de juny**:

  - Finalització de la documentació i entrega del Projecte de Final de Carrera.

Seguidament es presenta una representació en MS Project de la planificació d'aquest projecte, la qual s'adjuntarà també amb la memòria del Projecte:

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

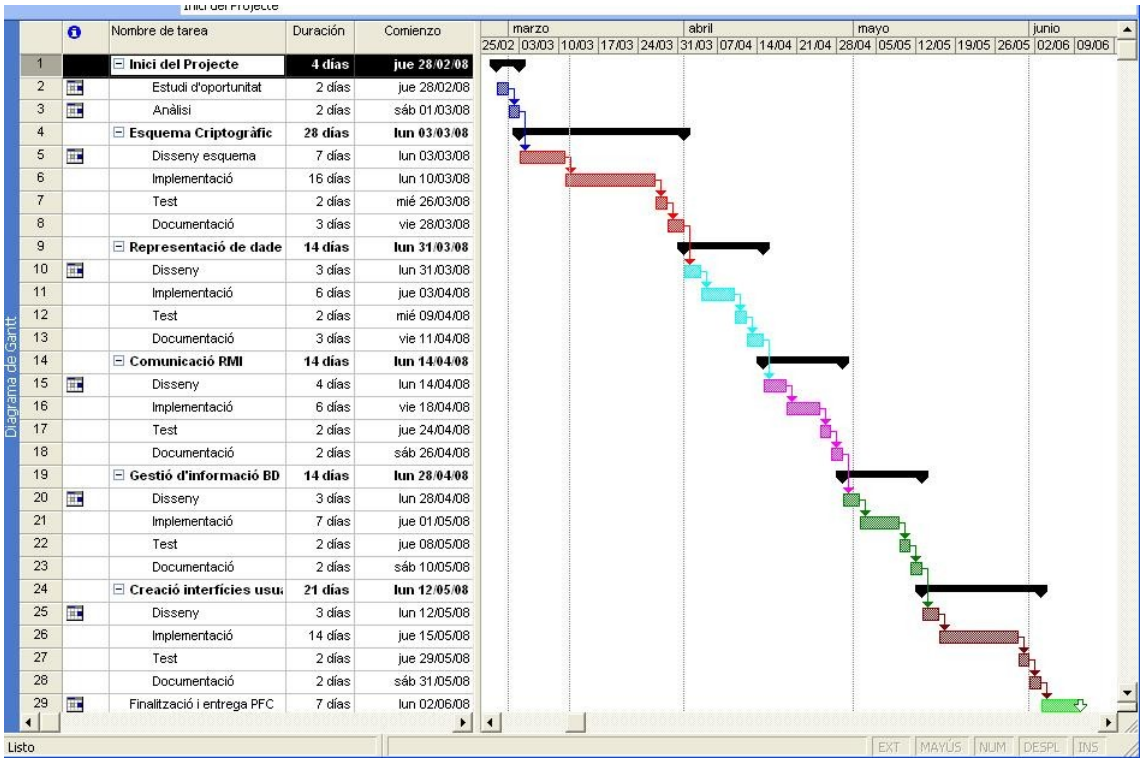


Figura 1.1: Planificació del Projecte.

1.5. Productes obtinguts.

Un cop desenvolupat i implementat tot el Projecte Final de Carrera obtindrem els següents productes:

- **Una infraestructura de clau pública (PKI)** la qual ens ha permés crear les parelles de claus dels usuaris pacient, metge i gestor. Si calgués l'entrada d'algun nou actor en el sistema podríem generar les peticion de certificats i les parellas de claus per a aquests nous actors.

- **Servidor RMI amb l'aplicació del gestor**, la qual donarà resposta a les operacions que faran metges i pacients, gestionant el repositori d'historials de forma centralitzada i segura, i possibilitant l'autenticació d'usuaris en el sistema, així com la marxa d'aquests del mateix de forma segura.

- **Aplicació del pacient** que possibilitarà a aquest demanar l'autentificació al gestor, realitzar consultes de les seves dades i de les dades de qualsevol de les seves visites, i li permetrà abandonar el sistema de forma segura.

- **Aplicació del metge** que permetrà a aquest accedir al gestor de forma segura per demanar autenticació, le consultes de dades i visites dels seus pacients, l'afegiment de visites a qualsevol historial de qualsevol dels seus

pacients, l'obtenció de la llista de pacients assignats, i la possibilitat d'abandonar el sistema de forma segura.

- **Una BD** que contindrà el registre de tots els usuaris, així com les dades d'aquests i els historials mèdics.

1.6. Descripció breu dels següents capítols de la memòria.

Els següents capítols de la memòria expliquen i mostren més abastament el disseny i la implementació del mòduls de l'aplicació, seguint així una implementació incremental que va integrant els diferents mòduls a mesura que es van enllestint, testejant i documentant. Els següent capítols són:

- **Infraestructura de clau pública (PKI):**
  Capítol que explica més ampliament el concepte de PKI i la seva necessitat i utilitat, així com la infraestructura creada per poder utilitzar aquest tipus de criptografia[2] en la gestió de la informació dels historials mèdics que es maneguen amb l'aplicatiu. Per a fer aquesta explicació he utilitzat la informació continguda en el mòduls didàctics de l'assignatura de Criptografia[2] següents: mòdul 5 (Xifres de clau pública), mòdul 6 (Signatura digital) i mòdul 7 (Infraestructura de clau pública).

- **Esquema criptogràfic:**
  Aquests esquemes venen descrits en la documentació del PFC. En aquest capítol es detallen i explica el seu funcionament.

- **Representació de dades: XML:**
  Aquest capítol descriu el format que tindran les dades que s'intercanviaran entre els diferents components de l'aplicació. Per a dur a terme aquesta fita del Projecte es farà servir l'API de Java JDOM [9].

- **Comunicació de components: RMI:**
  En aquest altre capítol s'explica com es porta a terme la comunicació entre els diferents mòduls de l'aplicació. En concret es descriu el mètode utilitzat i que s'ha escollit per minimitzar el temps de desenvolupament. Aquest mètode és Remote Method Invocation (RMI) [7] de Java.

- **Gestió de la informació: Base de Dades:**
  Donat que aquest component és fonamental per al desenvolupament d'aquest projecte, en aquest capítol es dóna el disseny de la BD en el seu model Entitat-Relació i s'expliquen les decisions de disseny. Per a la implementació de la BD i l'emmagatzemament de les dades s'utilitzarà MySQL [10] com a SGBD, fent servir Java [6] per manegar la BD.

- **Interfície gràfica dels diferents usuaris:**
  Ja que s'utilitzarà el mateix sistema per dissenyar les interfícies dels usuaris, s'ha unificat el desenvolupament de les diferents interfícies en un

sol capítol el qual estarà dividit en tres subcapítols on s'explicaran el disseny i el desenvolupament de la interfície del pacient, la del metge, i la del gestor del sistema.

- **Joc de proves:**
  Es mostren les proves del sistema que mostren el seu funcionament amb una exemplificació d'algunes de les operacions que es poden fer amb l'aplicatiu.

- **Conclusions:**
  En aquest capítol es fa un balanç del objectius assolits amb la implementació del Projecte Final de Carrera, comentant l'experiència adquirida.

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

2. PKI.

2.1. Introducció.

En la criptografia de clau pública [2] cada usuari té un parell de claus, una de privada, que només coneix l'usuari en qüestió, i una clau pública que està associada a la privada i que tothom pot tenir.

En el cas de les signatures digitals, les quals s'estan estenen i que aviat tothom podrà realitzar amb el nou DNI electrònic, estan basades en la criptografia de clau pública [2]. El signador signa un missatge amb la seva clau privada, i qualsevol que tingui la clau pública corresponent a aquesta clau privada podrà verificar la signatura digital. D'aquesta manera el verificador obté la certesa que el missatge signat no ha estat alterat, i el signador no es pot desdir de l'autoria d'aquesta signatura. Així, s'aconsegueixen dues propietats de seguretat criptogràfica, no-repudi, i l'autenticitat del missatge.

Però en aquest sistema tenim dos problemes principals: com es distribueix la clau pública d'un usuari, i com sabem que aquesta clau pública pertany a un usuari concret.

Bé, L. Kohnfelder, basat en la idea que dos anys enrera van introduir Diffie i Hellman d'autoritat central de confiança, el 1978 proposa la creació d'uns registres de dades signades, els certificats digitals, que solucionava aquests dos problemes, possibilitant la distribució de les claus públiques desde directoris públics sense requeriments de confiança.

Els certificats digitals són estructures de dades que contenen la informació del propietari d'una clau pública concreta i aquesta mateixa clau pública, tot signat per l'autoritat de certificació que ha fet l'emissió del certificat. L'autoritat de certificació és una entitat registrada i avalada per les institucions i en qui tothom hi confia. Al signar el certificat, dóna validesa a les dades que aquest encapsula, possibilitant així l'obtenció de les propietats d'autenticació, integritat, confidencialitat i no-repudi amb el seu ús.

Poden definir, llavors, una infraestructura de clau pública (PKI - Private Key Infrastructure en anglès) com el conjunt de maquinari, programari, persones, polítiques necessaris per a crear i gestionar certificats digitals basats en la criptografia de clau pública [2].

És, el certificat, el component clau d'una infraestructura de clau pública, el qual possibilita a aquesta el registre d'usuaris, emetre els certificats, signar-los, revocar-los, etc. Per a obtenir interoperabilitat entre aplicacions, cal establir uns estàndards en el tema d'infraestructures de clau pública. D'aquesta manera els fabricants poden aplicar una mateixa sintaxi i estructura de les dades que permetrà al mercat crèixer i millorar.

La situació del mercat avui dia està dividida en solucions propietàries i solucions obertes i de lliure distribució. Existeixen fabricants que amaguen les seves solucions, basant així la seguretat de les mateixes en l'obscuritat i el secret, fet aquest que no compleix la suposició de Kerckhoff (La suposició de Keckhoff diu que tot el mecanisme de xifratge és conegut pel criptoanalista enemic, excepte la clau secreta).

Els Laboratoris RSA han fet importants contribucions en els formats d'intercanvi de dades per a la infraestructura de clau pública. Aquestes especificacions de formats d'intercanvi de dades, anomenades PKCS (Public-Key Criptography Standards) [16], s'han convertit en estàndards *de facto*. En l'apartat 2.4 es descriuen els estàndards PKCS [16].

2.2. Components d'una infraestructura de clau pública.

Els components essencials de la infraestructura de clau pública que ens permetran la gestió de certificats són: l'autoritat de certificació, els subscriptors i els repositoris. Altres components no tan essencials, però que faciliten aquesta tasca de gestió de certificats són les autoritats de registre, l'autoritat de validació, i l'autoritat de segellat de temps.

L'**Autoritat de Certificació** (CA – Certification Authority en anglès) és la responsable d'emetre i revocar certificats. És l'entitat de confiança que dóna legitimitat a la relació entre la identitat d'un usuari o servei i la seva clau pública.

Una infraestructura de clau pública pot tenir una o més Autoritats de Certificació. La creació d'una Autoritat de Certificació necessita la generació d'un parell de clau, una de privada i una de pública, que aquesta farà servir per signar i validar els certificats que emetrà. És molt recomanable que aquest parell de claus de l'Autoritat de Certificació siguin fortes per minimitzar la possibilitat de trencament de les mateixes. La fortalesa de les claus dependrà de la seva longitud i de la qualitat de l'algoritme que les genera.

L'**Autoritat de Registre** (RA – Registration Authority en anglès) verifica el lligam entre les claus públiques i les identitats dels seus titulars.

Les Autoritats de Registre són components opcionals en les infraestructures de clau pública. La seva finalitat és descarregar a les Autoritats de Certificació de moltes funcions administratives. Un exemple d'Autoritats de Registre són les oficines que l'Agència Tributària té a cada localitat espanyola i que fan aquesta tasca per a l'Autoritat de Certificació de la Fàbrica Nacional de Moneda i Timbre d'Espanya. Les Autoritats de Registre són de gran utilitat per a organitzacions grans i geogràficament disperses.

Els **subscriptors** són els que poseeixen un parell de claus i un certificat associat a la clau pública. El seu parell de claus els possibilita realitzar

signatures digitals, xifrar i desxifrar documents, etc. També poden estar en posessió d'un parell de claus i el certificat associat a la clau privada entitats com ara empreses o organismes.

Els **repositoris** són estructures que emmagatzemen la informació relativa a la infraestructura de clau pública.

La infraestructura de clau pública té dos repositoris principals, el repositori de certificats i el repositori de llistes de revocació de certificats. Les llistes de revocació de certificats (CRL – Certificate Revocation List en anglès) inclouen els certificats que, per diversos motius, ja no són vàlids abans de la data de caducitat establerta en el mateix certificat.

El tipus de repositoris més usats són els directoris. Un directori és una base de dades especialitzada en la qual s'emmagatzema informació tipificada i organitzada sobre objectes. Un directori està optimitzat per fer operacions de lectura, de navegació i grans cerques, l'objectiu del qual és donar respostes ràpides a un gran volum de peticions.

L**'Autoritat de Validació** (VA – Validation Authority en anglès) s'encarrega de comprovar la validesa dels certificats digitals. Aquesta autoritat pot ser la mateixa Autoritat de Certificació, o bé una autoritat externa.

L'**Atuoritat de Segellat de Temps** (TSA – Time Stamp Authority) és l'encarregada de signar un missatge amb la finalitat de provar que un missatge existeix en un instant de temps determinat.

L'Autoritat de Segellat de Temps té gran importància per a la propietat de seguretat de no-repudi. Els serveis de no-repudi han de poder establir l'existència d'unes dades abans de determinats moments.

Per últim, i tot i que existeixen altres models d'organització, cal mencionar que les infraestructures de clau pública acostumen a estar organitzades utilitzant un model jeràrquic. En aquest model, els certificats dels subscriptors i entitats finals, estan signats per entitats externes que a la vegada s'identifiquen amb certificats emesos per una Autoritat de Certificació de jerarquia superior.

El certificat d'aquesta Autoritat de Certificació pot estar a la vegada signat per una altra Autoritat de Certificació i així successivament, fins a arribar a una Autoritat de Certificació anomenada arrel. Aquesta Autoritat de Certificació arrel té un certificat autosignat.

2.3. Certificats digitals: el certificat X.509.

El format més àmpliament acceptat per als certificats que emeten les Autoritats de Certificació a usuaris i entitats finals és el format que es coneix com

X.509v.3 [17], el qual també es publica amb el nom d'ITU-T Recommendation X.509 [17].

Aquests certificats s'estructuren en camps que poden ser de tres tipus diferents:

a) **Bàsics**: camps que aporten informació sobre l'autoritat de certificació que ha emès el certificat, l'entitat/subscriptor a què pertany la clau pública, la mateixa clau pública, el període de validesa i la identificació del certificat.

b) **Necessaris per a la signatura**: camps que utilitzarà qui rebi el certificat per a comprovar que el document està signat correctament.

c) **Ampliacions**: camps que han aparegut per cobrir les noves necessitats d'atributs en un certificat, com ara la millora de la gestió de l'herència de certificació, les polítiques de seguretat o les dades sobre l'usuari i la seva clau. Les ampliacions poden estar definides en estàndards o per una organització particular que faci ús de certificats.

2.4. Els documents PKCS.

Com he mencionat en l'apartat 2.1, els Laboratoris RSA, en el seu empeny d'accelerar la implantació de la criptografia de clau pública, desenvoluparen les especificacions del format d'intercanvi de dades en una estructura de clau pública anomenat PKCS (Public-Key Criptography Standards, en anglès) [16]. En el desenvolupament d'aquesta especificació van col·laborar empreses com ara Apple, Microsoft, DEC, Lotus, Sun, MIT, etc.

Des de la seva primera publicació l'any 1991 fins ara, desenvolupadors d'arreu del món els han estat referenciant i implementant àmpliament. Contribucions de les sèries PKCS [16] han estat adoptades com a estàndards formals i *de facto*, com PKIX, SET, S/MIME i SSL. Actualment hi ha 10 normes PKCS [16]: PKCS#1, PKCS#3, PKCS#5, PKCS#7, PKCS#8, PKCS#9, PKCS#10, PKCS#11, PKCS#12 i PKCS#15. El PKCS#2 i PKCS#4 han estat incorporats en el PKCS#1, i el PKCS#6 [16] s'ha retirat en favor de la versió 3 de l'estàndard X.509 [17].

A la Taula 2.1 podem veure el propòsit de cadascun d'aquests formats PKCS [16]:

PKCS	Descripció
1	Defineix els mecanismes per xifrar i signar dades amb el criptosistema de clau pública RSA. També defineix una sintaxi, idèntica a X.509 [17], per a les claus públiques i privades.
3	Defineix un protocol Diffie-Hellman per a l'intercanvi de claus.
5	Descriu un mètode per a xifrar una cadena de text amb una clau secreta derivada d'una frase de pas. El seu objectiu principal és permetre la transmissió xifrada de claus privades entre ordinadors, com es descriu en el PKCS#8 [16], encara que pot ser utilitzada per a xifrar missatges. Fa servir MD2 o MD5 per a produir una clau a partir d'una frase de pas. Aquesta clau s'utilitza per a xifrar amb DES (en mode CBC) el missatge en qüestió.
7	Defineix una sintaxi general per als missatges que inclouen millores criptogràfiques, com signatures digitals o xifratge.
8	Descriu el format de la informació de la clau privada. Aquesta informació inclou una clau privada per a algun algorisme de clau pública i, opcionalment, un conjunt d'atributs.
9	Defineix els tipus d'atributs seleccionats per utilitzar en altres estàndards PKCS [16].
10	Descriu una sintaxi per peticions de certificació.
11	Defineix una interfície de programació independent de la tecnologia, anomenada Cryptoki, per a dispositius criptogràfics com targetes intel·ligents i targetes PCMCIA.
12	Especifica un format portable per a emmagatzemar i transportar claus privades d'usuari, certificats, secrets diversos, etc.
13	Definirà els mecanismes per a xifrar i signar dades utilitzant criptografia basada en corbes el·líptiques [3].
14	Cobreix la generació de nombres pseudoaleatoris.
15	Complement del PKCS#11 [16] que defineix el format de les credencials criptogràfiques emmagatzemades en dispositius criptogràfics.

Taula 2.1: Docuements PKCS [16]

2.5. Ús d'una PKI en el projecte.

En aquest projecte l'ús de la PKI és molt important ja que serà qui emetrà els certificats de les tres parts que interactuen en aquest sistema, el pacient i el metge, tots dos mitjançant la mediació del gestor, per accedir-hi als historials emmagatzemats en el sistema. Cadascun d'aquests tres tipus d'usuaris dels sistema tindran el seu parell de claus, la privada i la pública, i un certificat emès

per l'Autoritat de Certificació de la nostra PKI, el qual encapsularà la identitat i la clau pública de l'usuari signades per aquesta Autoritat de Certificació.

Les claus de l'usuari i el certificats corresponents seran utilitzats per tots tres usuaris del sistema per a funcions d'intercanvi d'informació entre ells i per a les funcions d'autenticació que hauran de realitzar. D'aquesta manera es proporcionarà la seguretat suficient a totes aquestes operacions.

Per les necessitats concretes del Projecte, només tindrem en la jerarquia d'Autoritats de Certificació una única Autoritat de Certificació de confiança que serà l'Autoritat arrel, la qual tindrà el seu parell de claus i el seu certificat autosignat. A més, tot i que les Autoritats de Certificació realitzen més tasques, en el nostre cas, la nostra Autoritat de Certificació només tindrà la funció d'emetre certificats per als usuaris del sistema.

Com hem mencionat en el primer paràgraf d'aquest apartat, cada usuari tindrà un parell de claus i un certificat d'identificació, el qual contindrà la seva clau pública. Tot el conjunt estarà encapsulat en una estructura d'intercanvi d'informació de tipus PKCS#12 [16].

Per a la construcció d'aquesta petita PKI necessària en aquest Projecte utilitzem l'eina Open Source OpenSSL [13]. Mitjançant OpenSSL [13] obtenim un parell de claus per a l'Autoritat de Certificació de 2048 bits de llargada. La parella de claus de l'Autoritat de Certificació ha de ser suficientment forta per minimitzar les possibilitats de trencament, ja que si això passés tota la confiança depositada en l'Autoritat de Certificació es veuria compromesa. Després generem un certificat per a l'Autoritat de Certificació autosignat.

El següent pas és generar amb OpenSSL [13] un parell de claus per a cada usuari de l'aplicació, els pacients, els metges i el gestor del sistema. En aquest cas, la llargada dels parells de claus respectius serà de 1024 bits, ja que no caldrà que siguin tan fortes com les claus de l'Autoritat de Certificació.

Seguidament generarem, també amb OpenSSL [13], una petició de certificat per a cada pacient, una per a cada metge, i una per al gestor del sistema. Aquestes peticions de certificats són enviades a l'Autoritat de Certificació, la qual emetrà els certificats corresponents. Un cop tenim la parella de claus i el certificat de cadascun dels usuaris, generarem amb OpenSSL [13] el contenidor d'intercanvi d'informació PKCS#12 [16].

2.6. Passos a seguir per generar tots els arxius necessaris.

Seguidament es detallen les comandes necessàries i executades per cadascun dels pasos descrits en el apartat anterior:

- Generació de la parella de claus de l'Autoritat de Certificació:

```
head -c 300 /dev/random > aleatori
openssl genrsa -des3  -rand aleatori -out CA.key 2048
```

- Generació de la parella de claus del pacient:

```
head -c 300 /dev/random > aleatori
openssl genrsa -des3  -rand aleatori -out pacient.key 1024
```

- Generació de la parella de claus del metge:

```
head -c 300 /dev/random > aleatori
openssl genrsa -des3  -rand aleatori -out metge.key 1024
```

- Generació de la parella de claus del gestor del sistema:

```
head -c 300 /dev/random > aleatori
openssl genrsa -des3  -rand aleatori -out gestor.key 1024
```

- Generació del certificat autosignat de l'Autoritat de Certificació:

```
openssl req -new -sha1 -x509  -key CA.key -out CA.crt -days 300
```

- Generació de la petició de certificat del pacient:

```
openssl req -new -sha1 -config openssl.cnf -key pacient.key -out pacient.csr
```

- Generació de la petició de certificat del metge:

```
openssl req -new -sha1 -config openssl.cnf -key metge.key -out metge.csr
```

- Generació de la petició de certificat del gestor:

```
openssl req -new -sha1 -config openssl.cnf -key gestor.key -out gestor.csr
```

- Generació del certificat del pacient:

```
openssl ca -config openssl.cnf -out pacient.crt -infiles pacient.csr
```

- Generació del certificat del metge:

```
openssl ca -config openssl.cnf -out metge.crt -infiles metge.csr
```

- Generació del certificat del gestor:

```
openssl ca -config openssl.cnf -out gestor.crt -infiles gestor.csr
```

- Generació del contenidor d'intercanvi d'informació personal PKCS#12 [16] del pacient:

```
openssl pkcs12 -export -in pacient.crt -inkey pacient.key -certfile CA.crt -out pacient.p12
```

- Generació del contenidor d'intercanvi d'informació personal PKCS#12 [16] del metge:

```
openssl pkcs12 -export -in metge.crt -inkey metge.key -certfile CA.crt -out metge.p12
```

- Generació del contenidor d'intercanvi d'informació personal PKCS#12 [16] del gestor:

```
openssl pkcs12 -export -in gestor.crt -inkey gestor.key -certfile CA.crt -out gestor.p12
```

Aquests passos que s'han exemplificat en aquest punt de la memòria s'han de repetir per als metges o els pacients als quals es vulgui generar un certificat i un contenidor d'intercanvi PKCS#12 [16]. A l'annex A podem trobar el fitxer de configuració “openssl.cnf” que s'ha utilitzat per a generar la nostra estructura PKI.

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

3. Esquema criptogràfic.

3.1. Introducció.

Les possibilitats tecnològiques aplicades al món de les comunicacions ha fet possible l'accés a la informació d'una forma més ràpida i eficient. Les telecomunicacions aplicades a la medicina fan possible l'accés a historials mèdics des de qualsevol ordinador personal o terminal que estigui connectat a una xarxa de comunicacions.

La informació continguda en historials mèdics és molt més sensible que qualsevol altra informació. Accedir mitjançant una xarxa de comunicacions a aquests historials requereix prendre mesures de seguretat per protegir la intimitat del propietari de l'historial mèdic.

D'aquest motiu que apunta el paràgraf anterior prové l'objectiu d'aquest Projecte de Final de Carrera, el qual preten aplicar la criptografia simètrica i de clau pública [2] a la protecció d'historials mèdics que són accedits mitjançant xarxes informàtiques de comunicacions, o mitjançant la pròpia Internet.

En els següents apartats s'exposen i expliquen una sèrie de protocols criptogràfics, que també implementen aquest Projecte, que fan possible l'accés al historials mèdics d'una forma segura. El conjunt d'aquests protocols criptogràfics aplicats a les operacions d'accés als historials que seguidament definirem és el que anomenem esquema criptogràfic.

3.2. Actors i accions o serveis del sistema.

Tindem tres actors bàsics en el sistema:

- **Pacients**
- **Metges**
- **Gestor del sistema**

El sistema serà utilitzat bàsicament per pacients i metges, però aquest serà gestionat pel gestor del sistema.

Aquests actors utilitzaran el sistema per tal d'obtenir informació, que també podran modificar. Per a fer aquestes operacions determinarem el conjunt d'accions o serveis que es contemplaran en aquest Projecte:

- **Registre d'usuaris**: el gestor del sistema ha de poder donar d'alta nous pacients i metges en el sistema.
- **Autenticació d'usuaris**: cada usuari s'identificaran i aquesta identificació serà autenticada en el sistema per raons de seguretat.

- **Consultes de les dades generals d'un pacient**: qualsevol metge pot accedir a les dades generals d'un pacient registrat. El mateix pacient del qual són les dades també podrà consultar-les.

- **Consulta de les visites d'un pacient**: un pacient podrà consultar les visites del seu historial. Per als metges, podran consultar les visites de qualsevol historial de qualsevol del seus pacients assignats.

- **Consulta dels pacients assignats a un metge**: qualsevol metge podrà obtenir la llista de pacients que té assignats.

- **Afegir una visita a l'historial mèdic**: un metge podrà afegir noves visites a l'historial de qualsevol dels pacients que té assignats.

- **Modificació de dades dels pacients**: els gestors podran modificar les dades generals dels pacients. Als metges no se'ls permetrà modificar dades ja introduïdes de pacients, ni tan sol dels que té assignats.

- **Eliminació de dades**: els metges no podran eliminar dades del l'historial dels pacients, només els gestors podran afegir o eliminar usuaris del sistema, ja sigui pacients o metges.

3.3. Requisits de seguretat en la gestió de la informació.

Un cop definits els actors del sistema i les accions o serveis que es poden dur a terme en el sistema, seguidament especifico els requisits de seguretat que han de complir cadascuna d'aquestes accions:

- Per a les accions de **Registre d'usuaris**, **Modificació de dades dels pacients** i l'**Eliminació de dades**:

    - El fet que només el gestor és l'únic que pot registrar usuaris en el repositori de dades, modificar les dades generals d'un pacient i eliminar usuaris, metges o pacients, del sistema, proporciona la **integritat** de les mateixes.

    - La propietat de **no-repudi** no s'aplica en aquesta operació.

    - La **confidencialitat** de les dades queda protegida ja que només el gestor és qui les introduirà, entenent-se així que el mateix usuari és qui se les ha fet arribar directament i d'una forma segura.

    - Quant a l'**autenticitat**, queda avalada per la PKI del Projecte, la qual emetrà un certificat a cada usuari amb la seva identitat, tipus d'usuari, i el seu ID d'usuari que identificarà les seves dades.

- Per a les accions d'**Autenticació d'usuaris**, **Consultes de les dades generals d'un pacient**, **Consulta d'una visita d'un pacient**, **Consulta dels pacients assignats a un metge**:

  - La **integritat** de les dades queda garantida pel gestor que és qui fa l'autenticació de l'usuari i és l'únic que proporciona la informació dels historials.

  - La propietat de **no-repudi** no s'aplica en aquesta operació.

  - La **confidencialitat** la proporciona el protocol de l'acció que s'especifica en l'apartat corresponent 3.5, 3.6, 3.7 i 3.8 respectivament.

  - Quant a l'**autenticitat**, queda avalada pel certificat de l'usuari emès per la PKI del Projecte, i el protocol d'autenticació especificat en l'apartat 3.5.

- **Afegir una visita a l'historial mèdic**:

  - La **integritat** de les dades queda garantida pel gestor que és qui fa l'autenticació de l'usuari i és l'únic que proporciona la informació dels historials.

  - La propietat de **no-repudi** es garanteix per la signatura de les dades de la visita que fa el metge abans de fer-les arribar al gestor, el qual verificarà la signatura i, si tot és correcte, afegirà la nova visita a l'historial del pacient que especifica el metge.

  - La **confidencialitat** la proporciona el protocol de l'acció que s'especifica en l'apartat corresponent 3.9.

  - Quant a l'**autenticitat**, queda avalada pel certificat de l'usuari emès per la PKI del Projecte, i el protocol d'autenticació especificat en l'apartat 3.5.

3.4. Notació emprada.

En aquest apartat mostro la notació que emprarem per a la descripció dels protocols que farem servir per a algunes de les operacions especificades en l'apartat 3.2:

- K: clau d'un criptosistema simètric.
- E_K (M): xifratge simètric d'una missatge M amb la clau K.
- D_K (C): desxifratge simètric del criptosistema C amb la clau K.
- (P_E, S_E): parella de claus asimètriques propietat de l'entitat E, on P és la clau pública, i S és la clau privada.
- S_E [M]: signatura digital del missatge M amb la clau privada S de l'entitat E.

- P_E [M]: xifratge del missatge M amb la clau pública asimètrica P_E de l'entitat E.
- H (M): aplicació de la funció criptogràfica hash o resum al missatge M.

3.5. Identificació i autenticació d'usuaris.

Cada usuari s'identificarà en el sistema i, per raons de seguretat, aquesta identificació serà autenticada pel gestor del sistema. Les dades de cada usuari, pacient o metge, seran emmagatzemades junt amb el seu certifcat, així el gestor, al rebre l'identificador d'usuari, podrà contrastar-lo amb el que contingui el seu certificat.

Com a identificador de l'usuari, s'ha decidit prendre els bits que codifiquen el seu DNI concatenats als bits que codifiquen el seu Número de la Seguretat Social (NSS). Aquest identificador també estarà dins el certificat, de forma que el gestor del sistema podrà autenticar així a l'usuari, sigui metge o pacient. A l'annex A podrem veure el fitxer de configuració, “openssl.cnf”, utilitzat per generar la PKI i els certificats dels usuaris. En el cas del gestor, el seu identificador serà el hash del seu certificat, tot i que, com els certificats del pacient i del metge, portarà també a dins el seu DNI concatenat al seu NSS.

Cada usuari del sistema tindrà un certificat que l'identificarà de forma unívoca i que estarà generat per la PKI del Projecte. Els certificats que s'han creat per a aquest Projecte són els següents:

- **Pacient**:
  - PacientCountry Name (2 letter code): ES
  - State or Province Name (full name): Catalunya
  - Locality Name (eg. city): Barcelona
  - Organization Name (eg, company): Universitat Oberta de Catalunya
  - Organizational Unit Name (eg, section): pacient
  - Common Name (eg, YOUR name): Juanjo Rodriguez
  - Email Address: jrodriguezgue@uoc.edu
  - Identifier 2.5.4.5 dnQualifier (DNI|NSS): 55555555666666666

- **Gestor**:
  - PacientCountry Name (2 letter code): ES
  - State or Province Name (full name): Catalunya
  - Locality Name (eg. city): Barcelona
  - Organization Name (eg, company): Universitat Oberta de Catalunya
  - Organizational Unit Name (eg, section): gestor
  - Common Name (eg, YOUR name): Juanjo Rodriguez
  - Email Address: jrodriguezgue@uoc.edu
  - Identifier 2.5.4.5 dnQualifier (DNI|NSS): 11111111222222222

- **Metge**:

- PacientCountry Name (2 letter code): ES
- State or Province Name (full name): Catalunya
- Locality Name (eg. city): Barcelona
- Organization Name (eg, company): Universitat Oberta de Catalunya
- Organizational Unit Name (eg, section): metge
- Common Name (eg, YOUR name): Juanjo Rodriguez
- Email Address: jrodriguezgue@uoc.edu
- Identifier 2.5.4.5 dnQualifier (DNI|NSS): 33333333444444444

Hi ha dues possibilitats respecte a l'autenticació de l'usuari. L'usuari pot autentificar-se un únic cop al principi, o bé autentificar-se cada vegada que fa una operació en el sistema. En el primer cas, caldria mantenir l'estat de la connexió, així el gestor podria saber a quin usuari correspon cada connexió. En la segona opció, l'usuari serà autenticat pel gestor cada cop que realitzi una operació, és a dir, serà autenticat en el moment que sol·liciti l'acció.

La primera opció és la computacionalment més eficient, ja que només cal realitzar una sola autenticació, però, pel contrari, la implementació d'aquesta opció suposa una despesa de temps no menyspreable. És per aquesta última raó que, tenint en compte el temps de què es disposa per desenvolupar el PFC, s'ha optat per la segona opció, tot i que sabem que és menys eficient.

El protocol d'autenticació que seguidament es descriu es basa en el **protocol de clave pública Needham-Schroeder**[1], el qual van proposar Roger Needham i Michael Schroeder en un article el 1978.

Protocol 1 d'autenticació. Essent l'usuari P_i i el gestor del sistema G:

1. *P_i realitza les operacions següents:*
   a) *obté un valor de forma aleatòria N_i;*
   b) *xifra N_i i Id_{Pi} amb la clau pública de G, $P_G(N_i, Id_{Pi})$. Id_{Pi} és l'identificador de P_i;*
   c) *enviar $P_G(N_i, Id_{Pi})$ a G;*

2. *G realitza les operacions següents:*
   a) *desxifra $P_G(N_i, Id_{Pi})$ amb S_G, i obté N_i i I_{Pi};*
   b) *obtenir el certificat de P_i amb Id_{Pi}. A partir del certificat obtindrà P_{Pi};*
   c) *obtenir un valor de forma aleatòria , N_G;*
   d) *xifrar N_i, N_G, Id_G, amb la clau pública P_{Pi} de P_i, $P_{Pi}(N_i, N_G, Id_G)$;*

e) enviar $P_{Pi}(N_i, N_G, Id_G)$ a P_i;

3. *P_i realitza les operacions següents:*
   a) *desxifra $P_{Pi}(N_i, N_G, Id_G)$ amb la clau privada S_{Pi}, i obté N_i, N_G, Id_G;*
   b) *xifrar N_G amb la clau pública P_G de G, $P_G(N_G)$;*
   c) *enviar $P_G(N_G)$ 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.*

3.5.1. Disseny de l'operació.

Aquest protocol dóna lloc a l'operació d'autenticar-se per part de l'usuari, sigui metge o pacient, i el gestor. Hem de recordar que, per qüestions de temps, hem decidit que els usuaris s'autenticaran cada vegada que fan una petició al sistema mitjançant el gestor. Per tant, aquesta operació estarà inclosa en les següent operacions que es defineixen en apartats posteriors.

Seguidament es mostra el diagrama de casos d'ús per a aquesta operació en el sistema i la seva descripció:

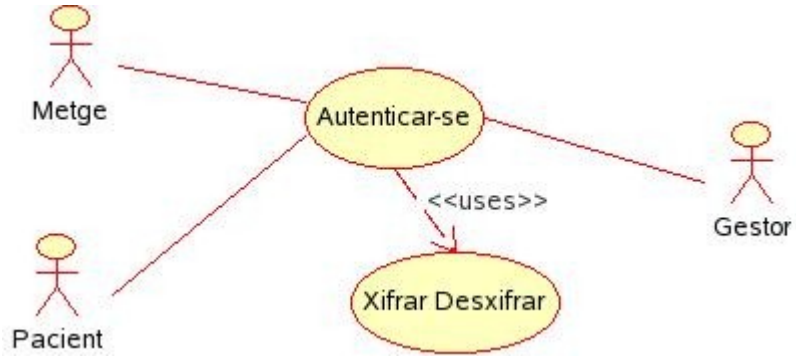


Figura 3.5.1.1: Diagrama cas d'ús Autenticar-se.

Cas d'ús: Autentificar-se.

Resum de la funcionalitat: Permetre a un usuari autentificar-se en el sistema.
Paper dins del treball de l'usuari: Habitual
Actors: Pacient, Metge i Gestor.

Casos d'ús relacionats: Xifrar Desxifrar,

Precondició: L'usuari no s'ha autenticat en el sistema.

Postcondició: L'usuari queda autenticat en el sistema per realitzar l'operació demanada.

Flux d'events principal:

1. L'usuari envia un token i el seu identificador xifrat (usant el cas d'ús Xifrar Desxifrar) amb clau pública al gestor. Dades enviades:
    1.1. Token aleatori
    1.2. DNI seguit NSS
2. El gestor desxifra (usant el cas d'ús Xifrar Desxifrar) les dades amb la seva clau privada, i li envia les dades següents a l'usuari, xifrades (usant el cas d'ús Xifrar Desxifrar) amb la seva clau pública:
    2.1. token de l'usuari
    2.2. token aleatori del gestor
    2.3. Id del gestor (hash del seu certificat)
3. L'usuari desxifra (usant el cas d'ús Xifrar Desxifrar) les dades amb la seva clau privada, i envia al gestor el token del gestor que acaba d'obtenir xifrat (usant el cas d'ús Xifrar Desxifrar) amb la clau pública del gestor. Dades enviades:
    3.1. token del gestor
4. El gestor desxifra (usant el cas d'ús Xifrar Desxifrar) el missatge de l'usuari amb la seva clau privada, obté el token aleatori i comprova que sigui igual al que ell havia enviat.
5. Usuari i gestor queden autenticats bilateralment.

Flux d'events alternatiu:

4.b. Si el token és diferent, no hi ha autenticació i es presenta error.

Per a aquest cas d'ús es necessitaran un seguit de classes que faran les diferents subtasques del cas d'ús. Per als actors del cas d'ús es crearan una classe "Usuari", la qual instanciarà un usuari del sistema, ja sigui metge o pacient. La classe "Usuari" tindrà un atribut amb el tipus d'usuari.

També es crearà una classe "Gestor" que instanciarà el gestor del sistema. Aquesta classe farà les operacions que realitza el gestor del sistema. Inicialment s'ha optat per tenir un repositori de claus públiques dels usuaris i del gestor a nivell local per tal que els diferents usuaris, metges o pacients, o el propi gestor puguin accedir a les claus públiques dels altres actors del sistema. Però quant s'implementi l'apartat d'emmagatzematge de les dades en BD, els certificats dels usuaris s'extreuran de la taula d'usuaris.

Per a tasques de maneig dels fitxers PKCS#12 [16] de cada usuari es crearà una classe "P12Manager", la qual gestiona i extreu les dades del contenidor PKCS#12 [16] de cada usuari. Les classes que instancien un actor del sistema

tindran un atribut que serà de tipus “P12Manager”, el qual contindrà el contenidor PKCS#12 [16] de l'actor.

Per a realitzar les tasques de xifrat i desxifrat de les dades que s'intercanvien els usuaris amb el gestor del sistema, s'ha creat una classe que permet aquestes operacions. La classe és “CipherManager”, la qual implementa un xifrador RSA que xifra amb clau pública i desxifra amb clau privada, tot i que això pot intercanviar-se.

Com que les dades que s'intercanvien els actors del sistema seran en format XML [12], i això pertany al proper capítol, i a més es guardaran en BD (cosa que també s'especificarà en el capítol 5), s'ha creat una classe, “Dades”, que ens servirà com a classe comodí per a aquesta fase de desenvolupament del PFC.

Aquesta classe emmagatzema les dades que s'intercanvien els actors en l'execució dels seus protocols. En els propers capítols, aquesta classe s'anirà adaptant a les necessitats del desenvolupament, fins que ja no sigui necessària quant s'implementi l'emmagatzematge de dades en BD.

També s'ha implementat una classe Main que fa les tasques d'instanciadora de classes i actors del sistema, i executa els protocols especificats en aquest capítol.

Seguidament es mostra un diagrama simple de les classes necessàries en aquest cas d'ús:

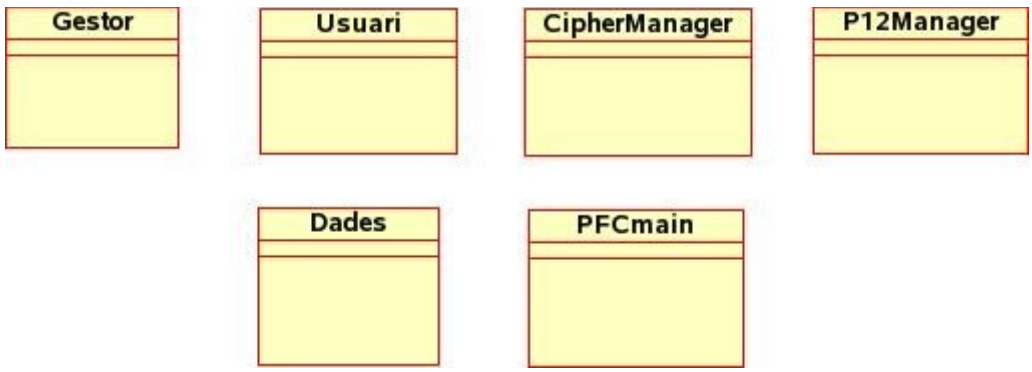


Figura 3.5.1.2: Diagrama de classes del cas d'ús Autenticar-se.

3.6. Consulta de les dades generals d'un pacient.

En aquest protocol, l'usuari U s'identifica com Id_U i disposa d'una parella de claus (P_U, S_U) amb el corresponent certificat $Cert_U$. L'identificador del gestor, que com he dit en el punt anterior és el hash del seu certificat, queda identificat dins el protocol amb Id_G. Aquest protocol podrà ser utilitzat per un metge o per

un pacient. G verifica en cada cas el tipus d'usuari i només facilita les dades generals de l'historial si l'usuari hi té accés:

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

Protocol 2 de consulta de dades generals d'un pacient:

1. *U realitza les operacions següents:*
   a) *Executar el Procedure 1 amb la clau pública P_G, i obtenir $P_G[N_i, Id_U]$;*
   b) *Enviar $P_G[N_i, Id_U]$ a G;*

2. *G realitza les operacions següents:*
   a) *Executar el Procedure 2 amb $P_G[N_i, Id_U]$, i obtenir $P_U[N_i, N_G, Id_G]$;*
   b) *Enviar $P_U[N_i, N_G, Id_G]$ a U;*

3. *U realitza les operacions següents:*
   a) *Desxifrar $P_U[N_i, N_G, Id_G]$ amb la clau privada S_U, i obtenir N'_i, N_G, Id_G;*
   b) *Si $N'_i = N_i$ fer:*
      - *Xifrar N_G, Consulta_dades_generals, Id_usuari amb la clau pública P_G de G, $P_G[N_G$, Consulta_dades_generals, Id_usuari]. Consulta_dades_generals indica que es volen consultar les dades generals de l'usuari identificat amb Id_usuari;*
      - *Enviar $P_G[N_G$, Consulta_dades_generals, Id_usuari] a G;*
   c) *Sino retorna 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_i i N_G han estat guardats a la BD;*
c) *Si $N'_G = N_G$ fer:*
   - *Si (Id_U = Id_usuari) o (Id_U és metge) fer:*
     - *Executar Procedure 3 amb Id_usuari i P_U, i obtenir $P_U[H]$;*
     - *Enviar $P_U[H]$ a U.*
   - *Sino retornar error;*
d) *Sino retornar error;*
e) *Borrar N_i i N_G de la BD;*

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

Procedure 1: conté una part del protocol d'autenticació de Needham-Schroeder. Passos que s'utilitzarà en d'altres protocols per metges i pacients.

Procedure 1 (P_G):

1. *Obtenir un valor de forma aleatòria N_i;*
2. *Xifrar N_i i Id_U amb la clua pública de G, $P_G(N_i, Id_U)$;*
3. *Enviar $P_G(N_i, Id_U)$ a G.*

Procedure 2: conté una altra part del protocol d'autenticació de Needham-Schroeder, però aquesta part està executada pel gestor.

Procedure 2 ($P_G[N_i, Id_U]$):

1. *Desxifrar $P_G(N_i, Id_U)$ amb S_G, i obtenir N_i i Id_U;*
2. *Obtenir el certificat de U a partir Id_U. Suposem que el sistema disposa d'una Base de Dades (BD) on per a cada identificador d'usuari trobem el seu certificat corresponent, a partir del qual obtenim la seva 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_G amb la clau pública P_U de U, $P_U[N_i, N_G, Id_G]$;*
6. *Retornar $P_U[N_i, N_G, Id_G]$.*

Procedure 3: el fa servir el gestor G per trobar l'historial que se li ha demanat i xifrar-lo amb la clau pública de l'usuari que el vol consultar.

Procedure 3 (Id_usuari, P_U):

1. *Buscar l'historial H corresponent a Id_usuari;*
2. *Xifrar H amb la clau pública P_U, $P_U[H]$;*
3. *Retornar $P_U[H]$.*

Procedure 4: l'utilitza l'usuari per tal de desxifrar un historial enviat pel gestor G i verificar si l'historial és correcte.

Procedure 4 (P_U[H]):

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.*

3.6.1. Disseny de l'operació.

Aquest protocol permet a pacients i metges consultar l'historial propi, o d'un pacient, respectivament. És un protocol que porta implícita l'autenticació dels usuaris mitjançant el Gestor del sistema. Hem de recordar que, per qüestions de temps, hem decidit que els usuaris s'autenticaran cada vegada que fan una petició al sistema mitjançant el gestor.

Seguidament es mostra el diagrama de casos d'ús per a aquesta operació en el sistema i la seva descripció:

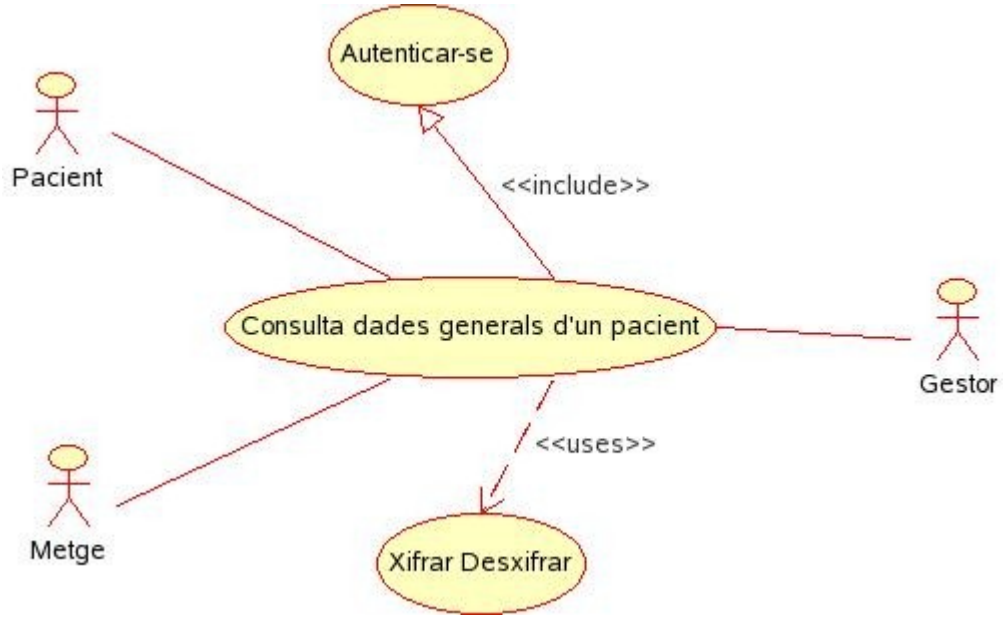


Figura 3.6.1.1: Diagrama cas d'ús Consultar dades generals d'un pacient.

Cas d'ús: Consulta dades generals d'un pacient.

Resum de la funcionalitat: Permetre a un usuari consultar les dades generals d'un historial, si aquest és metge, o el titular de l'historial.
Paper dins del treball de l'usuari: Habitual
Actors: Pacient, Metge i Gestor.
Casos d'ús relacionats: Autenticar-se, Xifrar Desxifrar..
Precondició: L'usuari vol consultar les dades generals d'un historial.
Postcondició: L'usuari s'ha autenticat i ha obtingut les dades generals de l'historial
Flux d'events principal:

1. L'usuari introdueix les dades d'identificació i les envia al gestor, xifrades (usant cas d'ús Xifrar Desxifrar) amb la clau pública del gestor:
   1.1.DNI seguit NSS
   1.2.token aleatori
2. El gestor desxifra (usant el cas d'ús Xifrar Desxifrar) les dades amb la seva clau privada, i li envia les dades següents a l'usuari, xifrades (usant el cas d'ús Xifrar Desxifrar) amb la seva clau pública:
   2.1.Id gestor
   2.2.token aleatori usuari
   2.3.token aleatori gestor

3. L'usuari desxifra (usant el cas d'ús Xifrar Desxifrar) les dades amb la seva clau privada i comprova que el token seu, que li ha enviat el gestor, correspon al que aquest li ha tornat.
4. L'usuari xifra amb la clau pública del gestor (usant el cas d'ús Xifrar Desxifrar) el token retornat del gestor, la petició de consulta, i l'identificador de l'usuari del qual vol consultar les dades generals de l'historial, i li envia al gestor. Dades:
   4.1.token aleatori del gestor
   4.2.petició de consulta
   4.3.Id d'usuari a consultar.
5. El gestor desxifra les dades (usant el cas d'ús Xifrar Desxifrar), recupera el seu token emmagatzemat i el comprova amb el que li envia l'usuari (cas d'ús inclós d'Autenticar-se).
6. Si el usuari és metge o té Identificador igual a l'identificador de consulta, busca l'historial corresponent, el xifra amb la clau pública de l'usuari (usant el cas d'ús Xifrar Desxifrar) i li retorna a l'usuari.
7. El gestor neteja els tokens emmagatzemats.
8. L'usuari desxifra l'historial amb la seva clau privada, obté la clau de sessió, desxifra la llista de descriptors de visites xifrades (usant el cas d'ús Xifrar Desxifrar) i comprova la signatura del gestor d'aquesta última llista.
9. Si l'usuari és pacient, obté la clau de sessió, desxifra la llista de metges (usant el cas d'ús Xifrar Desxifrar) i comprova la signatura del gestor.
10. Seguidament mostra l'historial.

Flux d'events alternatiu:

3.b. Si el token no coincideix el sistema presenta error.

5.b. Si el token no coincideix el sistema presenta error.

6.b. Si l'usuari no coincideix amb l'usuari del qual es vol consultar l'historial, i no és metge, el sistema presenta error.

8.b. Si l'usuari és metge, no es desxifra la llista de metges, ja que s'enten que pot consultar els metges del pacient per una altra via.

A més de les classes que s'han especificat en el punt anterior, per a fer les tasques de tenir dades amb les que treballar fins que s'implementi la Base de Dades, s'han implementat tres classes més que permeten instanciar un historial d'un pacient, una visita, i la fitxa d'un metge.

Per altra banda, la classe "Dades" que ja vam comentar en el cas d'ús anterior, Autenticar-se, punt 3.5, que s'aniria adaptant a les necessitats dels protocols fins a la implementació de la Base de Dades, instanciarà els objectes "Metge",

"Visita", i "Historial", que fan possible disposar de dades per fer proves en aquest punt del PFC. Les classes amb les qual treballem a partir d'aquest punt són:

Figura 3.6.1.2: Diagrama de classes del cas d'ús Consulta dades generals d'un pacient.

Per al disseny d'aquest protocol, com he mencionat, es dissenyaran tres classes, "Historial", "Visita" i "Metge", cadascuna de les quals albergarà les dades necessàries de l'historial, la visita o el metge, respectivament, que s'està consultant i tractant en un moment determinat.

Seguidament especifico les dades amb les quals treballa aquest protocol de consulta de dades generals d'un pacient, i amb les quals també treballaran els protocols restants. És possible que les classes anteriors que instanciaran les dades que estem consultant en un moment determinat, pateixin modificacions segons avança el Projecte i s'implementen els capítols següents d'XML [12] i Base de Dades:

- **Historial,** compost de:
  - Dades generals,
  - Llista de visites protegida,
  - Llista de metges protegida.

  - **Dades generals**:
    - DNI,
    - Numero de Seguretat Social,
    - Nom,
    - Cognom,
    - Número targeta sanitària,
    - Grup sanguini,
    - Al·lèrgies,
    - Vacunes,
    - Observacions,
    - Certificat del pacient X509 v.3

- **Llista de visites protegida**: s'ha pres la decisió que aquesta llista tingui la següent composició:
  - Una **llista de descriptors de visita** (els descriptors de visita es descriuen posteriorment) **xifrada** amb criptosistema simètric,
  - i una **llista d'accés** que es composa dels diferents resultats de xifrar la clau simètrica de sessió, utilitzada per xifrar la llista de descriptors de visita, amb la clau pública dels diferents actors que hi tenen accés, metges, pacient i gestor de sistema. S'ha pres aquesta decisió per no haver de desxifrar la llista i tornar-la a xifrar amb la clau pública del sol·licitant. Així, quan el sol·licitant fa la petició de consulta de les dades de l'historial, se li envia la llista de descriptors xifrada i, si té accés, ell mateix la desxifrarà.

- **Llista de metges protegida**: aquesta llista està composta de tots el metges que poden accedir a l'historial del pacient, i fins quan poden accedir. Per aquesta llista s'ha decidit que és molt interessant per a un pacient que consulta les dades del seu historial poder veure la llista de metges que té assignats. Per tant, aquesta llista estarà signada pel gestor del sistema per a donar-li autenticitat, i xifrada amb sobre digital. La clau de sessió es signarà amb la clau pública del gestor i del pacient, així permetem l'accés als dos actors del sistema. Aquesta llista no es signa amb les claus públiques dels metges del pacient, ja que es considera que cada metge ja sap amb la seva llista de pacients si el pacient és seu. A més, si el metge volgués veure quins metges té un pacient assignats, en un sistema real, podria obtenir aquesta informació per una altra via.

També s'ha decidit per motius de seguretat, respecte a la llista de descriptors de visita xifrada, que, com el pacient i el metge que consulten un historial poden accedir a la clau de sessió de xifratge simètric de la llista de descriptors de visites, quan es dissenyi el protocol d'afegir una visita a l'historial del pacient, això es tindrà en compte, i al afegir la nova visita, la clau de sessió del moment serà substituida per una de nova.

- **Visita,** cada visita es guardarà en una taula de la BD, la qual si és llegida ailladament no es podrà saber de quin pacient és cada visita. Una visita es composa de:
  - Descriptor de visita,
  - Dades de la visita,
  - Signatura digital, del descriptor de visita i de les dades de la visita, realitzada pel metge que ha fet la visita.

  - **Descriptor de visita,** es composa de la informació següent:
    - Identificador aleatori únic obtingut aleatòriament,
    - Data de visita,
    - Hora de visita,

    - Tema de la visita,
    - Metge.

  - **Dades de la visita,** són les següent:
    - anamnesi,
    - diagnosi,
    - tractament.

- **Metge,** tindrà les dades següent en el sistema:
  - DNI,
  - NSS,
  - Nom,
  - Cognom,
  - Número de col·legiat,
  - Especialitat,
  - Certificat X509 v.3,
  - Llista de pacients protegida.

  - **Llista de pacients protegida:** aquesta llista es composa de tots els pacients que té el metge. La llista anirà signada pel gestor del sistema com a prova d'autenticitat. Després tot el conjunt, llista de pacients i signatura del gestor, aniran xifrades amb un criptosistema simètric amb la clau de sessió xifrada amb la clau pública del gestor, i la clau pública del metge.

3.7. Consulta d'una visita d'un pacient.

En aquest protocol, l'usuari U s'identifica amb Id_U i pot ser un metge, o un pacient. G verifica en cada cas el tipus d'usuari i només facilita les dades generals de l'historial si l'usuari hi té accés:

- L'usuari demana una de les seves visites.

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

Protocol 3 de consulta d'una visita d'un pacient:

1. *U realitza les operacions següents:*
   a) *Executar el Procedure 1 amb la clau pública* P_G*, i obtenir* $P_G[N_i, Id_U]$*;*
   b) *Enviar* $P_G[N_i, Id_U]$ *a G;*

2. *G realitza les operacions següents:*

a) *Executar el Procedure 2 amb $P_G[N_i, Id_U]$, i obtenir $P_U[N_i, N_G, Id_G]$;*
c) *Enviar $P_U[N_i, N_G, Id_G]$ a U;*

3. *U realitza les següents:*
   a) *Desxifrar $P_U[N_i, N_G, Id_G]$ amb la clau privada S_U, i obtenir N'_i, N_G, Id_G;*
   b) *Si $N'_i = N_i$ fer:*
      - *Xifrar N_G, Consulta_visita, Id_usuari, descriptor_de_visita amb la clau pública P_G de G, $P_G[N_G$, Consulta_visita, Id_usuari, descriptor_de_visita]. Consulta_visita indica que es vol consultar la visita indicada per descriptor_de_visita de l'usuari identificat amb Id_usuari;*
      - *Enviar $P_G[N_G$, Consulta_visita, Id_usuari, descriptor_de_visita] a G;*
   c) *Sino retornar error;*

4. *G realitza les 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, descriptor_de_visita;*
   b) *Recuperar N_G de la BD. En el pas 4 del Procedure 2 N_i i N_G han estat guardats a la BD;*
   c) *Si $N'_G = N_G$ fer:*
      - *Si Procedure 5 [Id_U, Id_usuari, descriptor_de_visita] retorna que totes les les verificacions són correctes fer:*
        - *Obtenir la visita identificada per descriptor_de_visita (V) i calcula $P_U[V]$;*
        - *Enviar $P_U[V]$ a U;*
      - *Sino retornar error;*
   d) *Sino retornar error;*
   e) *Borrar N_i i N_G de la BD;*

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

Procedure 5: aquest procedure farà totes les verificacions pertinents abans d'entregar la informació de la visita a l'usuari, o al metge que demana la consulta.

Procedure 5 (Id_U, Id_usuari, descriptor_de_visita):

1. *Si (Id_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_U;*
   b) *Retornar el resultat de la verificació.*

2. *Si (Id_U) és un metge fer:*
   a) *Verificar si Id_usuari està dins de la llista_de_pacients_protegida del metge identificat per Id_U;*
   b) *Verificar si el metge identificat per Id_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 llista_de_visites_protegides de l'usuari;*
   d) *Retornar el resultat de la verificació.*

3.7.1. Disseny de l'operació.

Com he mencionat anteriorment, aquest protocol dóna el servei de consulta d'una visita a un pacient, prèvia verificació pel gestor del sistema què el descriptor de visita indicat estigui dins de la llista de descriptor de visita del pacient; o bé a un metge, també amb la verificació per part del gestor que el pacient indicat estigui en la llista de pacients del metge, que el metge sigui dins la llista de metges del pacient, i que el descriptor de visita pertanyi a l'usuari del qual es vol consultar la visita indicada per aquest descriptor.

Seguidament es mostra el diagrama de casos d'ús per a aquesta operació en el sistema i la seva descripció:

Figura 3.7.1.1: Diagrama cas d'ús Consultar visita d'un pacient.

Cas d'ús: Consulta d'una visita d'un pacient.

Resum de la funcionalitat: Permetre a un usuari consultar una visita del seu historial, o al metge si l'usuari al qual pertany la visita és pacient seu.
Paper dins del treball de l'usuari: Habitual
Actors: Pacient, Metge i Gestor.
Casos d'ús relacionats: Autenticar-se, Xifrar Desxifrar, Verificacions accés a visita.
Precondició: L'usuari vol consultar una visita del seu historial, o d'un pacient seu, si aquest és metge.
Postcondició: L'usuari s'ha autenticat, compleix els requisits d'accés a la visita, i ha obtingut les dades de la mateixa.
Flux d'events principal:

1. L'usuari introdueix les dades d'identificació i les envia al gestor, xifrades (usant cas d'ús Xifrar Desxifrar) amb la clau pública del gestor:
   1.1. DNI seguit NSS
   1.2. token aleatori

2. El gestor desxifra (usant el cas d'ús Xifrar Desxifrar) les dades amb la seva clau privada, i li envia les dades següents a l'usuari, xifrades (usant el cas d'ús Xifrar Desxifrar) amb la seva clau pública:
   2.1. Id gestor
   2.2. token aleatori usuari
   2.3. token aleatori gestor
3. L'usuari desxifra (usant el cas d'ús Xifrar Desxifrar) les dades amb la seva clau privada i comprova que el token seu que li ha enviat el gestor correspon al que aquest li ha tornat.
4. L'usuari xifra les dades següents amb la petició de consulta de visita amb la clau pública del gestor (usant el cas d'ús Xifrar Desxifrar) i li envia a aquest. Les dades són:
   4.1. Token del gestor rebut,
   4.2. Cadena "consulta_visita" per indicar que vol consultar una visita,
   4.3. Id del usuari de la visita,
   4.4. Descriptor de visita a consultar.
5. El gestor les desxifra les dades amb la seva clau privada (usant el cas d'ús Xifrar Desxifrar), obtenim tots els paràmetres enviats per l'usuari i comprova el seu token amb l'enviat per l'usuari (cas d'ús inclós d'Autenticar-se).
6. El gestor verifica si l'usuari pot veure la visita (cas d'ús inclòs Verificacions d'accés a visita).
7. El gestor obté la visita indicada pel descriptor de visita, la xifra amb la clau pública de l'usuari (usant el cas d'ús Xifrar Desxifrar), i li envia a aquest. Dades enviades:
   7.1. Visita xifrada.
8. L'usuari desxifra la visita amb la seva clau privada (usant el cas d'ús Xifrar Desxifrar), obté la visita i presenta les dades.

Flux d'events alternatiu:

3.b. Si el token no coincideix el sistema presenta error.
5.b. Si el token no coincideix el sistema presenta error.
6.b. Si l'usuari no pot veure la visita el sistema presenta error.

Cas d'ús: Verificacions d'accés a visita.

Resum de la funcionalitat: Permet verificar si l'usuari pot accedir a la visita o no.
Paper dins del treball de l'usuari: Habitual
Actors: Pacient, Metge i Gestor.

Casos d'ús relacionats: Consulta d'una visita d'un pacient.
Precondició: El gestor vol verificar si l'usuari té accés a la visita que demana.
Postcondició: El gestor ha verificat i confirmat, o no, l'accés a la visita.
Flux d'events principal:

1. El gestor comprova que l'suari que fa la petició és igual al qual pertany la visita.
2. El gestor verifica que el descriptor de visita és dins la llista de visites protegida de l'usuari.
3. El gestor retorna el resultat de la verificació.

Flux d'events alternatiu:

1.b. El gestor comprova que l'usuari que fa la petició és metge.
2.b. El gestor comprova que l'usuari del qual es vol consultar la visita és dins la llista de pacients protegida del metge.
3.b. El gestor comprova que el metge que fa la consulta està dins la llista de metges de l'usuari.
4.b. El gestor comprova que el descriptor de visita pertany a l'usuari del qual es vol fer la consulta de la visita.
5.6 El gestor retorna el resultat de les verificacions.

Per a aquesta operació no hi ha hagut la necessitat d'implementar cap altra classe, a de les que s'especifiquen en l'anterior operació i que podem veure en el diagrama 3.5.1.2.

3.8. Consulta dels pacients assignats a un metge.

Aquesta és una operació que faran els metges, ja que quan necessitin l'historial d'un dels seus pacients hauran de comprovar prèviament si el pacient que sol·liciten és pacient seu.

Protocol 4 de consulta dels pacients assignats a un metge:

1. *U realitza les operacions següents:*
   a) *Executar el Procedure 1 amb la clau pública P_G, i obtenir $P_G[N_i, Id_U]$;*
   b) *Enviar $P_G[N_i, Id_U]$ a G;*

2. *G realitza les operacions següents:*

c) *Executar el Procedure 2 amb $P_G[N_i, Id_U]$, i obtenir $P_U[N_i, N_G, Id_G]$;*
a) *Enviar $P_U[N_i, N_G, Id_G]$ a U;*

3. *U realitza les operacions següents:*
   a) *Desxifrar $P_U[N_i, N_G, Id_G]$ amb la clau privada S_U, i obtenir N'_i, N_G, Id_G;*
   b) *Si $N'_i = N_i$ fer:*
      - *Xifrar N_G, Llista_pacients amb la clau pública P_G de G, $P_G[N_G$, Llista_pacients]. Llista_pacients indica que es vol un llistat dels pacients del metge identificat amb Id_U;*
      - *Enviar $P_G[N_G$, Llista_pacients] 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, Llista_pacients;*
   b) *Recuperar N_G de la BD. En el pas 4 del Procedure 2 N_i i N_G han estat guardats a la BD;*
   c) *Si $N'_G = N_G$ fer:*
      - *Si Id_U és metge fer:*
        - *Calcular P_U[llista_pacients_protegida];*
        - *Enviar a U P_U[llista_pacients_protegida];*
      - *Sino retornar error;*
   d) *Sino retornar error;*
   e) *Borrar N_i i N_G de la BD;*

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

3.8.1. Disseny de l'operació.

Aquest protocol permet a un metge que necessita consultar l'historia d'un pacient, si aquest és pacient seu. Per a saber això, el metge demanarà al gestor del sistema la llista de pacients protegida que té assignada, i un cop la tingui, la desxifrarà ja que està xifrada amb la clau pública del gestor, i d'ell mateix.

El diagrama de casos d'ús d'aquesta operació és el següent:

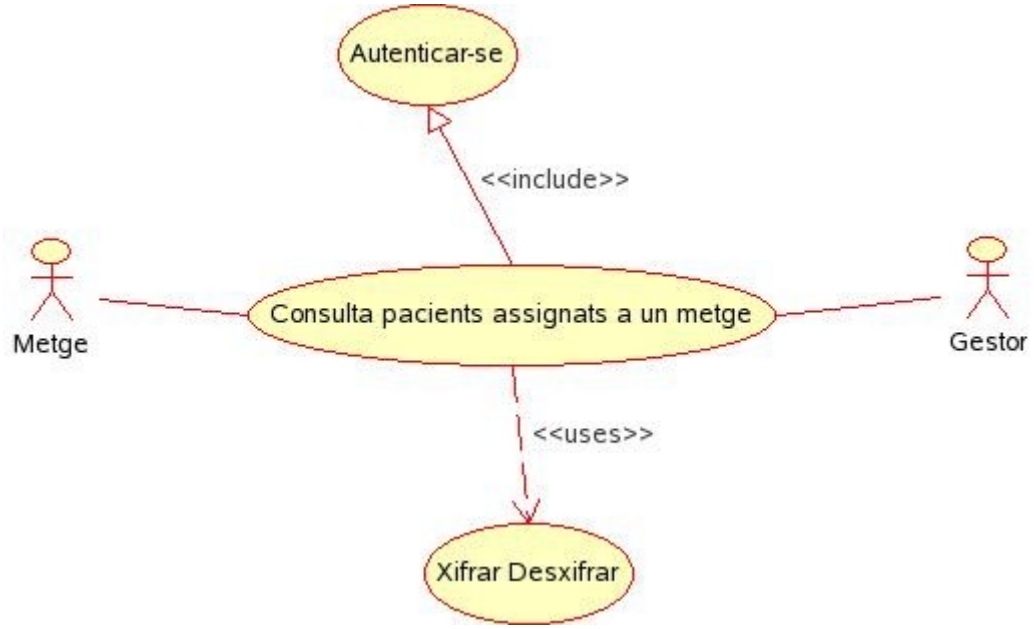


Figura 3.8.1.1: Diagrama cas d'ús Consulta pacients assignats a un metge.

Cas d'ús: Consulta dels pacients assignats a un metge.

Resum de la funcionalitat: Permetre a un metge consultar la llista de pacients que té assignats.
Paper dins del treball de l'usuari: Habitual
Actors: Metge i Gestor.
Casos d'ús relacionats: Autenticar-se, Xifrar Desxifrar.
Precondició: El metge vol saber els pacients que té assignats.
Postcondició: El metge ha obtingut la llista de pacients que té assignats.
Flux d'events principal:

1. L'usuari introdueix les seves dades d'identificació i les envia al gestor xifrades amb la clau pública del gestor (cas d'us Xifrar Desxifrar). Les dades enviades al gestor són:
   1.1. DNI seguit NSS

1.2.token aleatori

2. El gestor les desxifra, obté la clau pública de l'usuari, i envia les dades següents a l'usuari xifrades amb la clau pública del usuari (cas d'ús Xifrar Desxifrar):
   2.1. token aleatori del usuari rebut en pas anterior,
   2.2. token aleatori del gestor,
   2.3. Id del gestor.
3. L'usuari desxifra les dades amb la seva clau privada (cas d'ús Xifrar Desxifrar), i comprova que el seu token és igual al que li ha tornat el gestor.
4. L'usuari xifra amb la clau pública del gestor (cas d'ús Xifrar Desxifrar) el token rebut del gestor i la petició de consulta de la llista de pacients i li envia al gestor. Les dades enviades són:
   4.1. token aleatori del gestor,
   4.2. petició de consulta de llista de pacients.
5. El gestor desxifra les dades amb la seva clau privada (cas d'ús Xifrar Desxifrar) i obté les dades, recuperar el seu token emmagatzemat i el comprova amb el que li ha enviat l'usuari (fins aquí cas d'ús inclos Autenticar-se).
6. Seguidament comprova que l'usuari sigui un metge.
7. El gestor recupera la llista de pacients del metge protegida que ja està xifrada amb la seva clau pública i amb la de l'usuari i li envia a l'usuari. Dades enviades:
   7.1. Llista de pacients protegida.
8. L'usuari desxifra la llista de pacients protegida (cas d'ús Xifrar Desxifrar) amb la seva clau privada, tracta la llista i la presenta per pantalla.

Flux d'events alternatiu:

3.b. Si el token no coincideix el sistema presenta error.
5.b. Si el token no coincideix el sistema presenta error.
6.b. Si l'usuari no és metge, el sistema presenta error.

Per a aquesta operación no hi ha hagut la necessitat d'implementar cap altra classe a part de les que s'especifiquen en el punt anterior 3.5.1.2.

3.9. Afegir una visita a l'historial mèdic d'un pacient.

En aquest protocol se suposa que el metge ja ha consultat l'historial del pacient P, i per tant coneix Id_P, abans de la inserció de la visita en l'historial. El següent protocol, **Protocol 5**, està pensat únicament per afegir una nova visita

a l'historial. Quan el gestor rep una visita V d'un pacient P verifica que ha estat signada pel metge M assignat al pacient. Seguidament afegeix el descriptor de la visita a la llista de descriptors protegida i després la visita a la Base de Dades.

Protocol 5 per afegir una visita a l'historial mèdic d'un pacient:

1. *M realitza les operacions següents:*
   a) *Executar el Procedure 1 amb la clau pública P_G, i obtenir $P_G[N_i, Id_M]$;*
   b) *Enviar $P_G[N_i, Id_M]$ a G;*

2. *G realitza les operacions següents:*
   a) *Executar el Procedure 2 amb $P_G[N_i, Id_M]$, i obtenir $P_M[N_i, N_G, Id_G]$;*
   b) *Enviar $P_M[N_i, N_G, Id_G]$ a M;*

3. *M realitza les operacions següents:*
   a) *Desxifrar $P_M[N_i, N_G, Id_G]$ amb la clau privada S_M, i obtenir N'_i, N_G, Id_G;*
   b) *Si $N'_i = N_i$ fer:*
      - *Obtenir les dades de la visita V;*
      - *Signa V amb la clau privada S_M de M, $S_M[V]$;*
      - *Xifrar N_G, Afegir_visita, V, Id_usuari i $S_M[V]$ amb la clau pública P_G de G, $P_G[N_G$, Afegir_visita, V, Id_usuari, $S_M[V]]$. Afegir_visita indica que es vol afegir V a l'historial del pacient P indentificat per Id_usuari;*
      - *Enviar $P_G[N_G$, Afegir_visita, V, Id_usuari, $S_M[V]]$ 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 de G, i obtenir N'_G, Afegir_visita, V, Id_usuari, $S_M[V]$;*

b) Recuperar N_G de la BD. En el pas 4 del Procedure 2 N_i i N_G han estat guardats a la BD;

c) Si $N'_G = N_G$ fer;

- *Verificar que Id_M és metge;*
- *Verificar que Id_usuari és un pacient assignat al metge Id_M. Amb la llista_de_pacients_protegida del metge i la llista_de_metges del pacient es poden fer aquestes verificacions.*
- *Si les verificacions anteriors són correctes fer;*
  - *Verificar la signatura digital amb la clau pública P_M;*
  - *Obtenir el descriptor_de_la_visita de V;*
  - *Afegir el descriptor_de_la_visita a la llista_descriptors_de_visites;*
  - *Signar amb la clau privada del gestor, S_G, la llista_descriptors_de_visites;*
  - *Xifrar la llista_descriptors_de_visites amb una clau de sessió K, E_K(llista_descriptors_de_visites);*
  - *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_visites_protegida;*
  - *Afegir V a la BD.*
- *Sino retornar error;*

3.9.1. Disseny de l'operació.

Aquest protocol s'encarrega d'afegir una visita per part d'un metge a l'historial d'un pacient. Abans d'afegir-la, el gestor del sistema verificarà que l'usuari del sistema és metge, que el pacient estigui assignat al metge i que el metge ho sigui del pacient.

E verificarà la signatura del metge, s'obtindrà i afegirà el descriptor de visita a la llista de descriptors de visita de l'historia del pacient, i aquesta nova llista amb el nou descriptor derà signada novament pel gestor del sistema. Seguidament es tornarà a xifrar amb les claus públiques dels metges, del gestor del sistema,

i també del pacient, ja que com vam explicar en la consulta de l'historal del pacient, es permet al pacient que pugui accedir a aquesta llista.

Per seguretat de la mateixa llista, cada cop que s'afegeix una visita a l'historial del pacient, es canvia la clau simètrica que s'utilitza per xifrar la llista de descriptors de visita, així s'obté més seguretat ja que quan el pacient consulta el seu historial, se li facilita aquesta clau de sessió.

El més adient per això seria canviar la clau de sessió cada cop que el pacient consulta el seu historial, però aquesta última possibilitat no s'ha desenvolupat per qüestions de cost computacional.

En aquest punt, com que no tenim implementada la base de dades encara, el que es fa és crear una visita instanciant l'objecte Visita amb les noves dades. Això canviarà quan dissenyem i implementem la Base de Dades corresponent.

El diagrama de casos d'ús d'aquesta operació és el següent:

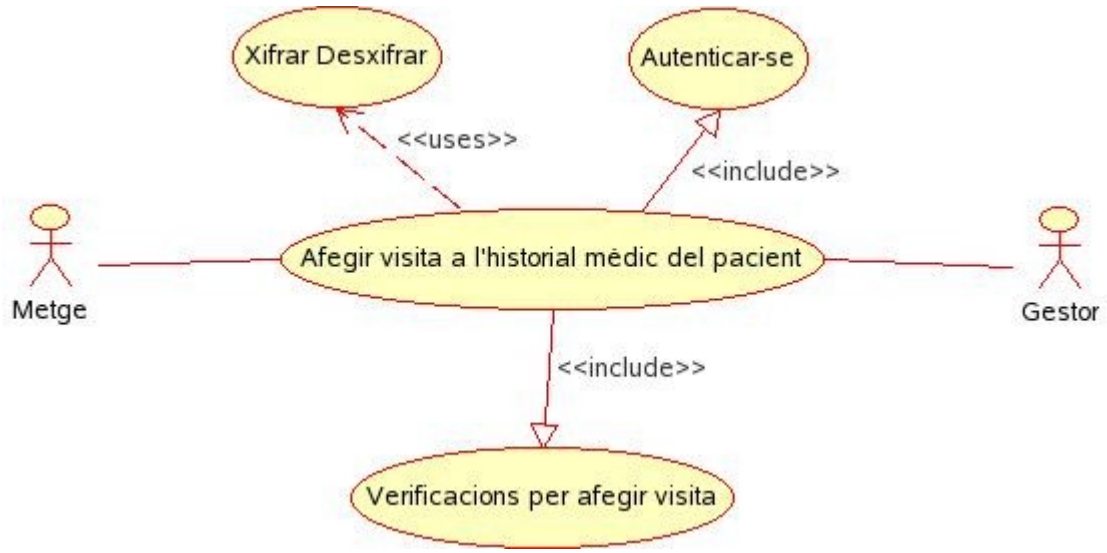


Figura 3.9.1.1: Diagrama cas d'ús Afegir visita a l'hitorial mèdic del pacient.

Cas d'ús: Afegir visita a l'historial mèdic d'un pacient.

Resum de la funcionalitat: Permet a un metge afegir una visita a l'historial mèdic d'un dels seus pacients.
Paper dins del treball de l'usuari: Habitual
Actors: Metge i Gestor.
Casos d'ús relacionats: Autenticar-se, Xifrar Desxifrar, Verificacions per afegir visita.
Precondició: El metge ha fet la visita al pacient i té les dades i les vol afegir a l'historial del pacient.

Postcondició: El metge ha afegit la visita a l'historial d'un del seus pacients.

Flux d'events principal:

1. El metge introdueix les dades d'identificació i les envia al gestor xifrades amb la clau pública del gesto (cas d'ús Xifrar Desxifrar). Dades enviades:
   1.1. DNI seguit NSS,
   1.2. token aleatori.
2. El gestor les desxifra amb la seva clau privada (cas d'ús Xifrar Desxifrar) i obté les dades.
3. Amb la clau pública del metge xifra (cas d'ús Xifrar Desxifrar) les següents dades i les envia al metge. Dades enviades:
   3.1. Id gestor,
   3.2. token aleatori usuari,
   3.3. token aleatori gestor.
4. El metge desxifra les dades amb la seva clau privada (cas d'ús Xifrar Desxifrar) i comprova que el seu token és igual al que li ha tornat el gestor.
6. El metge recull les dades de la visita i les signa amb la seva clau privada.
7. Xifra amb la clau pública del gestor (cas d'ús Xifra Desxifrar) el token rebut del gestor i la petició d'afegir una visita, l'identificador de l'usuari al qual vol afegir la visita, la visita i la signatura de la visita. Tot seguit envia tot el conjunt al gestor. Les dades enviades són:
   7.1. token aleatori del gestor,
   7.2. Afegir una visita,
   7.3. identificador d'usuari al qual se li vol afegir la visita,
   7.4. les dades de la visita que ha obtingut,
   7.5. la signatura de la visita.
8. El gestor desxifra les dades rebudes (cas d'ús Xifrar Desxifrar), recuperar el seu token emmagatzemat i el comprova amb el que li envia el metge (fins aquí cas d'ús inclòs Autenticar-se).
9. El gestor verifica que es compleixi que l'usuari que vol afegir la visita és metge i que l'identificador d'usuari rebut és un pacient seu (cas d'ús inclòs Verificacions per afegir Visita).
10. El gestor verifica la signatura del metge.
11. El gestor calcula el descriptor de la visita que s'ha d'afegir.
12. El gestor afegeix el descriptor de la visita a la llista de descriptors de visites de l'historial del pacient.

13. El gestor signa amb la seva clau privada la llista de descriptors de visites.
14. El gestor xifra la llista de descriptors de visites amb un clau de sessió i aquesta amb la seva clau pública, la del pacient i la dels metges que hi tenen accés, i s'obté una nova llista de descriptors de visites protegida.
15. Finalment afegeix la visita a la Base de Dades.

Flux d'events alternatiu:

4.b. Si el token no coincideix el sistema presenta error.
8.b. Si el token no coincideix el sistema presenta error.
9.b. Si les verificacions no són positives, retorna un error.

Cas d'ús: Verificacions per afegir visita.

Resum de la funcionalitat: Permet verificar si el metge pot afegir la visita o no.

Paper dins del treball de l'usuari: Habitual

Actors: Pacient, Metge i Gestor.

Casos d'ús relacionats: Afegir visita a l'historial mèdic d'un pacient.

Precondició: El gestor vol verificar si el metge pot afegir la nova visita.

Postcondició: El gestor ha verificat i confirmat, o no, que el metge pot afegir la nova visita.

Flux d'events principal:

1. El gestor verifica que l'usuari és metge.
2. El gestor verifica que el pacient està dins la llista de pacients del metge que vol afegir la visita.
3. El gestor verifica que el metge és dins la llista de metges del pacient.

Flux d'events alternatiu:

1.b. Si no és metge retorna error.
2.b. Si no és així retorna error.
3.b. Si no és així retorna error.

Per a aquesta operación no hi ha hagut la necessitat d'implementar cap altra classe a part de les que s'especifiquen en el punt 3.5.1.2. Cal recordar que la part d'afegir la visita a la Base de Dades no es podrà implementar fins a tenir aquesta definida i creada.

3.10. Diagrama de classes de l'esquema criptogràfic.

Arribats a aquest punt tenim les classes bàsiques que implementen i fan possible la funcionalitat de tots aquests protocols de seguretat especificats, és

a dir, del nostre esquema criptogràfic. Totes elles s'han implementat amb Java 1.5.0.06 [6]. A l'annex B s'explica la instal·lació de l'aplicatiu.

Seguidament mostraré un diagrama de classes més detallat de les classes que s'han implementat en aquest punt:

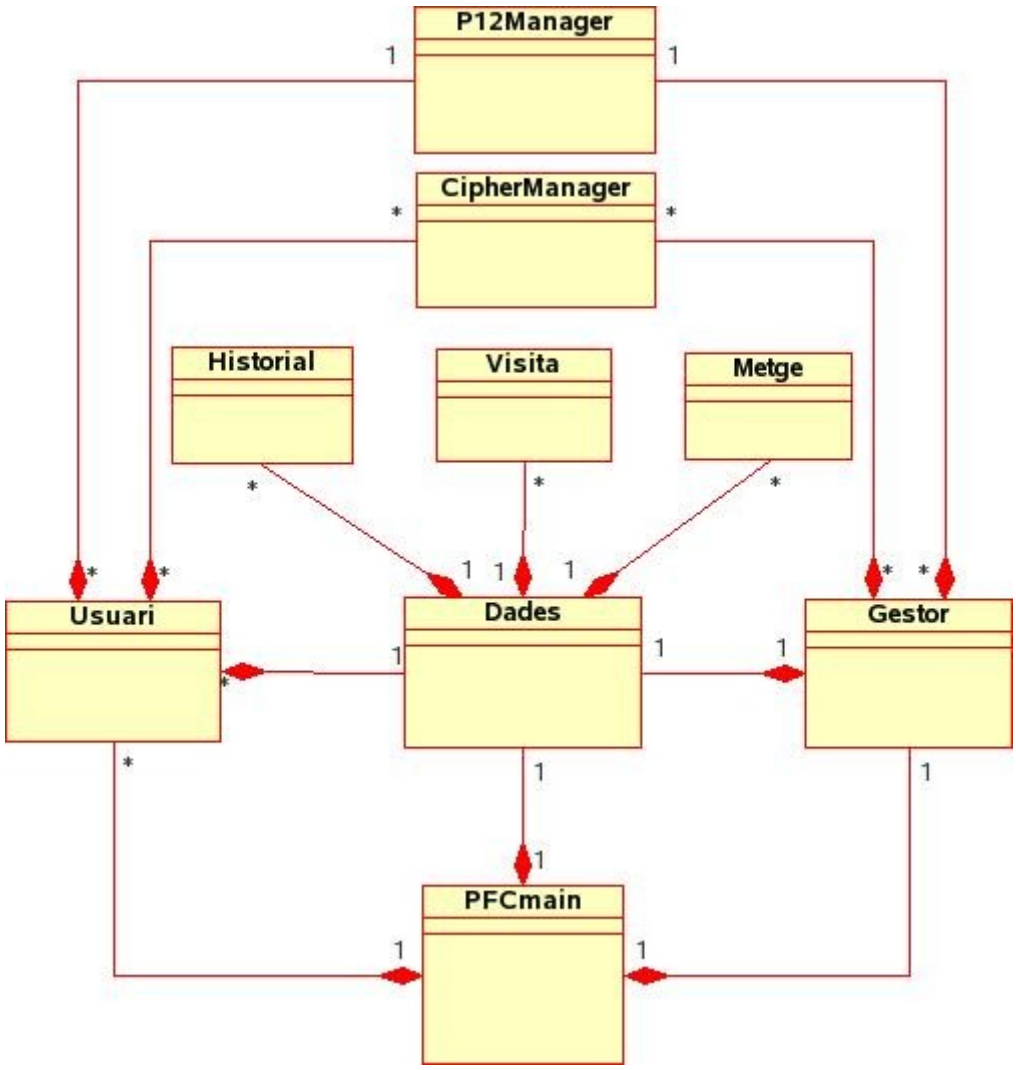


Figura 3.10.1: Diagrama de classes de l'esquema criptofràfic.

Alguns comentaria sobre el diagrama són els següents:

- La classe Dades fa una funció de magatzem de dades, i pot instanciar més d'un Historial, o Visita, o Metge. Encapsula les dades que estaran en la BD quan aquesta s'implementi.

- La classe P12Manager és el contenidor del PKCS#12 [16] de cada usuari, sigui metge, pacient o gestor del sistema. Cada pacient, o metge, o inclús el gestor tenen només una instància d'aquesta classe.

- La classe CipherManager és utilitzada per metges, pacients i gestor per a realitzar totes les tasques de xifrat i desxifrat. Tot i que només utilitzen una instància d'aquesta classe cada actor, podrien fer servir més d'una si calgués per realitzar diferents operacions de xifrat a la vegada.

- Per últim, la classe PFCmain, instancia els actors del sistema i les dades del sistema, que de moment s'encapsulen a la classe Dades, i llança totes les operacions que verifiquen l'esquema criptogràfic. Aquesta classe té quatre possibilitats d'execució:

  - Amb el número **"0"**: executa el protocol d'autenticació exclusivament.

  - Amb el número **"1"**: executa la consulta de les dades generals d'un pacient, i seguidament la consulta d'una visita del pacient.

  - Amb el número **"2":** executa el protocol de consulta dels pacients assignats a un metge.

  - Amb el número **"3"**: Executa el protocol de consulta de les dades generals de l'historial del pacient, i seguidament el protocol d'afegir una visita a l'historial del pacient.

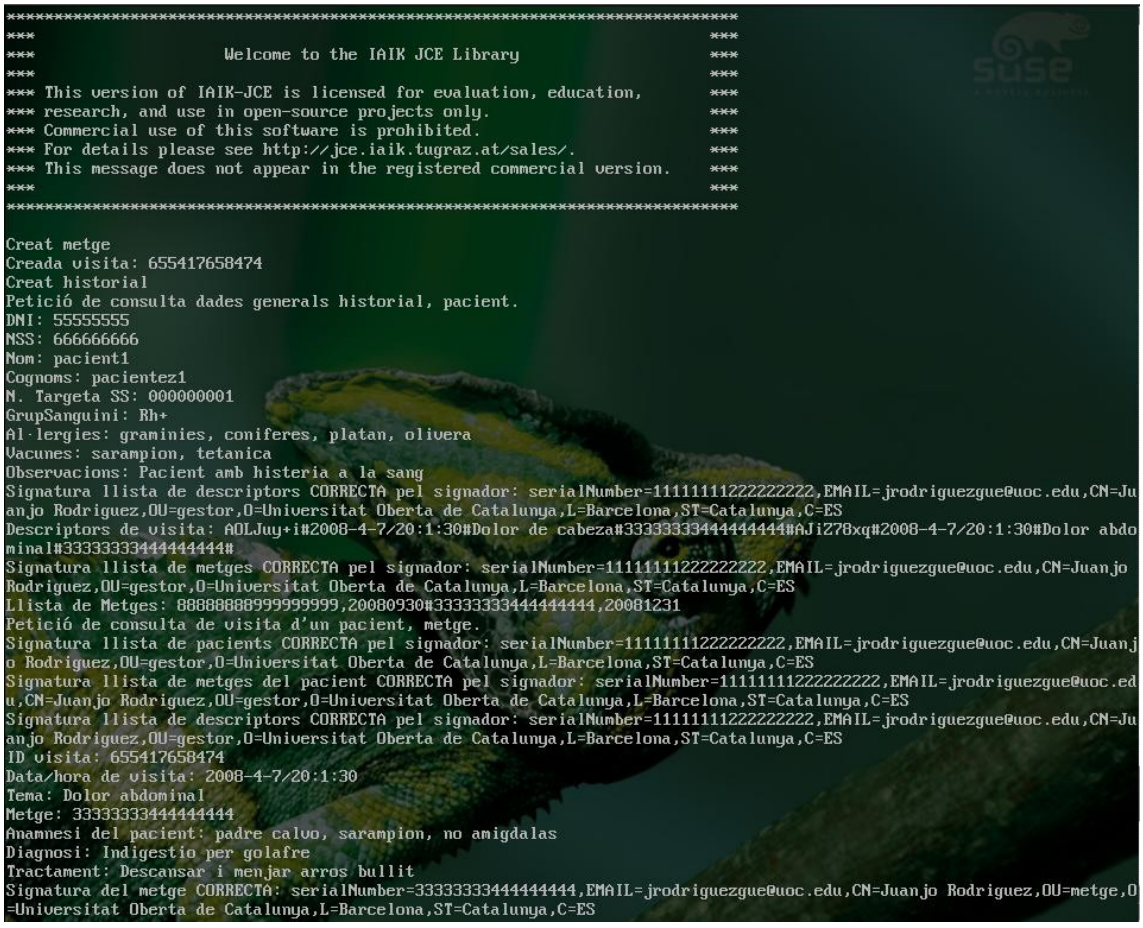
3.11. Proves realitzades.

Per a exemplificar les proves realitzades, i com he explicat anteriorment, s'ha construit la classe PFCmain que s'encarrega d'instanciar les dades i executar les crides als mètodes necessaris per a executar els protocols especificats. Seguidament es mostra les sortides de l'execució de cadascuna de les opcions esmentades en el punt anterior:

- Execució de l'aplicació amb opció "**0**":

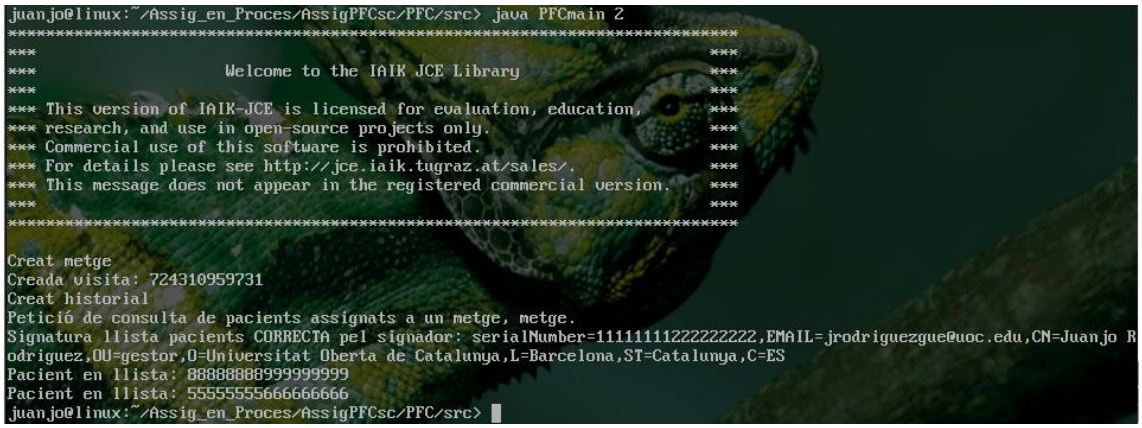
Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

- Execució de l'aplicació amb opció "**1**":

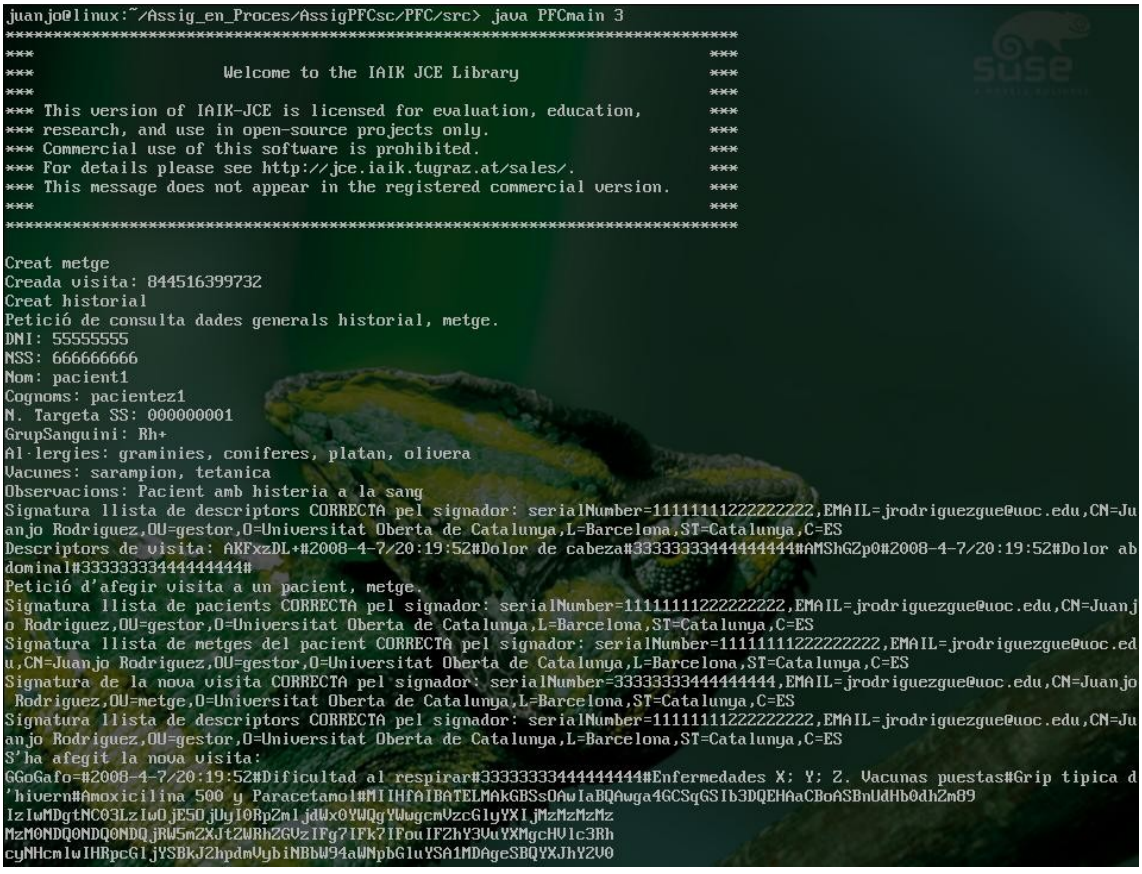


- Execució de l'aplicació amb opció "**2**":

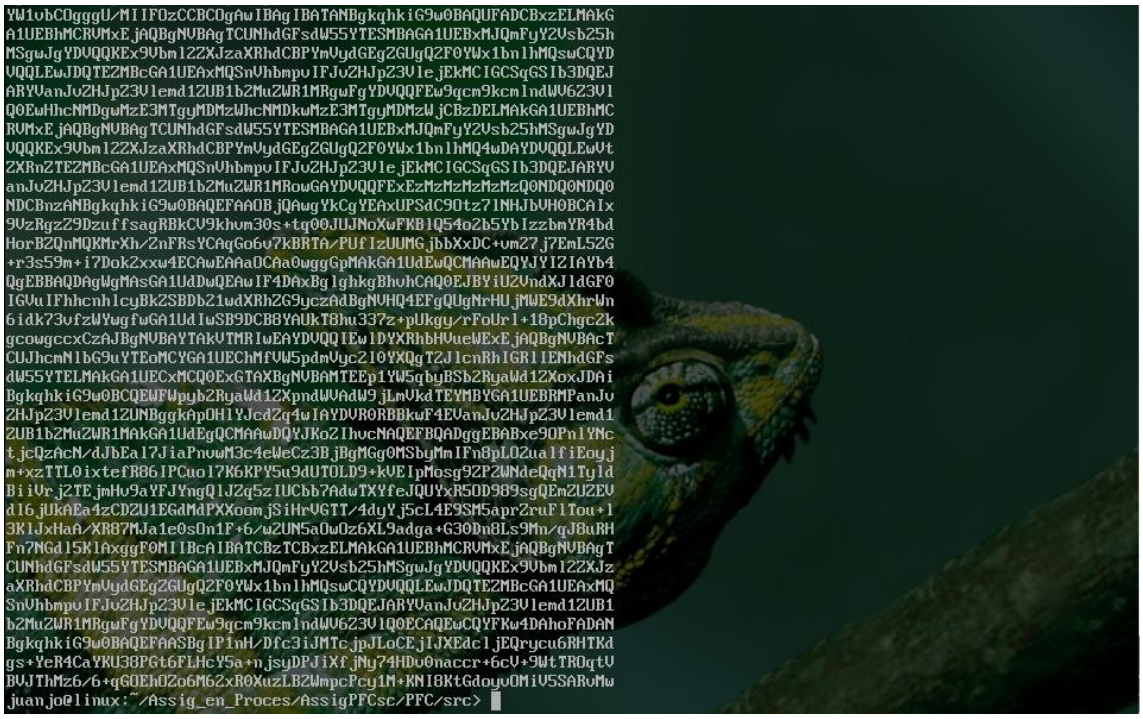
Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.



- Execució de l'aplicació amb opció “**3**”:



Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.



Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

4. Representació de les dades: XML.

4.1. Introducció.

XML és l'acrònim de eXtensible Markup Language [12]. Des de l'aparició d'aquest estàndard de W3C (World Wide Web Consoritum) [12], el seu ús a Internet com a mitjà per a la transmissió d'informació s'ha adoptat majoritàriament per a aquesta finalitat.

En aquest Projecte de Fi de Carrera s'utilitza l'XML [12] per a les transferències d'informació entre els pacients i el gestor del sistema per a totes les operacions que aquests poden fer, i entre els metges i el gestor del sistema per a les seves operacions. Tots els missatges entre aquests dos actors i el gestor del sistema aniran encapsulats dins de documents XML [12]. Igualment s'aplicarà aquesta tecnologia per a l'autenticació dels usuaris, o protocol 1 en el capítol anterior.

Es dissenyarà un document model que s'utilitzarà en cada cas, i que només transportarà en cadascuna de les fases de la comunicació entre l'usuari i el gestor del sistema, aquelles dades que es necessiten en cada moment per al desenvolupament del protocol que s'estigui duent a terme.

Per al desenvolupament d'aquesta fase del projecte s'utilitzarà l'API de Java JDOM (Java Document Object Model) [9]. El fet d'utilitzar la tecnologia XML [12] per a l'intercanvi de missatge entre les diferents parts de l'aplicatiu i els actors del sistema, facilitarà la integració de l'aplicació amb una altra, si en el futur es volgués fer.

4.2. Estructura dels documents XML.

Per a implementar la comunicació de les diferents parts de l'aplicació, amb vistes als propers capítols del Projecte, com són la comunicació mitjançant RMI [7] i la implementació de la BD on s'emmagatzemaran les dades de l'aplicació, necessitarem els següents documents XML [12] que seguidament passo a detallar.

El document que s'utilitzarà per a l'intercanvi de missatges entre el gestor i l'usuari, sigui pacient o metge, serà el document XML [12] genèric que tindrà la forma i estructura següent i que implementam mitjançant la classe de Java anomenada "XMLContainer". Aquest document encapsularà, com he dit, els missatges que s'intercanviaran els diferents actors del sistema.

"XMLContainer":

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE EncryptDocument SYSTEM "encdoc.dtd">
<EncryptDocument>
   <Document>
      <Title/>
```

```
      <Data/>
   </Document>
</EncryptDocument>
```

Pero abans d'explicar els camps d'aquest document XML [12] diré que els possibles missatges que s'intercanviaran els diferents actors del sistema, descrits en el capítol de l'esquema criptogràfic i segons la notació que es descriu en aquest mateix capítol, són els següents:

- *Pg (Nu, Idu)*: present en tots els protocols descrits.
- *Pu (Nu, Ng, Idg)*: present en tots els protocols descrits.
- *Pg (Ng, Consulta_dades_generals, Idusuari)*: present en el protocol 2 de consulta de dades generals d'un pacient.
- *Pu (H)*: present en protol 2 de consulta de dades generals d'un pacient.
- *Pg (Ng, Consulta_visita, Idusuari, descriptor_de_visita)*: present en el protocol 3 de consulta d'una visita d'un pacient.
- *Pu (V)*: present en el protocol 3 de consulta d'una visita d'un pacient.
- *Pg (Ng, Llista_pacients)*: present en el protocol 4 de consulta dels pacients assignats a un metge.
- *Pu (Llista_pacients_protegida)*: present en el protocol 4 de consulta dels pacients assignats a un metge.
- *Pg (Ng, Afegir_visita, Idusuari, V, Sm (V))*: present en el protocol 5 per a afegir una nova visita a l'historal d'una pacient.
- *Pg (Ng)*: present en el protocol 1 d'autenticació.

Dit això, el camps que poseeix el document XML "**EncryptDocument**" són:

- **<EncryptCocument>**: és l'arrel del document XML [12].
- **<Document>**: és l'element principal i únic que encapsula els camps del document XML [12].
- **<Title>**: aquest camp portarà en clar i segons la nomenclatura especificada en el capítol de l'esquema criptogràfic, la descripció del missatge que s'emmagatzemarà en Base64 en el camp de **<Data>**.
- **<Data>**: conté el missatge que es vol fer arribar a l'interlocutor codificat en Base64, ja que aquest missatge estarà, com em vist en la relació anterior, sempre xifrat amb la clau pública del receptor.

A més d'aquest document XML [12] genèric, per a l'intercanvi de missatges entre els actors del sistema s'han dissenyat una sèrie de documents XML [12] per a diferents propòsits, segons l'estructura d'informació detallada en el capítol de l'esquema criptogràfic. Aquests documents fan possible la implementació de les diferents llistes del model de dades, així com els contenidors per a l'intercanvi de registres de dades, i dels missatges que es transmeten els actors del sistema.

Aquests documents són els següents:

1. El primer d'aquests altres documents XML [12] que es faran servir en l'aplicació és el document “XMLMissatge”, el qual contindrà un missatge de comunicació entre actors, però sense xifrar. És a dir, el missatge concret que es passarà a l'actor receptor en el procés de comunicació, primer serà construit amb aquest document i després es xifrarà i encapsularà en un document “XMLContainer” (EncryptDocument) anteriorment descrit.

“XMLMissatge”:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Missatge SYSTEM "missatge.dtd">
<Document>
   <Missatge>
      <Title/>
      <Field>
         <Name/>
         <Value/>
      </Field>
      <Field>
         <Name/>
         <Value/>
      </Field>
      ...
   </Missatge>
</Document>
```

Els camps que poseeix el document XML “**Missatge**” són:

- **<Document>**: és l'arrel del document XML [12].
- **<Missatge>**: és l'element principal i únic que encapsula els camps del document XML [12] i que ens dona el tipus de document.
- **<Title>**: aquest camp portarà en clar i segons la nomenclatura especificada en el capítol de l'esquema criptogràfic, la descripció del missatge que s'emmagatzemarà en Base64 en el camp de **<Field>**.
- **<Field>**: aquest element té dos subcamps, i encapsula un element del missatge que es vol transmetre. Poden haver un o més d'aquests camps compostos en el document “XMLMissatge”.
- **<Name>**: és el nom de l'element del missatge que s'enviarà.
- **<Value>**: conté, en Base64, el valor de l'element del missatge que es transmetrà.

2. Altre d'aquests documents XML [12] que es faran servir en l'aplicació és el document “XMLLlistaP”. Aquest document s'utilitzarà per a representar

una llista de pacients del model de dades que seran part de les dades del metge.

"XMLLlistaP":

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE LlistaP SYSTEM "llistaP.dtd">
<Document>
   <LlistaP>
      <Title/>
      <Pacient/>
      <Pacient/>
      ...
      <Signature>
   </LlistaP>
</Document>
```

Els camps que poseeix el document XML "**LlistaP**" són:

- **<Document>**: és l'arrel del document XML [12].
- **<LlistaP>**: és l'element principal i únic que encapsula els camps de la llista de pacients.
- **<Title>**: aquest camp portarà en clar el títol de la llista de pacients.
- **<Pacient>**: és l'element que conté l'identificador d'un dels pacients del metge. Poden haver un o més d'aquests camps en el document "XMLLlistaP".
- **<Signature>**: aquest camp contindrà la signatura del document en Base64, signatura que com s'ha especificat en l'esquema criptogràfic, està realitzada pel gestor del sistema.

3. El següent document XML [12] és l'"XMLLlistaM". Aquest document encapsularà una llista de metges, la qual serà part d'un historial d'un pacient i especificarà els metges que tenen accés a l'historial i fins quan hi tenen accés.

"XMLLlistaM":

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE LlistaM SYSTEM "llistaM.dtd">
<Document>
   <LlistaM>
      <Title/>
      <Metge/>
         <Id/>
         <Data/>
```

```
      </Metge>
      ...
      <Signature>
   </LlistaM>
</Document>
```

Els camps que poseeix el document XML "**LlistaM**" són:

- **<Document>**: és l'arrel del document XML [12].
- **<LlistaM>**: és l'element principal i únic que encapsula els camps de la llista de metges.
- **<Title>**: aquest camp portarà en clar el títol de la llista de metges.
- **<Metge>**: és l'element que encapsula l'identificador del metge i la data fins a la qual tindrà accés a l'expedient. Poden haver un o més d'aquests elements a la llista de metges "XMLLlistaP".
- **<Id>**: és l'identificador del metge.
- **<Data>**: és la data límit fins a la qual tindrà accés a l'expedient del pacient.
- **<Signature>**: aquest camp contindrà la signatura del document en Base64, signatura que com s'ha especificat en l'esquema criptogràfic, està realitzada pel gestor del sistema.

4. El document següent encapsularà una llista de descriptors de visita, la qual serà també part de l'historial del pacient. El nom d'aquest document serà "XMLLlistaDV", i tindrà l'estructura següent.

"XMLLlistaDV":

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE LlistaDV SYSTEM "llistaDV.dtd">
<Document>
   <LlistaDV>
      <Title/>
      <Descriptor/>
         <Id/>
         <Data/>
         <Hora/>
         <Tema/>
         <Metge/>
      </Descriptor>
      ...
      <Signature>
   </LlistaDV>
</Document>
```

Els camps que poseeix el document XML "**LlistaDV**" són:

- **<Document>**: és l'arrel del document XML [12].
- **<LlistaDV>**: és l'element principal i únic que encapsula els camps de la llista de descriptors de visites.
- **<Title>**: aquest camp portarà en clar el títol de la llista de descriptors de visita.
- **<Descriptor>**: és l'element que encapsula el descriptor de visita pròpiament dit. Poden haver un o més d'aquests elements a la llista de descriptors de visita "XMLLlistaDV".
- **<Id>**: és el número aleatori que fa d'identificador de la visita a la qual fa referència.
- **<Data>**: és la data de la visita.
- **<Hora>**: és l'hora de la visita.
- **<Tema>**: és el tema de la visita.
- **<Metge>**: és l'identificador del metge que ha fet la visita.
- **<Signature>**: aquest camp contindrà la signatura del document en Base64, signatura que com s'ha especificat en l'esquema criptogràfic, està realitzada pel gestor del sistema.

5. L'últim d'aquests documents que s'utilitzaran en l'aplicació serà utilitzat per a compartir registres de dades complerts, com ara un historial, que un metge o pacient vulguin consultar. Parlem del document "XMLRegistre" que té l'estructura següent.

"XMLRegistre":

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Registre SYSTEM "registre.dtd">
<Document>
   <Registre>
      <Title/>
      <Field/>
         <Name/>
         <Type/>
         <Size/>
         <Value/>
      </Field>
      ...
      <Signature>
   </Registre>
</Document>
```

Els camps que poseeix el document XML "**Registre**" són:

- **<Document>**: és l'arrel del document XML [12].

- **<Registre>**: és l'element principal i únic que encapsula els camps del registre.
- **<Title>**: aquest camp portarà en clar el títol del registre que encapsula.
- **<Field>**: és l'element que encapsula un dels camps amb la seva descripció i valor corresponent. Poden haver tants elements <Field> com camps tingui el registre encapsulat en el document "XMLRegistre".
- **<Name>**: és el nom del camp.
- **<Type>**: és el tipus del camp.
- **<Size>**: és el tamany o longitud del camp.
- **<Value>**: és el valor en Base64 del camp.
- **<Signature>**: aquest camp contindrà, o no, la signatura del registre, com ara pot ser la signatura del metge d'una visita encapsulada en aquest document. La signatura, al igual que en tots els camps del document estarà codificada en Base64.

4.3. DTDs dels documents XML

En aquest apartat especifico els DTD de cadascun dels documents XML [12] especificats en l'apartat anterior. Els DTD o "*Document Type Definition*" tenen el propòsit de definir la legalitat i l'estructura del blocs XML [12] que composen el document concret.

Els DTDs dels nostres documents de l'apartat anterior, en ordre d'aparició, són:

1. DTD del document "**XMLContainer**", anomenat "encdoc.dtd":

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!ELEMENT EncryptDocument (Document)>
<!ELEMENT Document (Title, Data)>
<!ELEMENT Title (#PCDATA)>
<!ELEMENT Data (#PCDATA)>
```

2. DTD del document "**XMLMissatge**", anomenat "missatge.dtd":

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!ELEMENT Document (Missatge)>
<!ELEMENT Missatge (Title, Field+)>
<!ELEMENT Field (Name, Value)>
<!ELEMENT Title (#PCDATA)>
<!ELEMENT Name (#PCDATA)>
<!ELEMENT Value (#PCDATA)>
```

3. DTD del document "**XMLLlistaP**", anomenat "llistaP.dtd":

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!ELEMENT Document (LlistaP)>
<!ELEMENT LlistaP (Title, Pacient+, Signature)>
<!ELEMENT Title (#PCDATA)>
<!ELEMENT Pacient (#PCDATA)>
<!ELEMENT Signature (#PCDATA)>
```

4. DTD del document "**XMLLlistaM**", anomenat "llistaM.dtd":

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!ELEMENT Document (LlistaM)>
<!ELEMENT LlistaM (Title, Metge+, Signature)>
<!ELEMENT Metge (Id, Data)>
<!ELEMENT Title (#PCDATA)>
<!ELEMENT Id (#PCDATA)>
<!ELEMENT Data (#PCDATA)>
<!ELEMENT Signature (#PCDATA)>
```

5. DTD del document "**XMLLlistaDV**", anomenat "llistaDV.dtd":

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!ELEMENT Document (LlistaDV)>
<!ELEMENT LlistaDV (Title, Descriptor+, Signature)>
<!ELEMENT Descriptor (Id, Data, Hora, Tema, Metge)>
<!ELEMENT Title (#PCDATA)>
<!ELEMENT Id (#PCDATA)>
<!ELEMENT Data (#PCDATA)>
<!ELEMENT Hora (#PCDATA)>
<!ELEMENT Tema (#PCDATA)>
<!ELEMENT Metge (#PCDATA)>
<!ELEMENT Signature (#PCDATA)>
```

6. DTD del document "**XMLRegistre**", anomenat "registre.dtd":

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!ELEMENT Document (Registre)>
<!ELEMENT Registre (Title, Field+, Signature)>
<!ELEMENT Field (Name, Type, Size, Value)>
<!ELEMENT Title (#PCDATA)>
<!ELEMENT Name (#PCDATA)>
<!ELEMENT Type (#PCDATA)>
<!ELEMENT Size (#PCDATA)>
<!ELEMENT Value (#PCDATA)>
<!ELEMENT Signature (#PCDATA)>
```

4.4. Funcionament de la representació de dades mitjançant XML.

En el disseny d'aquests documents s'ha considerat, a excepció de les llistes, que seran purament contenidors de dades per a poder comunicar-se entre els diferents actors del sistema. És a dir, qualsevol missatge que s'intercanviïn els pacients o metges amb el gestor del sistema serà encapsulat dins d'un document XML [12] genèric.

Aquests documents, a excepció de les llistes de pacients, de metges, i de descriptors de visites, no s'emmagatzemaran a la Bases de dades quan aquesta s'implementi, ja que el seu objectiu és comunicatiu i temporal.

Com s'ha especificat en l'apartat 4.2, els missatges que es passen els actors del sistema, són missatges xifrats amb la clau pública del receptor i, per tant, quan aquests són rebuts pel destinatari, els desxifra i tracta segons el protocol que s'estigui executant. Així, el que guardarem a la Base de Dades seran parts de les dades que s'intercanvien els actors encapsulades i xifrades en aquests documents, i no els documents propis, exceptuant, com he dit, les llistes protegides de metges i de descriptors de visites en l'historial, i les de pacients en el registre del metge.

Les diferents parts de l'aplicatiu construiran els documents XML [12] de tipus “XMLMissatge”, segons el missatge que hagin d'intercanviar corresponent al protocol que s'estigui executant en cada moment. Aquest document XML [12] tindrà les dades necessàries del pas concret que s'està executant. Un cop construit, aquest document serà xifrat amb la clau pública del receptor. El resultat serà codificat en Base64 i encapsulat en un document XML [12] tipus “XMLContainer”. Llavors serà enviat al receptor.

Aquesta forma de procedir té un procés doble, que com he dit consta de la construcció de l'XML [12] missatge, i un cop xifrat es construeix el document XML [12] contenidor. S'ha considerat aquesta forma de fer-lo perquè el missatge que es transmet en cada moment és un missatge compacte, amb diferents dades, i l'estructura d'un XML [12] complet ens permet tenir aquestes dades perfectament controlades i separades per al seu posterior tractament. Per contra requereix una mica més de feina en el procés dels missatges, que queda totalment recoltzada i implementada gràcies a l'API de Java JDOM [9].

Tots els documents especificats en els punts anteriors tenen les seves classes de Java corresponents als noms que s'han descrit: “XMLContainer.java”, “XMLMissatge.java”, “XMLLlistaP.java”, “XMLLlistaM.java”, “XMLLlistaDV.java”, “XMLRegistre.java”. Cadascuna té els seus mètodes de construcció del document i retorn del document construit, així com de càrrega i retorn dels valors que encapsula el document. També tenen la possibilitat de carregar un document XML [12] a partir d'una cadena tipus String, utilitzant la classe “SAXBuilder”.

Quant a les funcions de codificació i decodificació a Base64 de les dades contenides en els documents, aquestes estan totalment recoltzades per la nostra classe “CipherManager.java” que ja s'ha especificat en l'apartat de l'esquema criptogràfic.

4.5. Diagrama de classes de la representació de dades mitjançant XML.

El diagrama següent mostra les classes que ja teniem, afegint ara al diagrama amb les classes que hem especificat en el capítol de l'esquema criptogràfic, les classes que permeten la implementació de la comunicació mitjançant XML [12], i la implementació de les llistes corresponents.

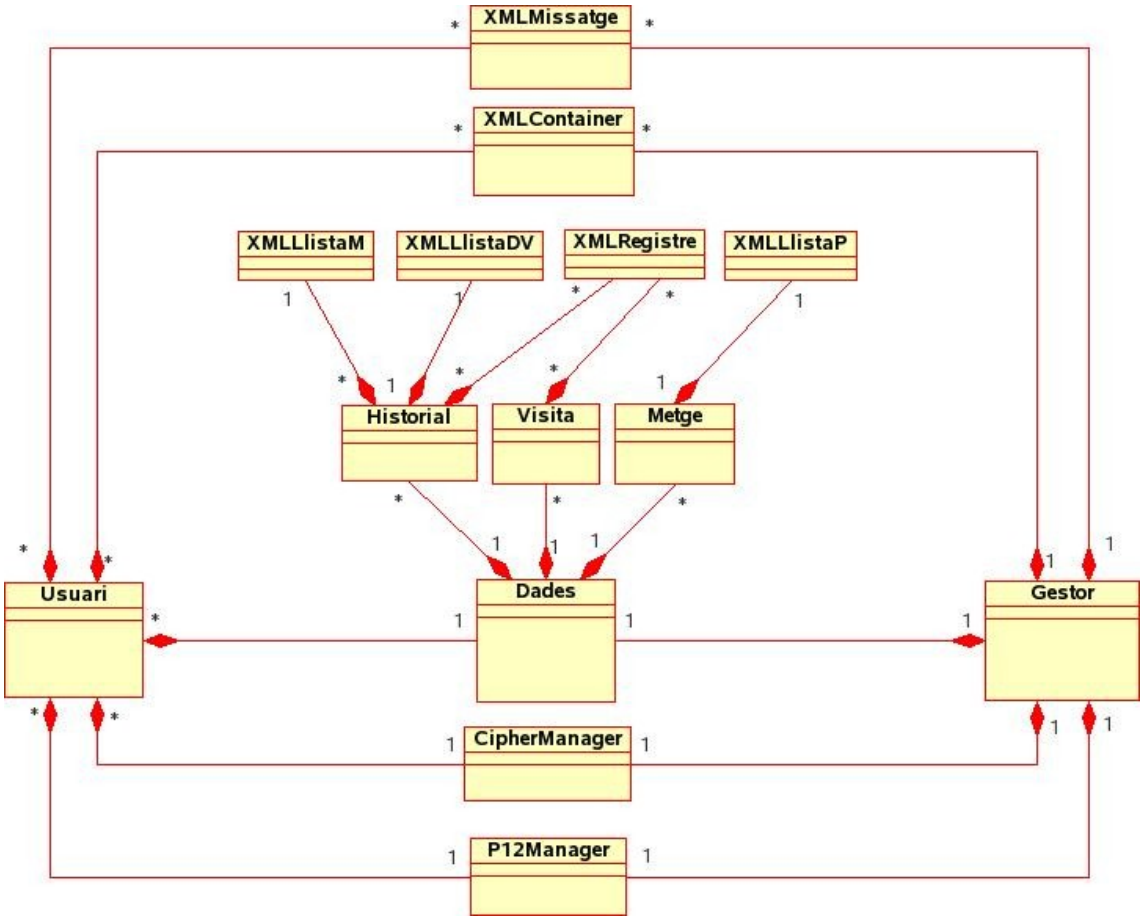


Figura 4.5.1: Diagrama de classes amb representació de dades XML [12].

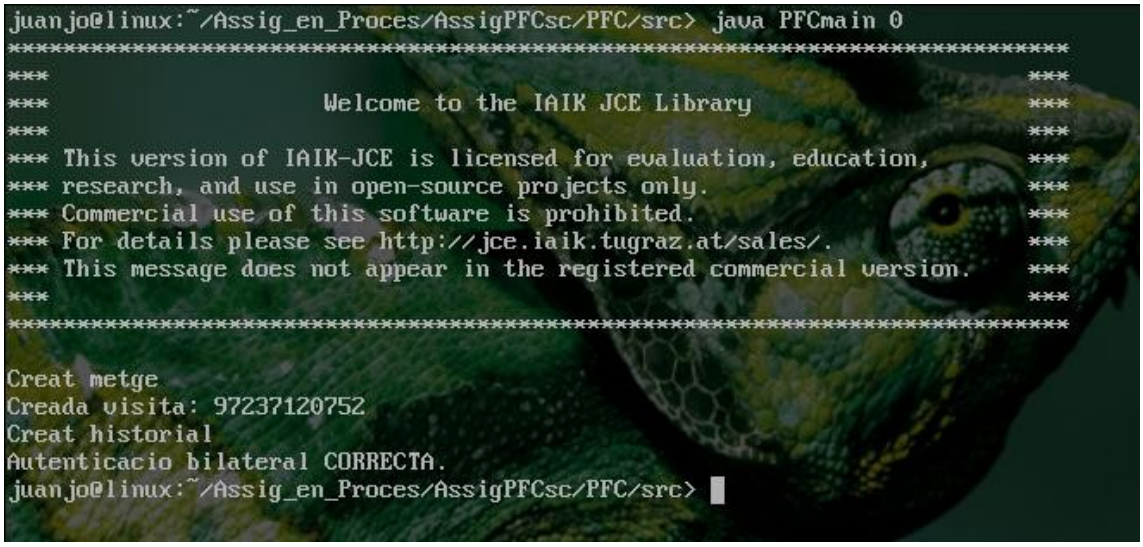
Com es pot veure al diagrama, les classes “XMLContainer” i “XMLMissatge” són les que fan servir els actors del sistema, el gestor i els pacient o els metges, instanciats els dos últims per la classe “Usuari”. Les altres classes implementen les llistes i un registre genèric que pot ser un historial, o una visita.

Aquests registres o llistes són de vegades, segons el protocol que s'estigui executant, encapsulades dins d'un missatge XML [12] i després xifrat i encapsulat dins d'un contenidor XML [12] de tipus “XMLContainer”

4.6. Joc de proves.

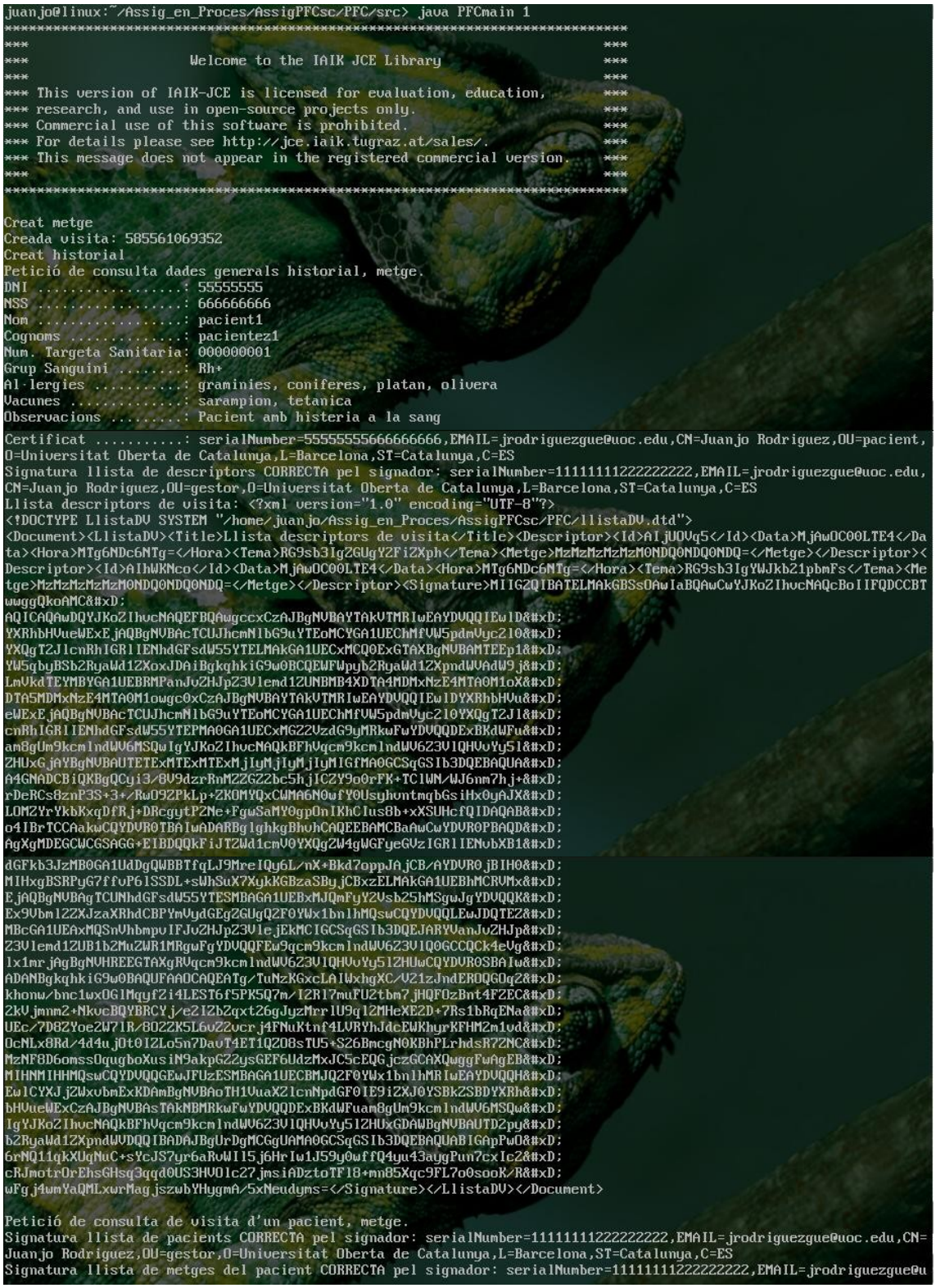
Com en el capítol de l'esquema criptogràfic, exemplificaré les proves realitzades amb la ja coneguda classe PFCmain que s'encarrega d'instanciar les dades i executar les crides als mètodes necessaris per a cadascun dels protocols que s'han implementat. Seguidament es mostra les sortides de l'execució de cadascuna de les opcions esmentades en el punt anterior:

- Execució de l'aplicació amb opció “**0**”, la qual executa el protocol d'autenticació de forma exclusiva (en l'execució d'aquest protocol no es pot apreciar l'ús de XML [12] en la comunicació dels actors):



- Execució de l'aplicació amb opció “**1**”, la qual executa el protocol que implementa la consulta de dades generals d'un pacient per part del pacient, o d'un metge, i seguidament també executa el protocol de consulta d'una visita per part del mateix usuari:

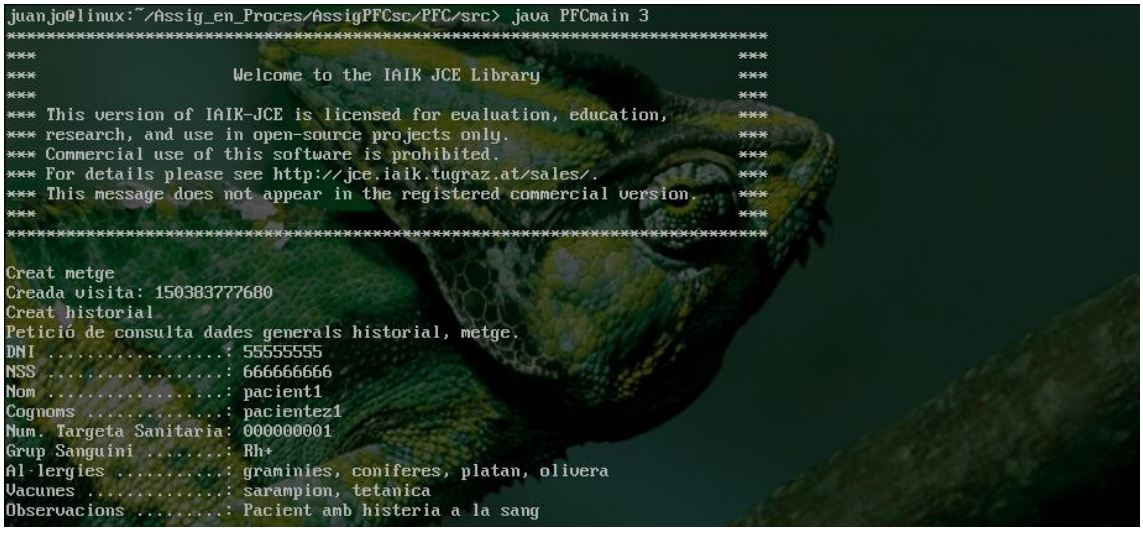
Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.



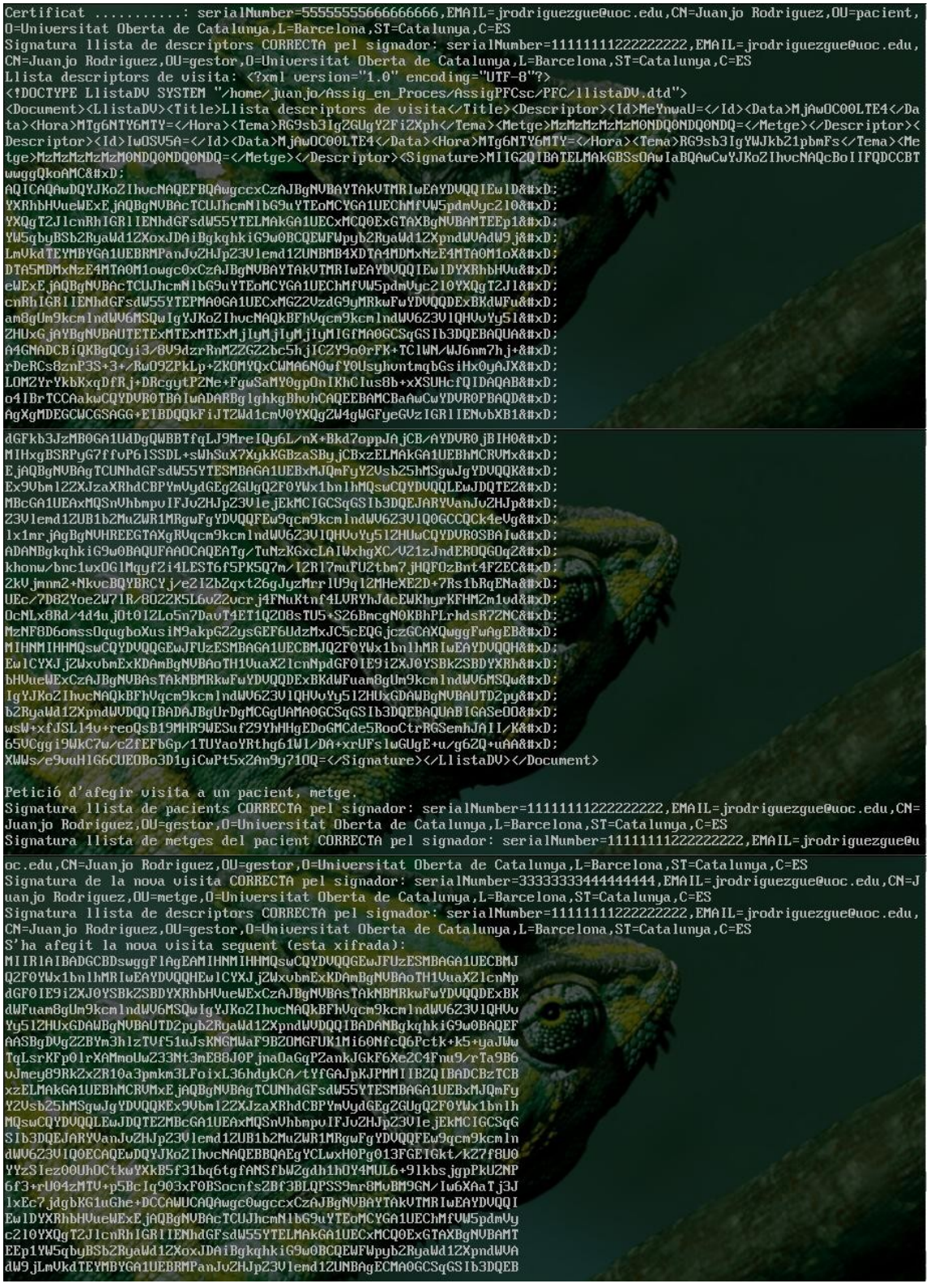
Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

- Execució de l'aplicació amb opció "**2**", la qual executa el protocol de consulta dels pacients assignats a un metge:

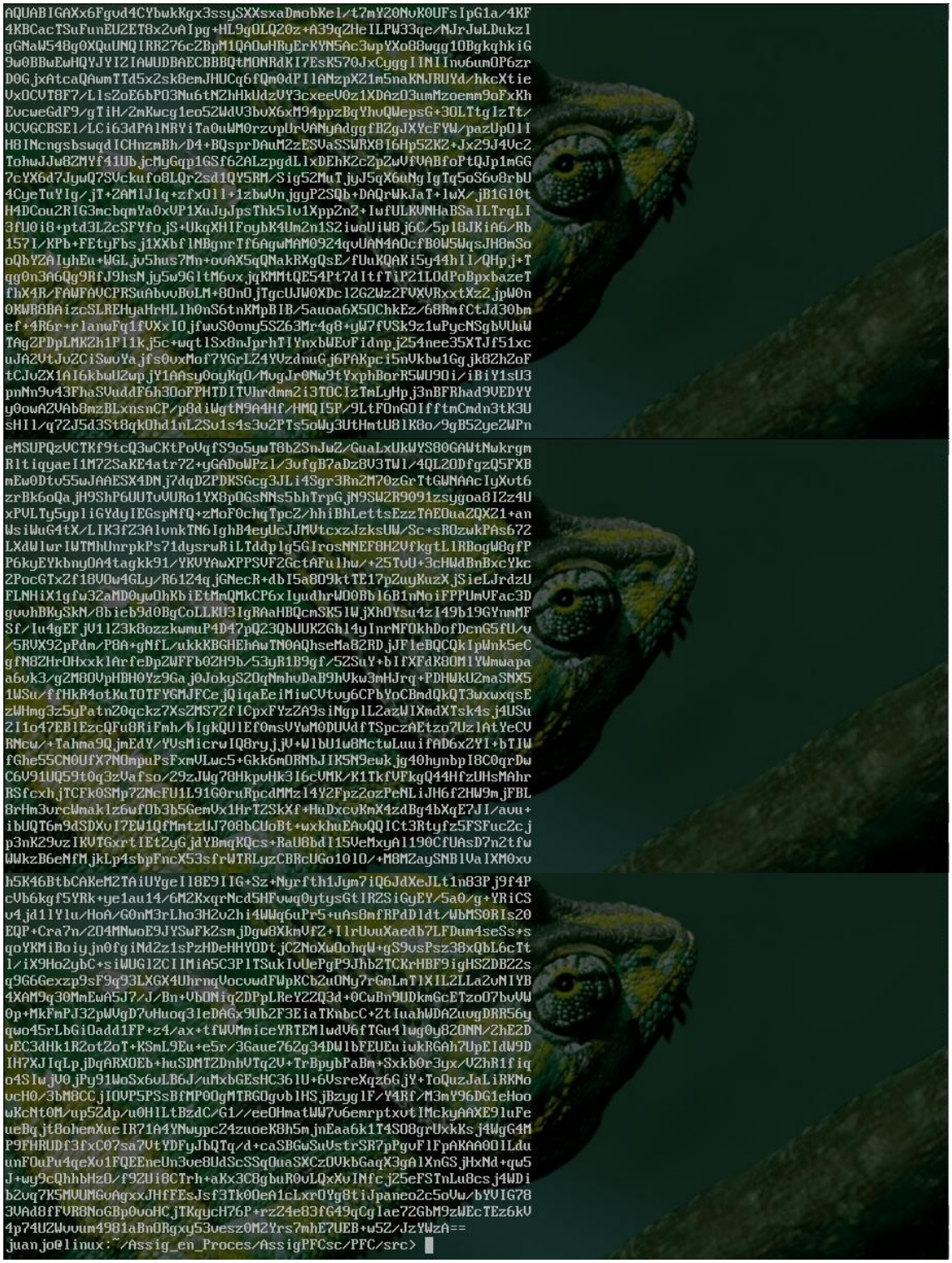
- Execució de l'aplicació amb opció "**3**", la qual torna a executar la consulta de les dades generals de l'historial d'un pacient, així s'obté la llista de descriptors de visita, i seguidament executa el protocol per a afegir una visita al mateix historial:



Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.



Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.



Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

5. Comunicació dels components: RMI.

5.1. Introducció.

La comunicació entre els diferents components de l'aplicació és una part essencial del Projecte. Tant els pacients com els metges han de poder comunicar-se i enviar les seves peticions de servei al gestor del sistema, i aquest últim ha de poder respondre a aquestes peticions amb els missatges corresponents. Això suposaria el disseny d'un protocol de comunicació entre les parts del sistema que implementés un mecanisme de transport d'aquests missatges i les seves corresponents respostes. Per evitar una sobrecàrrega de feina i donat que la part essencial és l'esquema criptogràfic, s'ha optat per utililtzar la tecnologia RML (Remote Method Invocation).

Java [6] incorpora aquesta tecnologia a la seva API estàndard. RMI [7] consta d'un servidor on s'executen diferents instàncies de les classes servidores que es necessiten. Les aplicacions que volen emprar els mètodes remots únicament necessiten saber la interfície del servidor, és a dir, els mètodes que ofereix la classe que està al servidor. La implementació d'aquesta interfície està oculta i el client no arriba mai a saber què és el que s'està executant.

Aquesta tecnologia de comunicació redueix notablement el temps de desenvolupament de les diferents parts de l'aplicatiu.

5.2. Funcionament de la comunicació amb RMI.

En la comunicació RMI [7] el servidor estableix un contracte amb les aplicacions remotes. Això vol dir que el servidor informa de les classes i mètodes que estaran disponibles per a l'ús per part d'aquestes aplicacions, mitjançant un servei de directori, l'rmiregistry. La forma d'aconseguir això passa per la creació d'unes interfícies que tothom conegui, les quals estan implementades pel servidor.

Quan la classe està implementada, s'han de crear els punts de connexió que utilitzaran els aplicatius remots. El resultat d'aquest procediment és la creació de dues classes que els clients han de conèixer. Aquestes classes es coneixen amb els noms d'Skeleton i Stub. Un cop realitzat això, les aplicacions remotes ja poden accedir a fer les instanciacions dels objectes remots. Aquests objectes remots són persistents.

L'stub és un representant local de l'objecte remot que està al servidor. Quan s'invoca un mètode de l'stub, aquest passa la petició a un representant remot, l'Skeleton, el qual invoca localment a l'objecte que implementa el mètode. Aquest objecte (Impl) o bé estarà esperant peticions, o bé serà activat automàticament en arribar una invocació. El resultat de la invocació segueix el mateix camí de tornada.

Per a l'objecte client, tot és gairebé invisible: ha invocat un mètode d'un objecte local (l'stub), i ha obtingut una resposta local. El que ha ocorregut entre stub client - stub servidor i objecte implementació és invisible per a ell.

5.3. Implantació d'RMI al sistema.

La utilització d'RMI [7] al sistema ha fet que l'ús de les dades quedi separat i centralitzat en el servidor, el qual serà qui accedirà a la Base de Dades que s'implementi en el capítol següent. D'aquesta manera, els usuaris remots, metges i pacients, només coneixeran els mètodes que poden cridar, però no sabran que és el que passa amb les dades en la màquina remota o servidor.

Per a la implantació de l'RMI [7] al sistema, s'ha creat un únic servidor. Per a la seva realització s'ha modificat la classe "Gestor" del nostre sistema per a convertir-la en un servidor RMI [7]. Aquesta classe implementa els mètodes que seran accessibles a tercers mitjançant la creació d'una interfície remota que hem anomenat "RemoteGestor", la qual inclou la descripció d'aquests mètodes accessibles a tercers.

Els mètodes que s'han publicat en aquesta interfície, implementats pel nostre servidor "Gestor", fan el tractament i la resposta dels missatges d'autenticació i les peticions que fan els usuaris al gestor en els protocols criptogràfics que ja s'han descrit. Aquests usuaris són els metges i els pacients, ambdos instanciats per la classe "Usuari".

L'ús d'RMI [7] redueix considerablement el temps de desenvolupament i la implementació de la comunicació entre el servidor i els clients de l'aplicació. A més, la portabilitat està totalment garantida entre diferents sistemes operatius, com ara Linux i Windows.

5.4. Diagrama de classes de la comunicació dels components.

El nou diagrama de classes un cop implementada la comunicació entre components del sistema mitjançant RMI [7] és el següent:

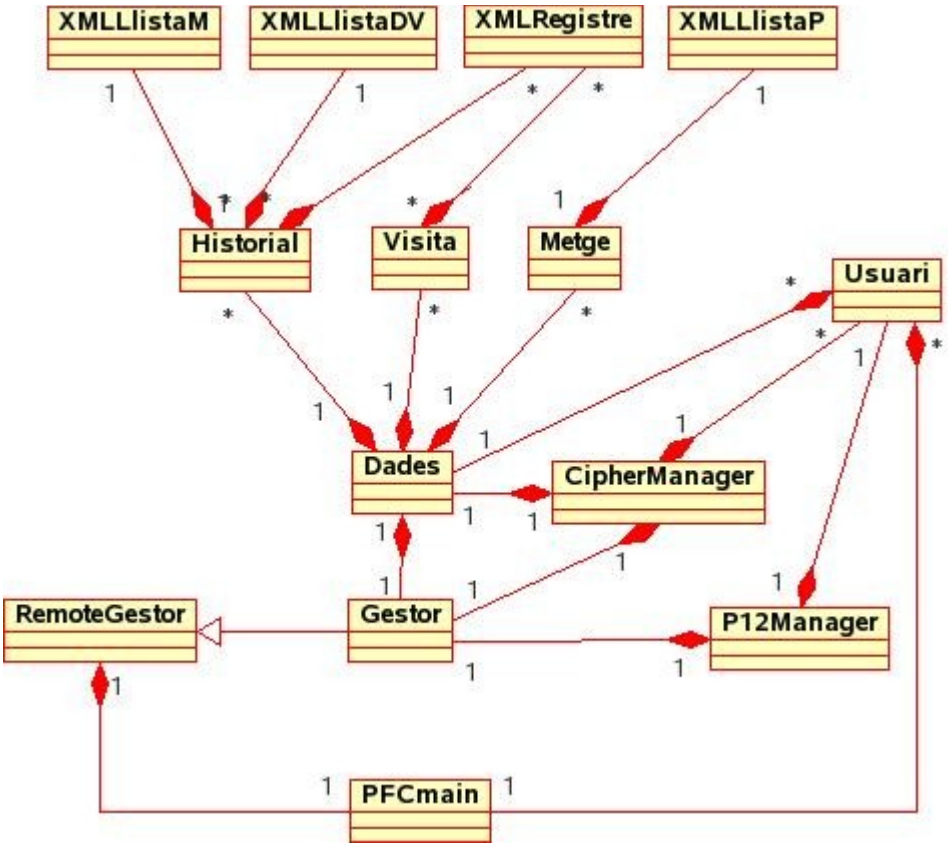


Figura 5.4.1: Diagrama de classes amb representació de dades XML [12] i comunicacions RMI [7].

5.5. Proves realitzades.

Les proves realitzades en aquest punt són exactament les mateixes que es poden veure en el punt 4.6 del capítol anterior. I les captures de pantalla que es poden veure en aquest punt són exactament les mateixes que produeixen les proves amb la implementació RMI [7]. Això és degut al fet que l'única cosa que ha canviat en l'aplicació és la transformació del la classe “Gestor” en un servidor RMI [7] que contesta a les preguntes que li fan els usuaris, pacients i metges.

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

6. Gestió de la informació: Base de Dades.

6.1. Introducció.

Fins a aquest punt les proves que s'han realitzat, de les diferents parts del Projecte que s'han enllestit, s'han fet utilitzant les dades que estan contenides en la classe “Dades” i que, cada vegada que es llança la classe que fa el testeig de les funcionalitats, “PFCmain”, es recreen per ser utilitzades en el test.

L'ús d'una Base de Dades permetrà l'emmagatzematge dels historials, les visites i els metges, les dades dels quals són la raó de ser de tot el Projecte.

El Sistema Gestor de Base de Dades que s'ha utilitzat per a aquesta finalitat és el sistema relacional MySQL [10]. Aquest SGBD relacional és programari de lliure distribució, així com la majoria del programari que s'ha utilitzat per al desenvolupament del Projecte. A més, podem aconseguir una versió per a Windows, per a Linux, o per a Mac OSX, cosa que permetrà utilitzar l'aplicatiu en qualsevol d'aquestes plataformes.

6.2. Utilitat de la Base de Dades.

La principal utilitat de la Base de Dades serà l'emmagatzematge de les dades que utilitza l'aplicació, ja siguin els historials mèdics amb les seves llistes de visites protegides i de metges autoritzats a accedir a l'historial, així com el certificat del propietari de l'historial i les visites amb la signatura del metge que les ha realitzades; així com les dades dels metges que estan registrats en el sistema amb la seva llista de pacients assignats i els seus certificats que els acrediten com a usuaris del sistema.

A més, la Base de Dades també guardarà dades temporals referents a una de les fases dels protocols criptogràfics que s'han implementat, i que utilitzarà el gestor del sistema per realitzar l'autentificació, i saber qui és l'usuari que fa la petició concreta, per així determinar si té accés o no a la informació que està demanant.

L'únic que podrà i accedirà a la Base de Dades és el gestor del sistema, el qual rebrà les peticions dels usuaris, farà les comprovacions adients i recuperarà o emmagatzemarà les dades pertinents.

S'ha de dir que amb la implementació de la Base de Dades, les classes que s'havien utilitzat fins al moment per al tractament de dades, “Dades”, “Metge”, “Historial”, “Visita”, han perdut la seva utilitat i per tant ja no s'utilitzaran.

6.3. Model de Base de Dades.

Per a la implementació de la Base de Dades que s'utilitzarà per a emmagatzemar la informació, s'utilitzarà una taula per a emmagatzemar les dades dels metges, una altra taula per a emmagatzemar les dades dels historials i una taula per a emmagatzemar les dades de les visites. S'ha

considerat el tenir una altra taula per emmagatzemar l'identificador del gestor i el seu certificat. Això s'ha fet perquè, tot i que només és un gestor l'encarregat de fer totes les transaccions en el sistema que demanen els usuaris mitjançant els protocols criptogràfics, podria haber més d'un usuari amb privilegis de fer aquestes operacions i és important tenir-los enregistrats en una taula de la Base de Dades. Per últim es crearà una altra taula amb la finallitat d'emmagatzemar part de la informació d'autenticació dels usuaris de forma temporal, tal i com s'especifica en el "Procedure 2" de protocol criptogràfic número 2 que s'ha especificat en el capítol de l'esquema criptogràfic.

Per tant el diagrama entitat-relació que surt és el següent:

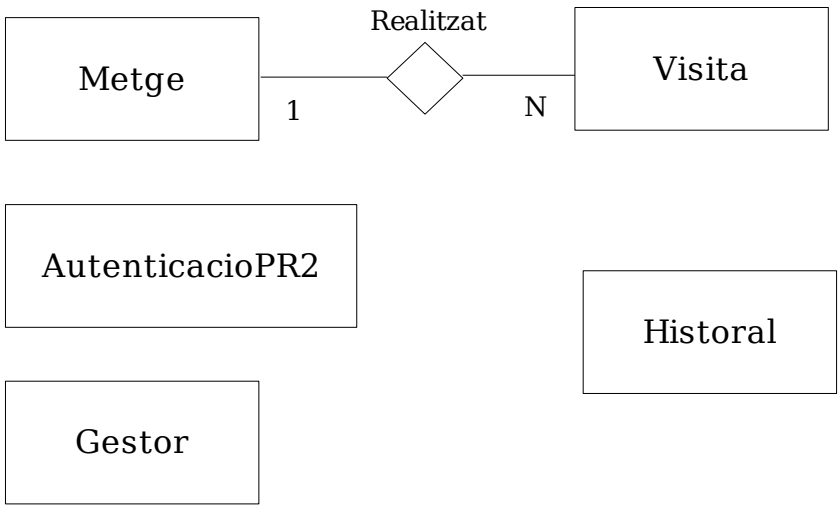


Figura 6.3.1: Diagrama Entitat-Relació de la Base de Dades.

D'aquest diagrama entitat-relació sorgeix el model relacional següent:

Visita (id_visita, datai, horai, tema, metge, anamnesi, diagnosi, tractament, signature)
on {metge} clau forana cap a Metge

Metge (dni_nss, nom, cognoms, ncolegiat, especialitat, certificate, llis_pac)

Historial (dni_nss, nom, cognoms, num_tars, grup_sang, alergies, vacunes, observacions, certificate, llis_vis, llis_met)

AutenticacioPR2 (num_user, num_gestor, user_cert)

Gestor (fingerprint, certificate)

6.4. Descripció de les taules de la Base de Dades.

La descripció de les taules que he definit en l'exemple anterior és la següent:

1. **Taula "Visita":**
   - **id_visita**: és un valor únic obtingut de forma aleatòria.
   - **datai**: és la data en la qual s'ha realitzat la visita.
   - **horai**: la hora en la qual s'ha realitzat la visita.
   - **tema**: és el tema del qual tracta la visita.
   - **metge**: és l'identificador del metge que realitza la visita.
   - **anamnesi**: part de la informació mèdica del pacient que pot ajudar en la visita.
   - **diagnosi**: diagnosi resultat d'aquesta visita realitzada.
   - **tractament**: tractament recomanat pel metge al pacient en aquesta visita.
   - **signature**: signatura digital del metge que ha realitzat la visita.

2. **Taula "Metge":**
   - **dni_nss**: número de DNI del metge seguit del seu número de seguretat social, ja que aquest és l'identificador que s'ha escollit per a identificar metges i pacients. Aquest identificador també anirà en el certificat del metge.
   - **nom**: nom del metge.
   - **cognoms**: cognoms del metge
   - **ncolegiat**: número de col·legiat del metge.
   - **especialitat**: especialitat mèdica.
   - **certificate**: certificat del metge.
   - **llis_pac**: llista de pacients del metge, protegida. Aquesta llista serà una llista XML [12] amb els números identificadors dels pacients, signada pel gestor i xifrada amb la clau pública del gestor i del metge.

3. **Taula "Historial":**
   - **dni_nss**: número de DNI seguit del número de seguretat social del pacient, els quals seran el seu identificador que també anirà en el seu certificat.
   - **nom**: nom del pacient.
   - **cognoms**: cognoms del pacient.

- **num_tars**: número de la targeta sanitària del pacient.
- **grup_sang**: grup sanguini del pacient.
- **alergies**: al·lèrgies rellevants que pugui tenir el pacient.
- **vacunes**: vacunes administrades al pacient.
- **observacions**: acumulacions d'observacions a tenir en compte del pacient d'aquest historial.
- **certificate**: certificat electrònic del pacient d'aquest historial.
- **llis_vis**: llista de visites protegides d'aquest historial. Les visites d'aquest pacient estaran referenciades pels descriptors de visita que estaran emmagatzemats dins d'aquesta llista, la qual serà un document XML [12] i estarà signada pel gestor i xifrada amb les claus públiques del gestor, del pacient, i dels metges que tenen accés a aquest historial.
- **llis_met**: conté un document XML [12] amb la llista de metges i la data final d'accés a aquest historial signada pel gestor del sistema, i xifrada amb la clau pública del gestor i la del pacient.

4. **Taula "AutenticacioPR2":** aquesta taula emmagatzemarà de forma temporal el número aleatori enviat per l'usuari, associat al número aleatori que generarà el gestor, i amb el certificat que identifica a l'usuari. Els seus atributs són:
   - **num_user**: és el número aleatori del usuari rebut pel gestor en el segon pas dels protocols criptogràfics.
   - **num_gestor**: és el número aleatori generat pel gestor del sistema en rebre el número de l'usuari i el seu identificador.
   - **user_cert**: és el certificat de l'usuari que l'identifica i que ha estat recuperat pel gestor.

5. **Taula "Gestor":** taula amb la finalitat de tenir registrats els possibles gestors del sistema, per a la nostra aplicació només treballarem amb un.
   - **fingerprint**: és l'identificador del gestor del sistema.
   - **certificate**: és el seu certificat digital.

6.5. Classe responsable de l'accés a la Base de Dades.

S'ha creat una classe que aglutina tots els mètodes necessaris d'accés a la Base de Dades que es necessiten per poder emmagatzemar, consultar i

actualitzar les dades d'aquesta Base de Dades. Aquesta classe es diu "DBManager".

Aquesta classe té també un mètode "main" que permet fer una crida a la classe per omplir la Base de Dades amb una informació inicial en cas que aquesta estigui buida. El mètode "main" introdueix a la Base de Dades un gestor, 6 historials mèdics, 18 visites, i 3 metges. Després fa una assignació de dos pacients per metge, i 3 visites per pacient.

Aquesta classe, "DBManager" és feta servir pel gestor del sistema per a fer tots els accessos a la Base de Dades que necessita en l'execució dels protocols criptogràfics.

6.6. Diagrama de classes amb implementació de Base de Dades.

Amb la nova restructuració de l'aplicatiu per la implementació de la Base de Dades, el diagrama de classes queda com segueix:

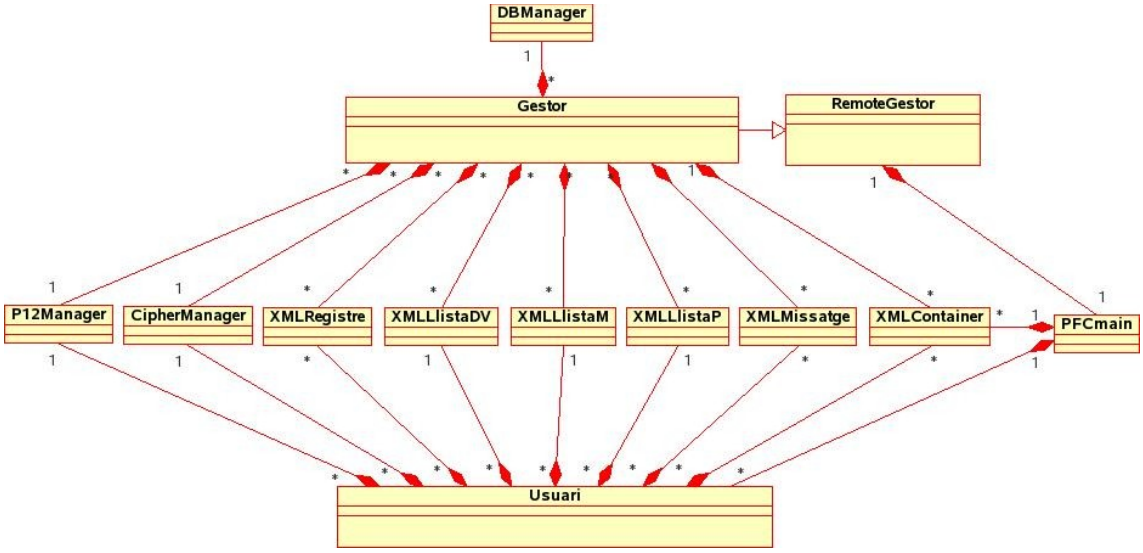


Figura 6.6.1: Diagrama de classes amb implementació de Base de Dades.

En aquesta representació, i amb motiu de la implementació de la Base de Dades, podem veure que les classes "Dades", "Metge", "Visita", "Historial", ja no apareixen en el diagrama de classes. La raó és evident, les dades que suportaven estan ara a la Base de Dades, i aquestes són gestionades i tractades per la nova classe "DBManager", a la qual només té accés la classe "Gestor" que implementa el nostre servidor RMI [7].

6.7. Parametrització de l'accés a la Base de Dades.

Amb la finallitat de facilitar l'accés a la Base de Dades i amb motiu d'aquest objectiu, s'ha creat un arxiu que estarà allotjat en el directori PFC, del qual penjarà tot l'arbre de directoris del PFC, i que tindrà el nom "**config.cfg**".

En aquest arxiu es recullen una sèrie de paràmetres per tal d'accedir a diferents arxius de treball de l'aplicatiu i, a més, els paràmetres que permeten l'accés a la Base de Dades. Entre els diversos camps que podem trobar en aquest arxiu, tenim els que permeten la parametrització de la Base de Dades i que especifico seguidament (a l'Annex C podem trobar l'especificació total d'aquest arxiu):

- **host**: nom del servidor de Base de Dades.
- **Database**: nom de la Base de Dades del PFC.
- **User**: usuari amb permís d'accés a la Base de Dades.
- **Password**: password de l'usuari amb accés a la Base de Dades.

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

7. Interfícies dels usuaris del sistema.

7.1. Introducció.

La interfície d'un aplicatiu és una de les parts més crítiques d'implementar del mateix. L'usuari que utilitzarà l'aplicatiu el farà servir mitjançant la intefície gràfica, i de vegades es passarà moltes hores utilitzant-la. És per això que si aquesta està mal dissenyada o és feixuga d'utilitzar, pot condemnar l'aplicatiu al fracàs.

En aquest Projecte la interfície ha de poder facilitar l'execució de les opcions que s'han implementat de la forma més senzilla possible, aplicant aquesta senzillesa també a la implementació. És per això que s'ha creat una única pantalla amb un menú d'opcions similar, tant per al pacient com per al metge, el qual conté les opcions que cada tipus d'usuari pot realitzar en el sistema. S'ha colocat també un menú d'ajuda senzill, i un “about” que mostrarà la informació bàsica de l'aplicatiu.

7.2. API utilitzada: AWT.

La API AWT o Abstract Window Toolkit [18] és una API que proveeix Java [6] per a la creació d'interfícies gràfiques. La API AWT [18] proveeix de molts components gràfics que poden ser afegits i posicionats en una àrea de pantalla que es visualitzi mitjanzant un administrador de capa.

Al igual que Java [6], AWT [18] és independent de la plataforma en la qual s'executa, tot i que depenent del servidor d'X natiu que s'estigui executant l'aparença de la interfície pot variar una mica d'un a l'altre. Això és degut a què l'aparença de la interfície segueix el “look and feel” del servidor concret on s'executa l'aplicatiu.

La construcció d'una interfície gràfica mitjançant AWT [18] es basa en la superclasse “Component”. Tots els components necessaris són afegits a un objecte “Container”, el qual conté tota la interfície.

7.3. Interfície del pacient.

Per a les interfícies del pacient, metge i gestor, s'ha utillitzat unes pantalles minimalistes quant a la seva elaboració. Aquest és un dels aspectes de l'aplicatiu desenvolupat en aquest Projecte que es podria millorar.

En la pantalla des d'on pot operar el pacient les opcions que pot fer contra el gestor del sistema es pot escollir qualsevol de les següents opcions que apareixen en el menú “File” de la seva interfície:

1. **Carregar contenidor PKCS#12**: aquesta opció demana al pacient que introdueixi el path i nom del seu contenidor PKCS#12 [16], i seguidament li demana el password d'aquest contenidor. Peces fonamentals per a identificar el pacient contra el gestor. Aquesta opció verifica que l'usuari

és un pacient, a més de controlar si el contenidor existeix i la seva password és correcta.

- **Autenticar-se contra el gestor del sistema**: aquesta altra opció executa el protocol número 1 que s'especifica en el capítol 3, apartat 5, i que s'utilitza per a autenticar el pacient contra el gestor del sistema.

- **Consulta dades generals del pacient**: aquesta opció executa el protocol 2 especificat en l'apartat 3.5 d'aquesta memòria. Permet al pacient consultar les seves dades generals que hi són en el seu historial. En la captura de pantalla següent podem veure el resultat de l'execució d'aquesta opció:

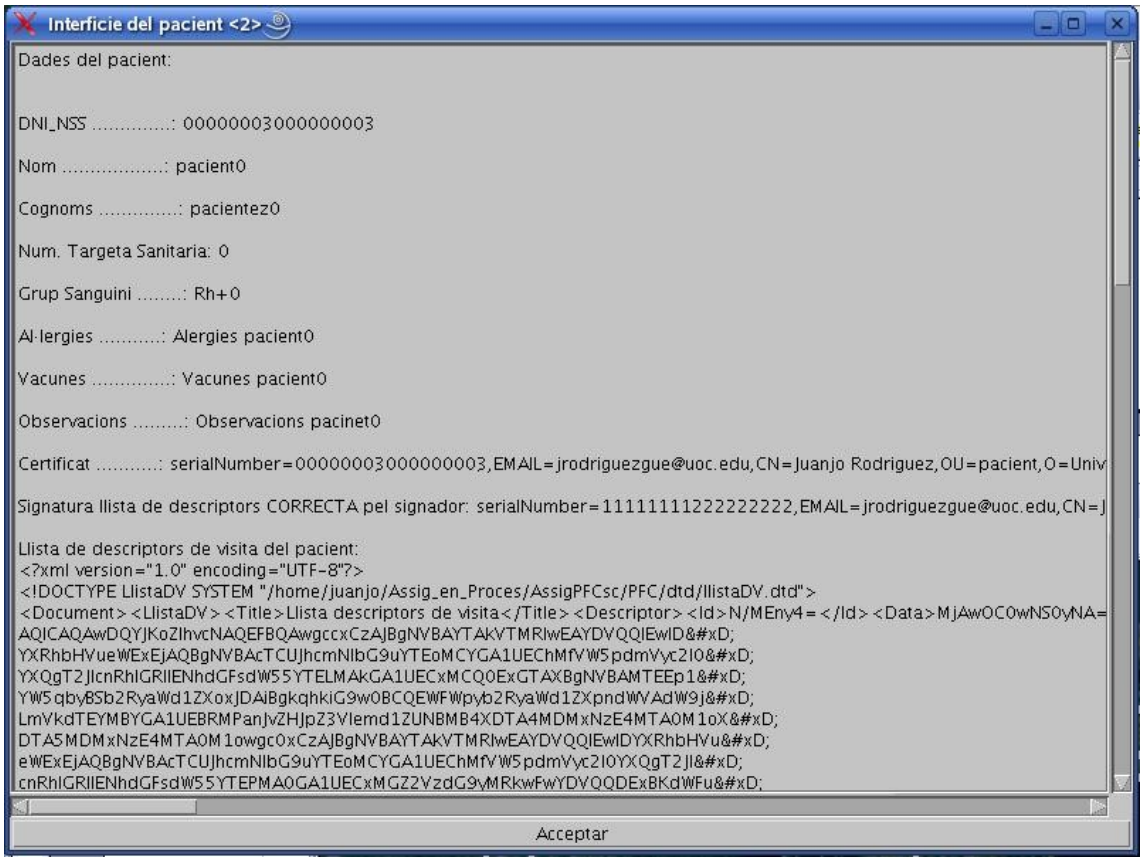


Figura 7.3.1: Captura de consulta dades general del pacient en la interfície del pacient.

- **Obtenir dades d'una visita**: aquesta opció presenta al pacient un rang de les visites que té emmagatzemades en la BD i li demana que esculli una d'elles. La opció s'ha desenvolupat així per senzillesa. Aquesta opció també és susceptible de millora en un sistema real. Seguidament veiem unes captures on podem veure com se li demana al pacient la visita a consultar i el resultat obtingut:

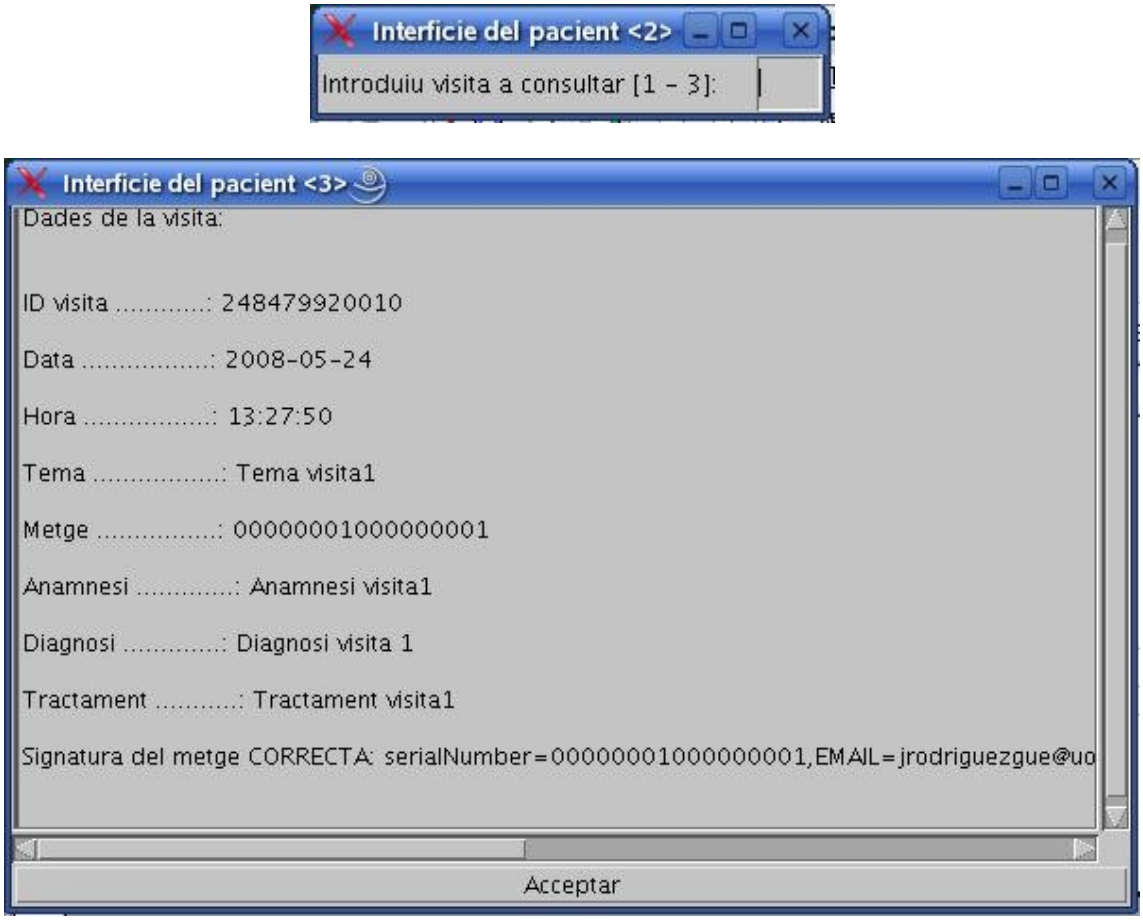


Figura 7.3.2: Captura de visita del pacient. Interfície del pacient.

- **Sortir de forma segura**: per últim, aquesta opció permet al pacient abandonar l'aplicació de forma segura.

7.4. Interfície del metge.

La interfície del metge segueix la mateixa filosofia que la del pacient, la qual acabem de veure. En ella podem trobar les següents opcions:

- **Carregar contenidor PKCS#12**: aquesta opció demana, com a la interfície del pacient, que el metge introdueixi el path i nom del seu contenidor PKCS#12 [16], i seguidament li demana el password d'aquest contenidor. Peces fonamentals per a identificar el pacient contra el gestor. Aquesta opció verifica que l'usuari és un metge, a més de controlar si el contenidor existeix i la seva password és correcta.

- **Autenticar-se contra el gestor del sistema**: aquesta altra opció executa el protocol número 1 que s'especifica en el capítol 3, apartat 5, i que s'utilitza per a autenticar el metge contra el gestor del sistema.

- **Consulta dades generals del pacient**: aquesta opció executa el protocol 2 especificat en l'apartat 3.5 d'aquesta memòria. Permet al metge consultar les dades generals de l'historial d'un pacient. L'opció demana al metge l'identificador de pacient, i seguidament mostra les seves dades generals. En la pantalla que es mostra, se li indica al metge si està, o no, autoritzat a veure les visites del pacient que ha consultat. En la captura de pantalla següent podem veure com se li demana al metge l'identificador del pacient, compost del seu DNI i el NSS seguits, i seguidament se li mostren les dades:

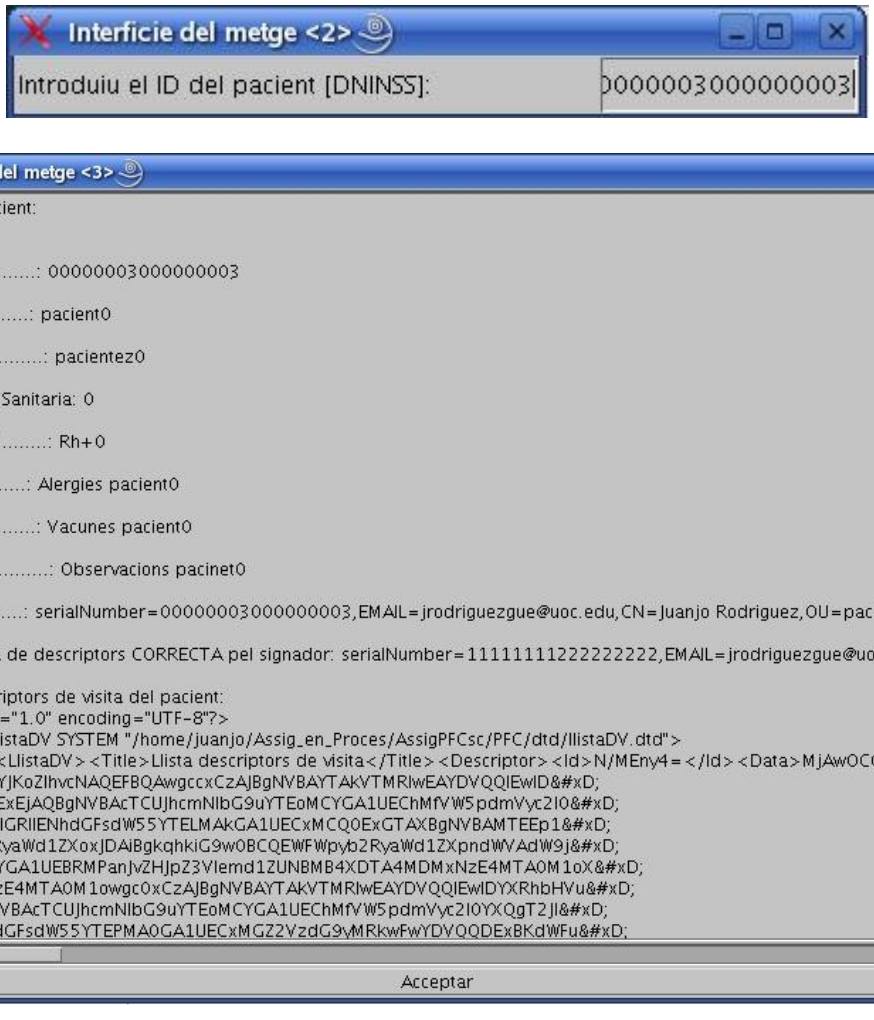


Figura 7.4.1: Captura de consulta dades general del pacient en la interfície del metge.

- **Obtenir dades d'una visita d'un dels seus pacients**: aquesta torna a demanar al pacient l'identificador del pacient, i seguidament fa les comprovacions que s'especifiquen en el protocol criptogràfic de consulta de visita d'un pacient, que es descriu en el capítol 3 d'aquesta memòria, per saber si el metge té accés a les visites del pacient que ha demanat o no. Si té accés, l'execució és la mateixa que en la consulta de visita de la interfície del pacient, se li presenta al metge el rang de visites possibles que pot consultar perquè esculli una d'elles. La opció s'ha desenvolupat així per senzillesa i és una opció que s'hauria de replantejar en un sistema real. Les captures de pantalla que podem veure d'aquesta opció en la interfície del pacient són igualment il·lustratives per a aquesta opció de la interfície del metge.

- **Afegir una visita al historial d'un dels seus pacients**: En aquesta opció se li presenta al metge una nova finestra on pot posar els camps que ha d'omplir per donar d'alta una nova visita del pacient. Si el metge no ha consultat l'historial del pacient prèviament, surt un missatge indicant que primerament consulti l'historial del pacient. Llavors el metge ha d'anar a l'opció de consulta de l'historial i consultar-lo, d'aquesta manera l'aplicatiu obté la llista de descriptors de visita i la de metges per verificar que el metge tingui accés a donar d'alta una nova visita en el pacient que s'està tractant. Seguidament podem veure unes captures de pantalla d'aquesta opció de la interfície del metge:

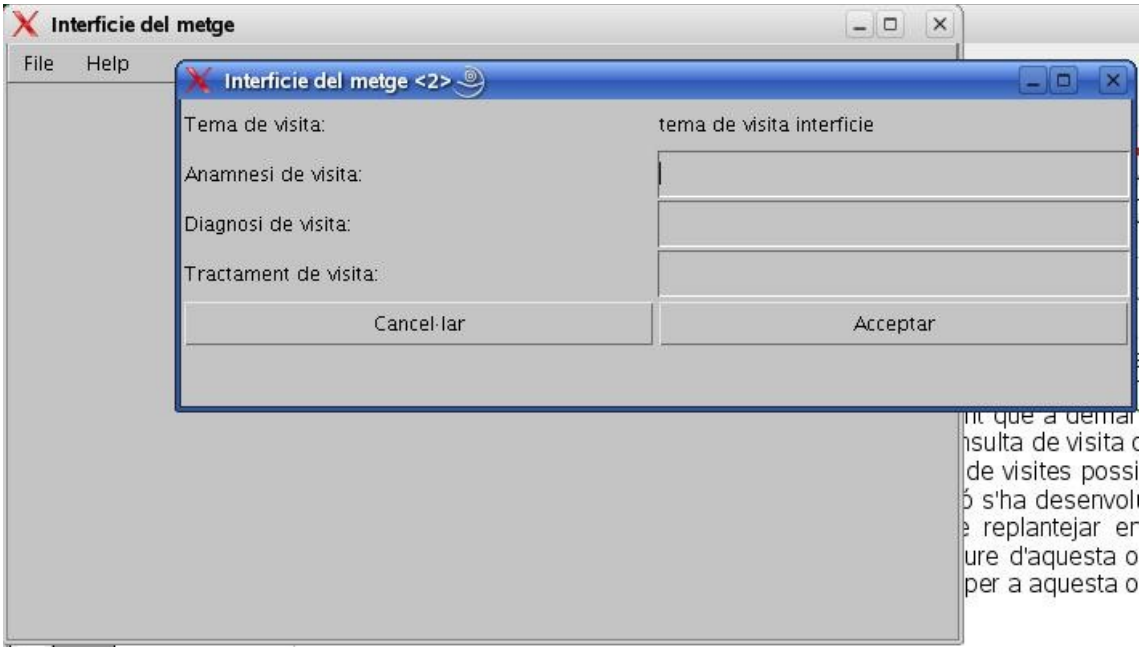


Figura 7.4.2: Captura de pantalla on es demanen les dades per afegir una nova visita en la interfície del metge.

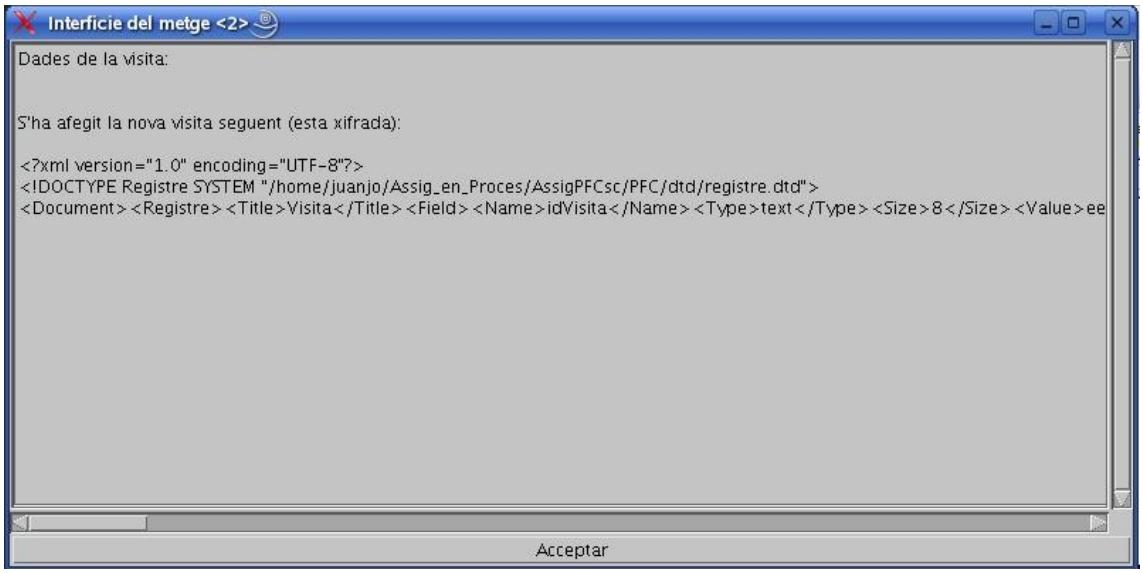


Figura 7.4.3: Captura de pantalla on veu el resultat d'afegir una nova visita a un pacient en la interfície del metge.

- **Obtenir llista de pacients assignats**: aquesta opció mostra la llista de pacients assignats al metge que està emmagatzemada en la fitxa del metge, i que està signada pel gestor del sistema. Seguidament veiem una captura de pantalla de l'execució d'aquesta opció:

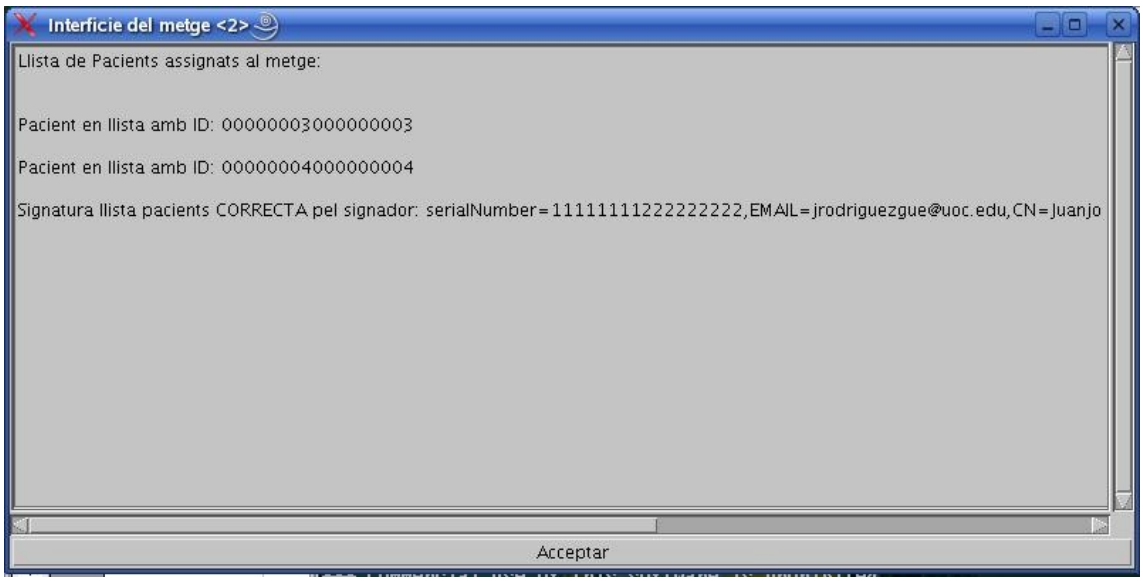


Figura 7.4.4: Captura de pantalla d'execució de mostra llista de pacients del metge en la interfície del metge.

- **Sortir de forma segura**: per últim, aquesta opció permet al metge abandonar l'aplicació de forma segura.

7.5. Interfície del gestor.

La interfície del gestor permetria 3 opcions possibles que s'especifiquen seguidament:

- **Registrar nou Metge**: aquesta opció permetria registrar nous metges en el sistema. Per falta de temps i perquè és una opció més aviat de manteniment de la BD de metges, aquesta opció no s'ha implementat. Junt amb aquesta opció es podrien implementar la baixa o modificació d'algun dels registres del metge.

- **Registrar nou Pacient**: com l'opció anterior, aquesta opció permetria afegir nous historials de nous pacients a la BD. I també, igualment, es podrien implementar opcions complementàries de modificació o baixa d'historials mèdics de pacients. Aquesta opció també queda pendent, ja que no és una opció crítica per a mostrar l'objectiu d'aquest Projecte.

- **Aturar gestor de forma segura**: aquesta opció ens permet aturar el gestor del sistema de forma segura, és a dir, prèvia comprovació que no hi ha cap usuari dins del sistema.

7.6. Gestió d'errors.

Quant a la gestió d'errors, no s'ha fet cap implementació extra que el control d'errors possibles que es poden donar en l'execució de les diferents opcions de cadascuna de les interfícies.

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

8. Joc de proves.

8.1. Introducció.

L'objectiu d'aquest capítol és donar un exemple complet d'execució de l'aplicació, tant del gestor del sistema, de l'aplicació del metge i de l'aplicació del pacient. En els diferents punts d'aquest capítol es passa per totes les fases d'execució de l'aplicatiu i es crea la infraestructura mínima per a la seva execució.

Primer de tot caldrà colocar el tar.gz del Projecte en el directori que es vulgui i descomprimir-lo amb la instrucció "`tar xvfz jrodriguezgue-PFC.tar.gz`". Un cop descomprimit canviarem al directori creat PFC amb la comanda "`cd PFC`", i a partir d'aquí ja podem seguir els passos que s'expliquen seguidament per a l'execució del PFC.

8.2. Generació de certificats.

En el punt 2.6 d'aquesta memòria s'expliquen els passos a seguir per a poder generar els certificats i els contenidors PKCS#12 [16] de l'Autoritat de Certificació necessària per a l'execució del projecte, així com el certificat i el contenidor PKCS#12 [16] d'un metge, un pacient i el gestor del sistema.

A més, tal i com s'explica en tot el capítol 2, per a la generació de tota la infraestructura de PKI que s'utilitza en aquest Projecte es fa servir l'eina de programari lliure OpenSSL [13], i d'aquesta eina per a la generació dels certificats fem servir un fitxer de configuració anomenat "openssl.cnf", el qual podem veure en l'annex A.

Amb l'entrega d'aquest Projecte, s'inclou, en l'estructura de directoris, el directori PKI, on podem trobar la PKI que tenim definida, així com els certificats i els contenidors PKCS#12 [16] del gestor, tres metges i sis pacients.

A més trobarem un arxiu "README" on estan especificades les contrasenyes de la parella de claus de cada usuari i del seu contenidor PKCS#12 [16].

8.3. Preparació de la Base de Dades.

Per a l'execució del Projecte s'utilitzarà el Sistema Gestor de Base de Dades (SGBD) de programari lliure MySQL 5.0 [10]. Se suposa que ja es té instal·lat un servidor de Base de Dades (BD) amb a una BD definida ja en aquest servidor.

Un cop tenim en marxa el SGBD MySQL [10] hem d'executar en la consola de MySQL [10] l'script que podem trobar en l'annex D i que també ve inclòs en el tar.gz que s'entregarà amb aquest Projecte anomenat "pfc.sql", el qual crearà la BD i les taules necessàries per a l'execució de l'aplicatiu.

Seguidament cal crear un usuari del SGBD que tingui permisos d'accés i modificació en la BD que hem creat. Per exemple podríem utilitzar les següents sentències per a realitzar aquest pas:

```
CREATE USER usuari IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES pfc.* TO usuari IDENTIFIED BY 'password';
```

Per tal de tenir parametritzat l'accés a la BD que ha de realitzar l'aplicatiu, en concret l'aplicatiu del gestor del sistema, s'ha utilitzat un arxiu anomenat "config.cfg" que també estarà inclòs en el tar.gz d'entrega i que conté uns camps amb el paràmetres següent:

- **host**: host on ens hem de connectar, per defecte "localhost".
- **database**: nom de la BD que conté les dades, per defecte "pfc"
- **user**: nom del l'usuari que té permís de connexió a la BD del PFC.
- **password**: password de l'usuari anterior.

A l'annex C podem veure el fitxer de configuració "config.cfg". En aquest arxiu hi ha d'altres camps que especifiquen directoris on s'emmagatzemen els contenidors i certificats dels pacients, metges i gestors que utlitzarà la "DBManager.java" per carregar les dades inicials per provar l'aplicatiu. És convenient deixar aquests camps amb els valors definits, ja que amb l'entrega del tar.gz del projecte s'inclou la PKI mínima necessària, junt amb tres metges i sis pacients amb els seus certificats i contenidors per a realitzar la càrrega incial de dades en la Base de Dades. L'únic camp que caldrà actualitzar serà el de "**dirbase**", el qual s'haurà d'actualitzar amb el directori base del Projecte. Quant el desplegui el tar.gz d'entrega, el directori base podria ser: "/directori_home_usuari/PFC".

8.4. Inserció d'usuaris i dades mínimes a la Base de Dades.

Un cop hem creat les taules i tenim en marxa el nostre SGBD, hem de carregar les dades mínimes a la Base de Dades per provar l'aplicatiu. Però primer **HEM DE POSAR EN MARXA EL NOSTRE SERVIDOR RMI [7]**, cosa que s'explica en el punt 8.6.

Un cop tenim en marxa el servidor RMI [7], des d'una còmsola del sistema ens coloquem dins la carpeta PFC del projecte i executem la següent sentència:

- **`java bin.core.DBManager`**

Aquesta sentència carregarà a la Base de Dades les dades necessàries per poder provar l'aplicatiu.

8.5. Configuració per a l'execució en Java.

Per a la correcta execució de l'aplicatiu en l'entorn Java [6], cal la configuració adient d'algunes llibreries, la principal de les quals és l'API IAIK [15]. Això es fa afegint en la variable CLASSPATH del sistema el fitxer “iaik_jce_full.jar”, i afegint dos arxius de configuració de les polítiques de seguretat de Java. Els arxius de polítiques de seguretat s'han de colocar dins el directori “`$JAVA_HOME/jre/lib/security`”. Aquests dos arxius són:

- local_policy.jar
- US_export_policy.jar

A més, caldrà el connector de MySQL [11] per a Java [6], fitxer “mysql-connector-java-5.0.8-bin.jar”, per poder operar desde Java [6] amb la Base de Dades, el qual també caldrà afegir-lo a la variable CLASSPATH del sistema, al igual que l'API de tractament de documents XML [12] JDOM [9], la qual quedarà configurada afegint a la variable CLASSPATH el seu arxiu jar “jdom.jar”.

8.6. Execució del servidor RMI.

Per a l'execució del nostre servidor RMI [7], caldrà tenir instal·lat correctament l'entorn Java [6], així com l'aplicació del Projecte compilada i instal·lada en els seus directoris corresponents, “bin/core” i “bin/interficie”. En l'entrega es proporcionarà ja l'aplicació correctament compilada i instal·lada en el seu directori corresponent, així com un fitxer “build.xml” per poder compilar-la mitjançant “Ant” [8]. Per a la compilació amb l'eina “Ant” [8] només caldrà colocar-se dins el directori “PFC” i executar la sentència següent: “ant -f build.xml”.

Seguidament caldrà executar les comandes següents estant en la mateixa carpeta “PFC”:

- `rmic bin.core.Gestor`
- `rmiregistry 2001 &`
- `java bin.core.Gestor contenidor_PKCS#12 password_contenidor &`

La primera instrucció crearà l'Stub corresponent per poder comunicar-se amb el servidor RMI [7], que en aquest cas serà el propi gestor.

La segona comanda posa en marxa el registre de servidors d'RMI [7], el qual en aquest cas quedarà escoltant pel port 2001.

Per últim, la tercera comanda executarà el nostre servidor RMI [7], el qual quedarà registrat en el registre RMI [7] i llest per a rebre peticions dels usuaris, en aquest cas els metges i pacients del sistema. Es pot observar que en

l'execució del nostre servidor RMI [7] cal passar-li per línia d'execució el path i nom del contenidor PKCS#12 [16] del gestor, seguit del password d'aquest contenidor. En el PFC que s'entrega i com que ja tindrem un gestor creat, l'ordre d'execució serà (suposem que estem dins el directori PFC): "`java bin.core.Gestor ./PKI/gestor/gestor.p12 PFCgestorp122008`".

Un cop executat el nostre servidor RMI [7], obtindrem també l'execució de la interfície gràfica del gestor, desde la qual podrem aturar el nostre servidor RMI [7], o gestor del sistema. La finestra que obtindrem en l'execució del nostre servidor RMI [7] tindrà l'aparença següent:

Figura 8.6.1: Captura de pantalla interfície del Gestor.

8.7. Execució de la interfície gràfica del pacient.

En el capítol 7 ja s'ha especificat la interfície del pacient, però en aquest punt especificarem com arrancar l'aplicació del pacient i l'operativa a seguir per poder executar les opcions que el pacient pot fer.

Primerament i suposant que estem al directori base del PFC, "PFC", executarem la següent comanda:

- `java bin.interficie.PacientApp`

Aquesta comanda mostrarà la pantalla principal de la interfície del pacient. Un cop es mostri aquesta pantalla hem de carregar el contenidor PKCS#12 [16] del pacient per poder executar qualsevol de les opcions possibles que el pacient pot llançar contra el gestor del sistema. Per a fer això ha d'escollir l'opció "File" de la barra superior de menú i seleccionar la primera opció, "Carregar contenidor PKCS#12 [16]". La interfície li demanarà el path i nom del seu contenidor PKCS#12 [16]:

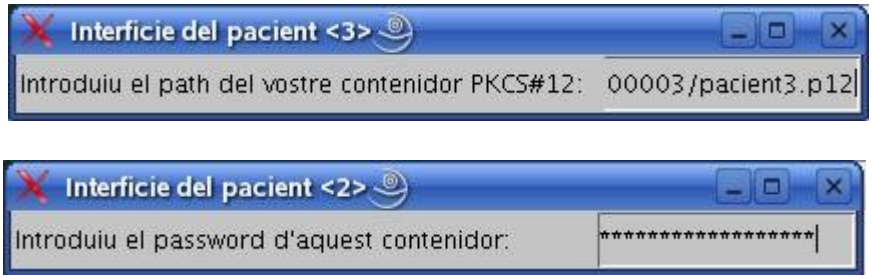


Figura 8.7.1: Petició contenidor PKCS#12 i password en interfície del pacient.

Un cop introduïdes el sistema busca i carrega. A partir d'aquí ja podem cridar qualsevol opció del pacient. Cal tenir present que tot i que el sistema és força ràpid, tenint en compte les operacions que ha de fer, és possible que la primera vegada que s'executa l'aplicatiu alguna operació trigui uns segons en executar-se, però només cal esperar i el resultat serà presentat.

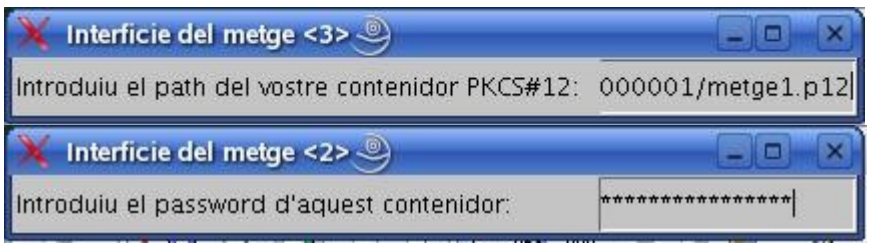
8.8. Execució de la interfície gràfica del metge.

Com en el cas de la interfície del pacient, en el capítol 7 ja s'ha especificat la interfície del metge, tot i que especificarem com arrancar l'aplicació del metge, així com els primers passos necessaris per a executar la resta d'operacions que pot demanar el metge. És aconsellable, al igual que amb el pacient, llegir el capítol 7 per a veure una descripció detallada de les opcions que pot executar el metge.

Primerament i suposant que estem al directori base del PFC, “PFC”, executarem la següent comanda:

- `java bin.interficie.MetgeApp`

Aquesta comanda mostrarà la pantalla principal de la interfície del metge. Un cop es mostri aquesta pantalla hem de carregar el contenidor PKCS#12 [16] del metge per poder executar qualsevol de les opcions possibles que el pacient pot llançar contra el gestor del sistema. Per a fer això s'ha d'escollir l'opció “File” de la barra superior de menú i seleccionar la primera opció, “Carregar contenidor PKCS#12”. La interfície li demanarà el path i nom del seu contenidor PKCS#12 [16]:



Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

Figura 8.8.1: Petició contenidor PKCS#12 i password en interfície del metge.

Un cop introduïdes les dades, el sistema buscarà el contenidor PKCS#12 [16] de metge i el carregarà. A partir d'aquí ja podem cridar qualsevol opció del metge. Cal tenir present que tot i que el sistema és força ràpid, tenint en compte les operacions que ha de fer, és possible que la primera vegada que s'executa l'aplicatiu alguna operació trigui uns segons en executar-se, però només cal esperar i el resultat serà presentat.

8.9. Apagar el sistema.

Per apagar el sistema amb seguretat, en la interfície del Gestor, en la barra del menú de la part superior, escollim l'opció “File” i del menú penjant que apareix, escollim l'opció “Apagar gestor de forma segura”. El gestor verificarà que no hi ha cap pacient ni metge funcionant en aquell moment i, si és així, aturarà el nostre servidor RMI [7], el qual és a la vegada el gestor del sistema.

Un cop aturat el nostre servidor RMI [7], podem aturar el “rmiregistry” matant el seu número de PID. En aquest moment, també podríem aturar el SGBD MySQL [10].

Aprofitem per a fer un apunt en aquesta secció per indicar que la gestió d'errors la fa la interfície de forma integrada amb la mateixa.

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

9. Treball futur.

9.1. Introducció.

És evident que aquest Projecte només pretén ser un exemple d'una possible aplicació de tractament d'historials mèdics de forma segura, la qual és susceptible de millora. Per tant, ni de bon troç, podem donar per finalitzada aquesta aplicació, ja que sempre podríem afegir noves prestacions, així com optimitzar les que ja s'ha implementat i que, per falta de temps, han quedat com es presenten en aquest Projecte de Final de Carrera. Per tant, seguidament exposem algunes de les millores possibles que podríem afegir a l'aplicatiu.

9.2. Millores a implementar.

L'aplicació implementada en aquest Projecte de Final de Carrera, pot servir de punt de partida funcional per a una aplicació en un ambulatori o hospital. Entre les possibles millores que podríem implementar estarien les següents:

- Una primera millora seria el redisseny i adaptació de la interfície gràfica a les necessitats de l'organització. Es podria redissenyar utilitzant altres eines, o bé convertir-la en una interfície web.

- Millorar el sistema de detecció d'usuaris actius en el sistema. Actualment es fa mitjançant un arxiu en el sistema local que indica al servidor RMI [7] que hi ha un usuari treballant, i això es fa a nivell d'interfície gràfica i perquè tot el sistema s'executa en una única màquina. Però una millora seria implementar un mètode públic en el servidor RMI [7] que els usuaris poguèssin utilitzar per indicar que el client està en marxa, passant informació de la seva identificació, a més de renovar el seu indicador d'activitat en cada operació que fes, i d'indicar en quin punt es trobaria en l'execució de la mateixa.

- Implementar les interfícies de manteniment d'usuaris, tant metges com pacients, en la interfície del gestor, ja que amb aquesta entrega, els usuaris són introduits mitjançant el procediment de càrrega de la Base de Dades, i la classe “DBManager”. Aquest manteniment pot contemplar les típiques opcions d'alta, modificació, consulta o baixa d'usuaris, i altres opcions que puguin ser útils.

- Realitzar una validació dels certificats mitjançant la cadena de certificació. Aquesta validació no s'ha implementat en aquesta primera versió de l'aplicatiu.

- Millorar el sistema de parametrització de l'aplicació per a optimitzar la càrrega de paràmetres des del sistema de fitxers de la màquina.

- Una possible millora seria l'ús de targetes amb xips integrats, les quals portarien a dins els contenidors PKCS#12 [16] dels usuaris. Amb aquestes targetes l'usuari podria autentificar-se en el sistema.

- En aquesta implementació s'ha utilitzat el programari lliure OpenSSL [13] per a la generació de la PKI i dels certificats dels usuaris. En un sistema real hauríem de tenir la nostra PKI implementada per alguna organització reconeguda, com ara l'Agència Catalana de Certificació.

- Una altra possible millora podria ser la integració en l'aplicació de la generació dels certificats a l'hora de registrar nous usuaris en el sistema, així com la generació de les seves claus i els seus contenidors PKCS#12 [16].

- Quant al tema de les signatures que s'utilitzen força en l'aplicació, seria molt convenient implementar el timestamp en les signatures mitjançant un servidor de temps. Això donaria plena validesa legal a les signatures del gestor i dels usuaris, metges o pacients.

- No s'ha inclòs en la parametrització de l'aplicació la parametrització del servidor RMI [7]. Si això s'inclogués, no caldria recompilar l'aplicació per a trobar el servidor RMI [7] que s'utilitzi.

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

10. Conclusions.

Hem arribat al final d'aquest Projecte i cal fer una valoració de la feina feta, així com de la feina que queda pendent. En aquest capítol es demostra que els requisits que trobem en l'enunciat d'aquest Projecte han estat assolits.

Inicialment podem dir que disposem d'un esquema criptogràfic que ens assegura l'objectiu principal d'aquest Projecte: la possibilitat de gestionar historials i visites mèdiques de forma remota amb una protecció i seguretat proporcionades per l'esquema criptogràfic implementat, així com una autentificació fiable dels usuaris que tracten amb el sistema.

Repassant les línies principals de l'enunciat, hem anat construint els requeriments del Projecte pas per pas, de forma incremental:

1. **Punt 1**: hem creat la nostra PKI necessària per a l'autentificació, integritat, confidencialitat i no repudi de les operacions contemplades en l'esquema criptogràfic.

2. **Punt 2**: s'ha implementat l'esquema criptogràfic necessari per a realitzar les operacions de gestió d'informació mèdica mitjançant un gestor de sistema, aconseguint els objectius marcats:

   a) Autenticitat.
   b) Privacitat.
   c) Correcció.
   d) Secret de la informació transmesa entre els actors del sistema.
   e) Integritat
   f) No-repudi de les accions de l'esquema criptogràfic que així ho requerien

3. **Punt 3**: s'ha aconseguit representar les dades en el format estàndard que es fa servir avui dia en Internet, l'XML [12], per a la transmissió d'informació i documents entre els diferents actors del sistema.

4. **Punt 4**: també hem implementat un sistema de comunicació entre els components del Projecte, concretament s'ha utilitzat la implementació de Java [6] per a la invocació remota de mètodes, RMI [7]. Aquest s'ha escollit per practicitat, comoditat i facilitat d'ús, acomplint així l'objectiu de minimitzar l'esforç d'aquesta part del Projecte.

5. **Punt 5**: Hem implementat l'ús i l'emmagatzematge de la informació en un SGBD de programari lliure com és MySQL [10], fet que assoleix els requeriments del Projecte, com la necessitat de guardar la informació en Base de Dades i que aquest programari fos programari lliure.

6. **Punt 6**: per a la interacció dels usuaris amb l'aplicació, s'ha implementat les respectives interfícies gràfiques per a cadascun dels actors del sistema, com requeria el Projecte, per tal de facilitar l'ús de l'aplicatiu als

usuaris del mateix. En aquest punt cal mencionar que, tot i la senzillesa de la interfície gràfica i la facilitat d'ús de les opcions implementades i suportades per l'esquema criptogràfic, al darrera d'aquesta simplicitat hi ha una complexitat força gran per a proporcionar els requeriments del Projecte.

Per acabar aquesta memòria voldria donar la meva opinió quant a la realització d'aquest Projecte Final de Carrera. La meva experiència ha estat molt positiva i gratificant al veure com a poc a poc s'ha anat construint tot un sistema de peces que en unir-les anaven assolint cada cop més prestacions.

En aquesta experiència he anat integrant els coneixements que en tantes assignatures he aprés i aplicat per separat, i he pogut experimentar la sensació de veure que sé una mica d'informàtica, ja que en un camp com aquest, i crec que en gairabé cap camp del coneixement, mai es pot dir que se sap molt.

Quant al disseny, crec que el fet d'utilitzar un disseny incremental ha facilitat molt tota la implementació, ja que, tot i haver de tenir o dissenyar classes de suport per al testeig de les parts de l'esquema criptogràfic i el funcionament de la implementació XML [12] fins a arribar a la implementació de la Base de Dades, he pogut experimentar que un cop la part de l'esquema criptogràfic o l'ús de dades amb XML [12] han estat implementades, només han requerit adaptacions mínimes per a integrar la part RMI [7], la Base de Dades i per últim la interfície gràfica.

També cal mencionar l'ajuda del consultor, Jordi Castellà-Roca, que molt eficaçment ha respost als dubtes plantejats i ha orientat la feina a desenvolupar en la implementació global del Projecte.

He de dir que si hagués de tornar a desenvolupar el Projecte desde l'inici, canviaria algunes coses i milloraria d'altres, però suposo que això és una qüestió d'experiència i es va millorant a mesura que es van desenvolupant Projectes.

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

11. Glossari.

API: Sigles de Application Programming Interface. Interfície a través de la qual un programa accedeix als serveis del sistema operatiu i d'altres. Una API proveeix un nivell d'abstracció entre l'aplicació i el kernel per tal d'assegurar la portabilitat del codi.

Aplicatiu: Conjunt d'elements de programari interrelacionats que porten a terme una o vàries tasques i que fan servir uns actors interactuant amb aquests elements.

Arquitectura client – servidor: sistema de maquinari i programari interelacionat en el qual hi ha un o més elements que fan peticions al elements anomenats servidors. En general, les màquines del client i del servidor són diferents, però no es descarta que pugui funcionar sobre la mateixa.

Autoritat de Certificació: Entitat que emet certificats digitals d'usuaris o companyies, de manera que aquests es poden identificar davant d'un tercer. És de vital importància que l'Autoritat de Certificació comprovi que la part que demana un certificat és realment qui diu ser.

AWT: Sigles d'Abstract Window Toolkit. És una API que proveeix Java per a la creació d'interfícies gràfiques.

Base 64: Codificació que empra només 6 bits per caràcter. Molt utilitzada en la transmissió d'informació mitjançant xarxes de comunicacions i Internet.

Base de Dades: Estructura de dades persistents, normalment associades a programaris que les consulten i actualitzen. Una base de dades és un component d'un Sistema Gestor de base de dades.

Cas d'ús: Diagrama pertanyent a les especificacions UML que permet veure gràficament i per a cada actor del sistema, les accions que pot dur a terme i les relacions d'aquestes accions amb el sistema.

Certificat: Arxiu que conté les dades que donen fe de l'autenticitat de la persona o entitat que el presenta.

Clau: Peça d'informació que s'utilitza en criptografia simètrica per a xifrar i desxifrar un missatge. En criptografia asimètrica la clau pot ser pública o privada. La clau pública s'utilitza per xifrar missatges o verificar una signatura. La clau privada s'utilitza per desxifrar o per signar unes dades. La longitud en bits de la clau sovint ens dóna una idea de la robustesa del sistema que emprem.

Document Type Definition (DTD): Definició d'un document XML o SGML. Consisteix en unes regles per interpretar els documents i per establir les regles de construcció dels mateixos.

Fingerprint: Funció de hash que s'aplica sobre un certificat d'un usuari per a obtenir un identificador únic i més còmode d'utilitzar. En el projecte s'utilitzem fingerprints per identificar de manera única el gestor del sistema .

Funció de Hash: Funció unidireccional que relaciona un contingut concret amb un resum de mida limitada i determinada.

IAIK: Sigles de "Institute for Applied Information Processing and Communication". Aquests són els desenvolupadors de la llibreria criptogràfica amb el mateix nom.

Interfície: Punt d'interacció i/o comunicació entre un ordenador i una altra entitat, ja sigui persona o un altre equip.

Java: Llenguatge de programació multi-plataforma, robust, interpretat, distribuït, orientat a objectes, portable, desenvolupat per Sun Microsystems a mitjans dels 90.

JDOM: Sigles de Java Document Object Model. Solució complerta basada en Java per accedir i modificar documents XML des de codi Java.

Lliure distribució: Se'n diu així del programari que s'ofereix lliurement sense la necessitat de que l'usuari final aboni una quantitat de diners per a la seva utilització. Normalment, amb la distribució s'inclou el codi font al que l'usuari hi te accés a modificar-lo i redistribuir-lo si així ho desitja.

Longitud de clau: En termes de criptografia indica el nombre de bits de la clau que utilitzem per a xifrar i desxifrar les dades. Les claus simètriques i privades s'han de mantenir en llocs segurs.

Màquina Virtual: Màquina abstracta per a la que existeix un intèrpret. En general s'utilitza en sistemes operatius per a assegurar la portabilitat dels aplicatius entre els mateixos. Java té la seva Java Virtual Machine (JVM), molt popular avui dia.

MySQL: Sistema Gestor de base de dades de lliure distribució.

Número aleatori: Número generat sense que l'usuari tingui contacte en el procés.

PKCS: Sigles de Public-Key Cryptography Standards. Són un conjunt d'estàndards definits pels laboratoris RSA que especifiquen els estàndards de clau pública.

PKI: Sigles de Public Key Infrastructure corresponent a infrastructura de clau pública.

Protocol d'autenticació: Conjunt d'operacions que duen a terme dues o més parts de manera que al final almenys una de les part queda autenticada davant de la resta.

Proves d'integració: Conjunt de test que es duen a terme sobre un aplicatiu en fase de desenvolupament per tal d'assegurar el seu correcte funcionament amb d'altres aplicatius o sistemes amb els que interactua.

RMI: Sigles de Remote Method Invocation. API propietària de Java que permet a les aplicacions locals executar codi que es troba allotjat en una altra màquina remota. Aquesta última posa a disposició uns mètodes públics que seran accessibles a través d'una interfície.

Signatura: Document generat a partir d'un missatge i la clau privada d'un usuari. Al xifrar les dades del missatge amb la clau privada es genera un missatge xifrat que una tercera persona pot desxifrar amb la clau pública i comprovar que és el mateix que les dades originals. D'aquesta manera s'assegura que l'origen de les dades no ha estat modificat i que la persona que l'envia és qui diu ser, ja que només ella te accés a la seva clau privada.

Sobre digital: Mètode emprat en la criptografia que utilitza els dos tipus de criptosistemes. D'una banda el missatge a transmetre es xifra utilitzant un xifratge de clau simètrica. A continuació, la clau que s'ha emprat per dur a terme aquest xifratge es xifra utilitzant un sistema de xifratge asimètric. El resultat de les dues operacions és el document que s'envia a l'altra part. L'avantatge d'utilitzar aquest sistema és que redueix dràsticament el temps emprat en xifrar el document original ja que el xifratge simètric és molt més ràpid que l'asimètric.

Sistema Gestor de Base de Dades (SGBD): Conjunt d'aplicacions que normalment porten control d'un conjunt de dades persistents, oferint alhora facilitat d'accés i consulta als usuaris finals.

UML: Sigles de Unified Model Language. Llenguatge no propietari d'especificació. És un mètode utilitzat per a especificar, visualitzar, construir i documentar un sistema orientat a objectes en fase de desenvolupament.

www: Sigles de World Wide Web o Xarxa d'ordinadors que contenen llocs d'Internet que ofereixen text, imatges, so i animacions a través del protocol de xarxa HTTP (HyperText Transfer Protocol).

Xifratge simètric: Mètode emprat en la criptografia que utilitza la mateixa clau per a xifrar i desxifrar un missatge.

Xifratge asimètric: Mètode emprat en la criptografia que utilitza dues claus, una pública i una privada, per dur a terme el xifratge i posterior desxifratge d'un

missatge. En general, s'utilitza la clau pública de l'usuari destí per xifrar i aquest utilitza la clau privada per desxifrar.

XML: Sigles d'eXtensible Markup Language, metallenguatge creat per l'organització W3C (World Wide Web Consoritum) que permet a un usuari de crear el seu propi llenguatge de tags. Freqüentment utilitzat per a facilitar l'intercanvi de documents en entorns de xarxa.

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

12. Bibliografia.

[1] Protocol asimètric de Needham-Schroeder:
http://es.wikipedia.org/wiki/Protocolo_de_Needham-Schroeder.
[2] Apunts de Criptografia de la UOC. Versió de febrer del 2004.
[3] Criptografia basada en curves elíptiques
http://en.wikipedia.org/wiki/Elliptic_Curve_Cryptography
[4] Apunts d'Enginyeria del Programari Orientat a Objectes de setembre 2005.
[5] Apunt d'Enginyeria del Programari de febrer 2004.
[6] The J2SE Development Kit (JDK),
http://java.sun.com/reference/api/index.html
[7] Java Remote Method Invocation (Java RMI),
http://java.sun.com/javase/technologies/core/basic/rmi/index.jsp
[8] The Apache Ant Project
http://ant.apache.org
[9] The JDOM XML API.
www.jdom.org
[10] The MySQL database server
www.mysql.com
http://dev.mysql.com//doc/refman/5.0/en/
[11] MySQL java connector.
http://dev.mysql.com/downloads/connector/j/5.0.html
http://dev.mysql.com/doc/refman/5.0/en/connector-j.html
[12] Extensible Markup Language (XML)
www.w3.org/XML,
www.w3schools.com/xml,
www.w3schools.com/dtd
[13] Openssl: The open source toolkit for SSL/TLS,
www.openssl.org,
www.openssl.org/support/faq.html
[14] The Unified Modelling Language: UML,
www.uml.org
[15] The "Institute for Applied Information Processing and Communication",
http://jce.iaik.tugraz.at
https://jce.iaik.tugraz.at/crm/freeDownload.php
[16] PKCS #12: Personal Information Exchange Syntax Standard, PKCS: Cryptographic Message Syntax Standard,
http://www.rsa.com/rsalabs/node.asp?id=2138
[17] RFC 2510 - Internet X.509 Public Key Infrastructure Certificate Management Protocols,
http://www.ietf.org/rfc/rfc2510.txt
http://www.ietf.org/rfc/rfc2560.txt
[18] API AWT o Abstract Window Toolkit
http://roseindia.net/java

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

13. Annexes.

13.1. Annex A: Fitxer de configuració “openssl.cnf”.

El següent arxiu “openssl.cnf” és l'arxiu de configuració de l'eina OpenSSL de programari lliure que s'ha utilitzat per a la generació dels certificats mitjançant la nostra PKI. Seguidament podem veure aquest fitxer:

```
#
# 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
dnQualifier=2.5.4.5

####################################################################
[ 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/private/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
localityName            = 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             = supplied

####################################################################
[ 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                     = Identificador Usuari [DNI+NSS]
dnQualifier_max                 = 25

# 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
```

13.2. Annex B: Instal·lació del l'aplicació.

En aquest annex s'expliquen els passos que cal seguir per descomprimir l'aplicatiu i desplegar-lo en el directori que es vulgui:

1. Primer es coloca l'arxiu que conté el PFC, "jrodriguezgue-PFC.tar.gz" en el directori que es vulgui.
2. Segon s'executa la comanda: "**tar xvfz jrodirguezgue-PFC.tar.gz**".
3. Ara tenim un directori "PFC" nou en la nostra carpeta. Canviem al directori amb la comanda "**cd PFC**".
4. Executem la comanda següent per a compilar l'aplicatiu: "**ant -f build.xml**"
5. Creem l'stub amb la comanda següent: "**rmic bin.core.Gestor**".
6. Seguidament arranquem el nostre servidor de BD MySQL amb l'ordre similar a la següent: "**$MYSQL_HOME/bin/mysqld_safe &**"
7. Després posem en marxa el nostre registre RMI: amb l'ordre: "**rmiregistry 2001 &**".
8. Ara ja podem executar el gestor amb el seu contenidor que s'inclou en el directori "./PKI/gestor/gestor.p12" i la seva password. La comanda és la següent: "**java bin.core.Gestor ./PKI/gestor.p12 PFCgestorp122008 &**"
9. I per últim podem arrancar la interfície del pacient i la del metge amb les ordres respectives per poder provar el Projecte:

    9.1. "**java bin.interficie.PacientApp**"

    9.2. "**java bin.interficie.MetgeApp**"

10. Per a provar cadascuna de les opcions del PFC cal seguir el capítol 8 de "Joc de proves".

13.3. Annex C: Arxiu de parametrització de l'aplicació del PFC.

Seguidament mostro l'arxiu de parametrització de l'aplicació, "**config.cfg**", que caldria modificar per a connectar-nos a la Base de Dades on es provarà l'aplicatiu. L'arxiu és el següent:

```
dirbase#./
gestor#PKI/gestor/
metge1#PKI/metge00000001000000001/
metge2#PKI/metge00000002000000002/
metge3#PKI/metge33333333444444444/

pacient1#PKI/pacient00000003000000003/
pacient2#PKI/pacient00000004000000004/
pacient3#PKI/pacient00000005000000005/
pacient4#PKI/pacient00000006000000006/
pacient5#PKI/pacient00000007000000007/
```

```
pacient6#PKI/pacient55555555666666666/

host#localhost
database#pfc
user#NOM_USUARI_BD
password#PASSWORD_USUARI

dtdfiles#dtd/
```

Com es pot veure, caldrà posar el nom d'usuari que es connectarà a la Base de Dades, i el password correcte d'aquest usuari. S'ha deixat com a nom de la Base de Dades, “pfc”, tot i que també es pot substituir pel nom de la base de dades que es tingui o es crei en el servidor de Base de Dades.

És important mantenir tal i com estan configurats els directoris base, del gestor, dels metges i dels pacients, ja que són els que ja estan creats i s'han de carregar automàticament en la Base de Dades per poder fer les proves amb unes dades mínimes.

En cas que s'implementés en el gestor el manteniment de pacients i metges, no caldria afegir en aquest arxiu els paths dels seus contenidors i certificats, ja que això quedaria solventat pel manteniment d'usuaris.

13.4. Annex D: Arxiu SQL de creació de Base de Dades i Taules del PFC.

Per últim, seguidament es pot veure l'arxiu SQL, “**pfc.sql**”, de creació de la base de dades i les taules necessàries per a l'execució del PFC:

```
CREATE DATABASE IF NOT EXISTS pfc;

USE pfc;

/* Taula del Metge */
CREATE TABLE IF NOT EXISTS metge (dni_nss VARCHAR (17) NOT NULL,
                                  nom VARCHAR (25) NOT NULL,
                                  cognoms VARCHAR (30) NOT NULL,
                                  ncolegiat INTEGER NOT NULL,
                                  especialitat VARCHAR (30) NOT NULL,
                                  certificate TEXT,
                                  llis_pac TEXT,
                                  CONSTRAINT pk PRIMARY KEY (dni_nss))
                                  ENGINE=InnoDB;

/* Taula per a les visites */
CREATE TABLE IF NOT EXISTS visita (id_visita BIGINT NOT NULL,
                                   datai DATE NOT NULL,
                                   horai TIME NOT NULL,
                                   tema VARCHAR (80) NOT NULL,
                                   metge VARCHAR (17) NOT NULL,
                                   anamnesi TEXT,
```

Esquema criptogràfic per gestionar de forma segura els historials mèdics dels pacients a través d'una xarxa de comunicacions.

```
                                    diagnosi TEXT,
                                    tractament TEXT,
                                    signature TEXT,
                                    CONSTRAINT pk
                                              PRIMARY KEY (id_visita),
                                    CONSTRAINT fk_met
                                            FOREIGN KEY (metge)
                                            REFERENCES metge (dni_nss))
                                    ENGINE=InnoDB;

/* Taula per a l'historial mèdic del pacient */
CREATE TABLE IF NOT EXISTS historial (dni_nss VARCHAR (17) NOT NULL,
                                      nom VARCHAR (25) NOT NULL,
                                      cognoms VARCHAR (30) NOT NULL,
                                      num_tars INTEGER NOT NULL,
                                      grup_sang VARCHAR (5) NOT NULL,
                                      alergies TEXT,
                                      vacunes TEXT,
                                      observacions TEXT,
                                      certificate TEXT,
                                      llis_vis TEXT,
                                      llis_met TEXT,
                                      CONSTRAINT pk
                                                PRIMARY KEY (dni_nss))
                                      ENGINE=InnoDB;

/* Taula per a dades temporals d'autenticació */
CREATE TABLE IF NOT EXISTS autenticacioPR2 (num_user BIGINT NOT NULL,
                                            num_gestor BIGINT NOT NULL,
                                            user_cert TEXT,
                                            CONSTRAINT pk
                                            PRIMARY KEY
                                            (num_user, num_gestor))
                                            ENGINE=InnoDB;

/* Taula per a gestors */
CREATE TABLE IF NOT EXISTS gestor (fingerprint VARCHAR (50) NOT NULL,
                                   certificate TEXT,
                                   CONSTRAINT pk
                                   PRIMARY KEY (fingerprint))
                                   ENGINE=InnoDB;
```