ÍNDEX

1. INTRODUCCIÓ

La gestió dels serveis mèdics que s'ofereixen al ciutadà és de gran importància ja que tracta un tema tan essencial com és la salut de les persones. Tot i això, la definició d'un sistema que permeti atendre ràpidament i eficientment les necessitats de l'individu suposa un gran esforç d'organització degut a la gran quantitat de factors que hi intervenen (gestió de centres de salut, horaris, material, personal mèdic, llistes d'espera...). A més, molts d'aquests elements estan físicament allunyats els uns dels altres i en un moment donat es pot requerir una comunicació entre ells (com per exemple per manegar l'establiment d'una visita amb un especialista en un hospital en funció dels resultats obtinguts per un metge de medicina general en un CAP, o per consultar la diagnosi o els tests mèdics realitzats en una visita anterior d'una altra especialitat mèdica).

En aquest cas, la utilització d'un sistema informàtic que gestioni tots aquests elements podria facilitar la interacció entre ells i en milloraria l'eficiència (permetent per exemple l'accés ràpid a l'historial d'un pacient o la planificació òptima dels horaris del personal mèdic).

Tot i que avui dia molts centres ja estan informatitzats, encara manquen temes importants com són:

- La coordinació eficient entre diversos centres per a la gestió dels recursos mèdics. Avui dia cada centre (CAP, hospital...) té una organització interna pròpia independent de forma que, quan s'ha de canviar d'un a un altre, es fa necessari enviar les dades de forma explícita i manualment (els programes de gestió no són compatibles i no hi ha una comunicació directa a través de xarxa).

- El manegament de la informació (ex: historials mèdics) es realitza de forma manual i descentralitzada, cosa que dificulta la consulta i modificació de les dades que es troben disperses en vàries fonts de dades (algunes en paper, altres en BD locals...).

- El servei de cara al ciutadà que ha de traslladar-se al centre en qüestió o trucar per telèfon per realitzar certes tasques d'àmbit administratiu que es podrien fer de forma remota (a través d'internet) i autònoma (cosa que requeriria, per altra banda, unes certes garanties de seguretat).

Una millora en la gestió d'aquests aspectes repercutiria en una reducció del personal necessari per encarregar-se de les feines administratives, un millor aprofitament dels recursos sanitaris i facilitaria l'accés a la informació. Tot això contribueix a millorar l'accés del ciutadà als serveis mèdics i fer que el personal mèdic treballi de forma més eficient.

Un cop exposada la problemàtica, ens trobem amb la dificultat d'escollir el sistema informàtic que més s'adapti a les necessitats del problema. En aquest cas, es podrien pensar aproximacions tradicionals com seria una aplicació distribuïda amb una BD

centralitzada on accedeixin el personal mèdic i els pacients o un sistema basat en un entorn web. Tot i això, aquests models tenen l'inconvenient de que són difícilment adaptables a un entorn dinàmic com és aquest cas (com per exemple quan un metge no està disponible, la durada de les visites s'allarga o s'escurça més del previst o es produeix alguna alteració en la planificació inicial de les tasques). A més, no existeix cap mecanisme de personalització del servei a banda de la pròpia informació proporcionada.

Per aquests motius, seria interessant que el sistema implementat fos capaç de:

- Adaptar-se ràpidament als canvis que es produïssin en l'entorn (variació d'horaris, alteració de visites preprogramades, indisposició d'un metge...) de forma que els elements que hi intervenen replanifiquessin de forma coordinada les tasques a realitzar (reestructurant les visites amb els pacients, buscant un substitut...).

- Permetre un accés a la informació ràpid i eficient de forma centralitzada de manera que, per exemple, un metge d'un centre qualsevol pogués tenir accés complet en tot moment a la historia mèdica completa d'un pacient i pogués actualitzar-la immediatament amb els resultats d'una nova observació.

- Implementar un model d'execució distribuïda en el que cadascun dels elements que formen part del sistema mèdic (metges, infermers, personal administratiu...) tinguessin a la seva disposició un terminal d'accés independent i personalitzat amb les seves necessitats però en permanent comunicació amb la resta del sistema. Així, un metge tindria a la seva disposició un ordinador personal on s'executaria un mòdul del programa especialment adaptat als requeriments que aquest pot demanar (ex: consultar i actualitzar l'historial d'un pacient, demanar la realització de certes proves o anàlisis mèdiques...).

- Adaptar-se automàticament a les necessitats que sorgeixen, proposant una solució el més òptima possible. Així, per exemple, si el resultat d'una visita requereix la realització d'una radiografia, seria convenient que automàticament es proposés i es concertés la data i el centre més adequats.

- Emmagatzemar totes les dades de forma informàtica de manera que s'agilitzés l'accés complet a la informació de diverses fonts (ex: dins de l'historial mèdic podrien emmagatzemar-se les radiografies realitzades, els resultats dels anàlisis de sang...).

- Oferir un mètode d'accés als usuaris senzill i còmode des del que es poguessin realitzar totes les accions que es necessitin de forma personalitzada per tal que:

    - Manegui les seves dades personals de forma transparent, proporcionant-les a la resta del sistema sempre que es necessitin sense intervenció de l'usuari però de forma segura.
    - Prengui una certa iniciativa en determinats aspectes com és la gestió d'una visita mèdica quan es requereixi (oferint per exemple les millors opcions d'entre les propostes rebudes).

- Doni accés a tota la informació que es pugui requerir de tot el sistema que es pretén modelar.
- Faciliti l'accés de la informació que es proporciona a l'individu realitzant un cert pretractament (ordenant-la o filtrant-la segons certs criteris en funció de les característiques i preferències de l'usuari).
- Permeti l'accés des de qualsevol lloc a les funcionalitats que s'ofereixen (ex: ordinadors, agendes electròniques, telèfons mòbils...).

Quan tractem d'implementar alguns d'aquests comportaments complexos és el moment de recórrer a tècniques d'IA que realitzin un cert manegament "intel·ligent" de les dades. Per aconseguir aquest objectiu, existeixen diverses tècniques com són per exemple: representacions del coneixement, planificació, sistemes d'ajuda a la pressa de decisions... Tot i això, aquests mecanismes suposen un complement als mètodes de desenvolupament tradicional però no pemetrien implementar un sistema complert (amb gestió de dades, execució distribuïda, coordinació entre els diversos elements...).

És precisament en aquestes situacions on els *Sistemes Multi-Agent* (SMA) (una de les tècniques més recents de l'IA) poden suposar una bona alternativa doncs permeten modelar de forma completa problemes complexos a base de definir unitats més petites (agents) que realitzen tasques de forma autònoma (no es requereix personal de gestió) i que es comuniquen i coordinen entre ells per tal d'arribar a aconseguir la solució global.

Així, els SMA no només implementen tècniques d'IA (com el tractament de dades intel·ligent, l'aprenentatge...) sinó que a més aporten una solució distribuïda (on cada agent es pot executar en una màquina diferent), una gestió de la xarxa transparent i un mecanisme de comunicació molt elaborat.

Donat que una de les principals característiques dels agents és la col·laboració, es fan necessaris certs mecanismes estàndards que assegurin la interoperabilitat dels sistemes. En aquest cas, es requereix d'una entitat que s'encarregui de definir l'especificació d'aquests estàndards de forma que diversos SMA puguin interactuar entre ells.

1.1 FIPA

La FIPA (*Foundation for Intelligent Physical Agents*) [27] és una fundació sense ànim de lucre amb seu a Ginebra (Suïssa). La seva missió fonamental és establir totes les regles que han de governar el disseny i implementació d'un SMA per tal d'aconseguir interoperabilitat entre sistemes. Des de 1997 treu especificacions que a poc a poc s'han anat imposant arreu i s'han convertit en estàndards *de facto* entre la comunitat.

Aquest projecte s'ha realitzat amb l'ajut d'una eina de desenvolupament de SMA que compleix les especificacions de la FIPA, per la qual cosa, es posarà èmfasi en aquestes regles de disseny i implementació.

1.2 Agentcities

Agentcities [2] és un ambiciós projecte que té com a principal objectiu la construcció d'una xarxa mundial pública de sistemes oberts que ofereixin serveis similars als que es poden trobar a una ciutat real. Dins d'aquests entorns, s'inclouen les plataformes d'agents basades en els estàndards de la FIPA.

Els agents que ja han estat inclosos en aquesta xarxa es centren en oferir serveis sobre cinemes, teatres, restaurants, hotels, taxis, turisme... Actualment, hi ha al voltant de 50 plataformes actives.

S'espera que en un futur pròxim, serà possible implementar serveis complexos que inclouen un conjunt de funcionalitats simples de forma coordinada per aconseguir una tasca (ex: planificar un cap de setmana incloent tasques com reserva de bitllets per desplaçaments, selecció d'habitació en hotels, compra d'entrades per teatre o reserva de restaurants segons els gustos de l'usuari).

1.3 Grup de recerca Banzai – GruSMA

El grup de recerca d'IA Banzai [6] de la URV és un dels equips d'investigació del departament d'Enginyeria Informàtica i Matemàtiques creat al 1992. Està format per diferents professors i estudiants de Doctorat.

Banzai té com una de les línies d'investigació el disseny d'aplicacions d'IA sobre temes mèdics (ex: trasplantaments d'òrgans) i dins d'aquest trobem el grup de treball GruSMA [30] (Grup de Sistemes Multi-Agent creat per Aïda Valls i Toni Moreno) que s'encarrega de desenvolupar diversos projectes de SMA.

1.4. El Prototipus

Quan l'iniciativa d'Agentcities es va fer pública, els coordinadors de GruSMA es van interessar en la possibilitat d'oferir serveis mèdics a través d'una xarxa de SMA doncs aquests ofereixen un comportament que permet modelar de forma eficient problemes distribuïts complexos, tal com s'ha indicat en el primer apartat.

Així, es va decidir dissenyar i construir un primer SMA que donés solució a alguns dels problemes anteriorment exposats, modelant el comportament i l'organització de certs dels elements que intervenen en un entorn mèdic i servís de base per realitzar futures ampliacions. Les característiques d'aquest prototipus inicial eren:

- L'usuari podria demanar informació sobre centres mèdics disponibles en una determinada ciutat.
- Si l'usuari coneix un determinat centre mèdic, podrà demanar informació sobre els serveis i doctors disponibles en aquest.
- Cal que es puguin concertar visites amb un metge.

- Durant la visita, cal que el doctor tingui accés a l'historial mèdic del pacient per tal d'actualitzar-lo (incloent el resultat, les proves realitzades...).
- Tal com indica la llei Catalana d'informació mèdica [29], cal que tot pacient tingui accés al seu historial mèdic.

Amb aquests objectius es va desenvolupar un PFC de segon cicle realitzat per David Isern [35] i que és la base d'aquest projecte.

1.5. Objectius del projecte

A partir del prototipus, es va proposar realitzar un conjunt de millores que permetessin apropar el sistema a un entorn d'execució real demostrant la seva utilitat. Així, les noves metes que es van marcar van ser les següents:

- Completar la funcionalitat bàsica, incloent l'execució concurrent i distribuïda d'agents en diversos ordinadors. D'aquesta forma, l'usuari podria accedir als serveis del sistema de forma remota a través d'un agent *Personal* extern.

- Actualitzar la implementació del prototipus a l'ultima versió de l'eina de desenvolupament d'agents per tal d'aprofitar les noves característiques (referents a eficiència, protocols de comunicació i definició millorada de dominis de comunicació (Ontologies), així com la possibilitat d'integrar certs mecanismes de seguretat).

- Oferir aquests serveis accessibles per a tothom a través d'una plataforma d'agents (servidor proporcionat per Agentcities).

A més, també es van marcar un conjunt de metes que representin una innovació en el camp dels agents com són:

- Estudiar diferents alternatives per al disseny d'Ontologies (dominis de representació de les dades per intercanviar-les de forma eficient) en el camp mèdic [44] per tal d'aplicar-les en el projecte i aproximar-lo el més possible a un entorn real. El resultat es proporcionarà a la xarxa Agentcities en un llenguatge de representació estàndard com RDF [71] de forma que sigui fàcilment aplicable a altres projectes de l'estil. Els missatges creats amb aquesta ontologia també es codificaran amb un llenguatge estàndard per tal que puguin ser fàcilment interpretables per altres aplicacions no directament relacionades amb agents.

- Proporcionar un model de seguretat per al xifratge de la informació i l'accés a dades confidencials (historials mèdics) en un SMA [53]. Aquest és un punt molt important doncs, fins ara, les aplicacions basades en agents no l'havien tingut en compte i la FIPA encara no ha definit un estàndard sobre el tema.

Aquesta proposta es va presentar a Agentcities, que va proporcionar els recursos tècnics (Beca amb suport econòmic i Servidor per col·locar l'aplicació) que l'han feta possible.

Aquest projecte doncs, engloba el treball realitzat durant els 5 mesos en els que es van desenvolupar els punts anteriors.

A dia d'avui es pot dir que aquests objectius s'han acomplit, i una versió funcional del sistema està corrent permanentment sobre el servidor proporcionat per Agentcities dins de la xarxa mundial d'agents. A més, el sistema també disposa d'una pàgina web [31] en la que es descriuen detalladament totes les seves característiques i on s'explica la forma en què qualsevol usuari pot accedir i demanar serveis remotament al sistema des del seu ordinador connectat a Internet.

Aquesta aplicació ha estat presentada a *l'Agentcities Agent Technology Competition 2003* [1] en la que participaven 39 grups de recerca d'arreu del món i on va guanyar el tercer premi dins de la categoria d'aplicacions i el premi especial del públic atorgat pels assistents a l'acte.

1.6. Estructura del document

En aquest document es pretén realitzar una descripció detallada de tota l'aplicació, des de les característiques i funcionalitats del prototipus de partida fins a les noves característiques implementades, així com els resultats de les investigacions realitzades en els àmbits d'innovació (ontologies, seguretat i migració del codi a les noves versions de JADE).

En primer lloc es realitzarà una introducció als conceptes fonamentals sobre els Agents i els Sistemes Multi-Agent. A continuació es passarà a descriure l'eina de desenvolupament de Sistemes Multi-Agent (JADE). En el següent capítol es realitzarà una justificació del sistema i s'explicarà l'arquitectura bàsica del prototipus en la que es basa el present projecte. Seguidament es parlarà sobre la investigació i els resultats obtinguts durant la recerca de l'Ontologia que es farà servir en el projecte, per després passar a descriure la problemàtica de la seguretat en aquests tipus de sistemes i el model que finalment s'ha implementat. Més tard, es descriuran detalladament tots els serveis que ofereix el sistema. A continuació s'inclourà un manual d'usuari que explica la forma d'accedir fàcilment als serveis de forma remota indicant les característiques del sistema mèdic que s'està executant actualment al servidor. Finalment s'exposaran les conclusions del projecte i les línies de treball futur.

A més de tot això, també es disposa d'un apèndix en el que es descriu de forma detallada les claus per programar i migrar codi a l'última versió de l'eina de desenvolupament (JADE), fruit de la investigació realitzada en aquest camp i que pot resultar especialment útil per als programadors. En l'apèndix també s'inclou l'especificació completa de l'Ontologia definida en un llenguatge de representació estàndard, un exemple de fitxer de configuració per als agents i el contingut de la BD implementada.

2. AGENTS I SISTEMES MULTI-AGENT

Un agent intel·ligent és un procés computacional capaç de realitzar tasques de forma autònoma i que es comunica amb altres agents per resoldre problemes mitjançant cooperació, coordinació i negociació. Habiten en un entorn complex i dinàmic amb el qual interaccionen en temps real per aconseguir un conjunt d'objectius [76].

2.1. Propietats dels agents

Les propietats més importants que poden tenir els Agents són les següents:

- Autonomia: és la capacitat d'operació sense la intervenció directa dels humans o altres, i tenir algun tipus de control sobre les pròpies accions i l'estat intern.
- Sociabilitat/cooperació: els agents han de ser capaços d'interactuar amb altres agents i/o humans a través d'algun tipus de llenguatge de comunicació.
- Reactivitat: els agents perceben el seu entorn i responen en un temps raonable als canvis detectats.
- Pro-activitat o capacitat de prendre iniciativa: han de ser capaços d'exhibir algun tipus de comportament encaminat a assolir algun tipus d'objectiu prenent la iniciativa.
- Mobilitat: possibilitat de moure's a altres entorns a través d'una xarxa electrònica.
- Continuïtat temporal: els agents estan contínuament executant processos.
- Veracitat: es l'assumpció que un agent no comunicarà informació falsa premeditadament.
- Benevolència: es l'assumpció que els agents no tenen objectius conflictius, i que cada agent intentarà fer allò pel qual és requerit.
- Racionalitat: l'agent ha d'actuar per tal d'aconseguir el seu objectiu.
- Aprenentatge: milloren el seu comportament amb el temps.
- Intel·ligència: utilitzen tècniques d'IA per resoldre els problemes i aconseguir els seus objectius.

2.2. Arquitectures d'agents

Les arquitectures utilitzades en el procés d'implementació dels agents poden ser de tres tipus:

- Deliberatives: donada una representació del món real, es poden executar accions intel·ligents. Segueix el model BDI (*Beliefs Desires and Intentions*).

- Reactives: estructura on les accions es duen a terme responent a estímuls. La combinació de diversos agents reactius proporciona un comportament "intel·ligent".

- Híbrides: estructures que barregen diferents tècniques.

2.3. Tipus d'agents

Segons la finalitat amb la que s'implementi un agent, podem classificar-lo en diferents tipus:

- D'informació: gestionen i manipulen dades; poden respondre a requeriments del usuaris o altres agents. Un exemple molt comú són els cercadors d'informació a Internet.

- D'interfície: sistema on els agents col·laboren amb l'usuari per resoldre un problema. La interacció amb l'individu permet a l'agent desenvolupar un aprenentatge basat en les accions que s'executen.

- De col·laboració: la característica principal és la comunicació i cooperació amb altres agents per resoldre un problema comú. Utilitzen tècniques d'IA distribuïda.

- Mòbils: la seva característica principal és la possibilitat de poder moure's per una xarxa electrònica recollint informació o interactuant amb altres hosts.

- Reactius: el seu comportament es basa en la resposta a estímuls segons un patró de l'estat actual en què es troben. Són molt simples, però la combinació de diversos agents reactius por generar comportaments complexos.

- Híbrids: són combinacions de diversos tipus anteriors intentant reunir els avantatges d'uns i altres.

- Heterogenis: es refereixen a un conjunt integrat d'almenys dos agents que pertanyen a dues o més classes diferents.

2.4. Sistemes Multi-Agent (SMA)

Un sistema Multi-Agent serà aquell en el que un conjunt d'Agents cooperen, es coordinen i es comuniquen per tal d'aconseguir un objectiu comú.

2.4.1. Avantatges d'un SMA

Les principals avantatges de la utilització d'un sistema Multi-Agent en la gestió d'un problema són:

- Modularitat: es redueix la complexitat en treballar amb unitats més petites, permetent una programació més estructurada.

- Eficiència: la programació distribuïda permet repartir les feines entre els agents, aconseguint paral·lelisme (agents treballant en diferents màquines).

- Fiabilitat: el fet que un element del sistema deixi de funcionar no ha de significar que la resta també ho facin; a més, es pot aconseguir més seguretat replicant serveis crítics i així aconseguir redundància.

- Flexibilitat: es poden afegir i eliminar agents dinàmicament.

2.4.2. Gestió d'un SMA

L'administració d'agents estableix el model lògic per a la creació, registre, comunicació, establiment, mobilitat i destrucció d'agents. La FIPA estableix el següent model per aquest entorn d'administració d'agents [19]:

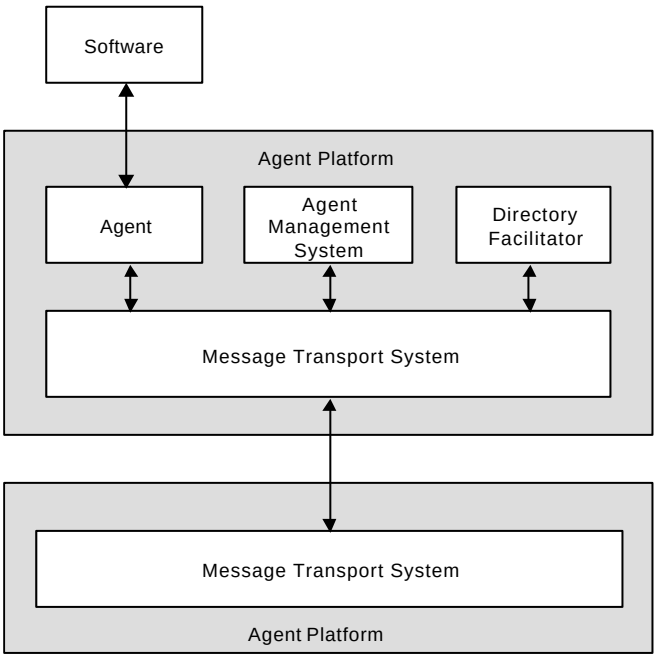

Figura 1 - Aspecte intern d'un Sistema Multi-Agent

Els components lògics principals que formen part del sistema de la **figura 1** són:

- *Agent*: unitat bàsica. Es pot descriure com un programa que encapsula una sèrie de serveis

- *Directory Facilitator* (DF): és un agent que proporciona un servei de Pàgines Grogues dins del sistema (coneix els serveis que poden oferir els diferents agents del SMA). Els agents han de registrar-se en el DF per a oferir els seus serveis.

- *Agent Management System* (AMS): és l'agent que controla l'accés i ús de la plataforma. Emmagatzema les direccions dels agents, oferint un servei de Pàgines Blanques.

- *Message Transport System* (MTS): facilita la comunicació entre agents de diferents plataformes.

- *Agent Platform* (AP): proporciona la infrastructura bàsica en la que poden crear-se i executar-se els agents.

- *Software*: qualsevol sistema informàtic accessible des d'un agent (p.e. una base de dades).

2.4.3. Llenguatge de comunicació entre agents (ACL)

Els agents individualment proporcionen una funcionalitat interessant, però el que els fa tant adequats per a determinats sistemes és la seva capacitat de cooperar per a resoldre problemes. Per poder fer-ho, els agents s'han de comunicar utilitzant un llenguatge comú : ACL (*Agent Comunication Language*).

Els ACL estan basats en la "teoria de la parla", en la qual les comunicacions individuals entre agents poden reduir-se a un petit nombre d'idees o, més generalment, actes comunicatius. Aquests actes donen forma al significat bàsic de la comunicació.

L'element bàsic de la comunicació és el missatge; la FIPA determina quina forma ha de tenir un missatge i la seva utilització [18]. En la **figura 2** observem els components bàsics que ha de tenir un missatge.

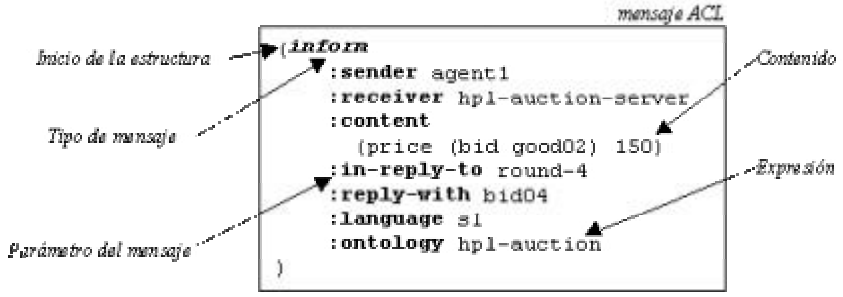

Figura 2 - Aspecte d'un missatge

2.4.3.1. Elements d'un missatge

Els elements que de forma estàndard formen part d'un missatge de comunicació entre agents són:

- Definició de tipus de missatge (*performative*): indica el tipus d'acte de comunicació. Les accions més habituals són:

    - **accept-proposal**: acceptació d'una proposta feta anteriorment per realitzar una acció.
    - **agree**: acceptació d'una acció.
    - **cancel**: cancel·lació d'una acció demanada anteriorment.
    - **cfp**: crida per proposar la realització d'una acció.
    - **failure**: indica que l'acció demanada anteriorment ha fallat per algun motiu.
    - **inform**: l'emissor informa al receptor que una acció s'ha realitzar correctament.
    - **not-understood**: l'emissor informa al receptor que no ha entès una petició realitzada anteriorment.
    - **propose**: acció de proposar la realització d'una acció, donades certes precondicions.
    - **query-if**: acció de preguntar a un altre agent si un fet és cert o no.
    - **refuse**: negació a la realització d'una acció donada.
    - **request**: l'emissor demana al receptor realitzar un acció.
    - **subscribe**: l'emissor demana ser avisat en el moment que es compleixi una condició.

- Interlocutors: són identificadors d'agents. N'hi ha 3: **sender** (qui envia el missatge), **receiver** (qui el rep), **reply-to** (a qui ha d'anar destinat el següent missatge de la conversa).

- Contingut del missatge: existeixen quatre tipus de continguts predefinits que s'utilitzen en funció de les necessitats que es tinguin.

    - FIPA-CCL (*Constraint Choice Language*): defineix una semàntica que permet especificar predicats amb satisfacció de restriccions (CSP).
    - FIPA-SL (*Semantic Language*): permet formar expressions lògiques, intercanviar coneixement de forma òptima i expressar accions a realitzar. És el llenguatge més general de tots i pot ser aplicat a molts dominis diferents.
    - FIPA-KIF (*Knowledge Interchange Format*): permet expressar objectes com a termes i proposicions com a sentències.
    - FIPA-RDF (*Resource Description Framework*): permet representar objectes, interaccions i accions entre ells. És el que s'utilitzarà en aquest projecte.

- Descripció del contingut: permet que l'agent que rebi el missatge pugui identificar què se li està enviant i en quin format. Es defineixen 3 descriptors a tal efecte:
    - **language**: en què està escrit el contingut, en aquest cas utilitzarem el FIPA-RDF.
    - **encoding**: indica si s'utilitza algun tipus de codificació especial.
    - **ontology**: és el camp més important, ja que defineix la ontologia utilitzada per donar significat al contingut del missatge.

- Control de la conversa: està destinat a identificar quin tipus de converses es mantenen. Els seus camps informatius són:
    - **protocol:** utilitzat en el missatge.
    - **conversation-id:** identificador de conversa.
    - **reply-with:** identificador de missatge que s'utilitza per seguir els diferents passos de la conversa.
    - **in-reply-to:** identifica una acció passada a la qual aquest missatge és la resposta.
    - **reply-by:** marca de *timeout*: data/hora de caducitat del missatge.

2.4.3.2. Protocols de comunicació

Abans s'ha indicat que un dels camps de control de conversa identifica el tipus de protocol que es fa servir. El protocol utilitzat és el que defineix una sèrie de regles o passos que s'han de seguir per a desenvolupar una conversa. N'hi ha de diferents tipus depenent de la seva finalitat. El conjunt de protocols disponibles i el seu funcionament també són especificats per la FIPA [22]:

- FIPA-Request: permet demanar la realització d'accions i retornar els resultats.

- FIPA-Request-When: variant de l'anterior (amb condició de final).

- FIPA-Query: s'utilitza per fer preguntes.

- FIPA-Contract-Net: és un protocol de negociació (es proposen i s'avaluen propostes).

- FIPA-Iterated-Contract-Net: variant de l'anterior que permet la renegociació.

- FIPA-Propose: permet gestionar propostes.

- FIPA-Brokering: gestiona els serveis disponibles pels agents.

- FIPA-Recruiting: variant de l'anterior.

- FIPA-Subscribe: permet la notificació de fets importants.

- FIPA-Propose: simplificació del FIPA-Contract-Net.

3. JADE

JADE (*Java Agent Development Enviroment*) [63] és una eina de programació que conté un conjunt de llibreries escrites en JAVA per al desenvolupament de Sistemes Multi-Agent. A més de proporcionar les funcions de manegament d'agents a baix nivell i les interfícies que faciliten la programació, també ens presenta un entorn d'execució on els agents podran executar-se i interactuar.

La versió utilitzada per a la implementació d'aquest projecte és la 2.61 juntament amb el *plug-in* de seguretat JADE-S [64] i el *parser* de RDF [65].

3.1. Paquets de JADE

JADE està composat per una sèrie de paquets escrits en JAVA. Els principals són els següents:

- **jade.core**: és el nucli del sistema. Proporciona la classe **jade.core.Agent** que és imprescindible per a la creació de SMA. També conté el paquet **jade.core.behaviour** que inclou les classes de comportaments suportats.

- **jade.content**: conté les classes específiques per a suportar un llenguatge de comunicació entre agents (ACL) així com la definició i utilització d'ontologies. Està format principalment pels paquets **jade.content.lang** i **jade.content.onto**.

- **jade.domain**: conté totes les classes per representar la plataforma d'agents i els models del domini, tals com entitats per a l'administració d'agents, llenguatges i ontologies (AMS, DF, ACC ...).

- **jade.proto**: paquet que conté els protocols d'interacció entre agents FIPA.

- **jade.tools**: trobem algunes eines que faciliten el desenvolupament d'aplicacions i l'administració de la plataforma:
    - *Remote Management*: Agent (RMA): actua com una consola gràfica per a l'administració i control del sistema.
    - *Dummy Agent*: és una eina de monitorització i depuració, composta per una interfície gràfica i un agent JADE. Permet crear missatges ACL i enviar-los a altres agents, podent visualitzar-los.
    - *Introspector Agent*: és una eina que permet la monitorització del cicle de vida d'un agent.
    - *Sniffer*: agent que intercepta missatges ACL i els visualitza.
    - *SocketProxyAgent*: agent que actua com un *gateway* bidireccional entre la plataforma JADE i una connexió TCP/IP.

3.2. Components principals de JADE

Per JADE, els agents resideixen en un entorn predefinit anomenat plataforma. Cada plataforma està dividida en contenidors i cadascun d'ells contindrà agents. Els contenidors poden entendre's com dominis d'agents.

La plataforma s'engega en una màquina determinada, mentre que els agents podran executar-se en qualsevol estació.

Tal com es defineix a la FIPA [16], en cada plataforma s'inicien dos agents que tenen unes funcions vitals per al funcionament del sistema:

- *Domain Facilitator* (DF): pot ser definit com un servei de "pàgines grogues", és a dir, contindrà tots els serveis que proporcionen els agents que hi estan donats d'alta.

- *Agent Management System* (AMS): es podria considerar com un servei de "pàgines blanques", on es registren les adreces dels agents que s'han donat d'alta.

Gràcies a aquests serveis, es proporciona molta flexibilitat i transparència. D'aquesta manera, si un agent A vol enviar un missatge a un agent B que desenvolupa la tasca T, l'agent A preguntarà al DF quin o quins agents poden fer-se càrrec de la tasca T. Quan es vulgui conèixer l'adreça d'una agent determinat, preguntarà a l'AMS. Per tal que aquest sistema funcioni correctament caldrà que, durant el procés d'inicialització de l'agent, aquest informi al DF de quin o quins serveis proporciona i quin tipus d'agent és (per tal de poder identificar-lo); en fer-ho, l'AMS li donarà una adreça física (depenent d'on s'executi), per tal que altres es puguin posar en contacte amb ell.

3.3. Agents i comportaments

Per poder crear agents en JADE, haurem d'extendre la classe **Agent** que ja es troba definida i anar omplint els camps i les funcions bàsiques que ens proporciona la interfície. Dos mètodes molt importants que s'han d'implementar necessàriament són **setup()** i **takedown()** que inicialitzen i finalitzen l'agent respectivament.

Un cop creat l'agent, haurem de donar-li funcionalitat. Això s'aconsegueix mitjançant la definició de *behaviours* (comportaments) que són els que controlen les seves accions. Un *scheduler* intern serà l'encarregat d'anar controlant automàticament l'execució de tots els comportaments.

Arribats a aquest punt, JADE ens proporciona un conjunt de comportaments bàsics que permeten implementar diferents protocols de comunicació.

Dins del grup de comportaments simples (sense fills) trobem els següents:

- **SimpleBehaviour**: representa un comportament atòmic que s'executa sense interrupcions. Pot ser de tipus **OneShotBehaviour** (si s'executa una sola vegada), o **CyclicBehaviour** (si executa una acció cíclicament).
- **ReceiverBehaviour**: permet engegar un comportament que espera la recepció d'un missatge.
- **SenderBehaviour**: és equivalent a l'anterior però des del punt de vista del qui envia el missatge.

El primer tipus ens permet implementar accions bàsiques que durà a terme l'agent de forma independent (consultar dades a una BD, interactuar amb l'usuari...), mentre que el segon i el tercer ens permeten implementar protocols de comunicació entre agents.

A més, també existeixen comportaments complexos que poden crear fills i atendre diverses peticions alhora. S'agrupen dins de la classe abstracte **CompositeBehaviour,** que té el següents subtipus:

- **SequencialBehaviour**: executa tots els seus fills de manera seqüencial i acaba quan tots els mètodes finalitzen correctament.
- **ParallelBehaviour**: executa concurrentment tots els seus fills i finalitza quan es compleix una condició determinada.
- **FSMBehaviour**: executa els seus fills segons una Maquina d'Estats Finits (FSM) definida per l'usuari.

3.4. Missatges

Pel que fa a la comunicació, JADE segueix les especificacions de la FIPA en quant a format dels missatges [18]. Així, un missatge estarà dividit en *slots* que ens donen una informació (llenguatge, protocol, contingut...), i sobre aquests, es defineixen funcions de creació (**set**) i consulta (**get**).

Els mètodes bàsics de què disposem per al manegament dels missatges entre agents són:

- **Send (ACLMessage)**: utilitzat per enviar un missatge un cop omplerts tots els seus paràmetres.
- **ACLMessage receive()**: rep un missatge de la cua de l'agent. Es pot especificar quin tipus de missatge es vol rebre mitjançant els patrons de cerca (*MessageTemplate*).

Com a llenguatge de codificació de missatges es pot utilitzar qualsevol dels definits per la FIPA [21]. JADE suporta de forma nativa el llenguatge FIPA-SL (*Semantic Language*) [24], tot i que se'n poden afegir d'altres mitjançant *plug-ins* que implementin el *parser* corresponent. En el nostre cas, tal com veurem més endavant, utilitzarem el llenguatge FIPA-RDF [23].

3.5. Ontologia

Un dels camps que haurem de definir en el moment d'enviar un missatge, és l'ontologia que utilitzarem per escriure el contingut. Aquest punt és molt important, ja que defineix el vocabulari emprat (i per tant el significat i la forma d'accedir a les dades).

Una ontologia permet definir els següents punts:

- *Frames*: objectes sobre els quals es podrà parlar i que, per tant, estaran disponibles per omplir el missatge.
- *Slots*: camps d'informació en cada objecte o *frame*.
- Etiqueta (amb la que s'accedirà), tipus de dades (*String*, *long*, *float*, compost...) que conté cada *slot* i altres característiques com són l'opcionalitat o la cardinalitat de cada element.

A més de definir l'ontologia en un mòdul a banda (*frames*, *slots* i tipus de dades que contenen), també serà necessari definir i implementar una classe que s'encarregarà de manegar cada objecte o predicat (*frame*) creat. Aquesta classe haurà de contenir mètodes **set** i **get** per a cada *slot* que haguem definit. Aquestes funcions seran les que ens permetran consultar i omplir les dades del missatge.

Amb tots aquest elements, podrem crear totes les entitats que formaran la nostra ontologia i que es poden classificar en tres grans grups:

- Conceptes: defineixen els objectes amb els que tractem (ex: hospital, metge...).
- Predicats: defineixen una sintaxi lògica que permet demanar peticions en base a un conjunt de restriccions (ex: "Obtenir pacient amb ID=1111").
- Accions: es refereixen als serveis que poden oferir-se els agents (ex: "Donar d'alta un pacient").

Per obtenir una explicació més detallada i informació sobre la implementació d'Ontologies en JADE 2.61, consultar l'**apèndix A.3**.

3.6. Protocols de comunicació

Un cop definida i implementada la ontologia, ens caldrà escollir el protocol de comunicació entre agents que més s'adapti a les nostres necessitats. En aquest punt, JADE també ens proporciona els element necessaris (*behaviours* i *performatives*) per definir els protocols descrits per la FIPA [22].

Per a cada conversa, JADE distingeix dos rols: el que la inicia (*initiator*) i el que la segueix (*responder*) i cada agent haurà d'extendre un comportament en funció del seu paper.

3.6.1. Tipus de protocols

Des de les últimes versions, JADE suporta la majoria dels protocols definits per la FIPA tot i que els més importants i els que hem utilitzat en aquest projecte són: FIPA-Request, FIPA-Query i FIPA-Contract-Net.

3.6.1.1. FIPA Request

S'utilitza per demanar la realització d'accions. El comportament que tindran els agents quan utilitzen aquesta especificació és el següent (veure **figura 3**):

- L'agent *initiator* envia un missatge de tipus **request** cap a l'agent amb el que vol interactuar (*responder*) per demanar-li la realització d'una acció.

- El receptor del missatge llegeix el contingut del missatge i obté l'acció que se li demana. L'agent estudiarà la petició i determinarà si la pot fer o no. Si la pot realitzar, enviarà un missatge **agree** a l'agent inicial i començarà a realitzar l'acció. Si per alguna raó no pot realitzar-la, enviarà un missatge **refuse** que informi de la raó d'aquesta negació.

- En qualsevol cas, si l'*initiator* no ha formulat bé el missatge i, per tant, no es pot desxifrar correctament, s'enviarà un missatge **not-understood**.

- En cas d'haver enviat un missatge d'**agree**, el *responder* haurà començat a realitzar l'acció. Quan l'acabi, enviarà un missatge de tipus **inform**. Aquest pot ser de dos tipus: **inform-done** que només notifica que l'acció ha estat finalitzada i **inform-ref**, que retorna un resultat. Si l'acció no s'ha finalitzat correctament, s'enviarà un missatge **failure** indicant la raó.

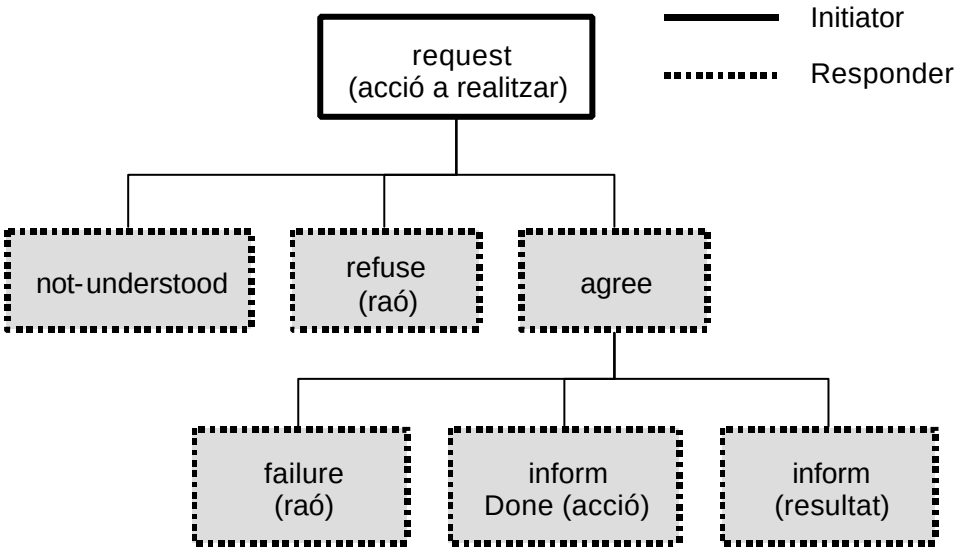

Figura 3 - Protocol FIPA-Request

3.6.1.2. FIPA Query

S'utilitza quan es vol demanar algun tipus d'informació. Els passos que se segueixen durant la conversa són els següents (veure **figura 4**):

- L'agent *initiator* formula una pregunta de tipus **query** o **query-ref**.

- L'agent *responder* podrà enviar la informació de la resposta o bé refusar la pregunta.

- Com sempre, si hi ha un error s'enviarà una fallada i si no se sap interpretar el contingut del missatge, un **not-understood**.

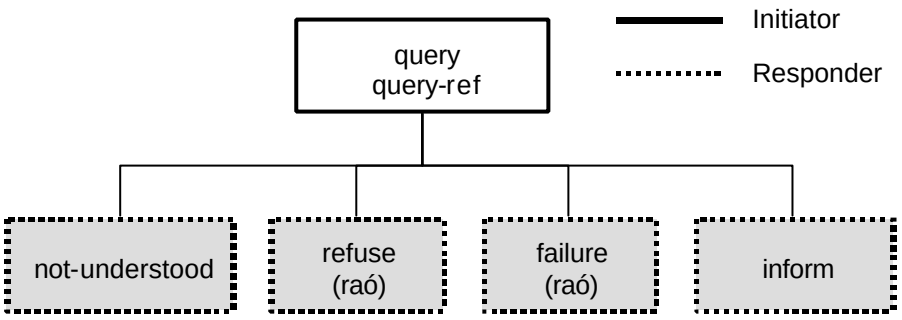

Figura 4 - Protocol FIPA-Query

3.6.1.3. FIPA Contract-Net

Aquest protocol s'utilitza quan es vol realitzar una negociació entre diversos agents per trobar aquell qui la pot realitzar millor segons els nostres interessos. Els passos que se segueixen són els següents (veure **figura 5**):

- L'agent *initiator* envia una petició de proposta d'acció passant-li unes precondicions a complir.

- L'agent *responder* avaluarà les premisses rebudes i, en funció de quines siguin proposarà realitzar o no l'acció. En el primer cas, enviarà una proposta d'acció amb unes condicions determinades; en el segon, un missatge de refús. Per tal que l'agent inicial sàpiga quan ha acabat de rebre totes les propostes a la seva petició, es disposa d'un *time-out* màxim.

- Si s'ha enviat una proposta, l'agent inicial l'avaluarà, i podrà o no acceptar-la enviant a l'agent *responder* el missatge corresponent.

- Finalment, l'agent *responder* començarà a realitzar l'acció, enviant un missatge de finalització (**inform**), una fallada (**failure**) en cas de produir-se algun error o una possible cancel·lació abans que acabi la tasca (**cancel**).

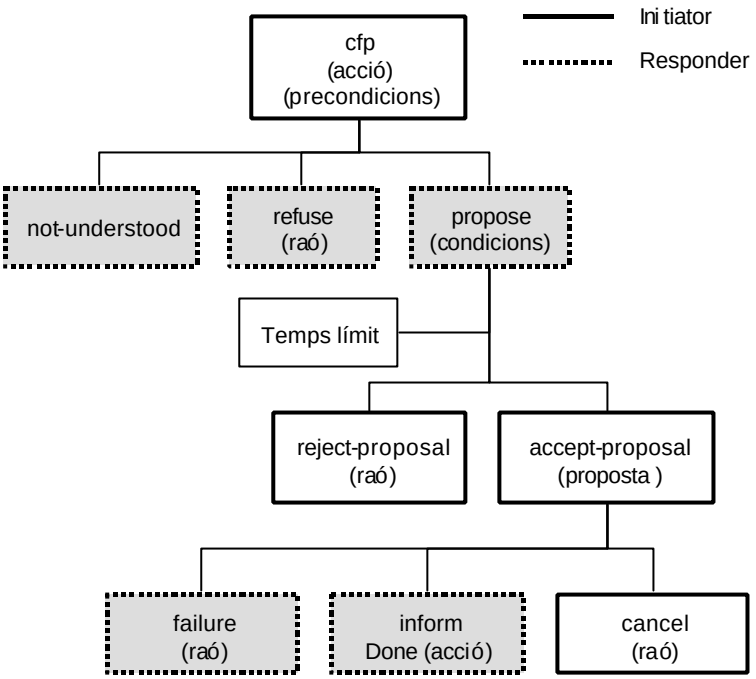

Figura 5 - Protocol FIPA-Contract-Net

3.6.2. Implementació de protocols

Tal com s'ha indicat abans, per poder implementar aquest protocols, JADE els divideix en dues parts: la que fa la crida inicial (*initiator*) i la que manega la resposta (*responder*). Per a cada comportament, el programador, només haurà d'omplir aquells mètodes que maneguen els diferents estats pels quals passa la conversa.

Les classes de què disposa JADE per implementar protocols són les següents:

- **AchieveREInitiator** / **AchieveREResponder**: implementen la banda de l'*initiator* i el *responder* respectivament per a protocols del tipus petició resposta. Concretament, se suporten els següents: FIPA-Request, FIPA-Query, FIPA-Propose, FIPA-Request-When, FIPA-Recruiting, FIPA-Brokering. Existeix una variant simplificada d'aquestes classes que són: **SimpleAchieveREInitiator** i **SimpleAchieveREResponder**.

- **ContractNetInitiator** / **ContractNetResponder**: implementen la banda de l'*initiator* i el *responder* per al protocol FIPA-Contract-Net.

- **SubscriptionInitiator** / **SubscriptionResponder**: implementen la banda de l'*initiator* i el *responder* per al protocol FIPA-Subscrive. Aquestes classes encara estan en fase de desenvolupament doncs la FIPA no ha donat una definició clara d'aquest protocol.

Per obtenir una explicació més detallada sobre els protocols de JADE 2.61, la seva implementació i les diferències amb versions anteriors, consultar l'**apèndix A.2**.

3.7. Consideracions generals

En resum els principals punts que s'han de considerar en la implementació d'agents en JADE són els següents:

- Definició i implementació d'un domini semàntic (vocabulari) comú als agents del sistema -> ontologia.

- Associació i implementació de comportaments (*behaviour*s) per a cada agent. La combinació d'aquests, ens permetrà definir protocols de comunicació.

- Cal que els agents publicitin al DF els serveis que proporcionen per tal que altres agents els puguin utilitzar.

3.8. Eines de JADE

JADE ofereix una interfície gràfica per a l'administració de la plataforma, així com eines per tal de facilitar la depuració i testeig de les aplicacions basades en ell.

Per poder iniciar una sessió i crear els agents, haurem d'escriure la següent comanda:

```
java jade.Boot [opcions] [llista d'agents]
```

En el camp d'opcions podem especificar que volem visualitzar l'entorn gràfic amb el paràmetre `-gui`. Pel que fa a la llista d'agents, serà una cadena de text on cadascun d'ells estarà separat per espais en blanc i seguirà el següent format:

```
<nomAgent>:ClasseJava
```

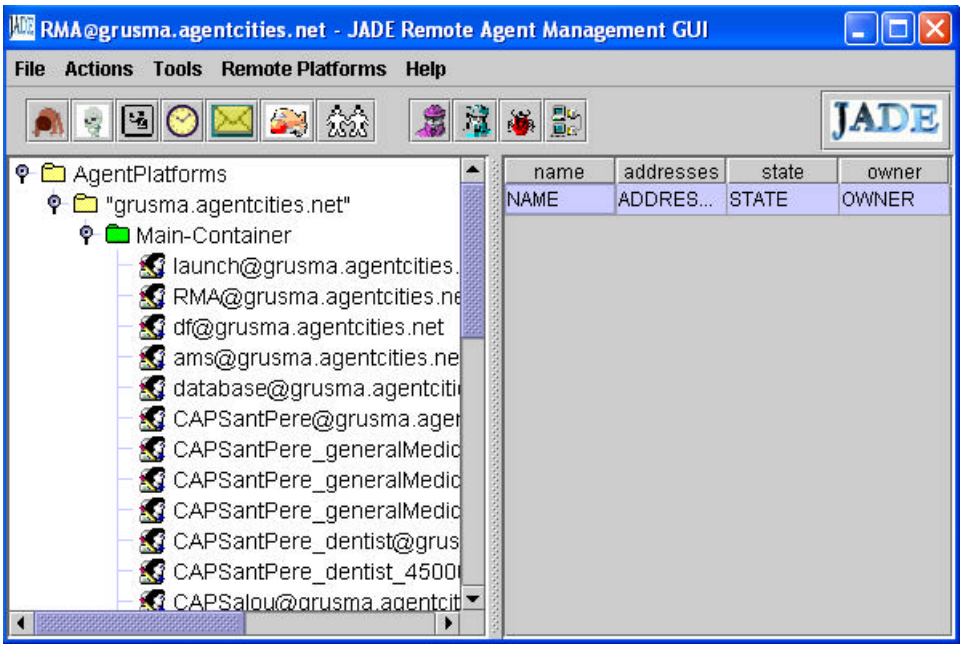

Figura 6 - Pantalla principal de JADE

Quan engeguem l'entorn gràfic, tal com es mostra a la **figura 6**, tot i que no s'inicialitzi cap agent propi, es creen uns quants automàticament:

- L'agent RMA serà l'encarregat de controlar la interfície.
- El DF, on se subscriuran els serveis dels agents.
- L'AMS on es guardaran les adreces dels agents.

A més de la llista d'agents que hi ha al contenidor i la informació sobre el que hem seleccionat, disposem d'una sèrie d'opcions (botons o menú *tools*) que ens facilitaran el manegament i testeig dels agents:

- Afegir un nou agent: haurem d'indicar el nom, la classe que l'implementa i el contenidor al que pertany.

- Eliminar els agents seleccionats.

- Suspendre els agents seleccionats.

- Continuar els agents seleccionats.

- Enviar un missatge als agents seleccionats: haurem d'omplir una finestra (veure **figura 7**) amb els camps corresponents: qui l'envia, qui el rep, protocol, acte comunicatiu, llenguatge, ontologia, contingut...

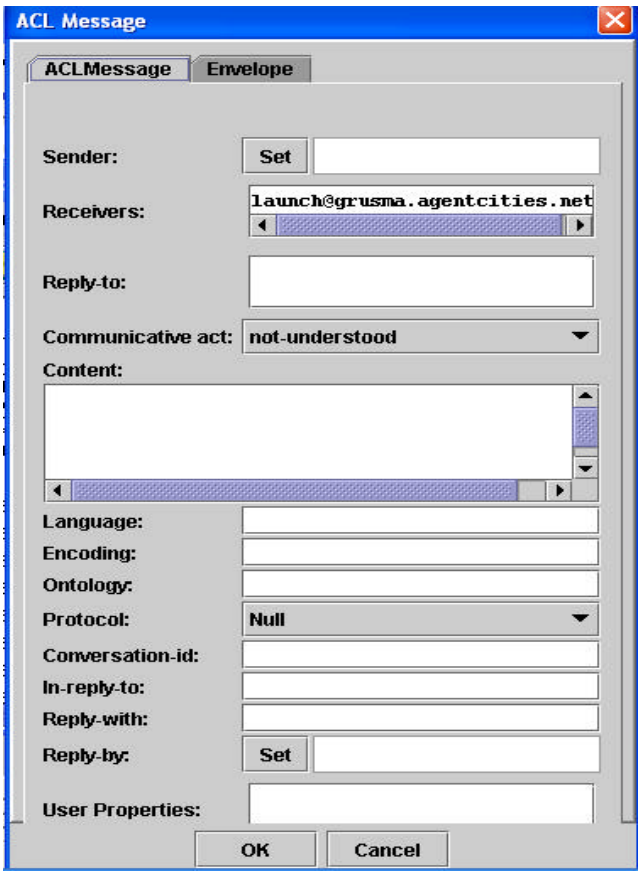

Figura 7 - Finestra per enviar un missatge

- Moure els agents seleccionats a un altre contenidor o plataforma.

- Clonar un agent creant-ne un d'igual.

- Executar l'agent *sniffer*: és una aplicació JAVA creada per rastrejar l'intercanvi de missatges entre els agents JADE. En activar-lo, s'obre una finestra (mostrada a la **figura 8**) on podem observar l'evolució temporal del pas de missatges. Si seleccionem alguna de les fletxes que simbolitzen aquest intercanvi, se'ns mostrarà el missatge amb tots els seus paràmetres. A més podrem seleccionar quins agents volem monitoritzar i guardar els missatges a disc.

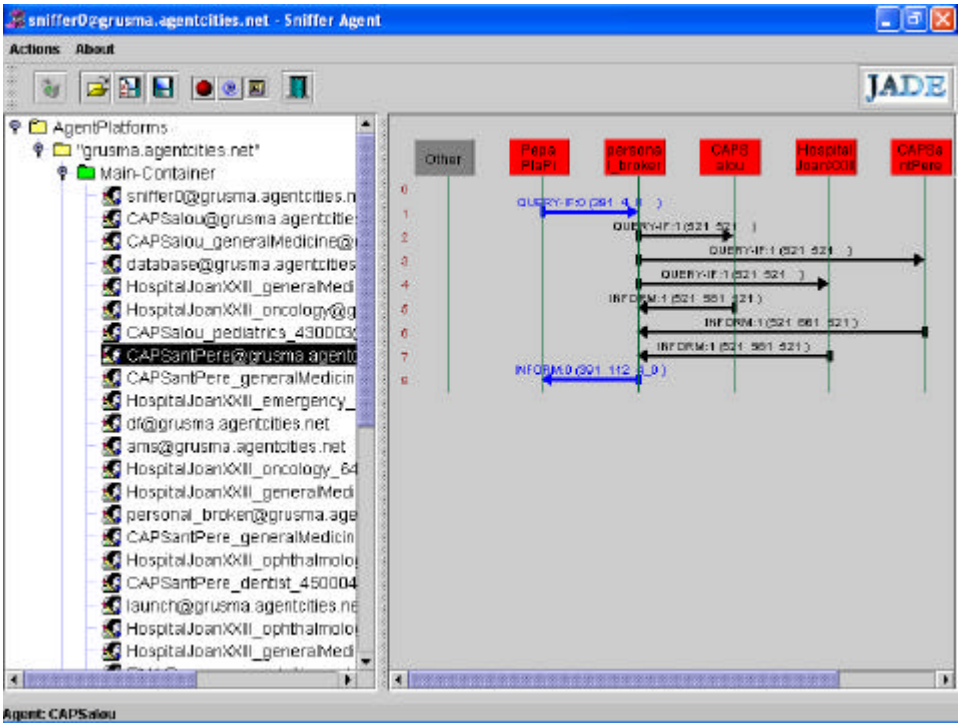

Figura 8 - Finestra de l'sniffer

- Obrir el *Dummy-Agent*: aquest permet enviar missatges ACL entre els agents, rebre'ls i inspeccionar-los i llegir i salvar-los a un fitxer (veure **figura 9**).

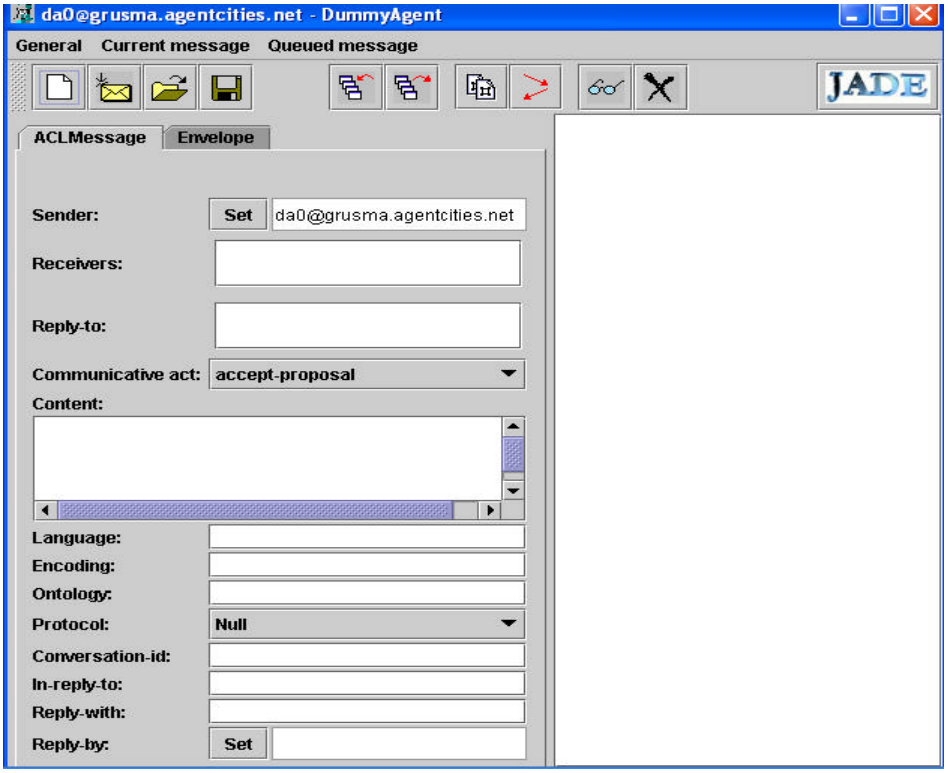

Figura 9 - Finestra de l'agent Dummy

- Engegar l'*Intropector Agent* que ens permetrà monitoritzar el cicle de vida de l'agent, tal com es mostra a la **figura 10**.

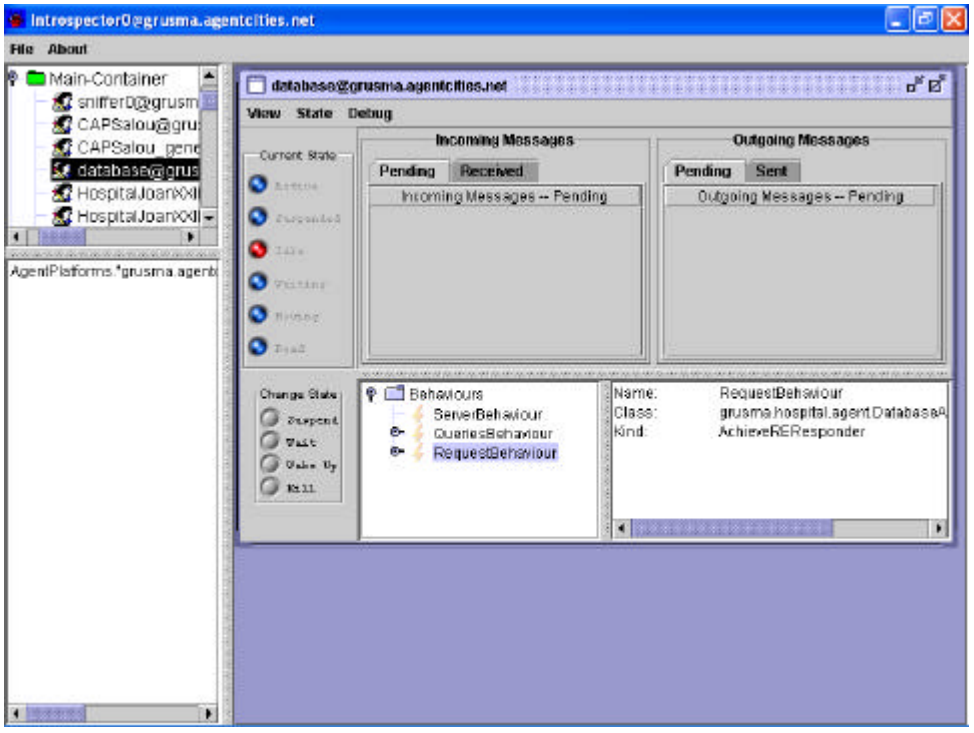

Figura 10 - Finestra de l'Introspector Agent

- Afegir una nova plataforma.

- A més, també disposem de la possibilitat d'activar la interfície de l'agent DF (des del menú *tools*), on podrem observar els agents que s'han registrat i quin serveis ofereixen, tal com es veu a la **figura 11**.

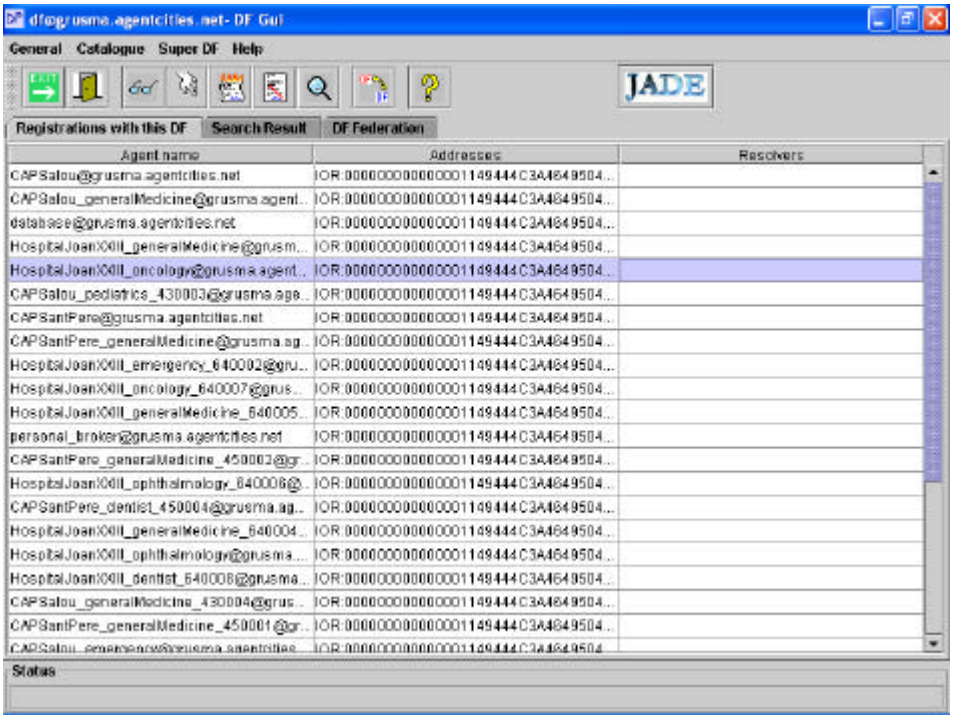

Figura 11 - Finestra amb la interfície del DF

4. DESCRIPCIÓ DEL PROTOTIPUS

En aquest apartat, es parlarà sobre les característiques del sistema realitzat com a PFC per David Isern [35] (en el que es basa aquest projecte) i es justificarà l'elecció dels SMA com a model d'implementació.

Tal com s'ha introduït a l'inici, els principals objectius d'aquest prototipus eren:

- Desenvolupar un sistema que permeti demanar informació sobre serveis mèdics i realitzar reserves d'hora amb un doctor.

- Modelar diferents entitats que proporcionen serveis mèdics (hospitals, CAPs, metges...).

4.1. Justificació del SMA

Aquests objectius poden traduir-se en diferents implementacions no basades en un SMA, com per exemple, una aplicació distribuïda tradicional, amb una BD central (servidor) i un conjunt de clients (doctors, pacients...) accedint-hi. També es podria pensar que els clients poguessin accedir via web a algun dels serveis.

El principal inconvenient d'aquests tipus d'aplicacions és que es tracten de models estàtics sense cap valor afegit més que el de la pròpia informació servida.

Per altra banda, les característiques que ens ofereix un SMA són:

- Modularitat: els diferents serveis o funcions que es volen realitzar poden ser distribuïts entre diversos agents segons la seva complexitat.

- Eficiència: els agents poden coordinar-se per a realitzar tasques més complexes, aconseguint que diferents parts d'un mateix procés es puguin realitzar alhora i així obtenint el màxim paral·lelisme (per al cas que s'executin en nodes diferents).

- Fiabilitat: tot procés distribuït té associat un augment de la fiabilitat en front d'un sistema centralitzat doncs una fallada en un sol element no provoca la caiguda de tot el sistema.

- Flexibilitat: els agents poden crear-se/eliminar-se segons les necessitats de l'aplicació. Les negociacions i l'intercanvi de coneixement permeten optimitzar els recursos.

- Escalabilitat: el nombre d'agents en el sistema no està acotat en principi doncs se'n poden afegir més i millorar el rendiment a base d'establir nodes amb una granularitat més fina (molt ordinadors interconnectats on en cadascun dels s'executa un nombre reduït d'agents).

- Independència en el desenvolupament: igual que si es tractés de programació Orientada a Objectes, els agents són fàcilment implementables per un conjunt de programadors. Un cop formalitzada l'ontologia i els serveis a implementar, cadascun pot posar-se a treballar en un agent diferent.

- Existència d'un estàndard: gràcies a les especificacions de la FIPA, qualsevol dels nostres agents pot ser compatible amb un altre que segueixi les mateixes especificacions.

- Valors afegits: els agents poden anar incorporant serveis a mesura que es van implementat. Així, es poden afegir més funcionalitats mèdiques a l'agent *Personal* o fins i tot d'un altre tipus (cinemes, viatges...).

Així doncs, aplicant aquestes propietats al nostre sistema, podríem acomplir un conjunt de requeriments de forma satisfactòria:

- La informació que hem de proporcionar ha d'estar distribuïda geogràficament, doncs cada centre mèdic ha de guardar les seves pròpies dades, cada doctor tindrà la seva informació personal en un ordinador propi, els registres mèdics dels pacients estaran guardats en diferents Bases de Dades... Per tant, es requereix una aproximació distribuïda per tal de modelar el problema.

- Cal que existeixi un flux d'informació entre els usuaris i els centres mèdics per tal de, per exemple, escollir la millor hora per a concertar una visita (avaluant horaris de diversos doctors). En aquest punt els Agents són especialment útils doncs són capaços de comunicar-se i realitzar una negociació que coordini les seves activitats.

- Els sistemes ja existents, tals com Bases de Dades de registres mèdics, es poden incloure fàcilment en un entorn Multi-Agent. El camí estàndard per "agentificar" una BD és col·locar un *wrapper* [11]. Aquest element és un agent que rep les peticions que cal realitzar a la BD en un llenguatge de comunicació estàndard entre agents (com el FIPA-ACL [18]) i és capaç de traduir aquests requeriments en sentències SQL. Un cop obtingut el resultat, l'agent és capaç novament de traduir aquesta resposta al llenguatge de comunicació d'agents i enviar-ho a l'agent corresponent.

- L'usuari ha de poder ser capaç d'accedir al seu historial mèdic des de qualsevol lloc. Mitjançant agents mòbils basats en Java, és possible implementar comportaments de consulta que s'executin sobre telèfons mòbils o assistents personals connectats a Internet.

- Finalment, cal que els diferents serveis mèdics puguin interactuar entre ells per poder realitzar tasques complicades (interoperabilitat). Els llenguatges de comunicació d'agents (FIPA-ACL [18]), llenguatges de codificació (FIPA-SL [24], FIPA-RDF [23]...), i les ontologies són molt útils per tal de descriure el procés comunicatiu entre diferents serveis. Sense aquest elements, seria molt complicat que diversos sistemes mèdics fossin capaços d'interactuar.

4.2. Arquitectura del prototipus

Després de realitzar l'anàlisi de requeriments, els punts a destacar són:

- Es pretén modelar un entorn mèdic amb un SMA.
- Caldrà que l'usuari sigui capaç de demanar serveis mèdics (reserva d'hora, informació de centres mèdics...) còmodament.
- Els centres mèdics estan organitzats jeràrquicament: un centre es composa d'un conjunt de departaments que corresponen a una certa especialitat mèdica i cadascun d'aquests l'integren una sèrie de doctors.
- Els pacients tenen associat un historial mèdic que caldrà emmagatzemar. Aquestes dades poden ser consultades de forma confidencial pel pacient i actualitzades pels doctors.

A partir d'aquest requeriments, es va dissenyar l'arquitectura del prototipus que es mostra en la **figura 12**.

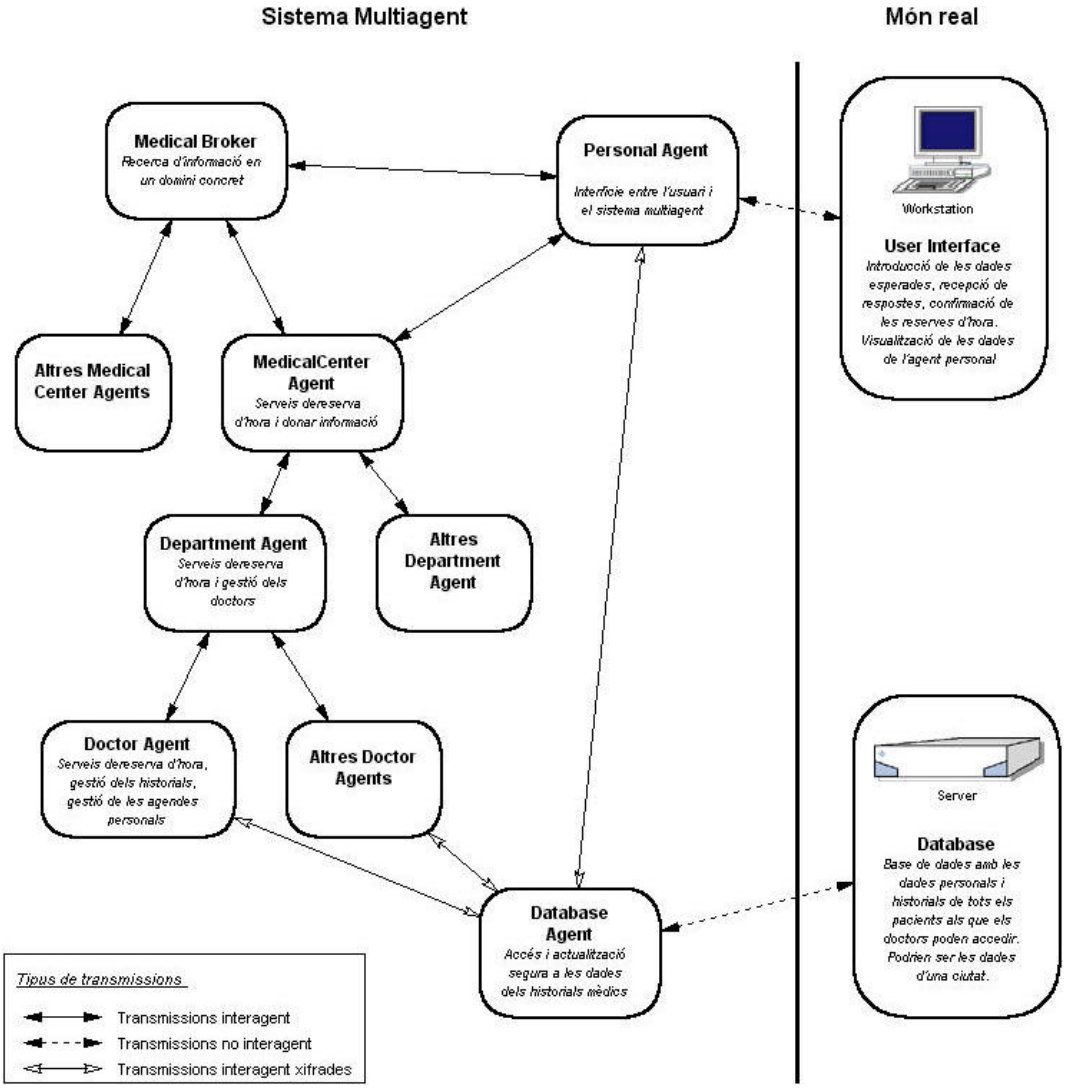

Figura 12 - Arquitectura del Prototipus

4.3. Agents i funcions

A continuació passarem a descriure les funcions que desenvoluparan cadascun dels agents del prototipus:

- *Personal Agent*: és únic i exclusiu a l'usuari. S'encarrega de facilitar l'accés als diferents serveis que proporciona el sistema a través d'una GUI.

- *Medical Broker Agent*: s'encarrega de facilitar les cerques d'informació realitzant un preprocessat dels resultats abans de retornar-los a l'usuari.

- *Medical Centre Agent*: simula el comportament d'un centre mèdic (hospital, CAP, clínica...) i engloba un conjunt de departaments. Té dues funcions principals: respondre les preguntes d'informació i rebre les peticions de reserva d'hora. Les primeres les contestarà ell directament i poden estar referides a una certa especialitat, localització, nom..., mentre que les segones les renviarà al departament corresponent a l'especialitat mèdica.

- *Department Agent*: correspon a una especialitat mèdica determinada que es troba en un centre mèdic i engloba un conjunt de doctors. La seva funció principal és atendre les peticions de reserva d'hora que li envia el *Medical Centre* i demanar-la a cadascun dels seus doctors, escollint la millor d'entre les possibles respostes.

- *Doctor Agent*: són els agents que simulen el comportament d'un doctor dins d'un departament d'un determinat centre mèdic. En aquest cas, funcionen de manera autònoma servint les peticions de reserva d'hora i simulant les visites en el moment adequat, consultant i actualitzant l'historial mèdic del pacient que visiten. Cada *Doctor* disposa d'una agenda personal on es van enregistrant les diferents cites de cada dia per poder engegar el *timer* que el desperti a l'hora fixada (temps real).

- *Database Agent*: les dades referents als usuaris (incloent el seu historial) han de mantenir la persistència, per la qual cosa s'emmagatzemen en una BD. El manegament dels historials es deixa en mans del *Database Agent* que gestiona totes les peticions d'accés sobre la BD. A més, fa d'embolcall de la BD física aïllant la implementació de la resta del sistema per tal d'aconseguir transparència.

4.4. Millores sobre el prototopius

A partir d'aquí i al llarg dels següents apartats, es passarà a descriure la feina realitzada en el present projecte, basat en el disseny i la implementació realitzada per al prototipus. Les millores afegides cobreixen els següents aspectes:

- Actualitzar la implementació del codi font a la darrera versió de JADE per tal d'incloure les característiques millorades en quant a Ontologies i Protocols de comunicació que s'ofereixen i assegurar que es podran aprofitar les noves funcionalitats de seguretat (JADE-S) i mobilitat (JADE-LEAP). El resultat d'aquest treball s'inclou en l'**apèndix A** en forma d'un manual en el que s'expliquen detalladament tots els canvis que s'han de realitzar.

- Completar la funcionalitat bàsica del sistema de forma que es permeti l'execució distribuïda i autònoma del *Personal Agent* en altres ordinadors. Per aquest motiu, s'han afegit nous serveis que fan referència a la gestió de les identitats dels usuaris i creació de perfils personals. En la **secció 7** s'ofereix una descripció de tots els serveis que ofereix el sistema.

- Estudiar diferents alternatives per al disseny d'Ontologies en el camp mèdic per tal d'obtenir una solució el més pròxima possible a un entorn real. A més, caldrà utilitzar un llenguatge de representació estàndard (tant per a l'Ontologia com per als missatges de comunicació) de forma que es faciliti la interoperabilitat del sistema. Aquest punt es tractarà de forma pormenoritzada en la **secció 5**.

- Proporcionar un model de seguretat que permeti el xifratge de la informació i el control de l'accés a les dades confidencials (historials mèdics) dins d'un SMA. Aquest tema es tractarà en profunditat a l'**apartat 6** i implicarà certes modificacions en el disseny arquitectònic bàsic del prototipus així com l'adició de nous serveis específics.

- Oferir els serveis mèdics accessibles per a tothom a través d'una plataforma d'agents. Les característiques d'aquesta i les instruccions detallades per utilitzar-la s'explicaran en l'**apartat 10**.

5. ONTOLOGIA

En aquest apartat, es descriuran totes les qüestions que fan referència a la definició i utilització de l'Ontologia creada per a aquest projecte.

Primerament parlarem dels models d'Ontologies estàndard disponibles en el mercat sobre els que hem basat el nostre disseny. A continuació es justificarà i descriurà el llenguatge de representació utilitzat tant per crear-la com per codificar els missatges. Seguidament, s'inclourà una referència a l'eina que hem fet servir per definir-la i la forma d'integrar-la sobre l'entorn JADE. Finalment, una exhaustiva definició de tots els elements, atributs i relacions definits a l'Ontologia. En l'**apèndix B** s'inlou a més, l'especificació completa en un format estàndard que asseguri l'interoperabilitat.

5.1. Ontologies estàndard

Els sistemes que tracten sobre temes de salut solen ser complexos, heterogenis i dispersos. Per tant, el desenvolupament i la utilització d'estàndards de comunicació, vocabularis i ontologies és un punt molt important quan tractem de sistemes Multi-Agent que implementen serveis mèdics.

De fet, ja existeixen un conjunt d'estàndards de comunicació. Des de fa temps en Europa i els Estats Units estan treballant diverses companyies d'estandardització que tracten sobre temes mèdics [52].

En Europa, el CEN/TC 251 [12] pretén aconseguir compatibilitat i interoperabilitat entre sistemes independents per tal de donar suport als procediments clínics i administratius proporcionant seguretat, garanties de funcionament i qualitat.

Per altra banda la *American Society for testing and Materials' Committee on Healtcare Informatics* (ASTM E31) [5] i el *Health Level Seven* (HL7) [33] estan treballant sobre un tema similar. ASTM E31 està desenvolupant estàndards relacionats amb l'arquitectura, contingut, emmagatzematge, seguretat, confidencialitat, funcionalitat i comunicació de la informació mentre que el HL7 es dedica principalment a la definició dels protocols per a aplicacions relacionades amb l'adquisició de dades i el seu processament.

Pel que fa a les ontologies, la situació no està tan clara. Els temes de salut en la informàtica és un camp en el què trobem diverses comunitats desenvolupant ontologies, com per exemple:

- OpenGALEN [50] que ofereix una terminologia comú però amb un domini limitat.
- *Unified Medical Language System* (UMLS) [66] que proporciona una gran base de coneixement però que li manca una estructura organitzativa.
- *Systematized Nomenclature of Human and Veterinay Medicine* (SNOMED) [56] que només ofereix una nomenclatura i codificació per a les diagnosis.

Tot i això, cada organització és completament independent de l'altra de manera que cada ontologia ha estat desenvolupada per tal de complir unes necessitats diferents. Per tant, a dia d'avui no existeix una definició d'ontologia comú.

En aquesta secció es realitzarà un estudi de les característiques de les ontologies estàndards que proporcionen cadascuna de les organitzacions mencionades. Amb això, pretenem desenvolupar-ne una que s'apropi el més possible a un entorn real basant-nos en els dissenys proposats per cada entitat. Cal tenir en compte però, que algunes d'elles no ofereixen els seus productes gratuïtament i, per tant, només hem pogut accedir a descripcions més o menys generals i publicitàries dels seus models.

5.1.1. UMLS: Unified Medical Language Sytem

Aquest projecte desenvolupa i distribueix bases de coneixement electrònic multipropòsit i programes lèxics associats.

El projecte UMLS intenta desenvolupar un sistema que faciliti l'adquisició i integració de la informació a partir de múltiples fonts de dades biomèdiques. Els punts d'interès més importants inclouen: descripcions de les dades biomèdiques, registres clínics, bases de coneixement i directoris de personal i organitzacions.

Les principals barreres en aquesta integració de la informació són:

- La varietat de vocabularis i classificacions que usen les diferents fonts i els usuaris.

- La quantitat i distribució de les fonts d'informació potencials.

Aquestes barreres dificulten la feina dels professionals mèdics i investigadors que tendeixen a evitar la utilització dels sistemes de tractament de la informació existents i impedeixen la creació de interfícies de cerca efectives que puguin assistir a l'usuari.

L'UMLS és una base de dades relacional que connecta conceptes de prop de 60 vocabularis i diccionaris. El seu objectiu es facilitar als professionals mèdics i investigadors l'obtenció i integració de la informació de diferents fonts tals com:

- Registres mèdics informatitzats.
- Bases de dades.
- Fonts bibliogràfiques.
- Sistemes experts.

La estratègia d'UMLS és la construcció de fonts de coneixement que puguin ser utilitzades en:

- Registres electrònics de pacients.
- Processament del llenguatge natural.
- Adquisició d'informació.
- Construcció de diccionaris de dades.

El nucli d'UMLS són les fonts de coneixement que l'integren i que concretament són:

- El *Diccionari* que conté informació semàntica sobre conceptes biomèdics com són els seus noms, sinònims, i relacions entre ells. Ha estat construït a partir d'altre diccionaris de conceptes, classificacions, sistemes de codificació i llistes de termes desenvolupades per altres organitzacions. Les característiques de què disposa la darrera versió són:
    - 1.900.000 termes extrets d'entre 60 fonts.
    - 800.000 conceptes.
    - 19 milions de relacions.
    - 13 idiomes.

- La *Xarxa Semàntica* defineix una sèrie de categories i tipus de significats als que pertanyen els diversos conceptes presents en el Diccionari. Està composada de 134 tipus semàntics i 54 relacions.

- L' *Especialista Lèxic* conté informació sintàctica sobre els termes biomèdics i cobreix la major part dels termes i conceptes presents al Diccionari. Per facilitar l'accés a aquestes dades es proporcionen una sèrie de programes lèxics que esdevenen potents eines de cerca, indexació i processament de la informació. Aquest àmbit cobreix prop de 140.000 registres.

El projecte UMLS ofereix la seva base de dades de coneixement a través d'Internet [66]. Per tal de poder accedir als recursos, és necessari complimentar una llicència d'ús per tal d'aconseguir el número de sèrie que permeti inscriure a l'usuari amb un *login* i *password*. Tota la especificació UMLS també està disponible en CDROM o ftp des del seu servidor de coneixement.

5.1.2. ASTM Committee E31

Aquest projecte desenvolupa estàndards relatius a l'arquitectura, contingut, emmagatzematge, seguretat, confidencialitat, funcionalitat i comunicació de la informació utilitzada en la medicina i el suport a la decisió incloent informació específica dels pacients.

Els estàndards del ASTM Committee E31 es poden adquirir a través de la seva pàgina web [5]. Cada document està valorat entre 30 i 60 $ i cobreixen les següents categories:

- Vocabularis per a aplicacions informàtiques que tracten temes mèdics.
- Manegament de la informació clínica.
- Intercanvi de senyals.
- Privacitat, confidencialitat i accés.
- Contingut i estructura dels registres mèdics electrònics.
- Informació i seguretat sobre dades mèdiques.
- Documentació i transcripció de dades mèdiques.
- Modelatge d'informàtica mèdica.
- Registres mèdics electrònics.

Concretament, del conjunt de guies que es troben contingudes en algunes de les categories anteriors, la que més s'apropa als nostres interessos de recerca és la "Guia estàndard per al Contingut i l'Estructura dels registre mèdics electrònics (EIIR) (E1384-01)" que té les següents característiques:

- Cobreix tots els tipus de serveis mèdics, incloent aquells que tracten sobre manegament d'hospitals, infermeria, especialitats mèdiques, atenció sanitària i entorns especialitzats d'assistència.

- Identifica el contingut i l'estructura lògica dels registres mèdics electrònics (EHR). Aquest registre conté informació relativa al pacient al llarg del temps. Inclou dades sobre observacions, descripcions del pacient, test de laboratoris, diagnosis, tractaments, administració de medicaments...

- Defineix les relacions de les dades que provenen de diferents fonts i la informació guardada en el Registre mèdic.

- Proporciona un vocabulari comú per a la definició d'EHR.

- Descriu exemples des de diferents punts de vista.

- Mapeja la informació mèdica actualment disponible (Estats Units) en els registres mèdics dels pacients.

5.1.3. Health Level Seven (HL7)

Aquesta és una de les Organitzacions Desenvolupadores d'Estàndards (SDO) que treballen en el camp de la medicina.

La majoria d'aquestes SDOs creen estàndards per a un cert domini concret com són farmàcia, dispositius mèdics, transaccions de segurs... El domini del *Health Level Seven* són les dades clíniques i administratives. La seva missió consisteix en: "Produir estàndards per intercanviar, manegar i integrar les dades que donen suport al tractament del pacient, i a la provisió, avaluació i manegament de serveis mèdics en general" [33].

Resum

"*Level Seven*" es refereix al nivell més alt del model per a la *Interconnexió de Sistemes Oberts* (OSI) creada per la *Organització Internacional d'Estàndards* (ISO), és a dir, el nivell d'aplicació. Aquest nivell tracta sobre la definició de la manera en que les dades han de ser intercanviades, la periodicitat i la comunicació de certs errors a l'aplicació. A més, suporta funcions com el control de la seguretat, identificació dels participants, comprovacions de disponibilitat, negociació i estructuració de les dades.

HL7 se centra en els requeriments d'interfície per a l'organització mèdica en general, mentre que la resta d'aproximacions només es fixen en un departament en particular.

La pàgina web de HL7 [33] ofereix la possibilitat de comprar els estàndards en diversos suports físics (CD-ROM, disquet, imprès...). L'última versió disponible és la 3.0, les característiques de la qual es descriuran a continuació.

Estàndard actual (Versió 3.0)

Les versions anteriors de HL7 (V2.x) oferien una gran flexibilitat definint una gran quantitat de camps opcionals que permetien adaptar les estructures al problema concret. Tot i això, aquest fet dificultava la interacció entre diferents implementacions ja que s'havien d'estudiar quins elements s'havien definit com a opcionals en cadascuna de les interfícies.

En la Versió 3, aquests i altres problemes s'intenten solucionar utilitzant una metodologia ben definida basada en un model d'informació de referència. Utilitza un mètode analític en la construcció dels missatges i incorpora events i una opcionalitat reduïda, produint un estàndard definitiu i estable que permet una interoperabilitat senzilla.

La Versió 3 utilitza un desenvolupament orientat a objectes i un *Model d'Informació de Referència* (RIM) per crear els missatges. El RIM és una part essencial del HL7 doncs proporciona una representació explícita de les connexions lèxiques i semàntiques que existeixen entre la informació continguda en cadascun dels camps del missatges creats per HL7.

RIM (*Reference Information Model*)

El RIM definit per HL7 consisteix en una gran quantitat de classes i relacions. Quatre d'aquestes són de gran importància doncs formen l'eix vertebrador del RIM. Aquestes classes són: **Entity**, **Role**, **Paticipation** i **Act**. Les relacions entre elles són les que es mostren en la **figura 13**:

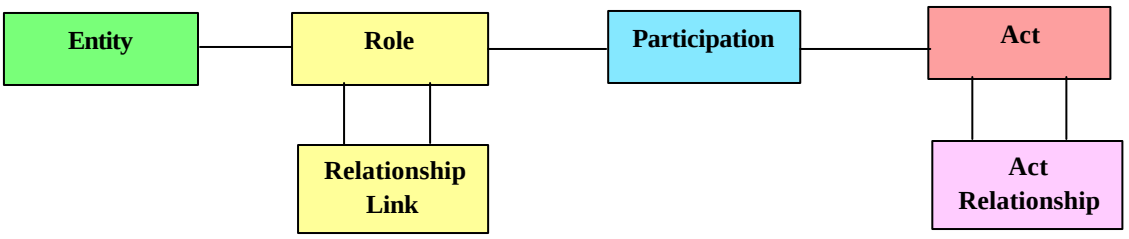

Figura 13 – Eix principal del RIM de l'HL7

Les entitats representen elements com persones, animals, organitzacions, productes mèdics, dispositius, llocs... El model d'una **Entity** i les seves relacions més importants són les que es mostren en la **figura 14**:

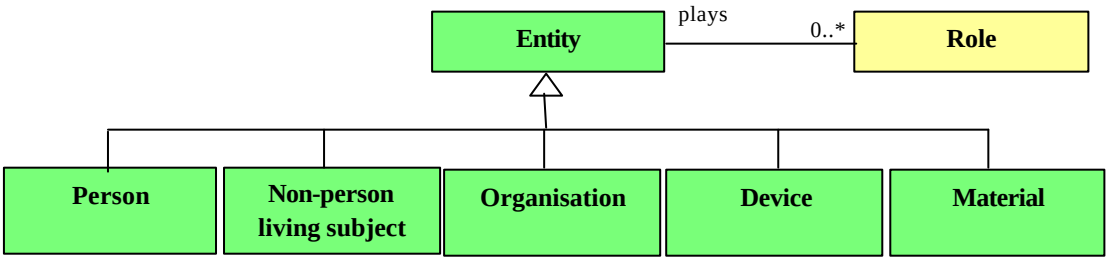

Figura 14 – Especialitzacions d'Entity

Els rols defineixen la relació entre una entitat i la seva participació en un cert acte. Un rol ha d'estar associat forçosament a una entitat; una entitat i un rol han d'estar associats a 0, 1 o més actes a través de la "participació" adequada; dos rols poden estar associats usant una instància de l'enllaç de relació. El model de la classe **Roles** i les seves relacions són les que es mostren a la **figura 15**:

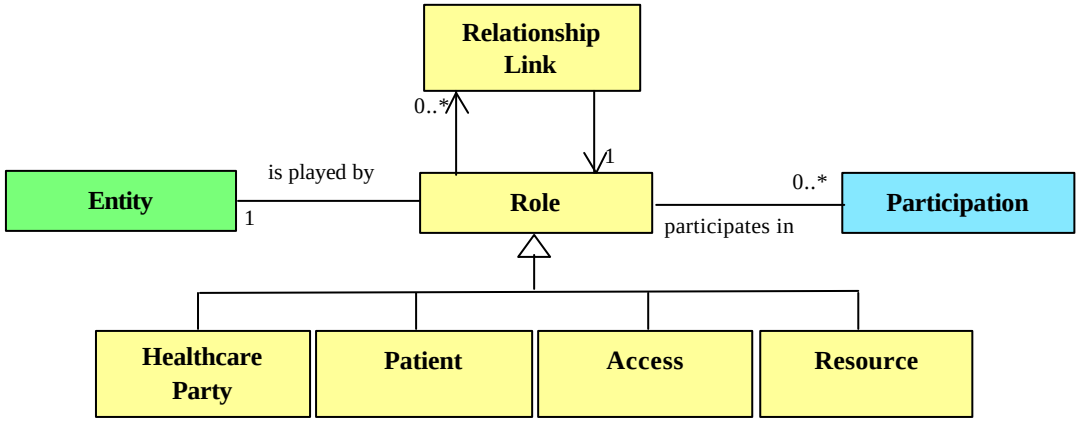

Figura 15 – Especialitzacions de Role

Una participació representa la part que s'encarrega de realitzar una entitat en relació a una certa activitat. El model de relació de les participacions és el que s'indica a la **figura 16**:

Figura 16 – Associacions de Participation

Un acte representa una activitat dins del camp de la medicina com són: observacions, investigacions, transports, visites a pacients... El model de relació per a un acte és el que s'indica a la **figura 17**:

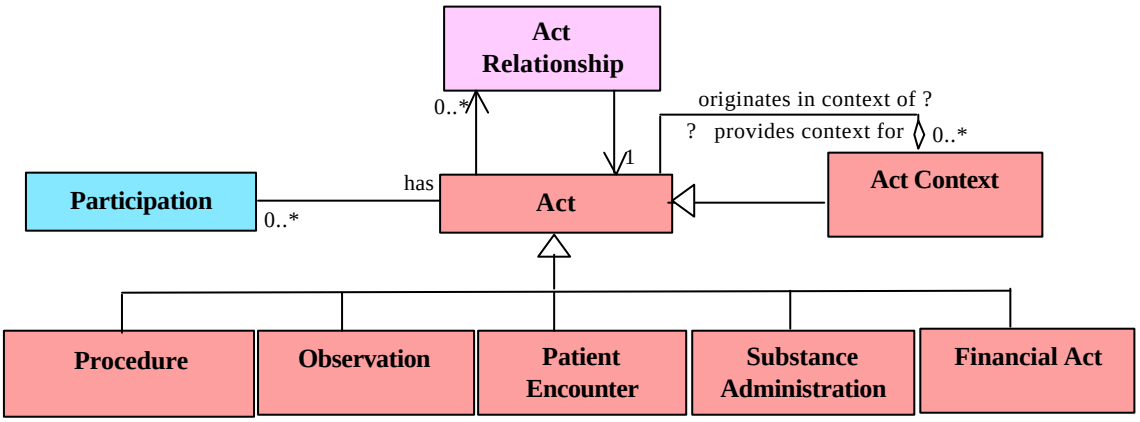

Figura 17 – Associacions i especialitzacions d'Act

5.1.4. SNOMED

El disseny d'SNOMED es basa en la premissa que utilitzar una detallada i específica nomenclatura és essencial per reflexar acuradament, en format electrònic, la complexitat i diversitat d'informació que podem trobar en un registre d'un pacient. El disseny assegura claredat de significat, consistència en l'agregació i facilitat en la creació de missatges.

SNOMED està composat per una estructura jerarquitzada que transforma la noció de vocabulari codificat en una potent eina per aplicacions com són la telemedicina, anàlisi de població, estudis de costos... Aquest disseny permet integrar la informació referent a registres mèdics electrònics en una única estructura de dades.

Els camps d'aplicació de SNOMED són:

- Registres mèdics electrònics.
- Repositori de dades estructurades.
- Sistemes patològics.
- Sistemes de laboratoris clínics.
- Sistemes radiològics.
- Avaluació clínica.
- Reports de malalties infeccioses.
- Codificació de reports d'investigació mèdica.
- Coordinació d'informació per a registres de càncer.
- Cerca de bibliografia.
- Repositori d'imatges.
- Telemedicina.
- Base de dades d'autòpsies.
- Informació mèdica a través de Web.
- Registres de malalties.

La pàgina web d'SNOMED (http://www.snomed.org) inclou informació sobre els productes desenvolupats. SNOMED RT i SNOMED CT són els entorns més moderns disponibles. Lamentablement, per poder accedir a aquesta informació, és necessària una subscripció anual amb un cost d'uns 2000 $.

SNOMED RT

SNOMED RT (*Systematized Nomenclature of Medicine Reference Terminology*) facilita la transició des de registres mèdics en paper cap l'emmagatzematge en registres electrònics.

Amb més de 340.000 interrelacions, SNOMED RT proporciona un mètode per comparar i agregar les dades de tot el procés mèdic. També facilita la transmissió d'informació del pacient a través de diversos canals de comunicació.

El seu disseny permet la completa integració de tota la informació mèdica electrònica en un sol registre de dades, facilitant la interoperabilitat entre una gran varietat de sistemes i registres mèdics.

SNOMED CT

SNOMED *Clinical Terms* (SNOMED CT) ha estat creat a partir de la col·laboració amb el NHS (*National Health Service*) de la secretaria de salut del Regne Unit per tal de poder afegir el diccionari de termes mèdics proporcionat pel NHS. El resultat és una aproximació computeritzada de termes científics utilitzats per doctors, infermers, psicòlegs i altres professionals, durant la comunicació de registres mèdics.

5.1.5. GALEN

La tecnologia GALEN ha estat dissenyada per representar informació clínica en un suport informàtic. El producte final és un sistema computeritzat multi-llenguatge per a la codificació de termes mèdics.

Per fer-ho, s'ha definit un *Model de Referència Comú* anomenat GRAIL (GALEN *Representation And Integration Language*) amb el fi de poder representar tots els conceptes mèdics. Aquests estan representats utilitzant un esquema accessible i manipulable informàticament pel personal mèdic. El *Model de Referència* de GALEN es proporciona a través d'un software anomenat *Servidor de Terminologia* que facilita la interacció amb els diversos termes tractats.

La tecnologia GALEN proporciona un llenguatge, terminologia i sistemes de codificació per a aplicacions mèdiques. La intenció és que sigui útil per als creadors de les aplicacions tant durant la construcció del programa com en temps d'execució. El *Servidor de Terminologia* proporciona mecanismes per a les aplicacions que permeten referenciar conceptes mèdics del *Model de Referència Comú* de GALEN i, per exemple, guardar aquestes referències com una part del registre electrònic del pacient.

A més, el *Servidor de Terminologia* proporciona accés a un gran nombre d'estructures ja dissenyades que utilitzen nombrosos, i sovint incompatibles, sistemes de codificació. L'objectiu és mapejar-los a través del *Model de Referència Comú* i proporcionar mecanismes per interactuar entre sistemes heterogenis que utilitzen aquests esquemes. Així, el *Model de Referència* de GALEN actua com un traductor de dominis diferents que utilitzen diversos sistemes de codificació o llenguatges naturals.

GRAIL

GRAIL (GALEN *Representation And Integration Language*) és el llenguatge de modelització conceptual que ha estat desenvolupat dins del projecte GALEN i ha estat utilitzat per construir el Model Comú de Referència.

GRAIL és el sistema formal que s'ha utilitzat per representar tots els conceptes mèdics de manera que siguin fàcilment utilitzables des d'un entorn informàtic.

El Model de Referència Comú

És el model amb el què han estat implementats els diversos conceptes mèdics. Aquest model forma la base estructural sobre la que treballen els *Servidors de Terminologia* de GALEN.

Servidor de Terminologia

El *Servidor de Terminologia* de GALEN és un software que implementa el llenguatge GRAIL i emmagatzema i manipula el *Model de Referència Comú*. Les seves funcions són dos: permet construir i mantenir el model i proporciona serveis per a aplicacions mèdiques.

OpenGALEN

OpenGALEN és una organització sense ànim de lucre que està formada per alguns dels creadors de la tecnologia GALEN i proporciona sota la llicència *Open-Source* accés gratuït al Model de Referència Comú de GALEN.

OpenGALEN ofereix accés a alguns models de GALEN que tracten sobre els següents temes mèdics:

- Malalties.
- Anatomia.
- Símptomes.
- Medicaments.
- Intervencions mèdiques.

Des de la pàgina web d'OpenGALEN [50] es poden descarregar aquests models gratuïtament i també una versió d'avaluació del software GALEN CRM creat per Kermanog [38] (un paquet de temps limitat que funciona sota Win32 i permet accedir les fonts ASCII dels models proporcionats per OpenGALEN, funcionant com un *Servidor de Terminologia*, tal com es pot veure a la **figura 18**).

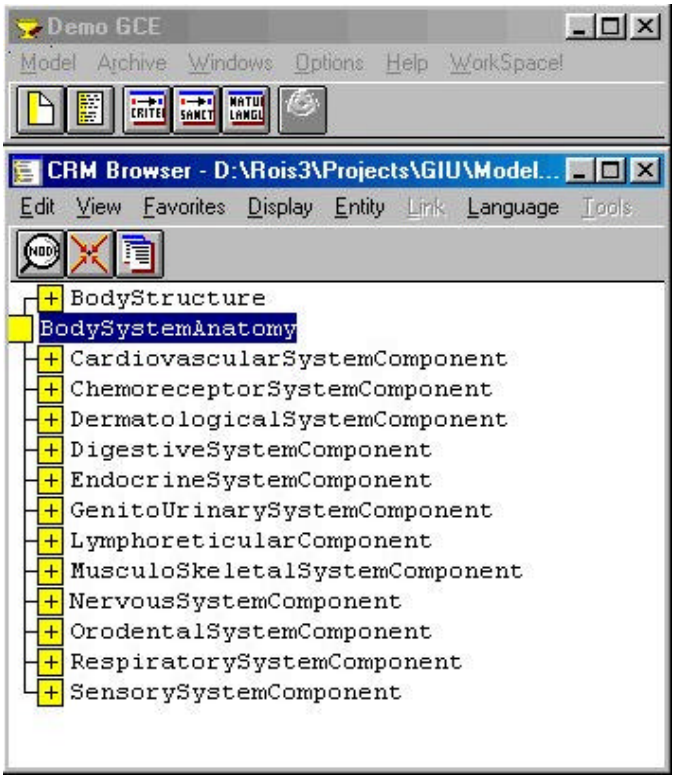

Figura 18 - Pantalla principal del software CRM de Kermanog

5.1.6. CEN/TC251

L'objectiu del CEN/TC251 és aconseguir un sistema estàndard en el camp de la *Informació de la Salut i la Tecnologia de Comunicacions* (ICT) per tal de permetre la compatibilitat i la interoperabilitat entre sistemes independents. Això exigeix definir una estructuració de la informació sanitària i dels procediments administratius, mètodes tècnics per suportar sistemes interoperables i requeriments referents a la seguretat i la qualitat.

La pàgina web del CEN/TC251 [12] proporciona accés als estàndards només per a l'ús intern del TC251 o a certes organitzacions de col·laboració (a través d'un *login* i *password*). No obstant, existeixen altres documents que poden ser descarregats gratuïtament i que són força útils per als desenvolupadors (contenen tutorials, estudis i esborralls d'estàndards).

Grups de treball

El treball del CEN/TC251 està essent dut a terme pels següents grups de treball:

Working Group I

L'objectiu del WGI és desenvolupar estàndards europeus que facilitin la comunicació entre sistemes d'informació independents dintre del camp de la medicina.

Aquests estàndards són essencials si els serveis mèdics han d'aprofitar els beneficis dels sistemes oberts evitant les restriccions de les interfícies propietàries. Els estàndards estan basats en models d'informació.

Una àrea important del treball del WGI són els registres mèdics electrònics. Aquests inclouen una arquitectura que estableix els principis per representar la informació continguda en una estructura de registre, un conjunt de conceptes i termes per als components, i regles i mecanismes per intercanviar informació.

A més d'això, el treball del WGI també inclou estàndards sobre els missatges que es fan necessaris durant el procés de la comunicació d'informació mèdica, proporcionant una sèrie de mecanismes de creació, recepció, validació i manteniment dels missatges.

També es treballa sobre els requeriments d'informació que poden ser aplicables a altres mecanismes d'emmagatzematge de les dades i de transferència de la informació mèdica.

Aquests requeriments s'han definit a partir de les consultes realitzades a professionals mèdics i institucions, així com als desenvolupadors de software.

Working Group II

Els objectius del WGII són:

- Organització semàntica de la informació i el coneixement de forma que es pugui utilitzar de forma pràctica en sistemes mèdics informàtics.
- Proporcionar informació i criteris per a suportar la consistència terminològica de les dades.

Això engloba aspectes de manegament i operacions clíniques sobre els registres mèdics.

El seu treball està centrat en:

- Termes, conceptes i interrelacions.
- Estructures per a sistemes de conceptes.
- Guies per a la producció de sistemes de codificació de bases de coneixement.
- Sistematització de l'estructura semàntica.

Working Group III

El treball del grup està centrat en desenvolupar sistemes per a la protecció de les dades tals com algoritmes de signatura digital, identificació segura utilitzant *passwords*...

Actualment, els principals treballs del grup són:

- Identificació segura de l'usuari utilitzant autenticació forta a través de targes amb microprocessador.
- Seguretat en la comunicació de sistemes mèdics.
- Requeriments de seguretat en la interconnexió de dispositius.
- Seguretat relacionada amb els estàndards de qualitat en el software mèdic.
- Modelatge de permisos de seguretat.
- Procediments per a l'identificació de persones i objectes.

Working Group IV

El propòsit d'aquest grup és desenvolupar i promoure estàndards que permetin la interoperabilitat de dispositius i sistemes d'informació en el camp de la informàtica mèdica.

Aquesta àrea cobreix 3 aspectes:

- Intercomunicació de dades entre dispositius.
- Integració de les dades per a aconseguir una representació homogènia.
- Comunicació de les dades entre els departaments i individus per tal de facilitar la provisió de registres mèdics electrònics.

5.2. Llenguatge de representació

A continuació es farà una breu introducció al llenguatge de representació utilitzat per a definir la ontologia i codificar els missatges enviats entre els agents: RDF. Primerament es descriuran les característiques del metallenguatge del que deriva (XML), per després explicar el funcionament bàsic de RDF així com una justificació de la seva utilització i la manera d'integrar-lo en un SMA.

5.2.1. XML

XML significa *Llenguatge de Marques Generalitzat* (*Extensible Markup Language*) [70]. És un llenguatge utilitzat per estructurar informació en un document o en general, en qualsevol fitxer de text, com per exemple, fitxers de configuració de programes o taules de dades.

Va ser proposat en el 1996 i la primera especificació va sorgir el 1998. Des de llavors ha crescut de forma molt ràpida al ser un estàndard obert i lliure, creat pel Consorci W3C [72] en col·laboració amb les principals companyies productores de software. Està basat en l'anterior estàndard de llenguatge de marques SGML (*Standard Generalized Markup Language*), que data de 1986 i del que també va sorgir l'HTML (veure **figura 19**)

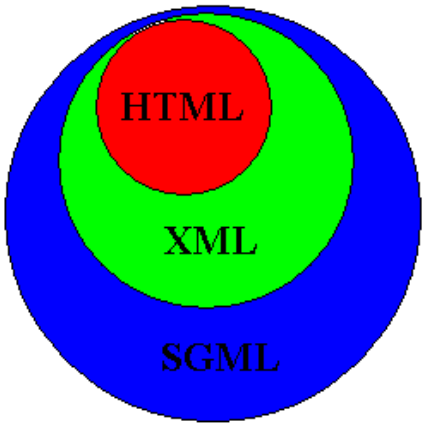

Figura 19 - Jerarquia de llenguatges de marques

Així, XML neix com un subconjunt simplificat del SGML en forma de metallenguatge, és a dir, és un llenguatge que en permet definir-ne altres. Aquesta és una de les principals característiques que el diferencien del HTML (llenguatge de gran implantació a Internet que especifica el model de representació de la informació d'una pàgina web mitjançant marques).

Tot i la gran popularitat del HTML, presenta un conjunt d'inconvenients importants:

- És un llenguatge de representació i, per tant, només defineix la forma en que es veuran les dades però no estructura el seu contingut.
- Això dificulta molt les cerques d'informació a les pàgines doncs la informació no està estructurada.
- No és extensible.
- La representació dependrà molt del visor/navegador que es fa servir per interpretar les marques doncs les regles que les defineixen són força relaxades.

Així, l'XML soluciona la major part de les mancances del HTML tot i que no està pensat com un substitut directe sinó com un complement que el millora.

Les característiques bàsiques del XML són les següents:

- És un metallenguatge: pot utilitzar-se per definir altres llenguatges de marques indicant quins seran els seus *tags* i les propietats de les dades.
- Està dissenyat per ser fortament estructurat mitjançant un conjunt de regles molt estrictes sobre com donar format a les dades.
- És extensible, doncs permet definir termes i *tags* propis.
- Separa el contingut de la visualització, i fa que aquesta última sigui homogènia independentment del visor utilitzat.
- Facilita molt l'intercanvi i les cerques de dades dins d'un document gràcies a l'estructuració.

En resum, l'HTML només reflexa com es deuria presentar el text:

```
<B>David Sanchez Ruenes</B><BR>
<I>dsr.si@estudiants.urv.es<I>
```

En canvi l'XML només indica com s'estructura la informació

```
<CLIENT>
  <NOM>
       <FIRST>David</FIRST>
       <COGNOM1>Sanchez</COGNOM1>
       <COGNOM2>Ruenes</COGNOM2>
  </NOM>
  <EMAIL>dsr.si@estudiants.urv.es</EMAIL>
</CLIENT>
```

L'XML per si sol només permet validar els documents des d'un punt de vista sintàctic (és a dir, que tots els *tags* estiguin correctament utilitzats), però no semàntic (és a dir, no poden indicar quins elements contindrà cada *tag*, camps obligatoris, cardinalitat...). Per tal de definir aquestes últimes regles que s'aplicaran durant la construcció i validació d'un document estructurat, XML utilitza un DTD (*Document Type Definition*).

Així, un DTD és un fitxer que conté una definició formal de l'estructura del document i descriu:

- Els noms que es poden utilitzar per als tipus d'elements.
- On poden apareixen aquest elements.
- Com s'interrelacionen.
- Els atributs d'un element.

Un exemple de DTD seria:

```
<?xml version="1.0"?>
<!ELEMENT facturas (factura+)>
<!ELEMENT factura (num_linea, cod_cliente,articulo+)>
<!ELEMENT articulo (cod_articulo,nom_articulo,cantidad,precioun)>
<!ELEMENT num_linea (#PCDATA)>
<!ELEMENT cod_cliente (#PCDATA)>
<!ELEMENT cod_articulo (#PCDATA)>
<!ELEMENT nom_articulo (#PCDATA)>
<!ELEMENT cantidad (#PCDATA)>
<!ELEMENT precioun (#PCDATA)>
<!ATTLIST factura facturaID CDATA #REQUIRED>
```

Un document XML que compleix les restriccions definides pel DTD anterior seria:

```
<facturas>
      <factura facturaID="0000">
            <num_linea>1</num_linea>
            <cod_cliente>1</cod_cliente>
            <articulo>
                  <cod_articulo>1</cod_articulo>
                  <nom_articulo>CD</nom_articulo>
                  <cantidad>2</cantidad>
                  <precioun>2200</precioun>
            </articulo>
      </factura>
</facturas>
```

5.2.2. RDF

RDF (*Resource Description Framework*) [71] és un DTD definit per W3C orientat a la descripció de recursos. El seu objectiu és facilitar l'intercanvi d'informació seguint un model estructurat que permeti classificar-la i trobar-la de forma senzilla quan sigui necessari.

Així, proporciona interoperabilitat entre aplicacions que intercanvien informació llegible per la màquina a través de web.

RDF pot utilitzar-se en els següents camps:

- Recuperació de recursos: per proporcionar millors prestacions als motors de cerca.

- Catalogació: per a descriure el contingut i les relacions de contingut disponibles en una Web o biblioteca digital.

- Agents de software intel·ligents: per facilitar l'intercanvi de coneixement

A un nivell més alt, el RDFS (*Resource Description Framework Schema*) [39] ha estat definit per tal d'oferir un vocabulari que modeli les classes i les relacions d'herència dels models RDF. Així, RDFS fa possible la definició d'una ontologia simple que permet que documents escrits RDF puguin ser validats per tal de determinar la seva consistència.

En l'**apèndix B**, es pot consultar un exemple de notació RDFS amb la que s'ha codificat l'Ontologia del sistema. En l'**apèndix C**, s'inclou un fitxer amb notació RDF que servei com a configurador per als agents que han d'adoptar un determinat perfil de comportament.

5.2.3. Llenguatge de codificació

Com ja hem vist, els estàndards XML-RDF estan orientats a facilitar l'intercanvi d'informació de forma senzilla i eficient entre diferents fonts. Si analitzem els entorns Multi-Agent, observem com aquests intercanvis són molt habituals i importants doncs suposen la base de la comunicació entre els agents.

Com ja s'ha dit, els agents poden utilitzar diferents tipus de llenguatges per comunicar-se (que s'especifiquen en el camp **language** dels missatges), i per a cadascun d'ells, la FIPA defineix les característiques i regles que han d'acomplir [21].

Abans d'escollir el llenguatge a utilitzar en aquest projecte hem estudiat detingudament els diferents llenguatges que se'ns oferien per tal d'escollir la millor opció. D'entre tots els que suporta la FIPA vam fer una primer selecció (la resta s'han desestimat doncs no disposaven d'una implementació per a JADE i, per tant, obligaven a construir-nos un *parser* ad-hoc):

- FIPA-SL [24]: és un llenguatge pensat específicament per a l'intercanvi d'informació entre agents. Està basat en la sintaxi de Lisp i destaca per la seva simplicitat i eficiència. A més, JADE el suporta de forma nativa i disposa de força funcions que faciliten el manegament dels missatges. El desavantatge principal és que es tracta d'un llenguatge exclusiu per a l'intercanvi d'informació entre agents i, per tant, no està gaire estès ni existeixen productes que el tractin.

- FIPA-XML [20]: com ja s'ha vist, és un llenguatge de marques molt utilitzat en Internet que permet estructurar les dades de forma eficient. La FIPA defineix la

forma de crear missatges amb XML on no només es codifica el contingut en si, sinó també la resta de camps que l'integren (emissor, receptor...). Tot i que no està suportat per JADE directament, existeix un *plug-in* que permet utilitzar-lo per crear missatges. La principal desavantatge es deu al fet que en codificar TOT el missatge (no només el contingut), no poden fer servir les funcions que proporciona JADE per al seu tractament (obtenció de l'emissor, implementació de protocols segons els tipus de missatges...).

- FIPA-RDF [23]: tal com s'ha explicat, és un llenguatge d'especificació basat en XML pensat per a l'intercanvi d'informació estructurada. En aquest cas, la FIPA defineix la forma d'utilitzar-lo per codificar el contingut dels missatges (la part que contindrà les dades a intercanviar), mantenint la resta de camps inalterats. Així, es pot fer servir en JADE fàcilment a través d'un *plug-in* que s'encarregui de codificar/descodificar el contingut dels missatges utilitzant aquesta sintaxi, però mantenint intactes la resta de parts del missatge, cosa que permet utilitzar les funcions natives de JADE per al manegament de la comunicació. A més l'eina de desenvolupament d'Ontologies que utilitzem permet guardar l'especificació de l'ontologia en RDFS.

Així, finalment vam escollir l'opció de l'RDF doncs ens aporta els següents avantatges:

- Pensat específicament per a l'estructuració de les dades (permet validar-les des d'un punt de vista estructural i semàntic).

- Suporta classes, relacions i instàncies.

- Àmplia implantació en el mercat, doncs es tracta d'una especificació W3C, que el converteix en un estàndard mundial per a l'intercanvi d'informació.

- Multitud d'aplicacions el suporten, com per exemple WebODE [73], OntoEdit [49], Protégé [54]...

- Permet comunicar una aplicació basada en agents amb una altra que treballi sobre un entorn Web de forma transparent.

- Manté la funcionalitat de JADE en quant al manegament de missatges, implementació de protocols..., doncs només codifica el contingut del missatge.

- Existeix un *parser* ja implementat que segueix la especificació de la FIPA i que permet utilitzar-lo sobre JADE.

Tot i això, no tot són avantatges per a aquest llenguatge respecte a SL (el que s'utilitza en la majoria d'aplicacions d'agents):

- Incorpora un *overhead* en la comunicació considerable degut a la multitud de *tags* que es fan servir per definir l'estructuració del missatge.

- El *parser* utilitzat és un exercici acadèmic i té algunes restriccions respecte a SL (en quant a la utilització de certs caràcters o a la definició de camps de dades opcionals).

- Hi ha poques aplicacions d'agents que l'utilitzin.

Per tal de poder fer servir aquest llenguatge sobre JADE necessitarem els següents elements:

- El *parser* de RDF per a JADE [51] que permeti codificar/descodificar (**setContent**, **extractContent**) el contingut dels missatges segons l'especificació de la FIPA.

- La definició de RDF que utilitza el *parser* per crear els missatges [67].

- Un *parser* de XML que implementi l'API SAX (dissenyada per Java), com per exemple Xerces [4].

A més de fer servir aquest llenguatge per a codificar el contingut dels missatges (veure **apèndix C**), aconseguint que la nostra aplicació sigui interoperable amb altres no relacionades directament amb els agents, també hem definit l'especificació de la nostra ontologia utilitzant aquesta representació (continguda en l'**apèndix B**). D'aquesta forma fàcilment l'accés d'altres aplicacions a la nostra definició per tal que la puguin incorporar o ampliar.

5.3. Eina per crear ontologies

Donat que definir i implementar els objectes d'una ontologia relativament gran pot resultat un treball considerable, existeix una eina que permet definir-les de manera gràfica (i independentment del llenguatge de programació o aplicació que finalment la farà servir) i després exportar-la a classes Java compatibles amb la definició d'ontologies de JADE.

L'eina en qüestió és Protégé 2000 [54] i el seu aspecte és el que es mostra a la **figura 20**.

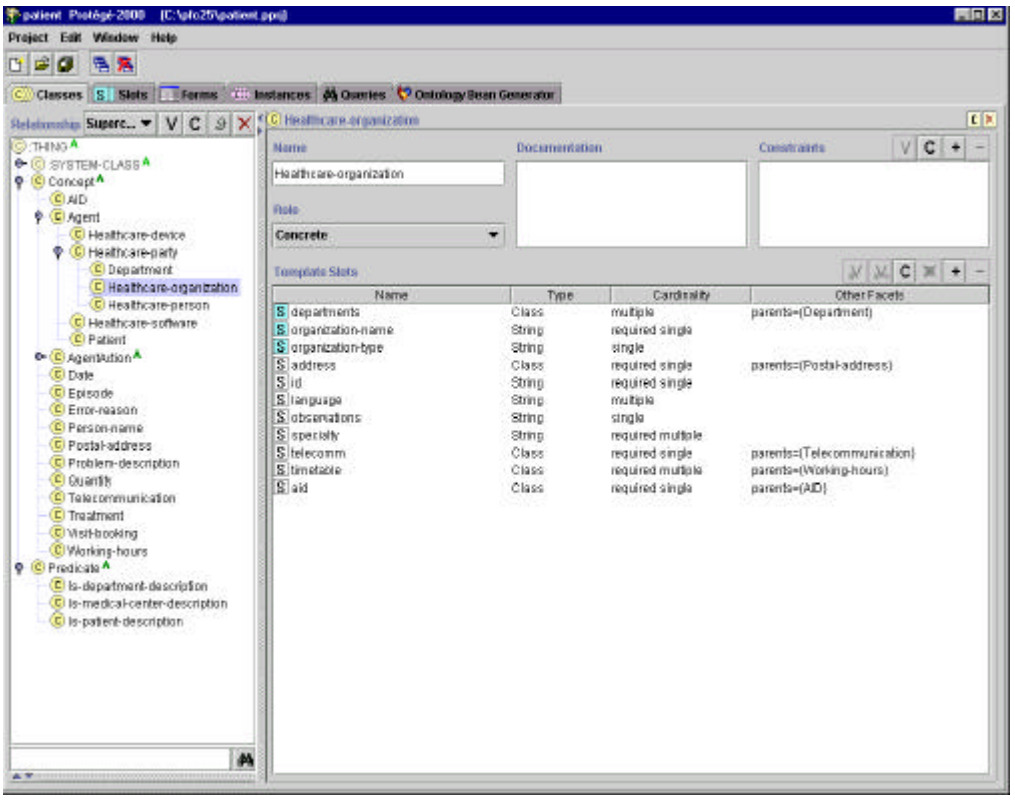

Figura 20 - Aspecte general d'una ontologia definida amb Protégé

Amb aquest programa es pot definir l'estructura de *frames*, herència, *slots* i els tipus de dades (*string*, *integer*, *boolean*...), tot i que algunes funcions avançades de JADE (com les generalitzacions de tipus) caldria definir-les a mà.

Per tal de crear una ontologia compatible amb JADE, serà necessari utilitzar un *plug-in* específic que realitzarà la traducció des de RDF (forma en que la guarda Protégé) a les classes Java corresponents amb la definició de cada *slot* i els seus mètodes d'accés (*read/write*). A més, juntament amb el *plug-in*, anomenat BeanGenerator [37] (**veure figura 21**), se'ns proporciona un esquelet que facilita la definició d'ontologies JADE ja que incorpora l'especificació abstracta dels conceptes, predicats i atributs així com la classe AID.

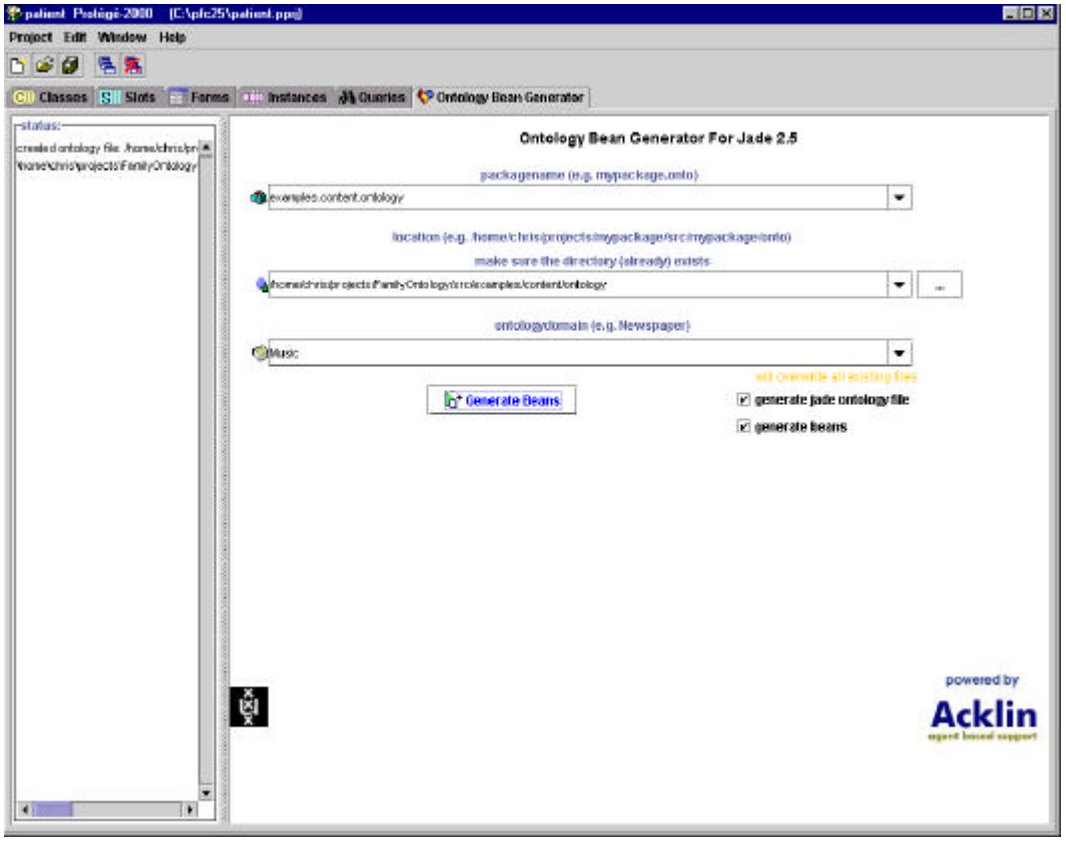

Figura 21 - Plug-in BeanGenerator per a Protégé

Si s'utilitza aquesta opció per a la creació d'ontologies, és convenient fer servir només les possibilitats que ofereix Protégé i el BeanGenerator (JADE suporta altres característiques més avançades), doncs sinó, caldrà modificar a mà els fitxers generats amb el risc que això suposa.

5.4. Especificació de l'Ontologia

En aquesta secció es descriurà l'Ontologia que finalment s'ha dissenyat i implementat per a la nostra aplicació.

5.4.1. Característiques generals

Per tal de crear-la, hem consultat els estàndards de diverses organitzacions que tracten sobre temes mèdics amb l'objectiu de construir un sistema que s'aproximi el més possible a un entorn real.

Les entitats que hem estudiat (descrites en l'**apartat 5.1**) són:

- CEN/TC251: *European Standardization of Health Informatics* [12].
- ASTM Committee E31 [5].
- HL7: *Health Level Seven* [33].
- OpenGALEN [50].
- UMLS: *Unified Medical Language System* [66].
- SNOMED: *Systematized Nomenclature of Medicine* [56].

Donat que algunes d'aquestes organitzacions no ofereixen els seus productes gratuïtament (ASTM, SNOMED) o els seus estàndards no cobreixen les nostres necessitats (OpenGALEN treballa sobre els temes d'anatomia, intervencions mèdiques, medicaments i símptomes, però no sobre registres de pacients o organitzacions mèdiques), hem centrat la nostra investigació sobre els estàndards de CEN/TC251, UMLS i HL7.

El Model de Referència d'Informació definit per HL7 l'hem aplicat a la nostra ontologia per tal d'utilitzar un sistema estructurat de representar les dades (amb herència i interrelacions entre els elements). A més, hem utilitzat alguns exemples d'atributs que han estat útils per guardar, per exemple, els registres dels pacients o el resultat de la visita amb un metge (diagnosi, tractament, proves mèdiques...).

L'estàndard CEN/TC251 sobre registres mèdics electrònics s'ha fet servir per especificar l'estructura, classes i atributs que fan referència a les entitats que proporcionen serveis mèdics (organitzacions, personal mèdic, dispositius...). A més, també ens ha especificat un model per guardar informació comú referent als noms de les persones, direccions, telecomunicacions...

Finalment, la base de coneixement del UMLS ofereix un extens vocabulari sobre termes mèdics agrupats segons la categoria semàntica. Hem utilitzat aquesta informació per crear alguns atributs i definir els seus rangs de valors (ex: especialitats mèdiques).

A més de tot això, hem inclòs també altres *frames* i *slots* que ens permetran afegir més funcionalitats, com són per exemple, l'horari de treball d'una organització mèdica, l'estructuració en departaments d'un hospital...
Pel que fa referència al registre de dades de cada pacient, hem consultat la llei Catalana que tracta sobre la definició i accés a les dades mèdiques [29].

En conclusió, hem creat una ontologia compacta però força general que ens permetrà manegar els principals aspectes d'una institució mèdica (organització general, historials mèdics, visites amb els metges, personal mèdic...).

5.4.2. Elements de l'Ontologia

A continuació s'inclou l'especificació de l'Ontologia (les seves característiques principals s'indiquen a la **taula 1**) utilitzada en el nostre sistema de manera que es puguin interpretar més fàcilment els missatges intercanviats entre els agents. Primer es descriuran els objectes (conceptes, predicats i accions) incloent-hi els atributs, els seus tipus i l'opcionalitat (**Mandatory** o **Optional**).

Nom	Medicalontology.rdf
Llenguatge de descripció	RDF(S)
Objectiu	Manegar dades d'un domini mèdic
Conceptes generals	Healthcare-device, Healthcare-party, Healthcare-software, patient
Ontologies integrades	Cap
Nombre de classes	19
Nombre de propietats	101
Nombre de classes independents	9
Mitjana de desenvolupament de branques	2
Profunditat mitjana	4
Nivell més profund	5

Taula 1 - Característiques de l'Ontologia mèdica

5.4.2.1. Proveïdors de serveis mèdics

Els següents registres i camps permeten modelar el comportament dels agents *Doctor*, *Department* i *Medical Centre*. Les dades es guardaran en fitxers de configuració estàtics a l'inici de l'execució de cada agent. La seva sintaxi es pot consultar a l'**apèndix C**.

Health-domain

Aquesta és la classe més general i representa un agent JADE (cadascun dels elements que formen el sistema) amb el seu identificador (veure **taula 2**).

Atribut	Tipus	Opt.	Descripció
:aid	AID	M	Identificador d'agent

Taula 2 - Atributs de Health-domain

Healthcare-device (hereta Health-domain)

Dispositiu o equipament involucrat directament o indirectament en la provisió de serveis mèdics com per exemple: ordinador, raig-X, quiròfan... (veure **taula 3**).

Atribut	Tipus	Opt.	Descripció
:device-type	String (Coded)	M	Tipus de dispositiu
:device-manufacturer	String (Coded)	O	Fabricant del dispositiu
:device-model	String	O	Nom del model
:device-version	String	O	Versió del dispositiu
:serial-number	String	O	Número de sèrie

Taula 3 - Atributs de Healthcare-device

Healthcare-software (hereta Health-domain)

Component Software involucrat en la provisió de serveis mèdics. Ex: eines de visualització, software de suport a la presa de decisions, bases de dades... (veure **taula 4**)

Atribut	Tipus	Opt.	Descripció
:software-name	String	M	Nom del producte de Software
:software-manufacturer	String (Coded)	O	Fabricant del Software
:internal-name	String	O	Nom intern
:filename	String	O	Arxiu executable
:software-version	String	O	Versió
:software-date	String	O	Data de creació (YY/MM/DD)

Taula 4 - Atributs de Healthcare-software

Healthcare-party (hereta Health-domain)

Informació sobre una organització o persona responsable de la provisió de serveis mèdics (veure **taula 5**).

Atribut	**Tipus**	**Opt.**	**Descripció**
:address	Postal-address	M	Direcció postal
:telecomm	Telecommunication	M	Telèfon, fax, mail...
:language	Set of String (Coded)	O	Idiomes amb els que es pot comunicar
:specialty	Set of String (Coded)	M	Disciplines o especialitats mèdiques proporcionades
:timetable	Set of working-hours	M	Horari de treball
:observations	String	O	Altres dades
:id-party	String	M	Identificador

Taula 5 - Atributs de Healthcare-party

La llista d'especialitats definides són:
- *ophthalmology*
- *pediatrics*
- *dentist*
- *emergency*
- *family psychologist*
- *oncology*
- *general medicine*

Healthcare-organization (hereta Healthcare-party)

Informació sobre una organització que proporciona serveis mèdics. Ex: hospital, clínica, laboratori... (veure **taula 6**).

Atribut	**Tipus**	**Opt.**	**Descripció**
:organization-name	String	M	Nom del centre mèdic
:departments	Set of departments	O	Llista de departaments
:organization-type	String (Coded)	O	Tipus d'organització

Taula 6 - Atributs de Healthcare-organization

Department (hereta Healthcare-party)

Divisió de l'organització mèdica en departaments segons la seva especialitat (veure **taula 7**).

Atribut	Tipus	Opt.	Descripció
:department-name	String (Coded)	M	Nom del departament
:staff	Set of healthcare-person	M	Personal del departament
:rooms	Set of room	O	Llista d'habitacions

Taula 7 - Atributs de Department

Healthcare-person (hereta Healthcare-party)

Persona que proporciona serveis mèdics. Ex: dentista, infermera, personal de laboratori, farmacèutic, doctor... (veure **taula 8**).

Atribut	Tipus	Opt.	Descripció
:person-name	Person-name	M	Nom de la persona
:person-type	String (Coded)	M	Tipus de personal (doctor, infermer...)
:position	String (Coded)	O	Posició dins l'organització (cap, treballador, aprenent...)
:qualification	String (Coded)	O	Qualificacions (estudis)

Taula 8 - Atributs de Healthcare-person

Room

Representa cadascuna de les habitacions d'un departament (veure **taula 9**).

Atribut	Tipus	Opt.	Descripció
:id-room	String	M	Identificador
:room-type	String (Coded)	M	Tipus d'habitació (dormitori, laboratori, quiròfan...)

Taula 9 - Atributs de room

Postal-address

Direcció postal expressada de forma estructurada (veure **taula 10**).

Atribut	Tipus	Opt.	Descripció
:country-code	String (Coded)	O	País
:zip-code	String	M	Codi postal
:city	String	M	Ciutat
:street	String	M	Carrer
:street-number	String	M	Número de carrer

Taula 10 - Atributs de Postal-address

Working-hours

Horari de treball de cada dia de la setmana per a personal o organitzacions en general (veure **taula 11**).

Atribut	Tipus	Opt.	Descripció
:day-week	String	M	Dia de la setmana
:begin	String	M	Hora d'inici de la jornada laboral
:end	String	O	Hora de finalització

Taula 11 - Atributs de Working-hours

Telecommunication

Telèfon, fax i altres medis de comunicació expressats de forma estructurada (veure **taula 12**).

Atribut	Type	Opt.	Descripció
:phone	String	O	Número de telèfon
:fax	String	O	Nombre de fax
:e-mail	String	O	e-mail
:web	String	O	Direcció Web
:Other	String	O	Altres

Taula 12 - Atributs de Telecommunication

Person-name

Nom d'una persona expressat de forma estructurada (veure **taula 13**).

Atribut	Tipus	Opt.	Descripció
:first-name	String	M	Nom
:first-surname	String	M	Primer cognom
:second-surname	String	M	Segon cognom
:title	String	O	Dr, Mr…

Taula 13 - Atributs de Person-name

5.4.2.2. Pacients

Les següents estructures permetran manegar les dades personal i l'historial mèdic dels pacients. Donat que cal que es mantingui la persistència d'aquesta informació, caldrà emmagatzemar-les a la BD del sistema. Tot i això, no tota la informació és necessària per implementar els serveis que ofereix el nostre sistema, per la qual cosa, a la BD només es guardaran les dades que es facin servir (ex: com que les visites són simulades, no es farà cap prova ni s'assignarà un tractament). El contingut específic de les taules i camps associats al següent disseny es pot consultar a l'**apèndix D**.

Patient (hereta agent)

Informació sobre el pacient: dades personals i mèdiques (veure **taula 14**).

Atribut	**Tipus**	**Opt.**	**Descripció**
:patient-name	Person-name	M	Nom del pacient
:sex	String (Coded)	M	Sexe (home/dona)
:date-of-birth	String	M	YY/MM/DD
:id-card	String	M	DNI
:id-medical	String	M	Nombre de la SS
:patient-address	Postal-address	M	Direcció postal
:allergies	Set of string	O	Llista d'al·lèrgies
:history	Set of episode	O	Llista d'episodis mèdics
:patient-phone	String	O	Número de telèfon
:job	String	O	Treball del pacient
:disability	Set of String (Coded)	O	Llista de discapacitats
:antecedents	Set of String	O	Llista d'antecedents familiars

Taula 14 - Atributs de Patient

Episode

Guarda la informació referent a una visita amb un doctor deguda a un cert problema mèdic (veure **taula 15**).

Atribut	**Tipus**	**Opt.**	**Descripció**
:date-visit	String	M	YY/MM/DD
:hour-visit	String	M	HH:MM
:episode-problem	Problem-description	O	Descripció del problema mèdic
:episode-id-medical-center	String	M	Identificador de la visita
:episode-id-department	String	M	Identificador del departament
:episode-id-doctor	String	M	Identificador del doctor
:episode-observations	String	O	Observacions

Taula 15 - Atributs d'Episode

Problem-Description

Guarda informació sobre un problema mèdic i els seus resultats: diagnòstic i tractament (veure **taula 16**).

Atribut	**Tipus**	**Opt.**	**Descripció**
:problem-type	String (Code)	M	Tipus de problema mèdic (especialitat)
:urgency	Integer	M	Rang 1 (urgent) – 5
:description	String	O	Descripció del problema
:problem-treatment	Treatment	O	Descripció del tractament
:results	String	O	Resultat de la visita (diagnòstic)
:tests	Set of String	O	Test realitzats (radiografia, anàlisi de sang...)

Taula 16 - Atributs de Problem-description

Treatment

Informació sobre el tractament aplicat al problema mèdic: medicaments i dosi d'administració (veure **taula 17**).

Atribut	**Tipus**	**Opt.**	**Descripció**
:drug-name	String	M	Nom del medicament
:drug-type	String (Coded)	M	Tipus de medicament
:route-administration	String (Coded)	O	Oral, intravenosa...
:dose	Quantity	M	Quantitat de medicament
:dosage-pattern	String (Coded)	M	Dosi d'administració
:duration	Quantity	M	Duració del tractament

Taula 17 - Atributs de Treatment

Quantity

Una quantitat expressada com un valor numèric i una unitat de mesura: duració del tractament o quantitat d'administració (veure **taula 18**).

Atribut	**Tipus**	**Opt.**	**Descripció**
:value	Integer	M	Valor numèric
:unit	String (Coded)	M	Unitat de mesura

Taula 18 - Atributs de Quantity

5.4.2.3. Manipulació de dades

Els següents elements ens permeten intercanviar la informació temporal necessària quan es porten a terme determinades tasques al sistema (ex: demanar hora amb un metge).

Visit-booking

Identifica una visita entre un doctor i un pacient en un centre mèdic en referència a un problema mèdic (veure **taula 19**).

Atribut	**Tipus**	**Opt.**	**Descripció**
:patient-aid	AID	M	Identificar de l'agent del pacient
:patient-id-medical	String	M	Número de la SS
:visit-problem	Problem-description	M	Descripció del problema mèdic
:visit-place	Healthcare-organisation	O	Lloc de la visita
:visit-date	String	O	Data de la visita (YY/MM/DD)
:visit-hour	String	O	Hora de la visita (HH:MM)
:visit-doctor	Healthcare-person	O	Atributs del doctor
:medical-center-aid	AID	O	Identificador de l'agent que representa el centre mèdic

Taula 19 - Atributs de Visit-booking

Error-reason

Informació referent als possibles errors que es poden produir (veure **taula 20**).

Atribut	**Tipus**	**Opt.**	**Descripció**
:reason	String	M	Descripció de la raó de l'error

Taula 20 - Atributs d'Error-reason

Llista d'errors definida:

- Ontologia no entesa per l'agent
- Format del missatge erroni
- Identificador d'agent erroni
- Condicions no satisfetes
- Agent no autoritzat
- Funció no suportada
- Funció no es pot realitzar
- No hi ha propostes
- Proposta no acceptada per l'usuari
- Identificador mèdic no trobat a la Base de dades
- Per accedir a un historial mèdic, es requereix d'un identificador
- Agent no registrat

5.4.2.4. Accions

Les accions expressen la petició de realització d'una determinada tasca per part d'un agent, i porten associada tota la informació necessària per a la seva execució.

Key-RSA

Guarda la clau pública i privada (conceptes tractats a la **secció 6**) associades a un agent... (veure **taula 21**).

Atribut	Tipus	Opt.	Descripció
:key-agent	AID	M	Identificador de l'agent
:pub-key	String	O	Clau pública
:priv-key	String	O	Clau privada

Taula 21 - Atributs de Key-RSA

Free-time

Permet donar el dia lliure a un doctor (veure **taula 22**).

Atribut	Tipus	Opt.	Descripció
:free-agent	AID	M	Identificador de l'agent
:free-working-hours	Working-hours	O	Hores lliures

Taula 22 - Atributs de Free-time

Register-agent

Permet registrar un agent en el sistema (veure **taula 23**).

Atribut	Tipus	Opt.	Descripció
:reg-agent	AID	M	Identificador de l'agent
:reg-info	ANY_TYPE	O	Descripció de l'agent

Taula 23 - Atributs de Register-agent

Booking-medical-center

Permet establir una visita amb un metge en un centre mèdic (veure **taula 24**).

Atribut	Tipus	Opt.	Descripció
:visit	Visit-booking	M	Informació sobre la visita

Taula 24 - Atributs de Booking-medical-center

Set-medical-record

Permet manegar l'historial mèdic d'un pacient (veure **taula 25**).

Atribut	Tipus	Opt.	Descripció
:medical-data	ANY_TYPE	M	Information about the medical record

Taula 25 - Atributs de Set-medical-record

Login

Permet registrar un pacient en el sistema, proporcionant la seva informació personal (veure **taula 26**).

Atribut	Tipus	Opt.	Descripció
:login-patient	Patient	M	Informació sobre el pacient

Taula 26 - Atributs de Login

5.4.2.5. Predicats

Els predicats expressen unes determinades condicions que han de complir les dades de resposta a una petició d'informació.

Is-medical-center-description

Permet realitzar cerques de centres mèdics que satisfan algunes condicions (veure **taula 27**).

Atribut	Tipus	Opt.	Descripció
:medical-var	String	M	Variable
:medical-attributes	Set of ANY_TYPE	M	Atributs del centre mèdic
:medical-conditions	Set of ANY_TYPE	M	Condicions del centre mèdic

Taula 27 - Atributs de Is-medical-center-description

Is-patient-description

Permet realitzar cerques de pacients que satisfan algunes condicions (veure **taula 28**).

Atribut	Tipus	Opt.	Descripció
:patient-var	String	M	Variable
:patient-attributes	Set of ANY_TYPE	M	Atributs dels pacients
:patient-conditions	Set of ANY_TYPE	O	Condicions del pacient

Taula 28 - Atributs de Is-patient-description

Is-department-description

Permet realitzar cerques de departaments que satisfan algunes condicions (veure **taula 29**).

Atribut	Tipus	Opt.	Descripció
:department-description	Department	M	Descripció del departament

Taula 29 - Atributs de Is-department-description

Is-agent-description

Permet realitzar cerques de agents que ofereixen serveis específics (veure **taula 30**).

Atribut	Tipus	Opt.	Descripció
:agent-description	String	M	Descripció del servei de l'agent

Taula 30 - Atributs de Is-agent-description

A més del predicats definits, també s'han utilitzat alguns dels predicats estàndards proporcionats per JADE:

- IOTA: representa una pregunta sobre una variable amb una resposta única.
- ALL: representa una pregunta sobre tots els valors d'una variable.
- ACTION: representa la petició d'una acció.
- DONE: representa la finalització correcta d'una acció
- EQUALS: representa el resultat d'una acció o petició.

5.4.3. Relacions entre els elements

A continuació es mostren les relacions d'herència i inclusió dels diversos elements que formen l'ontologia. Primer s'indicaran les que fan referència a les entitats proveïdores de serveis mèdics (veure **figura 22**) i a continuació les que formen el registre d'un pacient (veure **figura 23**).

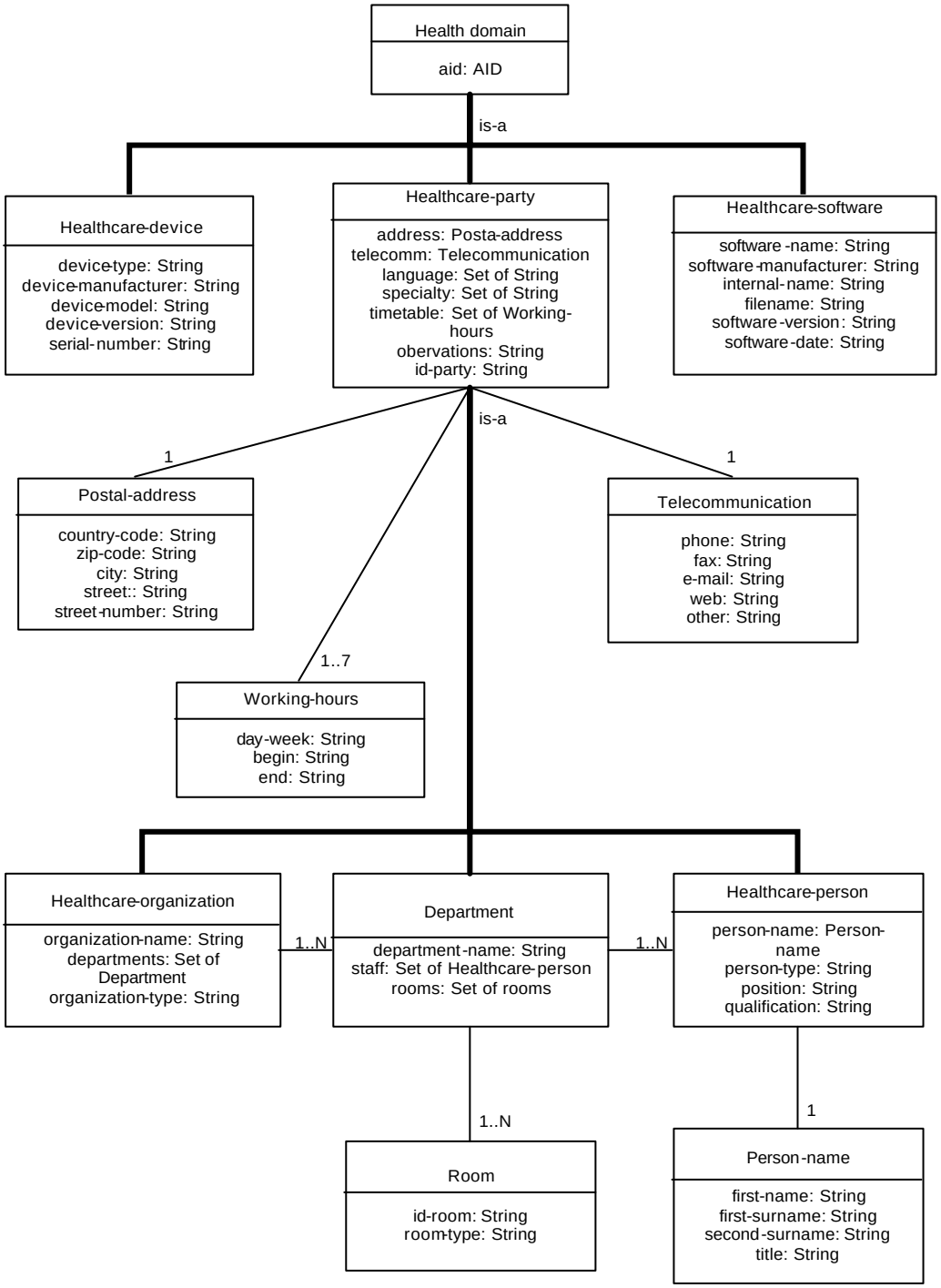

Figura 22 - Relacions entre entitats proporcionadores de serveis mèdics

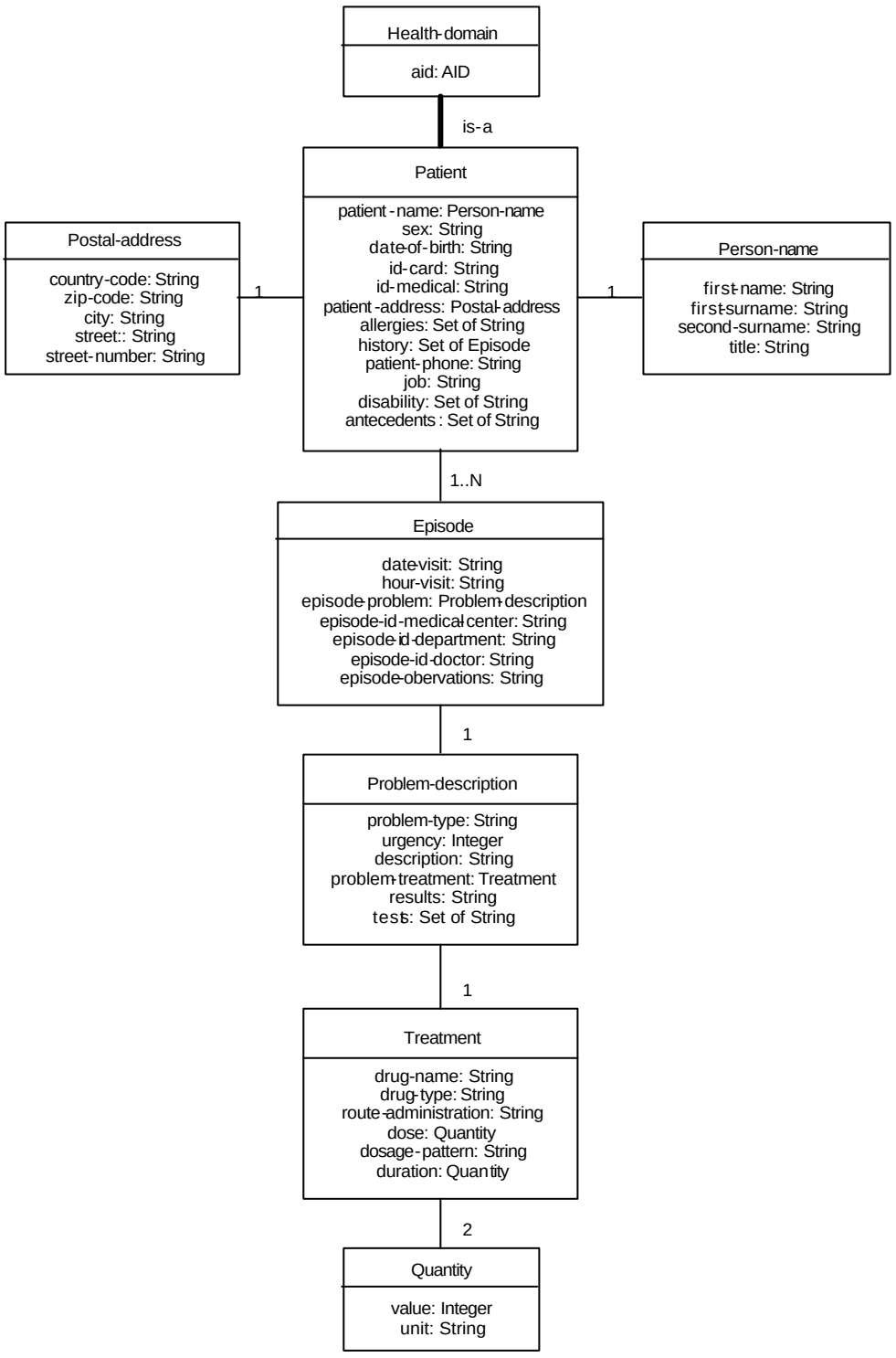

Figura 23 - Estructura del registre d'un pacient

6. SEGURETAT

El manegament d'historials mèdics és un procés molt delicat. Recentment, una llei Americana [68] ha definit les regles que ha de seguir el software en aquest camp i les sancions aplicables a aquells no les compleixin. No obstant, aquest text no defineix un estàndard específic per emmagatzemar o transmetre les dades. Per altra banda, també existeix una llei Catalana [29] que defineix els drets d'accés de qualsevol pacient a la seva informació mèdica.

Donat que el nostre sistema pretén donar accés a aquesta informació confidencial (a més de permetre certs serveis delicats, com la reserva d'hores amb un metge), la qüestió de la seguretat és un tema molt important. Així, en aquesta secció es pretén realitzar un estudi exhaustiu sobre aquest tema aplicat a un entorn de SMA [26], basant-nos en les funcionalitats que porta JADE des de la seva última versió (a través del *plug-in* JADE-S [64]).

Per fer-ho, primerament introduirem les qüestions de seguretat que cal tenir en compte quan es treballa amb un sistema obert que opera a través d'Internet [3], explicant també la base teòrica de les diferents tècniques que permeten implementar els mecanismes de seguretat.

A continuació es farà una introducció al model de seguretat proporcionat pel *plug-in* JASE-S on s'explicaran les seves funcionalitats i característiques però també les importants limitacions que presenta, ja que es tracta d'una versió en fase de desenvolupament.

Finalment es passarà a descriure el model de seguretat implementat en el nostre sistema, tant per al cas de les funcions proporcionades per JADE-S com les que s'han hagut d'implementar via programa per tal d'obtenir una solució ad-hoc.

6.1. Seguretat en els sistemes oberts

6.1.1. Conceptes bàsics de seguretat de la informació

Per a poder entendre quin paper té la seguretat en la transmissió d'informació privada i crítica (ex: historials mèdics) a través d'un entorn obert (Internet), primer és necessari establir diferents conceptes que fixen els diversos nivells de seguretat:

- Confidencialitat: és la propietat que assegura que només aquells que estan autoritzats tindran accés a la informació.

- Integritat: és la propietat que assegura la no-alteració de la informació. Aquesta modificació pot ser: inserció, esborrat o substitució de les dades.

- Autenticació: és la propietat que fa referència a la identificació. És el nexe d'unió entre la informació i l'emissor d'aquesta.

- El no-repudi: és la propietat que preserva que alguna de les parts negui algun compromís o acció presa amb anterioritat.

A part d'aquests termes bàsics, hi ha altres conceptes que inclou la seguretat de la informació i que es poden derivar dels ja citats:

- Control d'accés.
- Autenticació del missatge o d'entitat.
- Signatura.
- Autorització.
- Validació.
- Certificació.
- Confirmació.
- Revocació.

Per tal d'obtenir les quatre propietats bàsiques de seguretat de la informació descrites i també les extensions citades, l'eina bàsica que s'utilitza és la criptografia.

6.1.2. Conceptes bàsics de criptografia

La criptografia és la ciència que estudia les tècniques matemàtiques relacionades amb els diferents aspectes de la seguretat de la informació.

Existeixen dos grans tipus de criptografia: de clau simètrica i de clau pública.

6.1.2.1. Criptografia de clau simètrica

La criptografia de clau simètrica (secreta o compartida) inclou aquelles tècniques en les quals l'emissor i el receptor comparteixen la mateixa clau per xifrar i desxifrar. Alguns exemples d'aquest grup són: DES (*Data Encryption Standard*) , IDEA (*International Data Encryption Algorithm*) i AES (*Advanced Encryption Standard*).

El funcionament d'aquest criptosistema es mostra a la **figura 24**.

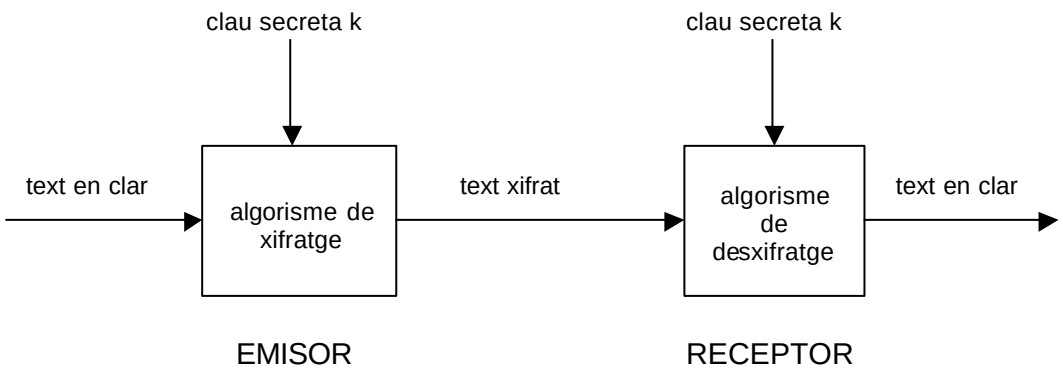

Figura 24 - Esquema de xifratge amb clau secreta

6.1.2.2. Criptografia de clau pública

La criptografia de clau pública engloba els criptosistemes en que cada usuari té un parell de claus: una pública, PK, i una privada (o secreta), SK. La clau pública i la secreta defineixen les funcions de xifratge, Epk(x) i de desxifratge Dsk(x) respectivament que són inverses.

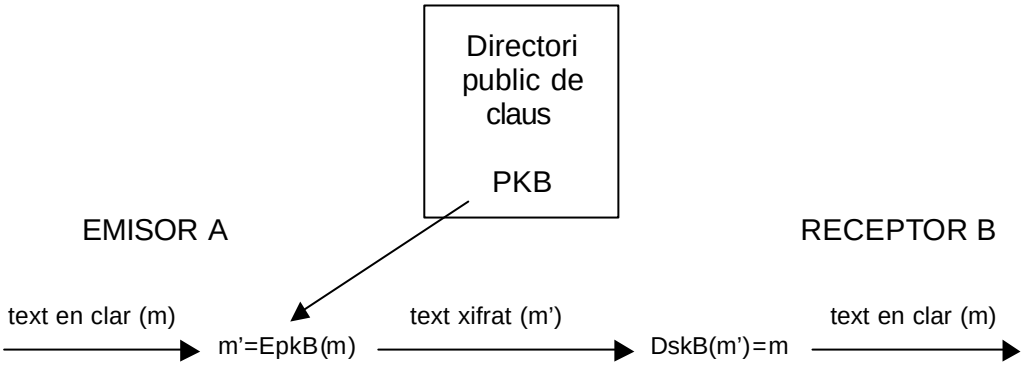

Figura 25 - Esquema de xifratge amb clau pública

La **figura 25** mostra com s'utilitzen els esquemes de clau pública per a xifrar. Quan A vol enviar un missatge, m, a B, obté la clau pública de B, PKB, l'utilitza per xifrar el missatge m aconseguint el text xifrat m'. El text m' només es pot desxifrar amb la corresponent clau privada que tan sols B coneix.

El punt important d'un esquema de clau pública és el fet que a partir de la clau pública no es pugui calcular la privada. Per tal d'aconseguir-ho, els criptosistemes es basen en la seguretat computacional. Es a dir, no es pot assegurar que no es pugui obtenir la clau privada a partir de la pública, sinó que el temps que requereix per fer-ho és prou gran.

A més del xifratge convencional, la clau pública permet autenticar els missatges que s'envien a través del mecanisme que es coneix com Signatura digital:

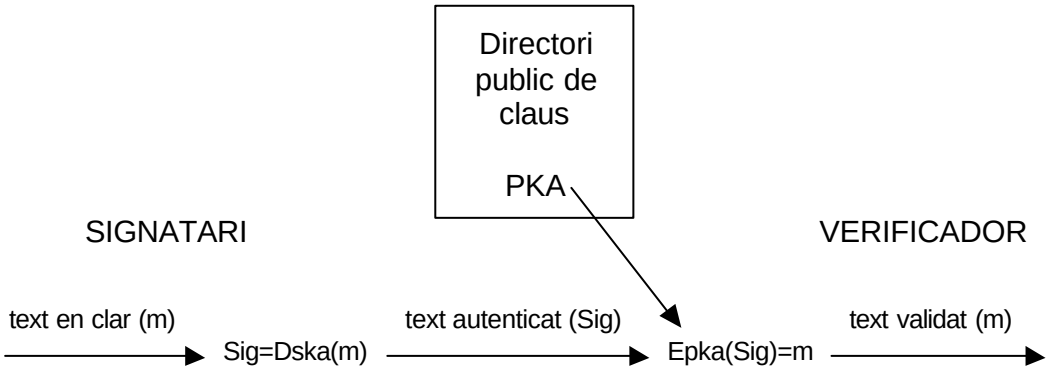

Figura 26 - Esquema de signatura amb clau pública

Tal com es veu a la **figura 26**, el signatari A vol signar un missatge m, per fer-ho utilitza la funció de desxifratge (que conté la seva clau privada SKA) i l'aplica a m obtenint el valor **Sig**. El destinatari obtindrà el missatge original a través de la clau pública d'A i comprovarà que el resultat és vàlid.

Així doncs, amb la criptografia de clau pública podem obtenir les següents propietats:

- Confidencialitat: xifrant els missatges tal com s'ha explicat anteriorment.
- Autenticitat: donat pel fet que la clau privada que s'utilitza per signar només la coneix el signatari.
- No-repudi: gràcies a que només el signatari coneix la seva clau privada, no pot negar haver signat un document.
- Integritat: si es modifiqués el missatge, la verificació de la signatura no seria correcta.

L'únic inconvenient d'aquest tipus de criptosistemes és que utilitzen unes claus més llargues que els de clau simètrica i que el temps de càlcul de les diverses operacions de xifratge/desxifratge és molt més elevat

El criptosistema de clau pública que més s'utilitza és l'RSA [14]. Va ser creat en l'any 1978 per Rivest, Shamir i Adleman i basa la seva seguretat en la dificultat de factoritzar un nombre gran.

6.1.2.3. Funcions de hash

Donat que els criptosistemes de clau pública són computacionalment molt costosos, existeix una altra alternativa que permet signar missatges: les funcions de *hash*.

Una funció de *hash* és un mètode que fa correspondre a un missatge 'm' de mida variable una representació H(m) de mida fixa.

L'esquema de signatura de les funcions de hash és el que s'indica a la **figura 27**.

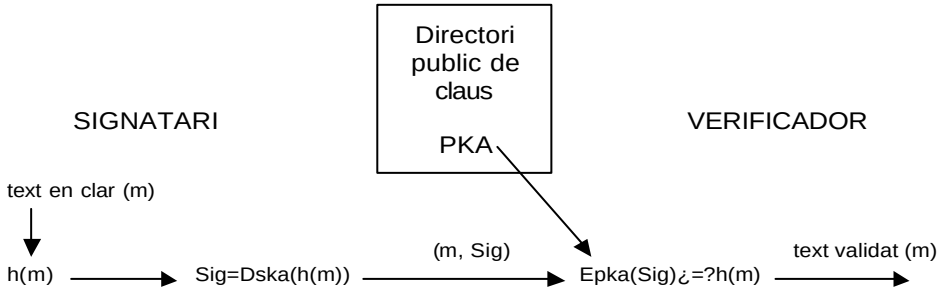

Figura 27 - Esquema de signatura amb funció de Hash

6.1.2.4. Certificats digitals i Autoritats de Certificació

Per tal d'assegurar-nos que una clau pública pertany realment al seu propietari, és necessari el que s'anomena Certificat Digital.

Un Certificat Digital és un document que vincula una determinada clau pública a un usuari.

La informació bàsica que conté és:
- Número de sèrie.
- Identificació de l'algorisme criptogràfic de signatura.
- Nom de l'entitat emissora.
- Període de validesa.
- Clau pública.
- Identitat i dades rellevants de la persona o entitat propietària de la clau pública.
- Signatura digital del certificat per l'entitat emissora.

A l'entitat emissora del certificat (que l'ha de signar per demostrar la seva validesa), se l'anomena Autoritat de Certificació i és un element en el que tots els usuaris de les claus han de confiar.

6.1.3. SSL

Donat que en el nostre sistema estem utilitzant una xarxa de computadors oberta (Internet) per transmetre informació crítica (registres mèdics), la seguretat en el transport de les dades és summament important.

Com és sabut, el protocol TCP/IP utilitzat en Internet es va dissenyar per a un entorn de comunicació obert i per aquest motiu no incorpora propietats de seguretat necessàries per a aplicacions d'aquest tipus. Per tant, quan s'utilitza aquest protocol caldrà aportar una seguretat extra per altre mitjans.

El protocol *Secure Socket Layer* (SSL) [47] és una especificació de nivell de transport que confereix diferents propietats de seguretat a les comunicacions.

L'SSL va ser creat per Netscape Communications per tal d'incloure seguretat al protocol TCP/IP. Així, per exemple, l'SSL resol el problema d'enviar dades confidencials a a un servidor des de un entorn web.

La versió d'SSL que està més implementada és la 3.0, desenvolupada a mitjans dels anys noranta, tot i que en l'actualitat s'ha definit la versió 1.0 del protocol anomenat *Transport Layer Security* (TLS), basat en SSL 3.0.

Des d'un punt de vista molt genèric, l'SSL estableix una comunicació xifrada entre el nostre ordinador (el client) i aquell del que en requerirem algun servei (servidor). Per fer-ho, utilitza tant la criptografia de clau simètrica com de clau pública.

A partir de la clau pública (normalment la del servidor), s'estableix una clau d'un esquema de clau simètrica que serà la clau de la sessió. Des d'aquest moment, tota la informació entre client i servidor va xifrada utilitzant aquesta clau de sessió i l'algorisme de xifratge simètric especificat.

Si repassem les propietats de seguretat que ens aporta el protocol SSL veiem el següent:

- Confidencialitat: en l'SSL la confidencialitat es dóna perquè les dades que circulen per la xarxa per mitjà d'aquest protocol ho fan en forma xifrada amb la clau de sessió que inicialment s'ha acordat. Aquesta clau s'eliminarà quan finalitzi la sessió i, en una altra, se'n negociarà una de nova. El criptosistema de clau simètrica utilitzat no està fixat per l'SSL, sinó que el protocol especifica diferents algorismes previstos.

- Integritat: l'SSL implementa la integritat utilitzant funcions de *hash* que permeten comprovar en el destí si el contingut del paquet ha estat alterat d'alguna forma.

- Autenticitat: l'autenticitat és la propietat que ens permet estar segurs de la identitat, en aquest cas, d'una de les parts de la comunicació. Aquesta propietat és la més important, perquè no serveix de res xifrar les dades (confidencialitat) i estar segurs que arriben tal com les hem enviat (integritat) si el destinatari que hi ha a l'altra banda de la comunicació no és el que pensem. Malgrat aquesta reflexió, l'autenticitat en el protocol SSL és en principi opcional tant per al servidor com per al client. L'SSL ofereix aquesta autenticitat per mitjà de criptografia de clau pública. Si la implementació ofereix autenticitat del servidor, aquest haurà de disposar d'una clau pública, d'una privada i d'un certificat digital obtingut d'alguna Autoritat de Certificació en la que tothom confiï.

- No-repudi: l'SSL no acompleix aquesta propietat completament. Això es deu al fet que es necessita criptografia de clau pública per assolir aquesta fita i, per a les implementacions que no incorporen autenticació en alguna d'aquestes bandes, no es complirà.

En el nostre cas concret, el protocol SSL proporcionat per JADE-S (verue **secció 6.2**) només ofereix claus públiques/privades a la banda del servidor, per la qual cosa haurem d'implementar algun tipus de mecanisme manualment per comprovar les identitats dels clients (pacients) que accedeixen al nostre sistema.

A més, l'SSL és un protocol de transport i, per tant, els mecanismes de seguretat que ofereix són transparents i no són accessibles des de programa, per la qual cosa, si volem establir diversos nivells de permisos segons la identitat dels individus, haurem de realitzar una implementació ad-hoc.

6.2. Model de seguretat de JADE (JADE-S)

Els sistemes distribuïts requereixen un alt nivell de seguretat tant a nivell d'infrastructura com d'aplicació. Per al cas de sistemes Multi-Agent, aquests requeriments [53] són encara més grans degut a les propietats d'autonomia i mobilitat.

D'aquesta forma, un sistema Multi-Agent sense suport de seguretat no es podria utilitzar en un entorn obert com Internet si tracta amb dades crítiques (ex: comerç electrònic, historials mèdics...), doncs les comunicacions podrien ser espiades o les identitats dels agents suplantades fàcilment (accedint als contenidors, registrant-se al DF fraudulentament, "matant" agents...).

JADE-S és un *plug-in* [64] que permet afegir certes característiques de seguretat al desenvolupament de sistemes Multi-Agent [10] de manera que es puguin començar a utilitzar en entorns reals. Així, la plataforma JADE es converteix en un entorn multi-usuari on tots els components (agents, contenidors...) són propietat d'usuaris autenticats, els quals estan autoritzats per l'administrador per realitzar certes accions privilegiades (crítiques).

JADE-S està basat en el model de seguretat de Java [62] extès per als sistemes Multi-Agent. Així, utilitza les avantatges de les següents tecnologies:

- JAAS (*Java Authentication and Authorization Service*) [58]: permet establir permisos d'accés per realitzar certes operacions sobre un conjunt de classes, llibreries o objectes determinades.

- JCE (*Java Cryptography Extension*) [13]: implementa una sèrie de funcions criptogràfiques per permetre manegar la creació i gestió de claus i utilitzar algoritmes d'encriptació.

- JSSE (*Java Secure Socket Extension*) [61]: permet l'intercanvi d'informació segura a través de xarxa utilitzant un model de transport de dades segur (SSL).

6.2.1. Introducció

Una plataforma JADE pot estar situada en diversos hosts i tenir varis contenidors. Per tal d'introduir seguretat en aquest tipus d'entorn distribuït i obert, JADE-S l'estructura com un entorn multi-usuari on cada element pertany a un usuari autenticat (a través d'un *login* i un *password*).

L'esquema general d'aquest entorn és el que es mostra a la **figura 28**:

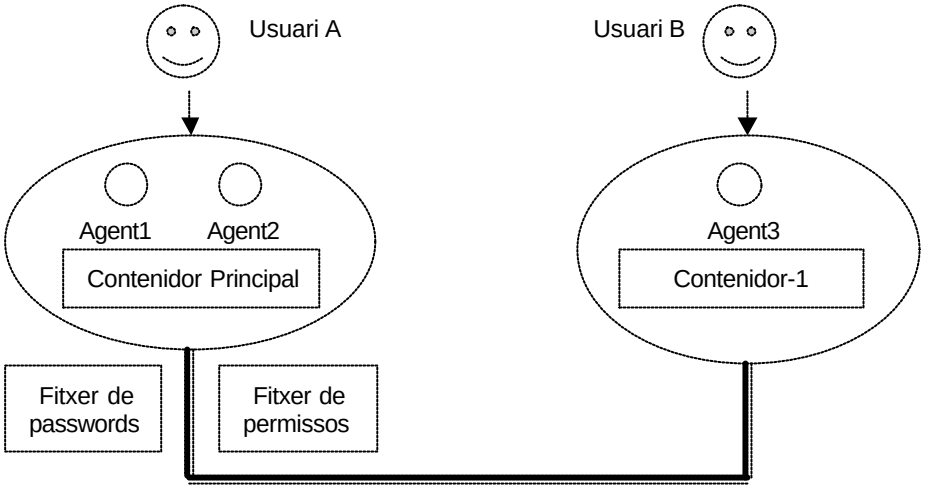

Figura 28 - Esquema general d'una plataforma d'agents segura

Així, a cada plataforma es defineix un fitxer de permisos que conté per a cada usuari el conjunt d'accions que està autoritzat a realitzar.

Internament, un agent prova la seva identitat mostrant un Certificat d'Identitat signat per l'Autoritat de Certificació (proporcionat de forma transparent a l'agent durant el procés d'alta a partir del *login*/*password* especificats). Utilitzant aquests certificats signats digitalment, la plataforma pot estar segura en qualsevol moment de la propietat dels elements i permetre o denegar certes accions.

A més de les accions bàsiques d'identificació d'un agent, també és possible realitzar un procés de delegació de forma que un agent "presta" les seves credencials a un a altre per poder realitzar una acció determinada. Aquestes credencials s'envien a través de Certificats de Delegació que estan signats novament per l'Autoritat de Certificació.

6.2.2. Autenticació

Com ja s'ha dit, cada component de la plataforma té com a propietari un usuari autenticat. Aquell que s'encarregui d'arrencar la plataforma a més, serà propietari dels agents AMS i DF i del Contenidor principal.

Quan un usuari vol accedir a la plataforma (a través d'un agent de la seva propietat per exemple), caldrà que proporcioni el seu *login* i *password* (amb un procés semblant a la gestió d'usuari en un SO multi-usuari). Aquestes dades es comprovaran respecte al fitxer de *passwords* contingut en la plataforma (es guarda de forma xifrada de la mateixa manera que els *passwords* Unix), que és únic i es carrega juntament amb el Contenidor Principal.

Cada agent que pertany a aquest usuari posseirà un Certificat d'Identitat associat a aquell usuari que conté el seu nom i el seu propietari juntament amb la signatura de l'Autoritat de Certificació.

6.2.3. Autorització

A una plataforma JADE-S, els permisos d'accés als recursos es proporcionen als diferents usuaris seguint el mecanisme definit per JAAS [58].

Així, per tal que una entitat pugui realitzar una acció (enviar un missatge, moure's a un contenidor...), caldrà que el manegador de seguretat de Java l'autoritzi a fer-la. El conjunt de permisos associats a cada identitat estan continguts en el fitxer de política d'accés de la plataforma (és únic i es carrega a l'inici).

6.2.4. Permisos i polítiques d'accés

JADE-S utilitza el nou sistema que proporciona Java (versió 1.4) per a l'autenticació basada en usuaris. Així es poden assignar permisos a porcions de codi i a qui les executa, limitant l'accés a certs mètodes, classes, llibreries... en funció de qui les crida.

Java proporciona una sèrie de permisos (a més dels que pot definir l'usuari) sobre els elements bàsics del llenguatge: **FilePermission**, **SocketPermission**, **AWTPermission**... A més d'aquests, JADE-S proporciona uns altres adaptats al comportament dels agents: **AgentPermission**, **ContainerPermission**... Per a cadascun d'aquests, es defineixen una llista d'accions que es poden permetre o denegar.

Així, a través de la definició d'un fitxer de permisos [59] es pot especificar el conjunt d'accions que podrà realitzar cada usuari. Per defecte, el manegador de seguretat de Java només donarà permís per realitzar aquelles accions que estan contingudes explícitament en el fitxer mentre que la resta seran denegades. Amb aquest sistema, es pot definir un fitxer de permisos associat a la plataforma que contindrà, per a cada usuari propietari de recursos, els mètodes que es podran realitzar. En aquest fitxer es poden combinar tant els permisos definits per Java com per JADE.

Per exemple, si volem indicar que un agent A pot enviar i rebre missatges d'un altre B, llavors el fitxer de permisos haurà de contenir la següent entrada (veure **figura 29**):

```
Grant principal jade.security.impl.PrincipalImpl "A"
{
  permission jade.security.impl.AgentPermission "B", "send-to,receive-
from";
}
```

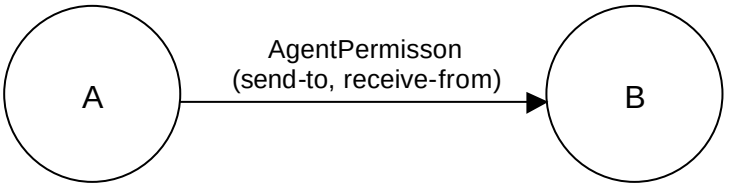

Figura 29 - L'agent A pot enviar i rebre missatges de B

Concretament, la llista de permisos proporcionats per JADE-S és la següent:

Jade.security.impl.AgentPermission

Create: 'A' pot crear agents que seran propietat de 'B' (normalment A = B).
Kill: 'A' pot matar agents propietat de 'B'.
Suspend: 'A' pot suspendre l'activitat d'agents propietat de 'B'.
Resume: 'A' pot retornar l'activitat a agents propietat de 'B'.
Take: 'A' pot prendre la propietat d'agents que eren de 'B'.
Send-to: 'A' pot enviar missatges a agents propietat de 'B'.
Send-as: 'A' pot enviar missatges com si fos 'B'.
Receive-from: 'A' pot rebre missatges d'agents propietat de 'B'.
Move: 'A' pot moure agents propietat de 'B'.
Clone: 'A' pot replicar agents propietat de 'B'.

Jade.security.impl.ContainerPermission

Create: 'A' pot crear un nou contenidor propietat de 'B' (normalment A = B).
Kill: 'A' pot eliminar un contenidor propietat de 'B'.
Create-in: 'A' pot crear un agent en un contenidor propietat de 'B'.
Kill-in: 'A' pot eliminar un agent que està en un contenidor propietat de 'B'.
Move-from: 'A' pot moure agents d'un contenidor propietat de 'B'.
Move-to: 'A' pot moure agent a un contenidor propietat de 'B'.
Clone-from: 'A' pot replicar agents des d'un contenidor propietat de 'B'.
Clonte-to: 'A' pot replicar agents cap a un contenidor propietat de 'B'.

Jade.security.impl.PlatformPermission

Create: 'A' pot engegar una plataforma propietat de 'B' (normalment A = B).
Kill: 'A' pot tancar una plataforma propietat de 'B'.

Jade.security.impl.AuthorityPermission

Sign-ic: 'A' pot signar certificats d'identitat (a través de l'Autoritat de Certificació) propietat de 'B' (normalment A = B).
Sign-dc: 'A' pot signar certificats de delegació de permisos cap a 'B' (a través de l'Autoritat de Certificació).

Jade.security.impl.AMSPermission

Register: 'A' pot registrar agents propietat de 'B' en l'AMS.
Deregister: 'A' pot desregistrar agents propietat de 'B' de l'AMS.
Modify: 'A' pot modificar la informació de registre d'agents propietat de 'B' en l'AMS.

6.2.5. Autoritat de Certificació i Certificats

L'Autoritat de Certificació és l'entitat que signa els certificats de tots els elements de la plataforma. Per fer-ho, posseeix una parella de claus pública/privada de forma que per a cada document acreditatiu (Certificat), crearà una signatura associada xifrant-lo amb la seva clau privada (que només ella coneix); així, quan es vulgui comprovar la identitat d'una entitat es podrà desencriptar la signatura amb la clau pública de l'Autoritat (coneguda per tothom) i veure que l'identitat que l'entitat pretén demostrar concorda amb la que realment l'Autoritat l'hi ha concedit.

La plataforma segura JADE-S proporciona una Autoritat de Certificació dins del Contenidor principal. Així, qualsevol certificat que es signi, només serà vàlid dins de l'àmbit intern de la plataforma on s'ha signat.

6.2.6. Delegació

Amb aquest mecanisme es permet "prestar" un conjunt de permisos d'un agent a un altre. Per fer-ho, l'agent prestador enviarà un Certificat de Delegació (signat per l'Autoritat de Certificació) amb el conjunt de permisos proporcionat i l'identitat del destinatari. L'agent receptor el podrà incorporar al seu propi certificat passant a tenir accés als privilegis proporcionats per l'emissor.

Aquests Certificats de Delegació estan limitats en el temps de forma que només són vàlids fins que el període expira.

6.2.7. Comunicació segura

Per tal de proporcionar comunicació segura entre agents situats en diversos contenidors o hosts, JADE-S utilitza el protocol SSL (*Secure Socket Layer*) que proporciona privacitat i integritat per a totes les connexions que s'estableixin a la plataforma. Així, s'aconsegueix protecció en vers d'*sniffers* de la xarxa.

6.3. Implementació

A continuació es descriuran detalladament tots els mecanismes de seguretat que s'han implementat en el nostre projecte, tant els que ens proporciona JADE-S de forma directa, com els que s'han hagut d'incloure manualment.

6.3.1. Control d'accés

Per tal de fer servir qualsevol mecanisme de seguretat proporcionat per JADE-S sobre una plataforma d'agents, primer cal definir una sèrie de permisos Java per defecte que ens permetran executar el codi bàsic de JADE (que inclou accés a la xarxa, interfície gràfica...). Recordem que només es permetran aquelles accions que estiguin indicades explícitament al fitxer de permisos. Aquestes indicacions hauran d'estar contingudes al fitxer "basic.policy", que es llegirà per defecte en arrancar la plataforma:

```
// Basic permission to allow JADE to start-up

grant codebase "file://classes/-" {
      permission                              java.util.PropertyPermission
"java.rmi.server.hostname", "write";
      permission java.util.PropertyPermission "jade.*", "read,write";
      permission                          java.security.SecurityPermission
"createAccessControlContext";
      permission java.security.SecurityPermission "getPolicy";
      permission              java.net.SocketPermission              "*",
"listen,connect,resolve";
      permission java.io.FilePermission "*", "read,write";
      permission java.lang.RuntimePermission "createClassLoader";
      permission                                      java.awt.AWTPermission
"showWindowWithoutWarningBanner";
      permission java.security.AllPermission;
};
```

Un cop fet això, a partir dels serveis de seguretat que ens proporciona JADE-S, hem creat un model que gestioni el control d'accés al nostre sistema basat en dos nivells de permisos:

- Administrador (**root**): aquest "usuari" (definit amb aquest nom al fitxer de *passwords*), tindrà accés a totes les accions que es poden realitzar a la plataforma (indicades explícitament una per una al fitxer de permisos). Així, tots els agents que pertanyin a aquest propietari tindran accés a qualsevol funció de JADE.

- Visitant (**guest**): en aquest cas, s'han limitat els permisos d'aquest usuari al mínim de forma que els agents que hi pertanyen només podran realitzar les accions bàsiques per demanar serveis al nostre sistema.

El contingut del fitxer de *passwords* amb aquests dos usuaris definits és el següent (vegeu que té el mateix format que els fitxers de *passwords* d'UNIX i que els *passwords* estan encriptats utilitzant DES: *Data Encryption Standard*):

```
root:pm2vWWNETbas6:11577:0:99999:7:0::
guest:gvs9QnChG/xEY:0:0:10000::::
```

Aquest fitxer es pot crear directament amb les funcions de manegament d'usuaris de UNIX o a través de les opcions que proporciona JADE (**jade.security.impl.User add/remove/passwd**).

Així, per a cadascun dels agents que formen el nostre sistema, els associarem a un dels propietaris anteriors:

- Agents interns: són els que s'executen dins de la plataforma (*Broker, Medical Centres, Departments, Doctors i BD*). S'han associat a l'usuari **root,** per la qual cosa no tindran cap restricció en les seves accions. Això ho fem per facilitar la interacció entre ells (podent-se enviar i rebre missatges de tothom, registrar-se al DF i AMS...) tenint en compte que com que són interns a la plataforma (s'executen al **Main-container**) i els hem programat nosaltres, no realitzaran cap acció deshonesta (matar altres agents, desregistrar-los, prendre una altra identitat...).

- Agents externs: en aquest cas englobem els agents *Personals* que s'executen des d'un altre host a través d'un Contenidor. Els hem associat a l'usuari **guest** de forma que no puguin accedir al **Main-container** (per executar-se juntament amb els agents interns), ni accedir al DF (per no modificar la informació dels altres agents) i que només puguin comunicar-se a través del *Broker* (per no suplantar la identitat d'altres agents o enviar-los missatges de *kill*). Totes aquestes restriccions es deuen al fet de que al tractar-se d'un agent extern sense control directe, pot haver estat modificat o reprogramat per realitzar accions perilloses.

El conjunt de permisos que defineixen els comportaments per a cada usuari, està contingut en un fitxer de control d'accés que és únic i es carrega quan s'engega la plataforma. Recordem que per a cada permís possible cal especificar-lo explícitament, ja que per defecte estan tots desactivats. Així, tal com es veurà a continuació, primer caldrà definir una sèrie de permisos globals que permeten que JADE s'executi (permetent l'accés a les llibreries i classes que integren el projecte). Un cop fet això, per a cada usuari, es definirà la llista de permisos adequada:

```
//
// --- These are the basic Java permissions to allow JADE to start up
//

// Allow access to the libraries used in the project

grant codebase "file:c:/jade/lib/jade.jar" {
      permission java.security.AllPermission; };
grant codebase "file:c:/jade/lib/jadeTools.jar" {
      permission java.security.AllPermission; };
grant codebase "file:c:/jade/add-ons/security/lib/jadeS.jar" {
      permission java.security.AllPermission; };
```

```
grant codebase "file:c:/jade/lib/iiop.jar" {
      permission java.security.AllPermission; };
grant codebase "file:c:/jade/lib/Base64.jar" {
      permission java.security.AllPermission; };
grant codebase "file:c:/jade/add-ons/RDFCodec/lib/rdf.jar" {
      permission java.security.AllPermission; };
grant codebase "file:c:/jade/add-ons/http/lib/http.jar" {
      permission java.security.AllPermission; };
grant codebase "file:c:/jade/add-ons/http/lib/sax2/sax2.jar" {
      permission java.security.AllPermission; };
grant  codebase  "file:c:/jade/add-ons/RDFCodec/rdf-api-2001-01-19.jar"
{
      permission java.security.AllPermission; };
grant codebase "file:c:/jade/lib/xerces.jar" {
      permission java.security.AllPermission; };
grant codebase "file:c:/jade/lib/crimson.jar" {
      permission java.security.AllPermission; };

// Allow access to auxiliar java classes used in the project

grant codebase "file:../classes/" {
      permission java.security.AllPermission; };

//
// --- Below all the JADE permissions allowed for the platform ---
//

// enable a set of permissions for 'root'
grant principal jade.security.impl.PrincipalImpl "root" {
      permission       jade.security.impl.AgentPermission       "root",
"create,kill,suspend,resume,take,send-to,send-as,receive-
from,move,copy";
      permission        jade.security.impl.AMSPermission        "root",
"register,deregister,modify";
      permission     jade.security.impl.PlatformPermission     "root",
"create,kill";
      permission     jade.security.impl.ContainerPermission    "root",
"create,kill,create-in";
      permission jade.security.impl.AuthorityPermission "root", "sign-
ic,sign-dc";
};

// Agents owned by guest only can send-to and receive-from messages
from/to the Personal_Broker
grant principal jade.security.impl.PrincipalImpl "guest" {
      permission                    jade.security.impl.AgentPermission
"/personal_broker", "send-to,receive-from";
};

// Agents owned by root can send-to and receive-from messages of guest
grant principal jade.security.impl.PrincipalImpl "root" {
      permission  jade.security.impl.AgentPermission  "guest",  "send-
to,receive-from"; };

// enable a set of permissions for 'guest'
grant principal jade.security.impl.PrincipalImpl "guest" {
      permission      jade.security.impl.AgentPermission      "guest",
"create,kill,suspend,resume,take,send-to,send-as,receive-
from,move,copy";
```

```
      permission    jade.security.impl.ContainerPermission    "guest",
"create,kill";
      permission    jade.security.impl.AuthorityPermission    "guest",
"sign-ic,sign-dc";
      permission       jade.security.impl.AMSPermission       "guest",
"register,deregister,modify";
};

// enable messaging from/to to ams
grant principal jade.security.impl.PrincipalImpl "guest" {
      permission  jade.security.impl.AuthorityPermission  "/ams",  "sign-
dc";
      permission   jade.security.impl.AgentPermission   "/ams",   "send-
to,receive-from"; };

grant principal jade.security.impl.PrincipalImpl "/ams" {
      permission   jade.security.impl.AgentPermission   "guest",   "send-
to,receive-from";
};
```

Més concretament, les accions que podran realitzar els agents que pertanyin a cada usuari seran les següents:

- Administrador (**root**):
    - Crear, eliminar, suspendre, activar, prendre el control, enviar, enviar com un altre, rebre, moure i copiar agents que també pertanyin a **root** (incloent l'AMS i el DF).
    - Registrar, desregistrar i modificar agents que pertanyin a **root** en l'AMS.
    - Crear i eliminar una plataforma que pertanyi a **root**.
    - Crear, eliminar contenidors i introduir-hi agents que pertanyin a **root**.
    - Obtenir certificats d'identitat i delegar-los a altres agents que pertanyin a **root**.
    - Enviar i rebre missatges d'altres agents que pertanyin a l'usuari **guest**.
- Visitant (**guest**):
    - Enviar i rebre missatges a l'agent anomenat **personal_broker**.
    - Crear, eliminar, suspendre, activar, prendre el control, enviar, enviar com un altre, rebre, moure i copiar agents que també pertanyin a **guest**.
    - Registrar, desregistrar i modificar agents que pertanyin a **guest** en l'AMS.
    - Crear i eliminar contenidors que pertanyin a **guest**.
    - Obtenir certificats d'identitat i delegar-los a altres agents que pertanyin a **guest**.

6.3.2. Comunicació segura

A més del mecanisme de seguretat implementat a través dels fitxers de permisos, també hem utilitzat la possibilitat que ens proporciona JADE-S per xifrar les comunicacions a través de SSL.

Tot i que en principi només ens caldria encriptar aquells missatges que entren o surten de la plataforma (entre el *Personal Agent* i el *Broker*) i que contenen informació confidencial (historials mèdics), l'activació de l'SSL només es pot realitzar de forma global: o es xifren tots els missatges o cap.

Així, a través de l'opció `'imtp=jade.security.impl.RMISSLIMTPManager'` que s'haurà d'incloure en els fitxers de configuració d'inici de JADE tant per al client (*Personal*) com per al servidor (Plataforma), totes les comunicacions que es realitzin estaran xifrades i, per tant, podrem enviar sense perill les claus d'encriptació o els historials mèdics. Per tal que aquest mecanisme funcioni, caldrà haver obtingut un certificat d'identitat per a la banda del servidor (el que engega la plataforma), de forma que l'SSL pugui implementar autenticació. A partir d'aquest certificat i mitjançant l'eina **keytool** (inclosa a la distribució de Java) s'hauran de crear un parell de fitxers anomenats: **keystore** i **truststore** que s'inclouran al directori d'execució. Els certificats els proporcionen de forma gratuïta (per avaluació) certes companyies de certificació, com per exemple Verisign [69].

Cal recordar però, que l'SSL és un protocol que funciona a nivell de transport de les dades i, per tant, és transparent a l'aplicació. Així, tot i que durant la programació ens despreocupem totalment d'ell, tampoc podem accedir als mecanismes de seguretat que incorpora (com la possibilitat d'autenticació a través de certificats).

6.3.3. Configuració de la plataforma

Per tal de poder utilitzar tots aquests mecanismes de seguretat, caldrà que la plataforma s'engegui amb les següents opcions (disponibles només a partir de JADE v.2.61):

`-Djava.security.manager`: activa el manegador de seguretat.

`-Djava.security.policy=nom.policy`: indica el nom del fitxer de permisos per a la plataforma.

`jade.security.passwd=nom.passwd`: indica el nom del fitxer de *login*/*passwords* per als usuaris definits a la plataforma.

`-owner login:password`: indica el nom i el *password* de l'usuari al qual pertany la plataforma que estem inicialitzant (es comprovarà vers el fitxer de passwords).

`imtp=jade.security.impl.RMISSLIMTPManager`: activa l'encriptació dels missatges i l'autenticació del servidor a través d'SSL, utilitzant el certificat contingut als fitxers **keystore** i **truststore**.

Per la seva banda, qualsevol client que vulgui accedir caldrà que s'engegui amb les següents opcions:

`-Djava.security.manager`: activa el manegador de seguretat.

`-Djava.security.policy=nom.policy`: indica el nom del fitxer de permisos bàsics que permeten executar JADE.

`-owner login:password`: indica el nom i el *password* de l'usuari que vol accedir (es comprovarà vers el fitxer de *passwords* de la plataforma).

`imtp=jade.security.impl.RMISSLIMTPManager:` activa el protocol SSL amb el servidor.

6.3.4. Limitacions de JADE-S

Tot i aquestes funcionalitats, cal destacar que JADE-S es troba en una fase molt preliminar del desenvolupament i, per tant, presenta força *bugs* i restriccions en el funcionament com són:

- Només funciona amb el JDK 1.4.0_02.
- L'SSL només es pot aplicar a tots els missatges en global (no de forma individual) i en ocasions no s'activa adequadament (quan s'indica l'opció a través de fitxer de configuració).
- Durant la fase d'inicialització dels agents (quan s'han d'identificar i obtenir els seus certificats de l'AMS), de vegades es produeixen excepcions de l'Autoritat de Certificació.
- El mecanisme de manegament d'usuaris (creació, eliminació, definició del *passwords*...) no està implementat (tot i que sí documentat). Només es podem manipular usuaris a través de les comandes de què disposa UNIX a tal efecte.
- Els permisos d'accés són força limitats i genèrics i no permeten per exemple definir les accions a realitzar en concret (ex: permetre realitzar cerques al DF però no registrar-se).
- No es pot accedir a la informació sobre la identitat de l'agent des del programa.
- Només es poden definir els permisos que estan especificats per JADE-S (no es poden afegir altres que facin referència als agents definits per l'usuari, com per exemple, la possibilitat de permetre demanar un determinat servei o no).

Així, degut a aquestes limitacions, hem hagut d'afegir certs mecanismes de seguretat des de programa per tal de crear un model de seguretat suficientment fiable i versàtil.

6.3.5. Model d'accés centralitzat

Un dels primers problemes amb que ens trobem és el fet que els permisos que es poden definir sobre el DF (agent que proporciona informació sobre la resta que s'estan executant a la plataforma), es redueixen a la possibilitat d'habilitar o no l'enviament o recepció dels missatges. Tot i això, les accions que es poden realitzar sobre el DF són:

- Registre d'un agent.
- Desregistre d'un agent.
- Modificació de la informació de registre d'un agent.
- Cercar informació sobre agents que compleixin una determinada condició (ex: que ofereixin un determinat servei o que siguin d'un tipus concret).

Donat que les operacions de registre/desregristre/modificació són crítiques i que no es realitza cap control de la identitat de l'agent que demana l'acció, hem hagut de prohibir l'accés a agents externs (*Personal Agents*) al DF per evitar que realitzin aquestes accions de forma maliciosa (fent-se passar per altres). Tot i això, el desavantatge que trobem és el fet que ja no podrem realitzar cerques d'agents directament, doncs no es permet la comunicació.

Així, la solució ha estat implementar un nou comportament al *Personal Broker* que simuli la funcionalitat del procés de cerca directament al DF. D'aquesta manera el *Personal Agent* disposarà de la possibilitat de cercar agents però sense haver de cridar directament al DF (el *Broker* farà aquest pas intermig).

Un altre inconvenient del model de seguretat de JADE-S és que no permet a l'usuari definir els seus permisos per a cada agent en concret (ex: controlar l'accés al servei de "consulta d'historials mèdics"). Així, tenint en compte que l'arquitectura inicial del prototipus (veure **secció 4.2**) permetia al *Personal Agent* accedir als *Medical Centres* i al *Database Agent* directament per concertar hora o demanar informació (a més del *Broker*), l'únic que podíem definir és l'enviament o no de missatges (però no especificar el servei que es demana en concret). A més, en aquest model descentralitzat resulta més difícil garantir la seguretat (doncs s'hauria d'implementar a diversos nivells) i, per tant, hem pres la decisió de què el *Broker* faci de passarel·la entre el *Personal* i el sistema, prohibint qualsevol comunicació amb altres agents. Així per exemple, podem evitar que el *Personal* es faci passar per un *Department* quan es comunica amb un *Medical Centre* o que cadascun d'aquests hagin de manegar les identitats dels primers.

Aquest model centralitzat ens permet incorporar tots els mecanismes de seguretat del programa (explicats a continuació) en un sol agent, de forma que deslliguem a la resta del sistema d'aquest control i, a més, aconseguim transparència de l'arquitectura interna del sistema respecte a l'usuari, doncs ell només haurà de comunicar-se amb el *Broker*. La contrapartida és que el *Broker* haurà d'implementar indirectament tots els serveis que se li ofereixen a l'usuari, complicant-se la seva implementació.

6.3.6. Autenticació via programa

Tot i que JADE-S manegui internament les identitats dels usuaris, aquesta informació no és accessible des de programa i, donat que ens cal conèixer inequívocament aquesta informació (per proporcionar per exemple els historials mèdics), caldrà que implementem un mecanisme d'autenticació.

Com ja s'ha explicat en la introducció a la seguretat, l'autenticació clàssica es basa en els algorismes de clau pública (en el nostre cas RSA), de manera que cada usuari posseeix un parell de claus pública i privada (aquesta última només la coneix ell i l'autoritat que l'hi ha proporcionat). Aquestes claus estan associades inequívocament a l'usuari a partir de les dades personals que proporciona durant el procés d'alta en el sistema i que li permetran posteriorment signar els missatges crítics que enviï per poder comprovar la seva identitat.

Així, quan finalitza el procés d'alta en el sistema, el *Database Agent* haurà creat les claus de l'usuari i l'hi enviarà a través del *Broker*. Aquest últim es guardarà la clau pública de cada usuari registrat i enviarà el parell al *Personal Agent*. La identitat del qui envia el missatges (*Database* o *Broker*) i la validesa d'aquest no cal comprovar-la via programa (amb certificats d'identitat), doncs es controla a un nivell més baix (gràcies a SSL i la gestió d'usuaris de JADE-S). D'aquesta forma, quan l'usuari vol enviar un missatge crític on ha de demostrar la seva identitat (ex: petició de l'historial mèdic o confirmació d'una visita amb un metge), l'encriptarà amb la seva clau privada (que només ell coneix). Quan el *Broker* rebi aquest missatge, comprovarà a quina identitat correspon utilitzant la clau pública associada a l'AID del que es rep el missatge. Si el contingut un cop desencriptat és vàlid, es considerarà que la identitat és correcta i es processarà la petició. En el cas que la clau no correspongui o s'hagi modificat el contingut del missatge, el resultat de la desencriptació serà incorrecte i provocarà una excepció durant el procés d'interpretació, que es reflexarà en una denegació de la petició.

Aquest mecanisme també s'utilitza per als *Doctors* (per comprovar que només ells poden modificar les dades de l'historial del seu pacient). En aquest cas, però, les claus no s'emmagatzemen (doncs les seves dades es llegeixen estàticament d'un fitxer de configuració i no es guarden a la BD). Així, les claus tindran validesa durant tot el cicle de vida de l'agent *Doctor* (funcionaran com "claus de sessió").

6.3.7. Resum de les característiques de seguretat

Com a resum, les característiques que presenta el nostre model de seguretat són les següents:

- Xifrat de tots els missatges intercanviats entre els agents a nivell de transport i autenticació del servidor amb SSL.

- Establiment de diversos nivells de seguretat a través de vàries identitats d'usuari als que pot pertànyer cada agent (controlat a través de *login* i *password*).
- Definició d'un conjunt de permisos per a cada nivell de seguretat a través de fitxers descriptors.

- Model d'accés centralitzat a través del *Broker* que implementa el control de seguretat via programa.

- Prohibició de comunicació directa amb qualsevol altre agent (incloent el DF).

- Autenticació de l'usuari mitjançant la signatura dels missatges crítics utilitzant un mecanisme de clau pública.

Així, l'arquitectura final del sistema (partint de la que s'havia dissenyat per al prototipus inclosa a la **secció 4.2**), un cop realitzades les modificacions pertinents per garantir la seguretat (accés centralitzat i autenticació dels missatges), és la que es mostra a la **figura 30** (totes les comunicacions representades utilitzen SSL, aquelles que s'autentiquen via programa s'indiquen amb una línia vermella discontínua):

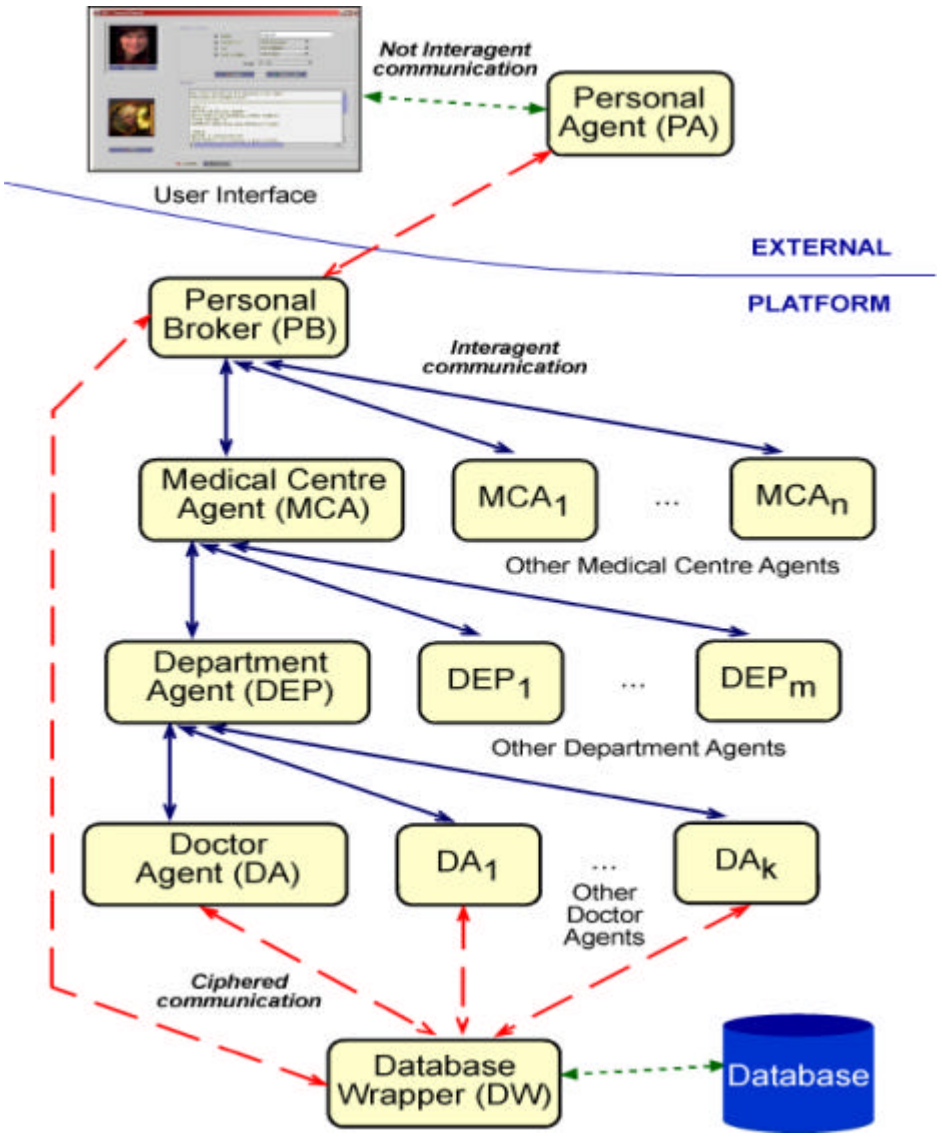

Figura 30 - Arquitectura final del Sistema Multi-Agent

7. SERVEIS PROPORCIONATS

Els serveis proporcionats són les funcions que un agent ofereix als altres agents del sistema o a un conjunt autoritzat. Cada agent implementarà un o més d'aquests serveis o bé els farà servir.

En el nostre cas, el conjunt d'agents que s'executen a la plataforma del servidor oferiran a través del *Personal Broker* una sèrie de serveis al *Personal Agent* que li permetran realitzar diverses accions sobre el sistema:

- Donar d'alta pacient.
- Buscar agents d'un cert tipus.
- Consulta de l'historial mèdic.
- Obtenir informació sobre centres mèdics.
- Negociar una cita amb un doctor en un centre mèdic a una data..

Altres serveis que implementen els agents de la plataforma que s'utilitzen per a la gestió interna de sistema són:

- Registrar un agent.
- Actualitzar l'historial mèdic.
- Demanar la clau pública/privada.
- Gestió dels horaris dels doctors.
- Obtenció de l'historial d'un pacient.

Cada servei defineix predicats amb una semàntica determinada que utilitzen els diferents objectes que s'han especificat en l'Ontologia.

A continuació s'inclourà una descripció detallada de tots els serveis del sistema, incloent les característiques (protocol, llenguatge, interlocutors...) de cadascun d'ells així com missatges d'exemple i diagrames temporals.

NOTA: degut a la complexitat d'alguns missatges, el tamany de l'Ontologia i l'utilització del llenguatge de codificació RDF (basat en etiquetes XML) el missatge final és força gran i resulta pesat d'interpretar. Per això, en els següents exemples, s'inclou una representació esquemàtica del contingut dels missatges que intervenen en cadascun dels serveis basada en el model SL i l'estructura general dels objectes de l'ontologia (sense incloure els atributs) de forma que siguin més fàcils de llegir (veure la definició de l'Ontologia per a més detalls).

7.1. Serveis oferts al Personal Agent

7.1.1. Alta d'un pacient

Quan un pacient (agent *Personal*) vol accedir al sistema, cal que proporcioni la seva informació personal per tal de comprovar la seva identitat. D'aquesta forma es permet manegar els usuaris de forma dinàmica sense que s'hagi d'introduir cada usuari manualment i s'aconsegueix que la resta d'agents puguin comunicar-se amb ell tot i que no estigui registrat en el DF (per motius de seguretat).

Si és la primera vegada que s'accedeix, l'identificador del pacient no estarà a la Base de Dades i caldrà crear i omplir una nova fitxa amb les dades personals proporcionades. A més, es crearan el parell de claus pública / privada que també es guardaran a la BD. Si ja s'havia donat d'alta, l'identificador estarà present a la Base de Dades i es podran recuperar les claus pública i privada corresponents. En qualsevol cas, amb el missatge de resposta, s'enviaran les claus per tal que l'usuari pugui "signar" els missatges crítics que enviarà posteriorment.

Per a aquest servei s'utilitza un protocol FIPA-Request i els missatges s'envien codificats amb RDF i encriptats amb SSL (doncs les claus i les dades personals són informació molt crítica).

A continuació s'inclou un exemple d'aquest servei que sempre ha de ser el primer que demani l'usuari per tal de poder accedir a la resta de funcionalitats del sistema. En aquest cas, el pacient *Personal* envia les seves dades juntament amb el primer missatge del protocol (**Request**) amb *Personal Broker*:

```
(REQUEST
  :sender
    (agent-identifier
    :name Personal@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name personal_broker@grusma.agentcities.net
  :language RDFCodec
  :ontology MedicalCenterOntology
  :protocol FIPA-Request
  :content
    (action
      :action-left
        (agent-identifier  //Agent que demana l'acció
          :name Personal@grusma.agentcities.net
      :action-right
        (login  //Nom de l'acció
          :login-patient
            (patient
              ... //Atributs del pacient (veure Ontologia)
            )
        )
    )
)
```

A continuació cal que el *Personal Broker* comprovi la identitat del pacient a la BD introduint-lo si fos necessari. Si el procés es fa correctament, enviarà un missatge d'**Agree**:

```
(AGREE
  :sender
    (agent-identifier
    :name personal_broker@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name Personal@grusma.agentcities.net
  :language RDFCodec
  :ontology MedicalCenterOntology
  :protocol FIPA-Request
  :content  //Mateix contingut que el missatge original
    (action
      :action-left
        (agent-identifier
          :name Personal@grusma.agentcities.net
      :action-right
        (login
          :login-patient
            (patient
              ...
            )
        )
    )
)
```

Finalment, s'enviarà un missatge d'**Inform** que adjuntarà el parell de claus pública i privada creades o recuperades de la BD.

```
(INFORM
  :sender
    (agent-identifier
    :name personal_broker@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name Personal@grusma.agentcities.net
  :language RDFCodec
  :ontology MedicalCenterOntology
  :protocol FIPA-Request
  :content
    (key-RSA  //Informació de les claus RSA
      :key-agent
        (agent-identifier
          :name Personal@grusma.agentcities.net
      :pub-key
        ... //Clau pública en format String
      :priv-key
        ... //Clau privada en format String
    )
)
```

L'esquema de comunicació general per a aquest servei tenint en compte el protocol utilitzat i els interlocutors s'indica a la **figura 31**:

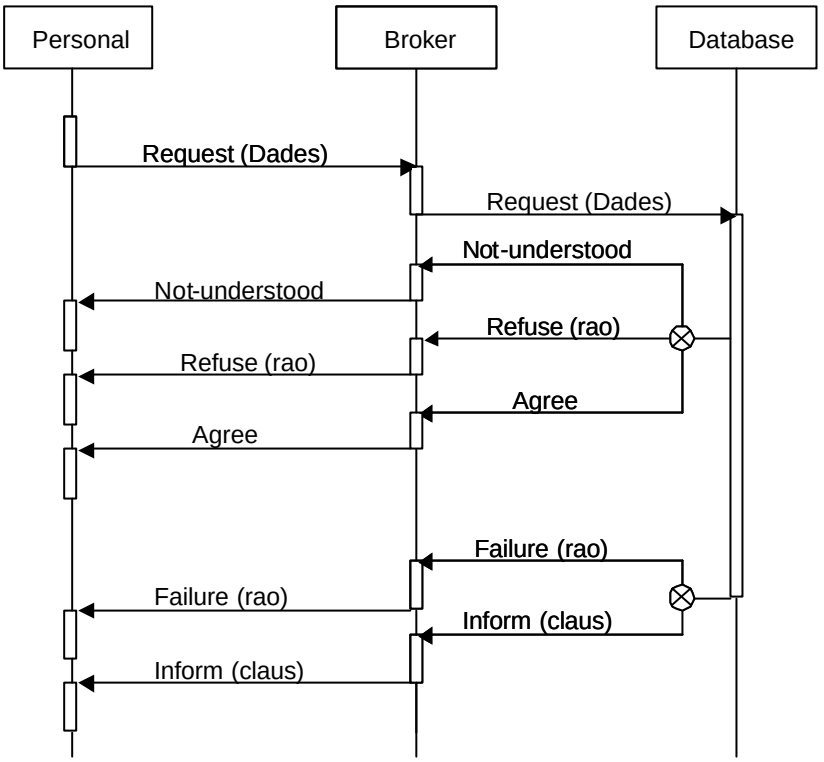

Figura 31 - Esquema de comunicació per a l'alta d'un pacient

7.1.2. Buscar agents d'un determinat tipus

Donat que l'agent *Personal* no es pot registrar en el DF (per evitar que pugui modificar informació crítica referent a altres agents) i que la única forma d'evitar aquesta situació a través del *plug-in* de seguretat Jade-S és prohibint la comunicació entre ells, tampoc es permeten les peticions de cerca d'agents que ofereixin algun servei. Tot i això, l'agent *Personal* cal que conegui quins *Medical Centres* hi ha en un moment donat en el sistema per tal de poder demanar hores de visita en un de concret.

Per aconseguir-ho, s'ha implementat un nou servei a través del *Personal Broker* de petició de cerca d'agents. D'aquesta forma, l'agent *Personal* envia uns criteris de cerca i el propi *Broker* la farà efectiva a través del DF (doncs ell sí que té permís de comunicació), retornant al *Personal* el resultat obtingut.

En aquest cas, s'utilitza un protocol FIPA-Query codificat en RDF i encriptat amb SSL però sense autenticació dels missatges doncs es considera una consulta "anònima".

En el següent exemple, el pacient *Personal* envia una petició de cerca d'agents del tipus **Medical-Centre-Agents** a través d'un missatges de tipus **Query** cap al *Personal Broker*:

```
(QUERY-REF
  :sender
    (agent-identifier
    :name Personal@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name personal_broker@grusma.agentcities.net
  :language RDFCodec
  :ontology MedicalCenterOntology
  :protocol FIPA-Query
  :content
    (all  //Buscar tots els agents
      :all-left ?x
      :all-right
        (is-agent-description
         :agent-description "medical-center-agent" //Servei buscat
        )
    )
)
```

El *Broker* fa efectiva la cerca en el DF i retorna al *Personal* el resultat:

```
(INFORM
  :sender
    (agent-identifier
    :name personal_broker@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name Personal@grusma.agentcities.net
  :language RDFCodec
  :ontology MedicalCenterOntology
  :protocol FIPA-Query
  :content
    (equals
      :equals-left
        (all  //Contingut del missatge original
        :all_left ?x
        :all_right
          (is-agent-description
            :agent-description "medical-center-agent"
          )
     :equals-right
       (set  //Llista d'agents "medical-center-agent"
          (agent-identifier
            :name center@grusma.agentcities.net
          )
          ...
        )
    )
)
```

L'esquema de comunicació general per a aquest protocol s'inclou a la **figura 32**:

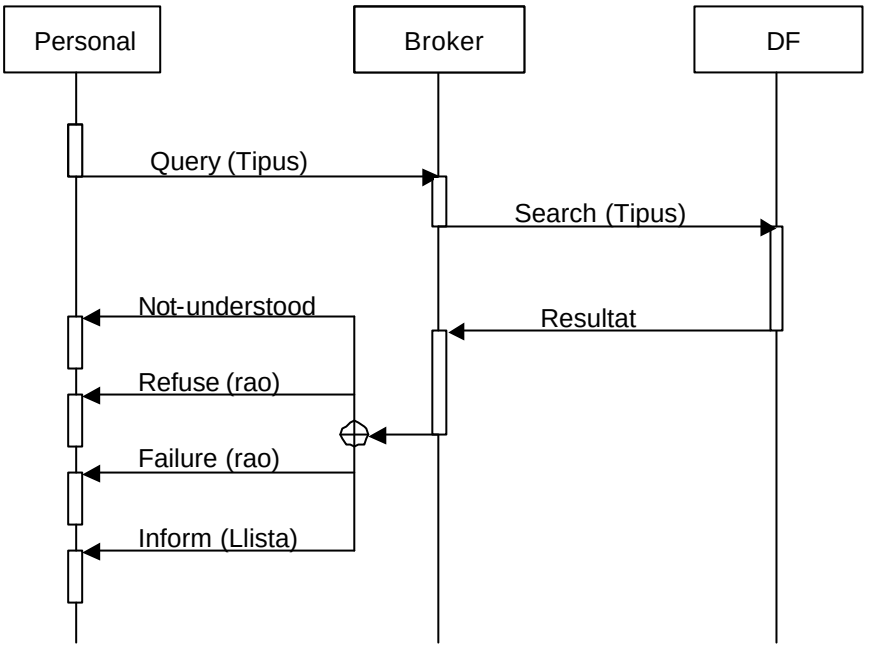

Figura 32 - Esquema de comunicació per a la cerca d'agents

7.1.3. Consulta de l'historial mèdic

En aquest cas, l'agent *Personal* voldrà accedir a les seves dades personals emmagatzemades a la BD, i més concretament al llistat d'episodis mèdics que estan continguts a la seva història.

Per fer-ho, s'utilitzarà un protocol de tipus FIPA-Query amb el *Personal Broker* que, per la seva part, el passarà a l'agent *Database,* el qual retornarà l'historial corresponent a la identitat del pacient que la demana.

En aquest cas, a més d'encriptar tots els missatges amb SSL, els que envia l'usuari cal que estiguin signats utilitzant la seva clau privada (que només ell coneix) per tal de comprovar inequívocament la seva identitat i donar-li accés a les dades crítiques contingudes al seu historial.

En el següent exemple, l'agent *Personal* envia la petició (**Query**) de l'historial al *Broker* encriptant el missatge amb la seva clau privada per tal de signar-lo:

```
(QUERY-REF
  :sender
    (agent-identifier
    :name Personal@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name personal_broker@grusma.agentcities.net
  :language message-codified  //Autenticació
  :ontology MedicalCenterOntology
```

```
  :protocol FIPA-Query
  :content
    (iota  //Buscar un element
      :iota-left "?x"
      :iota-right
        (is-patient-description
          :patient-var "?x"
          :patient-attributes
            (set)  //No hi ha atributs de cerca
          :patient-conditions
            (set :id-medical "39900002")  //Pacient buscat
        )
     )
)
```

El *Broker* comprovarà la identitat (desencriptant el missatge amb la clau pública associada a l'identificador del pacient i comprovant que el contingut és coherent) i enviarà la petició al *Database Agent*, que retornarà l'historial del pacient. A continuació, el retornarà cap a l'usuari amb el missatge d'**Inform**.

```
(INFORM
  :sender
    (agent-identifier
    :name personal_broker@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name Personal@grusma.agentcities.net
  :language RDFCodec
  :ontology MedicalCenterOntology
  :protocol FIPA-Query
  :content
    (equals
      :equals-left
        (iota   //Missatge original
        :iota-left "?x"
        :iota-right
          (is-patient-description
            :patient-var "?x"
            :patient-attributes
              (set)
            :patient-conditions
              (set id-medical "39900002")
          )
       )
     :equals-right
       (patient
         ... //Informació sobre el pacient (veure Ontologia)
        )
    )
)
```

L'esquema de comunicació per a aquesta situació es mostra a la **figura 33**:

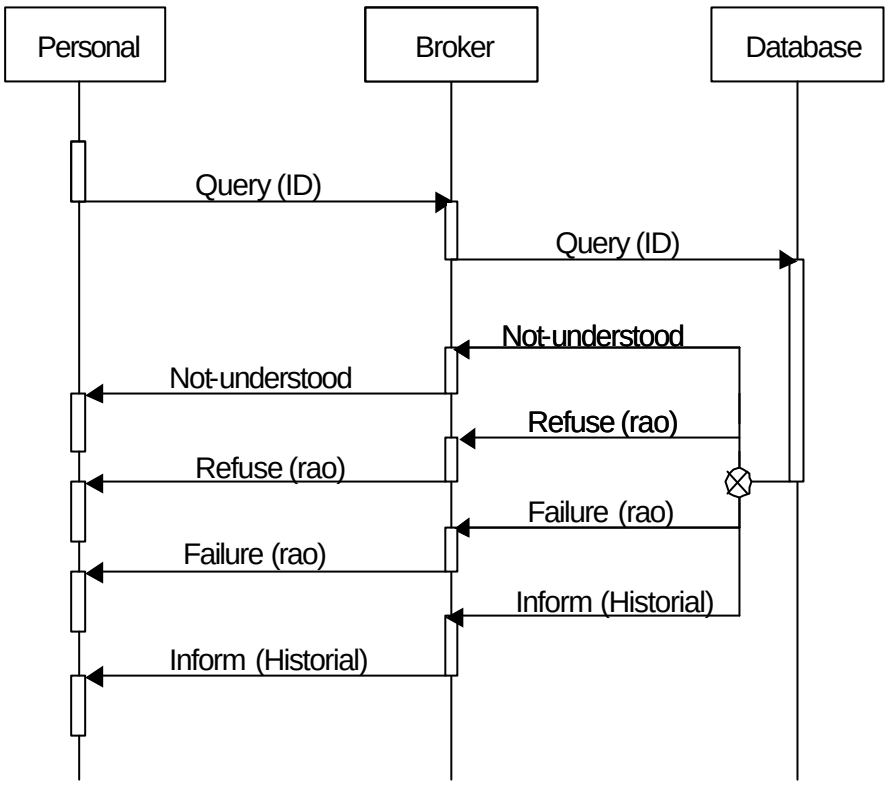

Figura 33 - Esquema de comunicació per a la consulta de l'historial mèdic

7.1.4. Consultar informació sobre centre mèdics

Un altre servei que se li proporciona a l'usuari és la possibilitat de demanar informació referent als *Medical Centres* presents al sistema que compleixin una sèrie de condicions (en quant a localització, especialitat mèdica, nom...).

Per fer-ho, l'agent *Personal* enviarà un missatge utilitzant el protocol FIPA-Query al *Broker* indicant les condicions a complir pels *Medical Centres*. Aquest consultarà els agents de tipus **Medical-Centre** del sistema, comprovarà quins compleixen les restriccions i enviarà la llista processada (ordenada segons criteris de proximitat) a l'usuari.

Els missatges s'encripten amb SSL però no s'autentiquen doncs es considera una consulta "anònima".

El servei s'inicia quan el *Personal Agent* envia un **Query** al *Broker* indicant les condicions definides per l'usuari:

```
(QUERY-REF
  :sender
    (agent-identifier
    :name Personal@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name personal_broker@grusma.agentcities.net
  :language RDFCodec
  :ontology MedicalCenterOntology
  :protocol FIPA-Query
  :content
    (all  //Buscar tots els centres mèdics
      :all-left ?x
      :all-right
        (is-medical-center-description
          :medical-var ?x
          :medical-attributes
            (set )  //No hi ha atributs de cerca
          :medical-conditions
            (set  //Llista de condicions a satisfer
              name="CAP"
              department="dentist"
              city_origin="Salou"
              city_dest="Reus"
            )
        )
     )
)
```

El *Personal Broker* avalua les condicions respecte als *Medical Centres* del sistema i retorna la llista processada adequadament a través d'un missatge d'**Inform**:

```
(INFORM
  :sender
    (agent-identifier
    :name personal_broker@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name Personal@grusma.agentcities.net
  :language RDFCodec
  :ontology MedicalCenterOntology
  :protocol FIPA-Query
  :content
    (equals
      :equals-left
        (all   //Missatge original
        :all-left ?x
        :all-right
          (is-medical-center-description
            :medical-var ?x
            :medical-attributes
              (set )
            :medical-conditions
              (set
                name="CAP"
```

```
                department="dentist"
                city_origin="Salou"
                city_dest="Reus"
              )
            )
          )
        :equals-right
          (set  //Llista de centres mèdics
            (healthcare-organisation
              ... //Informació del centre mèdic (veure Ontologia)
            )
            ...
          )
    )
)
```

En aquest cas, l'esquema d'intercanvi de missatges és el que s'indica a la **figura 34**:

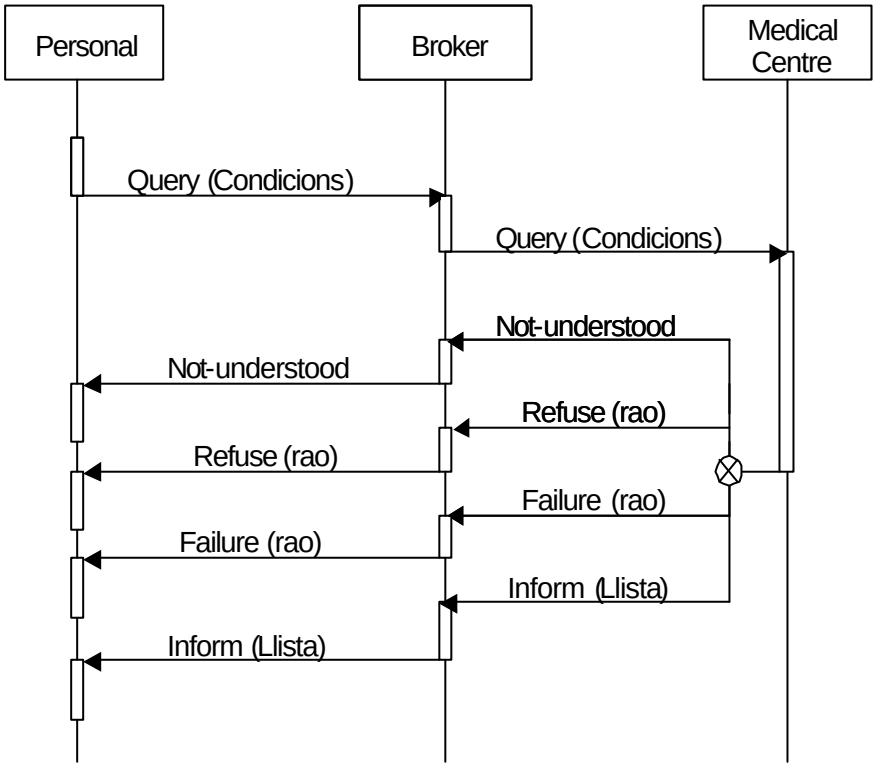

Figura 34 - Esquema de comunicació per a la consulta d'informació de centres mèdics

7.1.5. Negociació d'una visita amb un metge

Finalment, el servei més complex ofert a l'usuari és la negociació d'una visita amb un metge en un centre mèdic determinat a una certa hora per tractar un problema mèdic donat.

Per fer-ho, l'usuari estableix una sèrie de paràmetres que regiran la negociació (urgència, especialitat mèdica i centre mèdic). Aquestes dades s'enviaran al *Broker* mitjançant un protocol FIPA-Contract-Net amb un missatge de **Call-for-Proposals**. Aquest per la seva banda repetirà el procés en vers del *Medical Centre* escollit per l'usuari el qual passarà la petició al *Department* corresponent a l'especialitat mèdica (si estigués present) i finalment als *Doctors* que l'integren. Aquest últims retornaran una proposta de visita en funció del seu horari de treball, que es propagarà de forma inversa a la del primer missatge fins a arribar al *Broker*, que l'enviarà a l'usuari en forma d'un missatge de **Proposal**.

Arribats a aquest punt, l'usuari podrà escollir si acceptar o no la proposta (que serà la millor d'entre totes les que hagin formulat els *Doctors*), retornant un missatge de resposta (**accept** o **refuse**). En cas afirmatiu, el *Broker* el propagarà novament fins arribar als *Doctors* (que programaran la visita a l'hora fixada amb un *timer*) i retornarà una confirmació final a l'usuari. En cas negatiu, també s'indicarà a la resta d'agents i es finalitzarà la conversa.

Per tant, es tracta d'un complex protocol FIPA-Contract-Net de 4 nivells (*Personal-Broker-Medical_Centre-Department-Doctor*) en el que intervenen gairebé la totalitat dels agents del sistema per tal de trobar en cada cas la millor proposta possible per a les necessitats de l'usuari.

A més, donat que aquesta negociació implicarà la realització d'una visita, els missatges enviats per l'usuari estaran autenticats encriptant-los amb la seva clau privada (a més de l'SSL), de forma que el *Broker* pugui comprovar en tot moment si les peticions i les confirmacions o els refusos corresponents les fa l'usuari amb identitat esperada.

En el següent exemple l'usuari envia un **CFP** per demanar propostes de visita amb un metge autenticant-lo a través de la seva clau privada:

```
(CFP
  :sender
    (agent-identifier
    :name Personal@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name personal_broker@grusma.agentcities.net
  :language message-codified  //Autenticació
  :ontology MedicalCenterOntology
  :protocol FIPA-Contract-Net
  :content
    (action
      :action-left
        (agent-identifier  //Agent destí
          :name centre@grusma.agentcities.net
        )
      :action-right
        (visit-booking
```

```
          :patient-aid
            (agent-identifier
              :name Personal@grusma.agentcitites.net
            )
          :patient-id-medical "3990001"
          :visit-problem  //Atributs del problema mèdic
            (problem-description
              ... //Descripció del problema mèdic (veure Ontologia)
            )
          :... //La resta de camps estan buits
        )
    )
)
```

El *Broker* propagarà el missatge ¡ de totes les propostes que rebi, n'escollirà la millor (la més propera en el temps) que retornarà a l'usuari amb un missatge **Propose**:

```
(PROPOSE
  :sender
    (agent-identifier
    :name personal_broker@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name Personal@grusma.agentcities.net
  :language RDFCodec
  :ontology MedicalCenterOntology
  :protocol FIPA-Contract-Net
  :content
    (action
      :action-left
        (agent-identifier  //Agent destí
          :name centre@grusma.agentcities.net
        )
      :action-right
        (visit-booking
          :patient-aid
            (agent-identifier
              :name Personal@grusma.agentcitites.net
            )
          :patient-id-medical "3990001"
          :visit-problem  //Atributs del problema mèdic
            (problem-description
              ... //Descripció del problema mèdic (veure Ontologia)
            )
          :visit-place
            (healthcare-organisation
              ...  //Informació  del  centre  mèdic  escollit  (veure
Ontologia)
            )
          :visit-date "2003/2/1" //Data de la visita proposada
          :visit-hour "11:20" //Hora de la visita proposada
          :visit-doctor
            (doctor
              ... //Informació referent al doctor (veure Ontologia)
            )
          :medical-center-aid //Identificador del centre mèdic
            (agent-identifier
              :name center@grusma.agentcitites.net
```

```
            )
        )
    )
)
```

En aquest punt l'usuari podrà escollir si acceptar-la o no, autenticant el missatge a través de la seva clau privada. En el primer cas enviarà un missatge d'**Accept-proposal** com aquest:

```
(ACCEPT_PROPOSAL
  :sender
    (agent-identifier
    :name Personal@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name personal_broker@grusma.agentcities.net
  :language message-codified  //Autenticació
  :ontology MedicalCenterOntology
  :protocol FIPA-Contract-Net
  :content  //Mateix contingut que el missatge anterior
    (action
      :action-left
        (agent-identifier
          :name centre@grusma.agentcities.net
        )
      :action-right
        (visit-booking
          :patient-aid
            (agent-identifier
              :name Personal@grusma.agentcitites.net
            )
          :patient-id-medical "3990001"
          :visit-problem
            (problem-description
              ...
            )
          :visit-place
            (healthcare-organisation
              ...
            )
          :visit-date "2003/2/1"
          :visit-hour "11:20"
          :visit-doctor
            (doctor
              ...
            )
          :medical-center-aid
            (agent-identifier
              :name center@grusma.agentcitites.net
            )
        )
    )
)
```

El *Broker* el propagarà a la resta d'agents involucrats en la conversa i enviarà la confirmació final a l'usuari en cas de rebre el missatge **Inform** de la resta d'agents:

```
(INFORM
  :sender
    (agent-identifier
    :name Personal@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name personal_broker@grusma.agentcities.net
  :language RDFCodec
  :ontology MedicalCenterOntology
  :protocol FIPA-Contract-Net
  :content  //Mateix contingut que el missatge anterior
    (action
      :action-left
        (agent-identifier
          :name centre@grusma.agentcities.net
        )
      :action-right
        (visit-booking
          :patient-aid
            (agent-identifier
              :name Personal@grusma.agentcitites.net
            )
          :patient-id-medical "3990001"
          :visit-problem
            (problem-description
              ...
            )
          :visit-place
            (healthcare-organisation
              ...
            )
          :visit-date "2003/2/1"
          :visit-hour "11:20"
          :visit-doctor
            (doctor
              ...
            )
          :medical-center-aid
            (agent-identifier
              :name center@grusma.agentcitites.net
            )
        )
    )
)
```

L'esquema de comunicació per a aquest complex protocol es mostra a la **figura 35**:

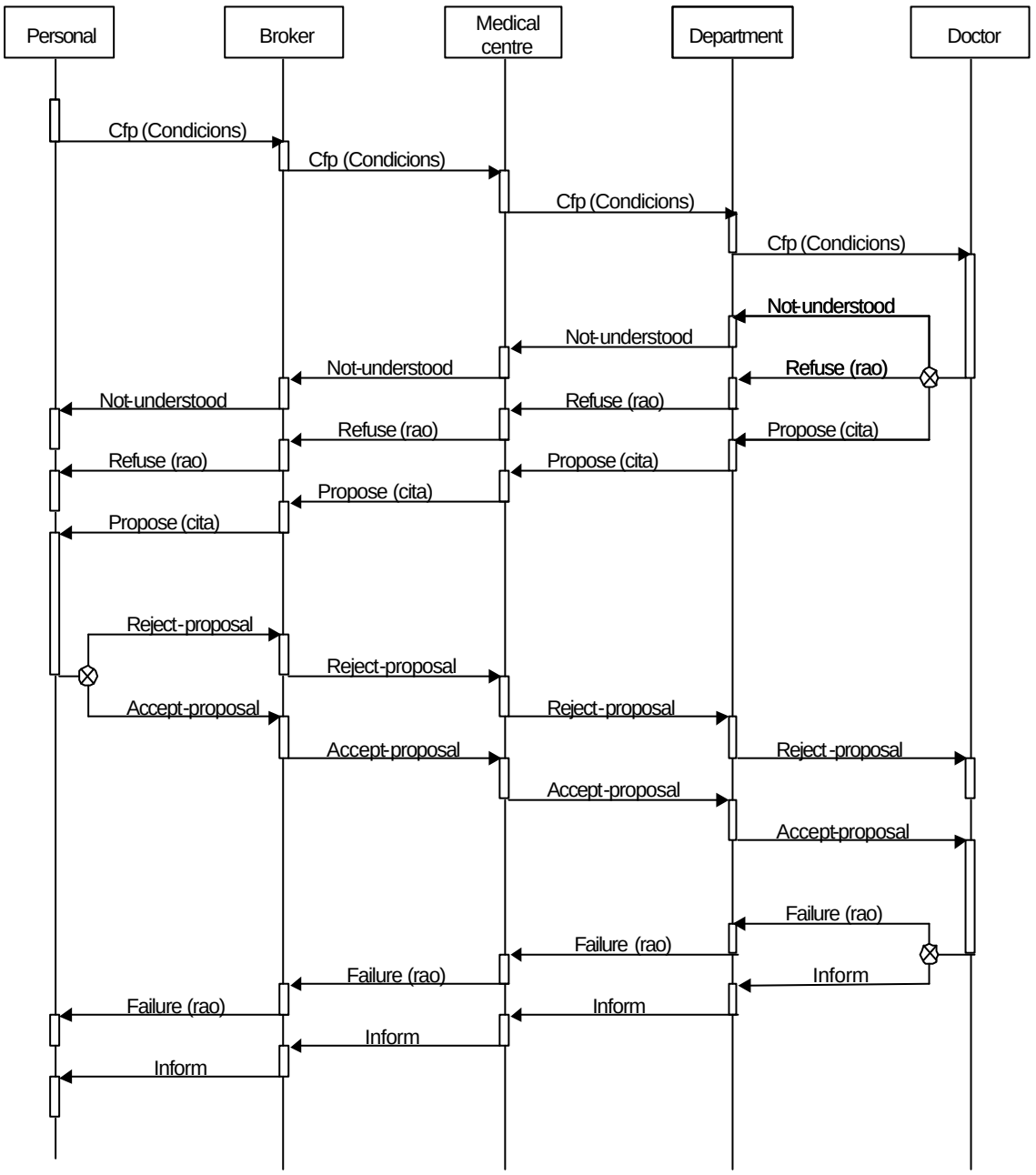

Figura 35 - Esquema de comunicació per a la negociació d'una visita

7.2. Serveis interns al sistema

7.2.1. Registrar un agent

Tal com s'ha explicat, els agents s'organitzen en una jerarquia (centres-departaments-doctors). Per tal d'oferir un bon servei, cada agent ha de conèixer els que estan per sota d'ell, per la qual cosa és necessari que el personal mèdic es registri en el seu departament i aquest en el seu centre mèdic.

Per fer-ho, s'utilitza el protocol FIPA-Request i juntament amb el missatge inicial (**Request**), l'agent a registrar envia la informació referent a les seves característiques.

En el següent exemple, un *Department* es registra en el seu *Medical Centre*:

```
(REQUEST
  :sender
    (agent-identifier
    :name department@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name centre@grusma.agentcities.net
  :language RDFCodec
  :ontology MedicalCenterOntology
  :protocol FIPA-Request
  :content
    (action
      :action-left
        (agent-identifier  //Agent que demana l'acció
          :name department@grusma.agentcities.net
      :action-right
        (register-agent  //Nom de l'acció
          :reg-agent
            (agent-identifier  //Agent que es vol registrar
              :name department@grusma.agentcities.net)
          :reg-info
            (is-department-description
              :department-description
                (department
                  ...  //Informació del departament (veure Ontologia)
                )
            )
        )
    )
)
```

En aquest cas l'agent *Department*, a més d'indicar que es dóna a conèixer, proporciona a l'agent al qual es vol subscriure una descripció seva. D'aquesta manera, l'agent *Medical Centre* sap quins són els *Departments* que el formen i la seva descripció. En el cas de la subscripció d'un *Doctor* al seu *Department*, aquesta informació no li cal perquè el *Department* només vol saber quins *Doctors* hi ha actius en un moment donat.
La resposta a aquest enregistrament, si la sintaxi era correcta, serà un missatge d'**Agree**:

```
(AGREE
```

```
  :sender
    (agent-identifier
    :name centre@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name department@grusma.agentcities.net
  :language RDFCodec
  :ontology MedicalCenterOntology
  :protocol FIPA-Request
  :content  //Mateix contingut que el missatge original
    (action
      :action-left
        (agent-identifier
          :name department@grusma.agentcities.net
      :action-right
        (register-agent
          :reg-agent
            (agent-identifier
               :name department@grusma.agentcities.net)
          :reg-info
            (is-department-description
              :department-description
                (department
                  ...
                )
            )
        )
    )
)
```

Finalment, s'enviarà un missatge d'**Inform** que indicarà que el procés de registre en el *Medical Centre* ha estat correcte.

```
(INFORM
  :sender
    (agent-identifier
    :name centre@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name department@grusma.agentcities.net
  :language RDFCodec
  :ontology MedicalCenterOntology
  :protocol FIPA-Request
  :content  //Mateix contingut que el missatge original
    (action
      :action-left
        (agent-identifier
          :name department@grusma.agentcities.net
      :action-right
        (register-agent
          :reg-agent
            (agent-identifier
               :name department@grusma.agentcities.net)
          :reg-info
            (is-department-description
              :department-description
                (department
                  ...
```

```
                    )
                )
            )
        )
)
```

L'esquema d'intercanvi de missatges en aquest cas és el de la **figura 36**:

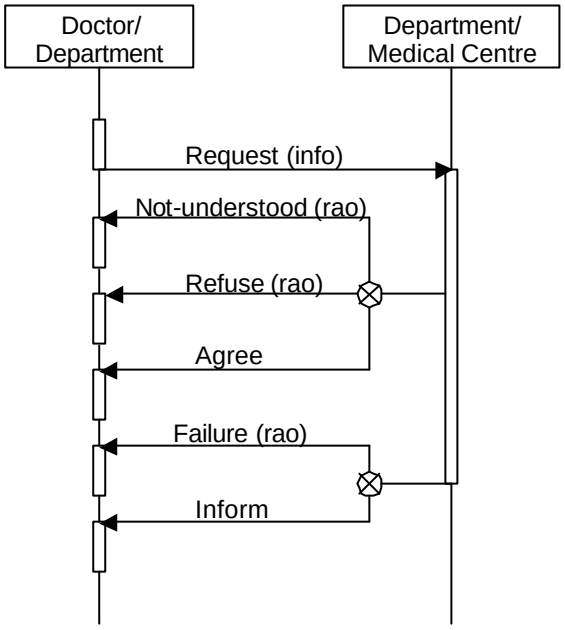

Figura 36 - Esquema de comunicació per al registre d'un agent

7.2.2. Preguntar la clau pública

A més del *Personal Agent*, els agents que representen el personal mèdic també cal que s'identifiquin abans de demanar accions o informació crítica en quant a consulta d'historials mèdics de certs pacients o fins i tot modificacions. Per tant, es fa necessària la utilització d'un mecanisme de signat dels missatges per tal de comprovar les identitats.

Novament s'utilitza el parell de claus pública i privada per realitzar el procés d'autenticació tal com s'ha descrit per al cas dels agents *Personal*.

En aquest cas, el personal mèdic demana al *Database Agent* que li generi un parell de claus (que seran vàlides durant el cicle de vida del sistema) i que les hi enviï. Per fer-ho, s'utilitza un protocol FIPA-Query amb encriptació dels missatges amb SSL.

Així, quan un *Doctor* s'ha donat d'alta, demana la seva clau amb el següent missatge de **Query**:

```
(QUERY-REF
```

```
  :sender
    (agent-identifier
    :name doctor@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name database@grusma.agentcities.net
  :language RDFCodec
  :ontology MedicalCenterOntology
  :protocol FIPA-Query
  :content
    (iota  //Buscar un element
      :iota-left "?x"
      :iota-right
        (key-rsa
          :key-agent
            (agent-identifier
              :name doctor@grusma.agentcities.net
            )
            .... //Informacio buida
        )
    )
)
```

La resposta del *Database Agent*, un cop obtingudes les claus, serà la següent:

```
(INFORM
  :sender
    (agent-identifier
    :name personal_broker@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name Personal@grusma.agentcities.net
  :language RDFCodec
  :ontology MedicalCenterOntology
  :protocol FIPA-Query
  :content
    (equals
      :equals-left //Missatge original
        (iota
          :iota-left "?x"
          :iota-right
            (key-rsa
              :key-agent
                (agent-identifier
                  :name doctor@grusma.agentcities.net
                )
                ....
            )
        )
      :equals-right //Resposta
        (key-rsa
          :key-agent
            (agent-identifier
              :name doctor@grusma.agentcities.net
            )
          :pub-key
            ....  //Clau publica
          :priv-key
```

```
            .... //Clau privada
        )
    )
)
```

En aquest cas, l'esquema de missatges intercanviats serà el que es mostra a la **figura 37**:

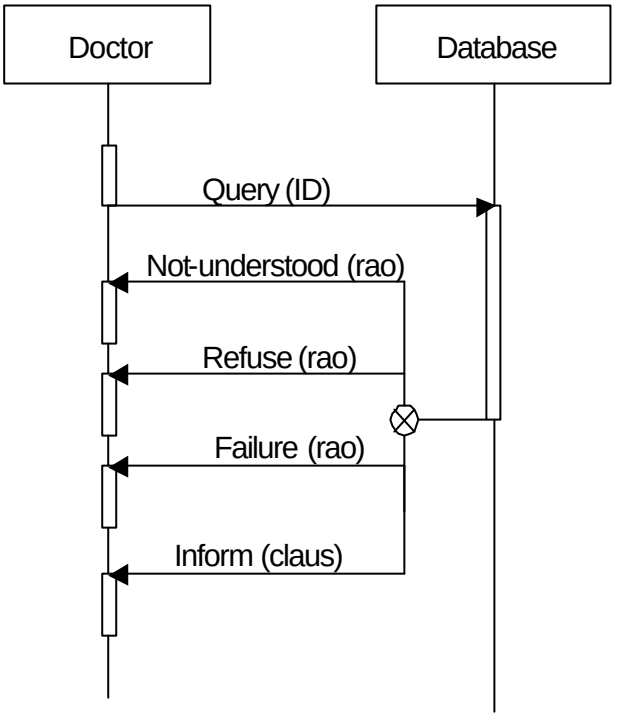

Figura 37 - Esquema de comunicació per a la petició de claus

7.2.3. Gestió dels horaris

El fet que els *Departments* coneguin al personal que hi treballa fa que d'alguna forma en pugui gestionar el seu treball. D'aquesta manera, s'ofereix un nou servei que permet donar dies lliures a algun dels agents subordinats a banda de l'horari setmanal prefixat a priori.

L'activació d'aquest servei per part del *Department* es realitza de manera aleatòria (doncs no hi ha intervenció de l'usuari) mitjançant un protocol FIPA-Request en el que s'indica al *Doctor* quin període de temps lliure se li dóna:

```
(REQUEST
  :sender
    (agent-identifier
    :name department@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name doctor@grusma.agentcities.net
  :language RDFCodec
  :ontology MedicalCenterOntology
  :protocol FIPA-Request
  :content
    (action
      :action-left
        (agent-identifier  //Agent que demana l'acció
          :name department@grusma.agentcities.net
      :action-right
        (free-time  //Nom de l'acció
          :free-agent
            (agent-identifier
              :name doctor@grusma.agentcities.net)
          :free-working-hours
            (working-hours    //Hores lliures
              :day-week "Monday"
              :begin "09:00"
              :end "19:00"
            )
        )
    )
)
```

El *Doctor* avaluarà el missatge i si està ben formulat respondrà amb un **Agree**:

```
(AGREE
  :sender
    (agent-identifier
    :name doctor@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name department@grusma.agentcities.net
  :language RDFCodec
  :ontology MedicalCenterOntology
  :protocol FIPA-Request
  :content  //Mateix contingut que el missatge original
```

```
    (action
      :action-left
        (agent-identifier
          :name department@grusma.agentcities.net
      :action-right
        (free-time
          :free-agent
            (agent-identifier
              :name doctor@grusma.agentcities.net)
          :free-working-hours
            (working-hours
              :day-week "Monday"
              :begin "09:00"
              :end "19:00"
            )
        )
    )
)
```

Finalment el *Doctor* enviarà un missatge d'**Inform** un cop hagi actualitzat el seu horari de treball eliminant les hores lliures. En el cas que ja tingués alguna visita programada dins de l'interval d'hores indicat, retornaria un missatge de **Failure**.

```
(INFORM
  :sender
    (agent-identifier
    :name centre@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name department@grusma.agentcities.net
  :language RDFCodec
  :ontology MedicalCenterOntology
  :protocol FIPA-Request
  :content  //Mateix contingut que el missatge original
    (action
      :action-left
        (agent-identifier
          :name department@grusma.agentcities.net
      :action-right
        (free-time
          :free-agent
            (agent-identifier
              :name doctor@grusma.agentcities.net)
          :free-working-hours
            (working-hours
              :day-week "Monday"
              :begin "09:00"
              :end "19:00"
            )
        )
    )
)
```

L'esquema de comunicació per a aquest servei s'inclou a la **figura 38**:

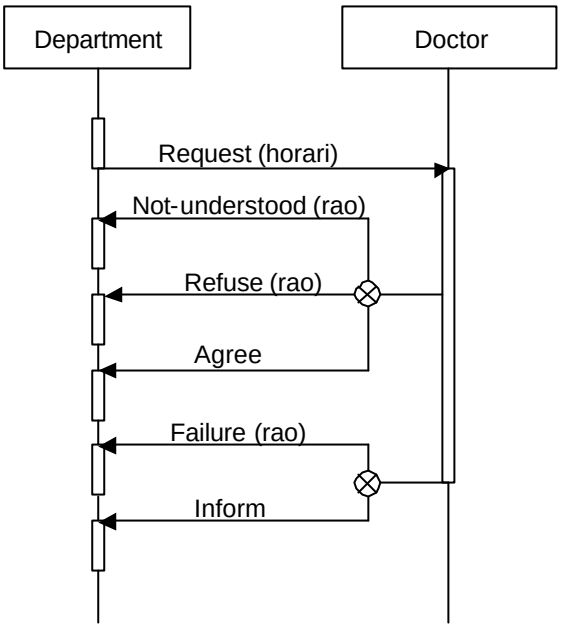

Figura 38 - Esquema de comunicació per a la gestió d'horaris

7.2.4. Actualització d'un historial mèdic

Durant el transcurs de una visita a un pacient, el *Doctor* pot voler actualitzar les dades del seu historial mèdic per tal d'incloure la informació del nou episodi (tractament, proves realitzades, diagnòstic...).

Donat que aquest és un procés crític i que només poden realitzar-lo els *Doctors*, es requereix que el missatge de petició estigui signat utilitzant la clau privada del *Doctor*. El protocol és el FIPA-Request i tal com s'ha dit, tots els missatges s'encripten amb SSL.

En el primer missatge, el *Doctor* envia la petició d'actualització al *Database Agent* juntament amb les noves dades que vol introduir a l'historial, autenticant el missatge de **Request** a través de la seva clau privada:

```
(REQUEST
  :sender
    (agent-identifier
    :name doctor@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name database@grusma.agentcities.net
  :language message-codified  //Autenticació
  :ontology MedicalCenterOntology
  :protocol FIPA-Request
  :content
    (action
      :action-left
```

```
        (agent-identifier  //Agent que demana l'acció
          :name doctor@grusma.agentcities.net
      :action-right
        (set-medical-record  //Nom de l'acció
          :medical-data
            (patient
              ... //Informació sobre el pacient
              :id-medical "39900002"  //Identificador del pacient
              :hitory
                (set
                  (episode
                    ... //Informacio del nou episodi mèdic
                  )
                )
            )
        )
    )
)
```

El *Database Agent* comprovarà la sintaxi del missatge així com l'identificador del pacient per comprovar si existeix un historial associat a modificar enviant a continuació un missatge d'**Agree**:

```
(AGREE
  :sender
    (agent-identifier
    :name database@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name doctor@grusma.agentcities.net
  :language RDFCodec
  :ontology MedicalCenterOntology
  :protocol FIPA-Request
  :content  //Mateix contingut que el missatge original
    (action
      :action-left
        (agent-identifier
          :name doctor@grusma.agentcities.net
      :action-right
        (set-medical-record
          :medical-data
            (patient
              ...
              :id-medical "39900002"
              :hitory
                (set
                  (episode
                    ...
                  ))))))
```

Finalment el *Database Agent* enviarà un missatge d'**Inform** un cop hagi actualitzat correctament l'historial del pacient incloent el nou episodi mèdic:

```
(INFORM
  :sender
```

```
   (agent-identifier
   :name centre@grusma.agentcities.net
 :receiver (set
   (agent-identifier
   :name department@grusma.agentcities.net
 :language RDFCodec
 :ontology MedicalCenterOntology
 :protocol FIPA-Request
 :content  //Mateix contingut que el missatge original
   (action
     :action-left
       (agent-identifier
         :name doctor@grusma.agentcities.net
     :action-right
       (set-medical-record
         :medical-data
           (patient
             ... //Informació sobre el pacient
             :id-medical "39900002"
             :hitory
               (set
                 (episode
                   ...
                 ))))))
```

L'esquema de comunicació per a aquesta situació és el que s'indica a la **figura 39**:

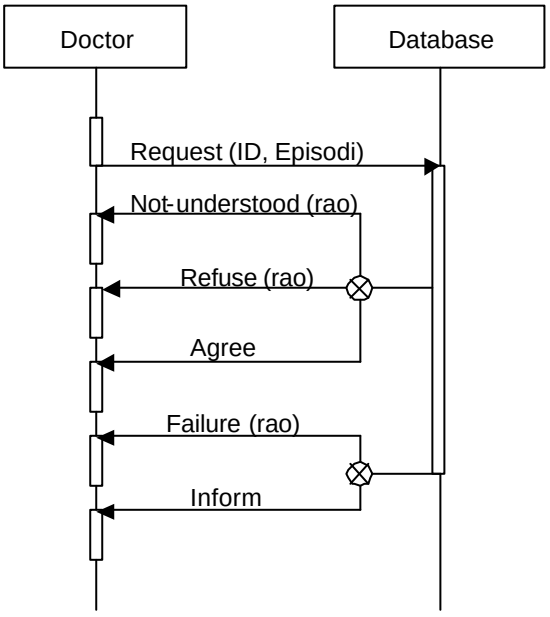

Figura 39 - Esquema de comunicació per a l'actualització d'historials mèdics

7.2.5. Obtenció de l'historial mèdic d'un pacient

Aquest servei és semblant al de la consulta de l'historial mèdic per part d'un pacient, però amb la diferència que en aquest cas la realitza un *Doctor* i no intervé el *Broker*; per tant, les peticions es realitzen directament sobre el *Database Agent*.

La situació que requereix d'aquest servei es produeix quan un *Doctor* s'activa per tal de simular una visita prèviament concertada amb un pacient. Llavors, haurà d'obtenir l'historial d'aquest pacient per tal de fer-lo servir durant la visita i poder actualitzar-lo adequadament amb el resultat d'aquesta.

Per fer-ho, s'utilitza un protocol de tipus FIPA-Query on se li passarà al *Database Agent* l'identificador del pacient, i aquest retornarà la seva informació personal i la llista d'episodis mèdics.

Com ja s'ha dit, a banda d'encriptar tots els missatges amb SSL, el *Doctor* caldrà que autentifiqui la petició d'aquest historial amb la seva clau privada, doncs només ell o el propi pacient han de tenir aquest a aquestes dades.

En el següent exemple, l'agent *Doctor* envia la petició (**Query**) de l'historial a l'agent *Database*, encriptant el missatge amb la seva clau privada per tal de signar-lo:

```
(QUERY-REF
  :sender
    (agent-identifier
    :name Doctor@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name Database@grusma.agentcities.net
  :language message-codified  //Autenticació
  :ontology MedicalCenterOntology
  :protocol FIPA-Query
  :content
    (iota  //Buscar un element
      :iota-left "?x"
      :iota-right
        (is-patient-description
          :patient-var "?x"
          :patient-attributes
            (set)  //No hi ha atributs de cerca
          :patient-conditions
            (set :id-medical "39900002")  //Pacient buscat
        )
     )
)
```

El *Database Agent* comprova l'autenticitat del missatge a través de la clau pública associada i retorna la informació demanada al *Doctor*.

```
(INFORM
  :sender
```

```
    (agent-identifier
    :name Database@grusma.agentcities.net
  :receiver (set
    (agent-identifier
    :name Doctor@grusma.agentcities.net
  :language RDFCodec
  :ontology MedicalCenterOntology
  :protocol FIPA-Query
  :content
    (equals
      :equals-left
        (iota   //Missatge original
        :iota-left "?x"
        :iota-right
          (is-patient-description
            :patient-var "?x"
            :patient-attributes
              (set)
            :patient-conditions
              (set id-medical "39900002")
          )
      )
     :equals-right
      (patient
        ... //Informació sobre el pacient (veure Ontologia)
       )
    )
)
```

El model de comunicació en aquest cas és el de la **figura 40**:

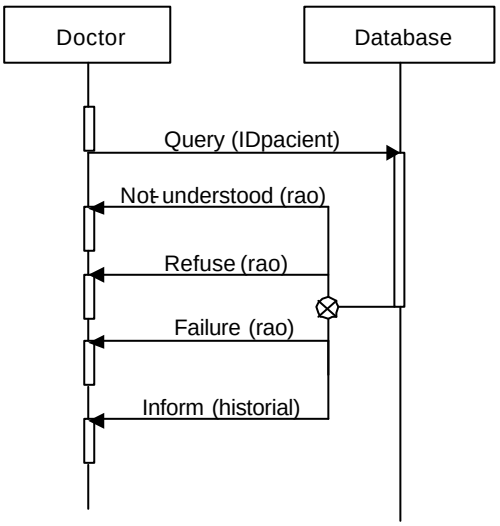

Figura 40 - Esquema de comunicació per a la consulta d'un historial mèdic

8. MANUAL D'USUARI

L'objectiu del nostre sistema és oferir una sèrie de serveis mèdics a l'usuari remotament de forma que es pugui comunicar des de qualsevol lloc i dispositiu (de moment només amb ordinadors estàndard). Aquest tindrà a la seva disposició un agent *Personal* que li permetrà accedir còmodament a la nostra plataforma i demanar certes funcions.

8.1. La plataforma d'agents

Actualment, els nostres agents s'executen en un servidor amb les següents característiques:

- Nom: grusma.agentcities.net
- Port: 10082
- Web: http://grusma.etse.urv.es/~agentcities
- Software utilitzat:
  - Java Development Kit (http://java.sun.com)
  - Jade 2.6.1 (http://sharon.cselt.it/projects/jade)
  - Jase-S security plug-in (http://sharon.cselt.it/projects/jade)
  - RDF add-on (http://sharon.cselt.it/projects/jade)
  - RDF parser (http://www-db.stanford.edu/~melnik/rdf/api.html)
  - Sax parser (http://xml.apache.org/xerces-j/index.html)
  - MySQL database (http://www.mysql.com)
  - Protègè 2000 (http://protege.stanford.edu)
  - BeanGenerator (http://gaper.swi.psy.uva.nl/beangenerator)
  - Cryptonite (http://logi.org/logi.crypto)
  - MySQL driver (http://mmmysql.sourceforge.net)
  - Netbeans (http://www.netbeans.org)

8.2. Descàrrega i configuració

El mòdul client que implementa el *Personal Agent* permet accedir al nostre sistema fàcilment i inclou una completa interfície gràfica; es pot descarregar de la direcció http://grusma.etse.urv.es/~agentcities/software.htm. Es requereix que es tingui instal·lat a la màquina client el *Java Runtime Environmet* versió 1.4.0_02 (aquest punt és molt important ja que els mecanismes de seguretat que implementa Jade-S només funcionen amb aquesta versió). La resta de llibreries utilitzades s'inclouen en el propi paquet de l'agent.

Un cop descarregat el mòdul, només caldrà descomprimir-lo en un directori del disc dur i executar un dels següents *scripts* de configuració que personalitzaran al client amb les dades personals de l'individu:

- **config.bat** per a sistemes operatius del tipus Win32
- **config.sh** per a sistemes operatius del tipus Linux/Unix

El programa demanarà que s'introdueixi certa informació personal (nom, DNI, direcció...) per tal de crear el perfil adequat per a l'agent del client. Aquest procés només cal executar-lo la primera vegada que s'accedeix al sistema ja que la informació proporcionada quedarà emmagatzemada a la BD (veure **apèndix D**).

Un exemple del resultat de l'execució d'aquest *script* és el següent:

```
**************************************************
**** WELCOME TO THE PERSONAL CONFIGURATION AGENT ****
**** (c) Grusma 2002-2003 (Agentcities project)  ****
**************************************************


IMPORTANT: you only have to execute this program the FIRST time

WARNING: only letters and digits allowed (non EMPTY or SPACE-starting
values);
Please, follow DATATYPES indications for each field

 - FIRST NAME (String, max 10 chars): David
 - FIRST SURNAME (String, max 10 chars): Sanchez
 - SECOND SURNAME (String, max 10 chars): Ruenes
 - DATE OF BIRTH (YYYY/MM/DD): 1980/09/23
 - PERSONAL IDENTIFICATION (Number, max 9 digits, UNIQUE!): 47763566
 - MEDICAL IDENTIFICATION (Number, max 10 digits, UNIQUE!): 2347263
 - LIVING CITY (String, max 20 chars): Salou
 - STREET NAME (String, max 30 chars): Faro
 - STREET NUMBER (Number, max 3 digits): 58
 - ZIP CODE (Number, max 5 digits): 43840
 - COUNTRY CODE (String, max 5 chars) (Ex: Spain -> SP): SP

Creating your patient identity.....
Writing file: DavidSanchezRuenes.txt
File closed succesfully.

Creating start scripts.....
Writing file: personal.bat
File closed succesfully.
Writing file: personal.sh
File closed succesfully.

All files created !

Type 'personal.bat' if you are using a Win32 OS
or 'personal.sh' if you have a Unix based OS
```

8.3. Accés als serveis oferts

Un cop el programa de configuració hagi finalitzat, haurà creat un fitxer descriptor (veure **apèndix C**) del l'usuari (Nom.txt) i un parell d'*scripts* que permetran l'accés al sistema a través de la creació d'un nou *Container* d'aquest usuari en concret. Així, quan es vulgui utilitzar algun servei mèdic, només caldrà executar:

- **personal.bat**: per a SO de tipus Win32
- **personal.sh**: per a SO de tipus Unix

L'agent *Personal* s'inicialitzarà amb les dades de l'usuari i la interfície gràfica es mostrarà amb la identitat corresponent. A partir d'aquest punt, es podran començar a demanar serveis al sistema, el funcionament dels quals s'explica a continuació.

8.3.1. Consulta d'informació

La **figura 41** mostra la interfície gràfica del *Personal Agent* amb l'àrea destinada a realitzar les consultes d'informació sobre centres mèdics.

Figura 41 – Finestra per consultar informació

La pantalla està dividida en dues parts: un formulari per introduir els criteris de cerca i una secció destinada a mostrar els resultats:

- El formulari té diferents camps que permeten concretar el resultat de les consultes. Els quatre camps disponibles són precedits d'un *checkbox*; si s'activa es tindrà en compte el valor d'aquell camp en la cerca (d'altra manera estaran inactius). Els camps possibles i el seu significat són els següents:

    - Nom del centre: podem introduir alguna paraula que formi part del nom del centre mèdic que volem buscar. Per exemple, si utilitzem la paraula "CAP", se'ns donarà la llista de Centres d'Atenció Primària.

    - Departament: podem indicar un departament en concret corresponent a una especialitat mèdica. Concretament tenim: *ophthalmology*, *emergency*, *dentist*, *family psychologist*, *general medicine*, *oncology* i *pedriatics*.

    - Ciutat origen: com que el sistema està pensat per a usuaris mòbils que poden estar en qualsevol lloc, aquesta ciutat representa la localització de l'usuari. La informació la utilitzarà el *Broker* per ordenar els resultats de la cerca segons un criteri de proximitat (de més propers a més llunyans).

    - Ciutat destí: si es vol restringir la ciutat on es troba el centre mèdic buscat, es pot indicar en aquest camp. Tant les ciutats d'aquest camp com les de l'anterior estan restringides al domini del projecte i es troben al voltant dels centre mèdics considerats, concretament: Tarragona, Reus, Salou, Les Borges del Camp, Vila-Seca, Cambrils.

- L'àrea de resultats és una zona de text on es presenten els resultats de les consultes de l'usuari: informació sobre el centre mèdic, localització, horari de treball, departaments...

El sistema està pensat per poder incorporar diversos *Personal Broker* que tractin amb subconjunts de *Medical Centres* (per repartir la carrega d'execució). Per tant, existeix un camp que permet la selecció del *Broker* al que s'enviaran les peticions de cerca.

Per fer efectiva la cerca, només cal polsar el botó **send**; amb el botó **reset** s'inicialitzen els valors de les condicions de cerca.

8.3.2. Consulta d'informació personal

Si es polsa el botó **Personal Info** situat just a sota de la foto de l'usuari, apareixerà una nova pantalla com la de la **figura 42** on es podrà veure la informació relativa a l'individu (nom, direcció, dni...) i la de l'agent *Personal* que s'està utilitzant (nom, ID...). A més, es pot consultar l'historial mèdic polsant el botó **Consult History**. Cada línia de l'historial representarà un episodi en el que s'indica informació relativa al doctor, data, problema mèdic...

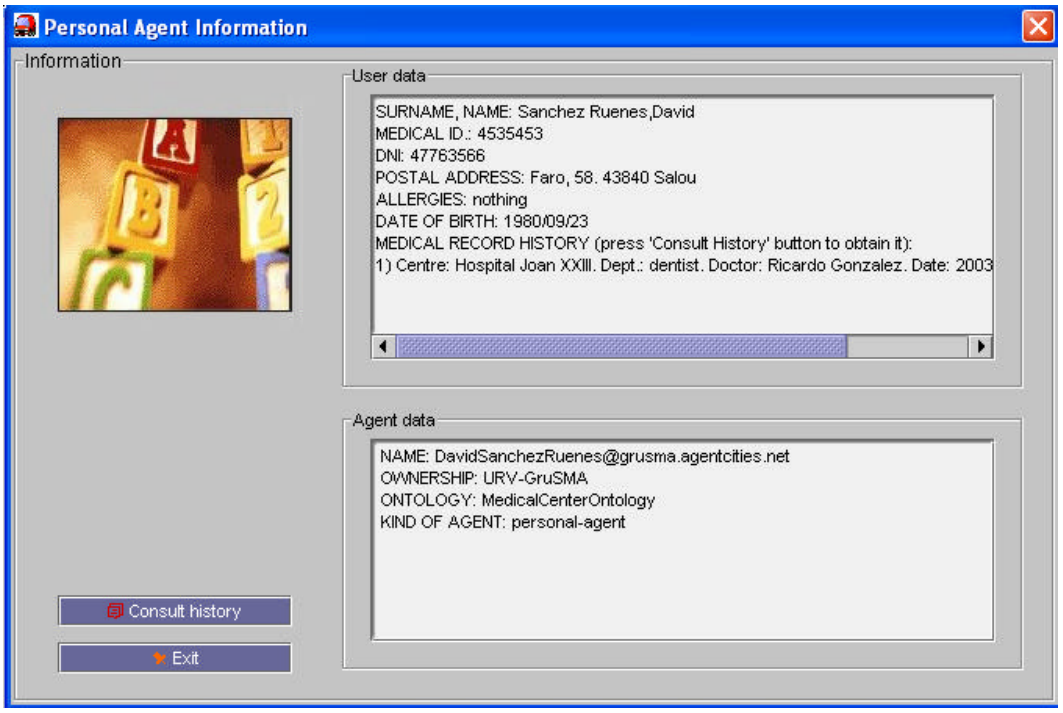

Figura 42 - Finestra per consultar l'historial mèdic

8.3.3. Agents presents al sistema

Polsant amb el ratolí a sobre la il·lustració que representa als agents (sota de la foto de l'usuari), s'accedeix a una nova secció que ens permet consultar la llista d'agents que s'estan executant en aquell moment al sistema (veure **figura 43**). Això ens permet fer-nos una idea de l'arquitectura interna i, per tant, dels resultats que podem esperar de les nostres consultes o peticions.

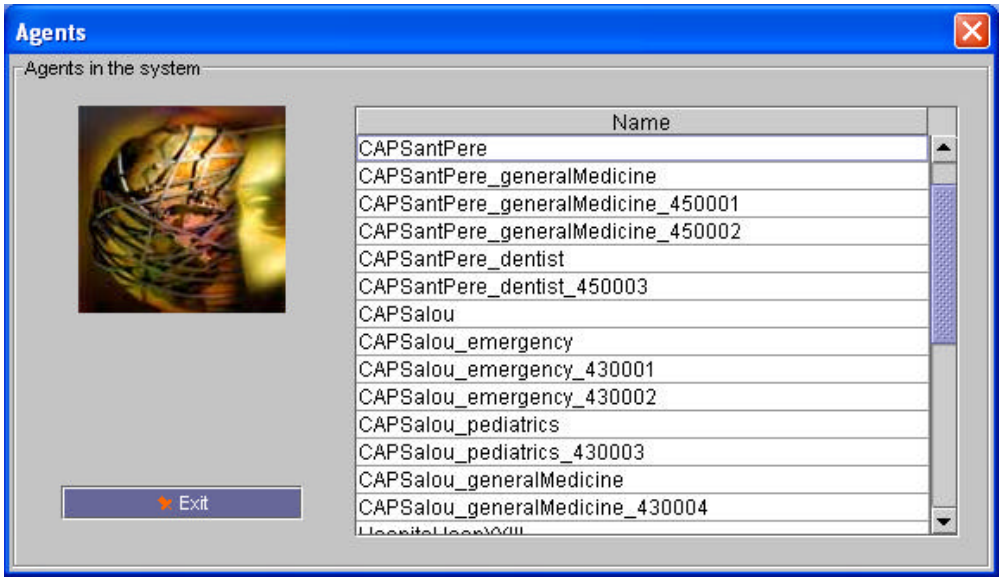

Figura 43 - Finestra amb els agents disponibles al sistema

8.3.4. Reserva d'hora

Si polsem sobre la carpeta **Bookings** accedirem a la secció que ens permet negociar visites amb els metges tal com es mostra a la **figura 44**.

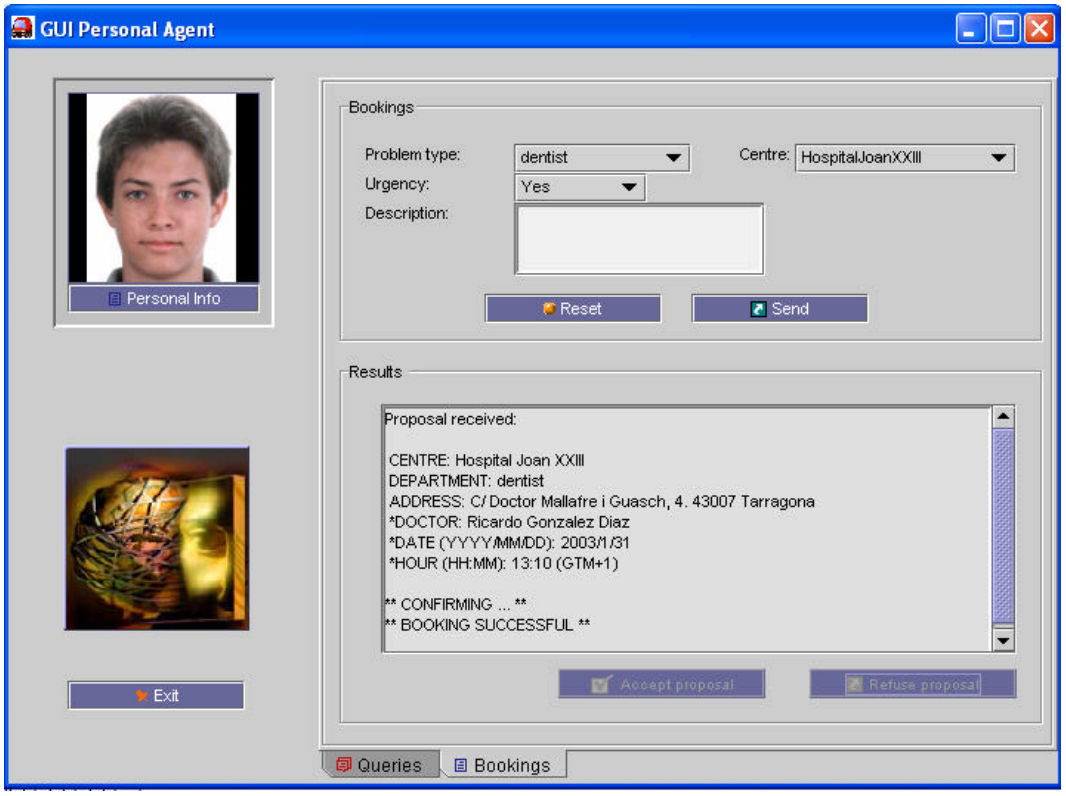

Figura 44 - Finestra per reserva hora amb el metge

En aquest cas, la pantalla està dividida en dues parts: una àrea superior amb el formulari i una d'inferior per mostrar/supervisar els resultats:

- La part superior conté el formulari per introduir les condicions de la reserva:
    - Tipus de problema: aquest camp és determinant per tal que el *Medical Centre* escollit comprovi si és capaç de satisfer el problema o no (associant-lo a una de les especialitats mèdiques considerades). Si no existeix un departament adequat, no es realitzaran propostes.
    - Urgència: aquest valor booleà determina si s'intentarà buscar una hora de visita a partir del mateix dia o bé a partir dels dos dies següents.
    - Descripció del problema: aquí indicarem una breu informació referent al problema mèdic per tal que el doctor es faci una idea del que es trobarà. Tot i això, aquest camp es opcional.
    - Centre mèdic: aquí se'ns proporciona una llista amb els *Medical Centres* presents al sistema de forma que puguem escollir aquell al que volem anar. La llista la proporcionarà el *Broker* que s'ha escollit en aquell moment.

- L'àrea inferior ens permet veure els resultats de les peticions de reserves. En un primer moment se'ns mostrarà una possible proposta (dia, hora, metge, centre mèdic), en cas d'existir, donant-los la possibilitat d'acceptar-la o no a través del parell de botons situats a la part inferior. Sigui quina sigui la decisió de l'usuari, finalment se'ns mostrarà el resultat del procés (èxit o refús).

8.4. Arquitectura del sistema

Per tal d'entendre correctament els resultats obtinguts de les diferents consultes, cal tenir en compte l'arquitectura del nostre sistema, principalment pel que fa als centres mèdics i la seva organització interna (departaments i doctors) que han estat modelats a base d'agents.

Així, actualment, s'estan executant els següents centres mèdics:

- CAP Sant Pere (Reus):
    - Departament *General Medicine*:
        - Doctor 450001: Sergio Lopez (Dilluns, Dimecres, Divendres)
        - Doctor 450002: Paco Montes (Dimarts, Dijous)
    - Departament *Dentist*:
        - Doctor 450003: Antonio Garcia (Dilluns, Dimecres, Divendres)
- CAP Salou (Salou):
    - Departament *Emergency*:
        - Doctor 430001: Juan Vazquez (Dilluns, Dimecres, Divendres)
        - Doctor 430002: Maria Sanchez (Dimarts, Dijous)
    - Departament *Pediatrics*:
        - Doctor 430003: Julian Sanz (Dimarts, Divendres)
    - Departament *General Medicine*:
        - Doctor 430004: Lluis Gonzalez (Dilluns, Dimarts, Dimecres, Divendres).
- Hospital Joan XXIII (Tarragona):
    - Departament *Emergency*:
        - Doctor 640001: Sonia Martinez (Dilluns, Dimecres, Divendres)
        - Doctor 640002: Pere Millan (Dimarts, Dijous)
    - Departament *Pediatrics*:
        - Doctor 640003: Miguel Angel Garcia (Dimecres, Divendres)
    - Departament *General Medicine*:
        - Doctor 640004: David Gonzalez (Dilluns, Dimecres, Divendres)
        - Doctor 640005: Marc Morell (Dimarts, Dijousy)
    - Departament *Ophthalmology*:
        - Doctor 640006: Anna Hernandez (Dimarts, Dimecres, Dijous, Divendres)
    - Departament *Oncology*:
        - Doctor 640007: Javier Lopez (Dilluns, Dimecres, Divendres)
    - Departament *Dentist*:
        - Doctor 640008: Ricardo Gonzalez (Dimecres, Dijous, Divendres)

9. CONCLUSIONS I VALORACIÓ

Els agents suposen un gran avenç en IA. El fet de que diversos individus treballin conjuntament per aconseguir un objectiu els confereix una capacitat de reacció i de decisió força complexes. Una entitat per sí sola no tindria gaire èxit imitant el comportament humà, però la combinació i cooperació d'un grup d'elles amb diversos comportaments fa que, globalment, les seves actuacions siguin força intel·ligents (tal com fan algunes comunitats d'insectes). És en aquest punt precisament on els Sistemes Multi-Agent (SMA) demostren com poden ser d'útils per resoldre determinades tasques que individualment serien massa complexes.

El camp dels SMA és una tècnica molt nova dins de l'IA i encara està en procés de desenvolupament però, gràcies a organismes internacionals sense ànim de lucre com la FIPA, que defineix estàndards, o projectes com Agentcities, que en promou la utilització, s'aconsegueixen avenços molt ràpidament.

Pel que fa a la realització d'aquest projecte, he de dir que ha estat una experiència molt positiva i que n'estic molt satisfet. Tot i que en principi la feina de recerca va ser força dura doncs requeria entendre i dominar un extens projecte realitzat per David Isern [35], reescriure gran part del codi a les noves versions de JADE (que no destaquen especialment per la seva bona documentació o exemples pràctics), i investigar sobre temes força dispersos i abstractes referents a Ontologies (moltes d'elles no accessibles directament), a mesura que s'anaven assolint els objectius marcats a l'inici l'interès anava creixent.

Així, per exemple, el tema de la seguretat va ser força interessant, doncs es tractava d'un tema pioner en els sistemes Multi-Agent i que permetia aplicar conceptes apresos en altres assignatures. De fet, tan aviat com va sortir la primera versió del plug-in JADE-S [64] el vam començar a utilitzar. Val a dir, que aquest tema va suposar força problemes degut a les limitacions i *bugs* pròpies d'una versió en desenvolupament tot i que també s'ha d'admetre que ens va facilitar força el treball en alguns aspectes que ja no calia implementar a mà.

Però el que finalment va suposar tot un repte va ser el concurs celebrat a l'ID3 [1] (el primer d'aquestes característiques), doncs requeria que tot el sistema estigués disponible i funcionant al servidor proporcionat per Agentcities a una data límit per tal de poder avaluar-lo. Això implicava testejar el sistema profundament ja que s'havia d'executar de forma permanent al servidor i que no presentés falles d'execució en cap circumstància. També s'havia de facilitar l'accés dels usuaris per poder demanar-hi serveis, per la qual cosa es van elaborar manuals, una pàgina web explicativa [31] i es va adaptar l'agent *Personal* per tal que funcionés de forma autònoma adaptant-se al perfil del pacient. Tot aquest treball va anat acompanyat d'extensos informes en els que s'explicava la feina realitzada (per justificar la beca d'Agentcities) i per presentar el nostre projecte al concurs.

Tenint en compte la magnitud de la competició (nivell mundial) i el nombre de competidors presentats (39), en principi no tenia gaires esperances i veia el concurs com un element circumstancial. Però a mesura que s'anaven superant les diferents fases fins a arribar a la final de Barcelona, les possibilitats en el projecte creixen juntament amb l'interès del mitjans de comunicació per l'aplicació (ràdio, premsa, televisió).

Així, els últims dies van ser frenètics preparant la demostració que s'havia de realitzar durant la final (configuració d'ordinadors, elaboració de documentacions, pòsters...); fins a arribar el dia de la competició on vaig poder viure en primera persona la realitat dels SMA: interès de les empreses, evolució del mercat, projectes en curs, personalitats importants... el que va suposar molt més que un simple projecte acadèmic.

El reconeixement final amb els premis aconseguits (tercer premi en la categoria d'aplicacions i premi especial del públic) ja va ser una experiència inoblidable doncs, veient el nivell dels participants, no tenia cap expectativa (tot i que Toni Moreno no va perdre mai la confiança).

Com a conclusió vull destacar el fet que aquest projecte ha estat essencialment un treball d'equip (tot que ara presenti aquesta documentació a títol individual), no només pel fet de basar-se en el projecte de David Isern [35] (el que fa que la meitat del mèrit sigui seu), sinó perquè durant tot el desenvolupament tant Toni Moreno com Aïda Valls i David Isern han estat plenament involucrats en la feina, dirigint-me i ajudant en totes les situacions on ha estat necessari. També cal mencionar l'ajuda prestada per Agentcities doncs ha estat l'encarregada de proporcionar tots els mitjans tècnics per poder realitzar el desenvolupament (a través de la Beca i el Servidor) així com per l'organització de la primera competició mundial d'Agents.

Així, les meves últimes paraules van adreçades a agrair als meus companys i caps el recolzament constant i la confiança dipositada per oferir-me la possibilitat de participar en aquest projecte.

10. TREBALLS FUTURS

El principal objectiu d'aquest projecte és demostrar que la tecnologia d'agents és una bona candidata per a utilitzar en un entorn real que proporcioni serveis mèdics als ciutadans i millori la gestió dels centres mèdics.

Tot i això, encara queden força punts a tractar per tal d'obtenir un model de simulació realista.

En aquest àmbit, seria interessant estendre el sistema per arribar a simular l'organització sanitària de Catalunya anomenada CatSalut [55]. En aquesta existeix una jerarquia d'àrees mèdiques de tres nivells. El nivell més baix és l'*Àrea Bàsica de Salut* (ABS). Cada ABS controla un servei mèdic que cobreix algunes zones d'una ciutat, o alguns pobles d'un àrea rural. El següent nivell és l'anomenat *Sector Sanitari* (SS). Un SS cobreix uns quants ABSs i té l'objectiu de coordinar diversos serveis d'un àrea (ex: especialitats mèdiques, accions de prevenció...). Finalment, tenim el servei més alt anomenat la *Regió Sanitària* (RS). Cada RS té suficients recursos per atendre de forma autònoma les necessitats sanitàries dels ciutadans d'una regió. Catalunya té 8 RSs que cobreixen els 6 milions d'habitants (veure **figura 45**).

Figura 45 - Organització sanitària de Catalunya

Així, una possible arquitectura [40] d'agents per simular un SS seria la que inclou a la **figura 46**:

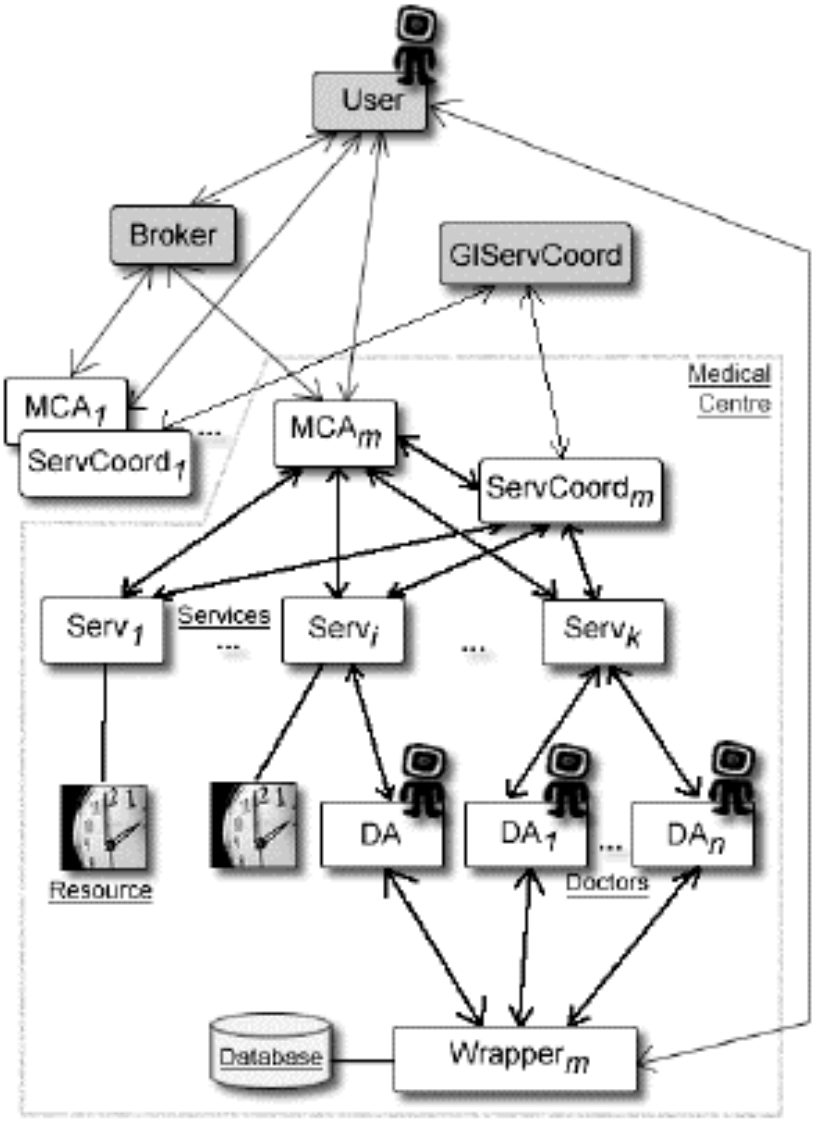

Figura 46 - Arquitectura extesa del SMA

S'observa com s'han afegit una sèrie d'agents nous que simulen els serveis mèdics addicionals i els recursos que estan associats a cada departament d'un centre, així com els Coordinadors de serveis de cada centre i el Coordinador Global, amb els quals es pretén simular la gestió dels recursos sanitaris (ex: raigs-X, quiròfans...). D'aquesta forma els doctors podran demanar la realització de diferents proves davant un cert problema mèdic trobat durant una visita i el sistema s'encarregarà de proposar la data en que el recurs requerit està lliure (coordinant diversos elements dins del propi centre o en vers altres centres). Els resultats dels tests (radiografies, anàlisis de sang...) s'emmagatzemaran en l'historial del pacient de forma que el doctor el pugui accedir fàcilment en una futura visita.

A partir d'aquest model, es podrien afegir diversos SS fent-los accessibles a través del seu *Broker* corresponent. Així, l'usuari podria escollir de quina zona o regió vol consultar informació mèdica de forma més precisa i estructurada aconseguint un model de simulació distribuït en zones.

Altres possibles millores que podríem afegir al nostre sistema són:

- Provar el sistema amb dades reals de centres mèdics, departaments, doctors i serveis a la ciutat de Tarragona.

- Ampliar els serveis que s'ofereixen a l'usuari.

- Possibilitat d'interacció amb els agents que simulen els centres mèdics, principalment els doctors, de forma que la simulació de les visites es realitzés realment a través d'un procés de consulta a un especialista mitjançant un agent personal per als metges.

- Execució distribuïda de cadascun dels elements del sistema (centres mèdics, Base de dades...) en ordinadors diferents, fet que en milloraria l'eficiència (avui dia, la complexitat del sistema simulat està restringida a la potència de càlcul d'un únic servidor). Tot i això, aquest model distribuït afegiria una nova problemàtica de seguretat que caldria estudiar.

- Possibilitat d'accés de la informació a través d'una interfície web (en lloc d'un agent *Personal*) que faria el sistema més "assequible" a tot tipus d'usuaris, tal com es fa per exemple a través de l'ICS (*Institut Català de la Salut*) [34] però amb tota la potència i versatilitat en quant a personalització del servei i gestió eficient d'un SMA.

- Implementació d'agents personals sobre dispositius mòbils (com PDAs o telèfons mòbils) per tal de permetre a l'usuari d'accedir-hi des de qualsevol lloc.

APÈNDIX

A. Escrivint codi per JADE 2.61

JADE és una eina força útil i eficaç per desenvolupar aplicacions basades en agents, però el fet que encara estigui en desenvolupament, provoca que a cada nova versió es realitzin multitud de canvis. De vegades, només es corregeixen errors o es millora el rendiment però altres, suposen una redefinició important de certs aspectes de l'aplicació. Concretament, des de la versió 2.3 a la 2.61, s'han dut a terme força canvis importants que fan referència als protocols de comunicació i la definició d'ontologies. Aquestes modificacions, tot i que no impedeixen que aplicacions programades en versions antigues funcionin, si en restringeixen la vida útil, doncs en versions futures, les antigues funcions s'arribaran a suprimir.

A més, la documentació de JADE no cobreix tots els aspectes que fan referència a aquests canvis de forma que hem hagut d'investigar per tal de trobar l'equivalència entre les funcions antigues i les noves. Per tal de facilitar el procés de migració d'aplicacions escrites per a versions antigues, hem realitzat aquest manual que cobreix els principals aspectes que afecten als canvis realitzats.

A.1. Integració amb el projecte LEAP

A partir de la versió 2.4, JADE inclou la funcionalitat de les llibreries del projecte LEAP que permet oferir una plataforma d'agents amb un consum reduït de recursos i compatible amb l'especificació de *mobile Java* J2ME.

Aquest fet ha provocat que determinades estructures disponibles en Java estàndard però que no estan suportades a l'especificació J2ME, hagin estat substituïdes per les equivalents en les llibreries LEAP (sobretot afecta a dades dinàmiques que consumeixen gran quantitat de recursos). Per aquest motiu, moltes de les estructures disponibles a la llibreria **`java.util`** han estat canviades per les la nova llibreria **`jade.util.leap`**.

Aquest canvi afecta sobretot a les llistes (**List**) que són molt utilitzades en el desenvolupament d'agents per crear i llegir els missatges que s'intercanvien. Així, la **`java.util.list`** i els mètodes que la utilitzen han estat *deprecated* en front de la nova **`jade.util.leap.list`** i en el futur seran eliminats completament.

El principal canvi en el codi, afecta als mètodes de definició i extracció del contingut dels missatges disponibles en la classe `Agent`; així, la següent seqüència de codi:

```
Import java.util.List;

myAgent.fillContent(ACLMessage msg, List l);
List l = myAgent.extractContent(ACLMessage msg);
```

S'haurà de substituir per les següents sentències equivalents:

```
Import jade.util.leap.List;

myAgent.fillMsgContent(ACLMessage msg, jade.util.leap.List l);
jade.util.leap.List l = myAgent.extractMsgContent(ACLMessage msg);
```

La nova llista de la llibreria LEAP té els mateixos constructors i mètodes manipulació que la antiga llista de Java, així que el codi que la resta de codi de programa no cal modificar-lo.

NOTA: en cas que es faci servir la nova definició de l'ontologia a partir del paquet **`jade.content`** (en lloc del **`jade.onto`**), aquest canvis no caldrà realitzar-los, doncs s'utilitzaran les funcions de **`fillContent`** i **`extractContent`** incloses en la classe **`ContentManager`** (en lloc de la classe `Agent` que es fa servir en els exemples anteriors). Aquest dos mètodes no utilitzen cap llista (reben/retornen directament l'objecte de l'ontologia) i, per tant, no els afecten els canvis descrits en aquest punt.

Un altre canvi important es produeix en l'accés als objectes **`DFServiceCommunicator`** i **`AMSServiceCommunicator`**, ja que internament utilitzen l'antiga **List** i han estat substituïts pels objectes equivalents **`DFService`** i **`AMSService`**. Tots els mètodes d'un i altre es mantenen iguals, tot i que les cerques (**search**) que abans retornaven una llista que s'havia de convertir a la corresponent **`DFAgentDescription`**, ara retornen directament un *array* de descriptors. Així, el següent codi que implementa la cerca d'agents al sistema:

```
List result = DFServiceCommunicator.search (Agent parent,
DFAgentDescription dfd, SearchConstraints sc);

if (result.size() > 0) {
      for (int k=0; k<result.size(); k++) {
            dfd = (DFAgentDescription)result.get(k);
            agentsBuscats.addElement(dfd.getName());
      }
}
```

S'haurà de substituir per les següents sentències:

```
DFAgentDescription[] result = DFService (Agent parent,
DFAgentDescription dfd, SearchConstraints sc);

if (result.length > 0) {
      for (int k=0; k<result.length; k++) {
            agentsBuscats.addElement(result[k].getName());
      }
}
```

A.2. Protocols de comunicació

A.2.1. Protocols pregunta/resposta

Des de la versió 2.4 de JADE, tots els protocols de tipus petició/pregunta – resposta (FIPA-Query i FIPA-Request), han estat agrupats sota una interfície comuna (inclosa a la **figura 47**).

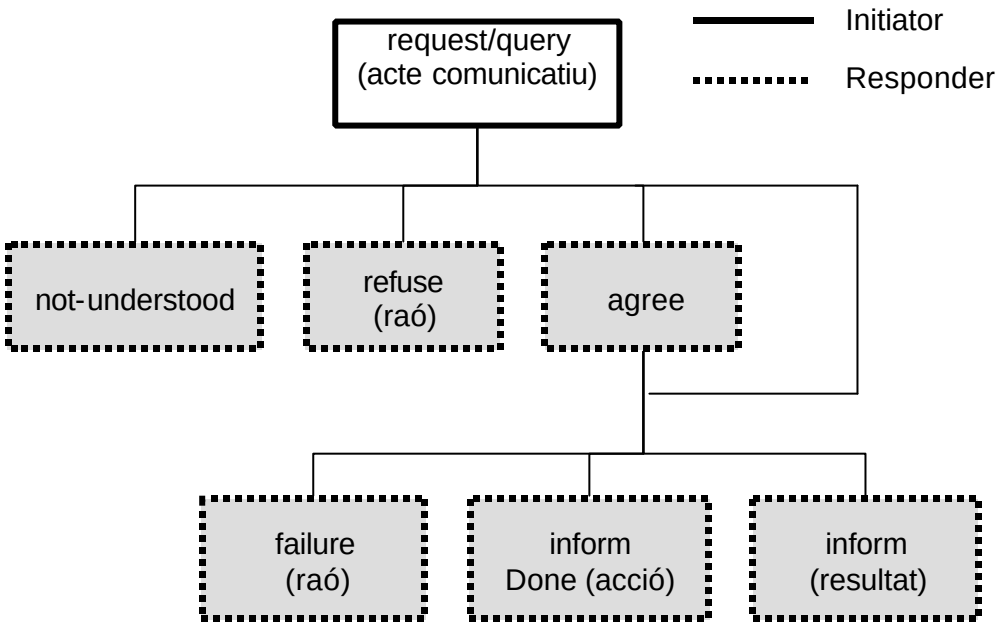

Figura 47 - Definició homogènia de protocols pregunta/resposta

Aquest fet permet que ara es puguin suportar de forma indirecta tots els protocols definits per la FIPA que tenen un comportament comú (doncs ara la implementació no fa referència a un protocol específic). Concretament, la llista de protocols que es poden utilitzar és: FIPA-Request, FIPA-Query, FIPA-Propose, FIPA-Request-When, FIPA-Recruiting, FIPA-Brokering.

Les noves classes de què es disposa dins del paquet **`jade.proto`** són:

- **`AchieveREInitiator`**: substitueix la classe **`FipaXXXInitiatorBehaviour`** (on XXX és el nom de protocol corresponent) que implementa la banda que inicia la conversa.

- **`AchieveREResponder`**: substitueix a la classe **`FipaXXXResponderBehaviour`** (on XXX és el nom de protocol) que implementa la banda que rep les peticions i envia respostes.

Alguns dels mètodes que l'antiga especificació no estan disponibles en la nova o han estat substituïts per altres que no són compatibles (sobretot a la banda del receptor), per tant, caldrà realitzar la implementació seguint les especificacions de la nova API tal com es descriurà més endavant.

Alguns dels canvis més importants són els següents:

- En protocols en els que es demana realitzar una acció (ex: **Request**), la fase intermitja en la que el destí envia un missatge d'**Agree** informant que està en disposició de realitzar-la (enviant més tard un **Inform**), és opcional de forma que directament es pot enviar el resultat de l'acció just després de la petició. Això és útil en el cas d'accions de cost reduït en les que el missatge d'**Agree** suposava un *overhead* extra, tot i que no se segueixi estrictament la definició de la FIPA.

- Ara existeixen mètodes que s'encarreguen de tractar de forma conjunta tots els missatges de la primera fase de la comunicació, on s'informa si es pot realitzar o no l'acció (`handleAllResponses`) i la segona on es retorna el resultat (`handleAllResultNotifications`).

- És possible registrar l'arribada d'un determinat missatge amb un comportament definit per l'usuari.

Per al cas d'un **Request** caldrà utilitzar la seqüència de missatges: **Request-Agree-Inform**, mentre que per a un **Query**, només serà necessari implementar la parella **Query-Inform**.

Les classes que implementen aquests protocols estan disponibles al paquet `jade.proto`:

- `AchieveREInitiator`: fa les vegades d'iniciador de la conversa (realitza la pregunta-petició). S'encarrega de crear el missatge inicial (**Request** o **Query**) i enviar-lo al destí; a continuació esperarà a rebre les respostes.

- `AchieveREResponder`: implementa la banda que respon a les peticions. Si el protocol es de tipus **Request**, caldrà que primer s'enviï una confirmació del missatge (**Agree**) i a continuació el resultat final (**Inform**).

Opcionalment, des de la versió 2.6 de JADE existeix una parella alternativa de classes que consumeixen menys recursos a costa de reduir la funcionalitat: `SimpleAchieveREInitiator` i `SimpleAchieveREResponder`. La principal restricció afecta al fet de que cada missatge només podrà ser enviat a un receptor.

Les seves característiques són les següents:

- Els seus mètodes i la funcionalitat de cadascun d'ells és la mateixa que per a les classes **`AchiveREInititator`** i **`AchiveREResponder`**.

- No disposa de mètodes per a registrar comportaments definits per l'usuari a cada tipus de missatge que pugui arribar.

- Tots els missatges que s'enviïn seran de 1:1, és a dir, el camp del destinatari del missatge només pot ser un sol agent. En cas d'haver-hi més d'un, només s'enviarà al primer de la llista i la resta s'ignoraran.

- El fet de ser una implementació més simple permet que la seva execució sigui més ràpida i es consumeixin menys recursos.

Tenint en compte aquestes característiques, l'usuari podrà decidir quin parell de classes utilitzar segons les seves necessitats.

NOTA: els missatges enviats en Jade tenen un conjunt d'*slots* que cal omplir correctament i completament abans d'enviar-los: protocol, llenguatge, ontologia, emissor, receptor... Mitjançant la classe **MessageTemplate** és possible especificar per a molts d'aquest *slots* un valor esperat (exemple: l'identificador d'un agent determinat en el camp d'emissor o un protocol específic en l'*slot* corresponent). Quan passem una instanciació d'aquesta classe amb una definició de valors específics per a alguns camps a un constructor de resposta d'un protocol, aquestes restriccions s'aplicaran a tots els missatges rebuts, de forma que només s'acceptaran els que concordin amb els valor especificats als camps del **MessageTemplate**.

A.2.1.1. Adaptació del protocol FIPA-Query

INITIATOR

El primer pas és crear una classe que estengui a **`AchieveREInitiator`** o bé a **`SimpleAchieveREInitiator`** (en lloc de la **`FipaQueryInitiatorBehaviour`** antiga).

El següent pas és la implementació dels mètodes de tractament dels missatges rebuts.

En la antiga especificació disposàvem de:

- **`void handleOtherMessages (ACLMessage msg)`**: s'executava cada cop que es rebia un missatge que no era de tipus **Inform**.
- **`void handleInformMessages (Vector msg)`**: s'executava un cop rebuts tots els missatge de tipus **Inform** (normalment només 1).

En la nova especificació haurem d'implementar els següents:

- **`void handleInform (ACLMessage msg)`**: s'executa quan es rep un missatge de tipus **Inform**. Aquest mètode és equivalent a l'antic **`handleInformMessages`** però s'executa quan només s'ha rebut un sol missatge, sense esperar a tenir-los tots (d'altra banda, aquesta sol ser la situació habitual).
- **`void handleNotUnderstood (ACLMessage msg)`**: s'executa cada cop que es rep un missatge del tipus **Not-understood**.
- **`void handleRefuse (ACLMessage msg)`**: ídem amb **Refuse**.
- **`void handleFailure (ACLMessage msg)`**: ídem amb **Failure**.

Aquest tres últims mètodes equivalen al comportament de l'antic **`handleOtherMessage`** però amb l'avantatge de poder distingir de forma immediata el tipus de missatge.

Alternativament, es poden fer servir aquests altres dos mètodes també disponibles, tot i que l'estructura acabada de descriure sigui més intuïtiva:

- **`void handleAllResponses (Vector responses)`**: espera a rebre tots els missatges del tipus **Not-understood** o **Refuse** abans d'executar-se.
- **`void handleAllResultNotifications (Vector notifications)`**: espera a rebre tots els missatges del tipus **Inform** o **Failure** abans d'executar-se.

NOTA: el mètode **`done()`** que calia implementar en l'antiga especificació indicant el moment en que s'acaba el protocol, ara és implícit i el manega el propi agent en funció del tipus de missatges rebuts.

Finalment, caldrà definir el constructor i instanciar-lo passant-li l'Agent que l'executarà i el missatge que s'enviarà en el primer pas del protocol. Aquest apartat no canvia en la nova especificació.

Com a resum, l'esquelet que s'haurà d'implementar per tal de utilitzar aquest protocol serà el següent (els mètodes són el mateixos tant per a la implementació "Simple" com normal):

```
class QueryInitiator extends (Simple)AchieveREInitiator {
```

```
      public QueryInitiator (Agent myAgent, ACLMessage queryMsg) {
            Super (myAgent, queryMsg);
      }

      protected void handleInform (ACLMessage msg) {...}

      protected void handleNotUnderstood (ACLMessage msg) {...}

      protected void handleRefuse (ACLMessage msg) {...}

      protected void handleFailure (ACLMessage msg) {...}

}
```

RESPONDER

De la mateixa forma, en la banda de les respostes, caldrà crear una classe que estengui a **`AchiveREResponder`** o bé a **`SimpleAchieveREResponder`** (en lloc de l'antiga **`FipaQueryResponderBehaviour`**).

Pel que fa als mètodes de manegament dels missatges, en l'antiga versió teníem:

- **`ACLMessage handleQueryMessage (ACLMessage msg):`** que rebia un missatge de tipus **Query** i retornava la resposta (**Inform**, **Failure**, **Not-understood** o **Refuse**).

En la nova especificació disposem del següent mètode:

- **`ACLMessage prepareResponse (ACLMessage msg)`**: que té el mateix comportament.

El constructor i la posterior instanciació es faran passant-li l'Agent que executa el protocol i un **MessageTemplate** que filtri els missatges acceptats.

En resum, l'esquema de mètodes a implementar per a aquest protocol serà el següent:

```
class QueryResponder extends (Simple)AchiveREResponder {

      public QueryResponder (Agent myAgent, MessageTemplate mt) {
            Super (myAgent, mt);
      }

      protected ACLMessage prepareResponse (ACLMessage msg) {...}

}
```

A.2.1.2. Adaptació del protocol FIPA-Request

INITIATOR

Per al cas de la banda iniciadora de la conversa per al protocol FIPA-Request, el primer pas serà la creació de la classe que estengui a **`AchieveREInitiator`** o bé a **`SimpleAchieveREInitiator`** (en lloc de l'antiga **`FipaRequestInitiatorBehaviour`**).

En l'antiga especificació disposàvem dels següents mètodes de manegament de cada tipus de missatge rebut:

- **`void hanldeAgree (ACLMessage msg)`**
- **`void handleInform (ACLMessage msg)`**
- **`void handleNotUnderstood (ACLMessage msg)`**
- **`void handleFailure (ACLMessage msg)`**
- **`void handleRefuse (ACLMessage msg)`**

En la nova definició, tots aquest mètodes es mantenen amb el mateix nom i comportament, de manera que no caldrà realitzar cap canvi.

Alternativament, es dóna la possibilitat de fer servir el següent parell de mètodes nous:

- **`void handleAllResponses (Vector responses)`**: s'executa quan es reben tots els missatges dels tipus **Agree**, **NotUnderstood** o **Refuse**.
- **`void handleAllResultNotifications (Vector notifications)`**: s'executa un cop rebuts tots els missatges del tipus **Failure** o **Inform**.

Tant en la versió antiga com la nova, el constructor de la classe rebrà l'Agent que executa el protocol i el missatge que s'enviarà en el primer moment (**request**).

En resum, l'estructura de mètodes a implementar per a aquest protocol serà la següent:

```
class RequestInitiator extends (Simple)AchieveREInitiator {

      public RequestInitiator (Agent myAgent, ACLMessage requestMsg) {
            Super (myAgent, requestMsg);
      }

      protected void handleAgree (ACLMessage msg) {...}

      protected void handleInform (ACLMessage msg) {...}

      protected void handleNotUnderstood (ACLMessage msg) {...}

      protected void handleFailure (ACLMessage msg) {...}

      protected void handleRefuse (ACLMessage msg) {...}

}
```

RESPONDER

La banda de resposta del protocol FIPA-Request era completament diferent de la resta de protocols i, per tant, en la nova versió s'ha adaptat per tal que s'utilitzi una definició comuna.

El primer pas és crear la classe que estengui a **`AchieveREResponder`** o bé a **`SimpleAchieveREResponder`** (en lloc d'entendre el manegador **`FipaRequestResponderBehaviour.ActionHandler`** i implementar l'interfície **`FipaRequestResponderBehaviour.Factory`**).

En l'antiga especificació, calia implementar l'**ActionHandler** que enllacés el missatge rebut amb el constructor de la classe per tal d'executar el mètode per crear les respostes. Ara només caldrà definir el constructor tal com s'ha fet en els apartats anteriors passant-li l'agent i el **MessageTemplate** que indica les característiques dels missatges que s'acceptaran. La instanciació i posterior creació del comportament, es farà de la mateixa forma que en la resta de protocols (sense haver de registrar la implementació del **Factory** per poder manegar missatges rebuts).

Com a mètode de manipulació de missatges, antigament disposàvem de:

- **`void action()`**: que era executat un cop cridat al constructor de la classe dels de l'**ActionHandler** corresponent. El missatge que s'havia rebut es llegia amb el mètode **getRequest** i calia que manegués la creació de les respostes: en un primer moment avaluava l'acció demanada i enviava un missatge indicant si la podia fer o no (**Agree** o **Refuse**) i en la segona part (només si s'havia enviat un **Agree**), creava el missatge amb el resultat (**Inform** o **Failure**).

En la nova especificació, es defineixen mètodes de manipulació que s'apropen més a la definició disponible per a la resta de protocols:

- **`ACLMessage prepareResponse(ACLMessage request)`**: s'executa un cop rebut un missatge de tipus **Request** i s'encarrega d'avaluar l'acció demanada i crear la resposta (**Agree** o **Refuse**).
- **`ACLMessage prepareResultNotification (ACLMessage request, ACLMessage reponse)`**: s'executa només si el missatge que s'ha retornat en el mètode anterior era del tipus **Agree** i es reben per paràmetres el missatge de **Request** original i la resposta d'**Agree** que s'acaba d'enviar. S'encarregarà de realitzar l'acció demanada i retornar un missatge amb el resultat.

L'esquelet que caldrà implementar per utilitzar aquest protocol serà el següent:

```
Class RequestResponder extends (Simple)AchieveREResponder {

      public RequestResponder (Agent myAgent, MessageTemplate mt) {
            Super (myAgent, mt);
      }

      protected ACLMessage prepareResponse(ACLMessage request) {...}
      protected   ACLMessage   prepareResultNotification   (ACLMessage
      request, ACLMessage reponse) {...}

}
```

A.2.2. FIPA-Contract-Net

Des de Jade 2.5, l'especificació del protocol FIPA-Contract-Net (veure **figura 48**) ha canviat per tal d'adaptar-se a l'especificació homogènia dels protocols FIPA-Query i FIPA-Request.

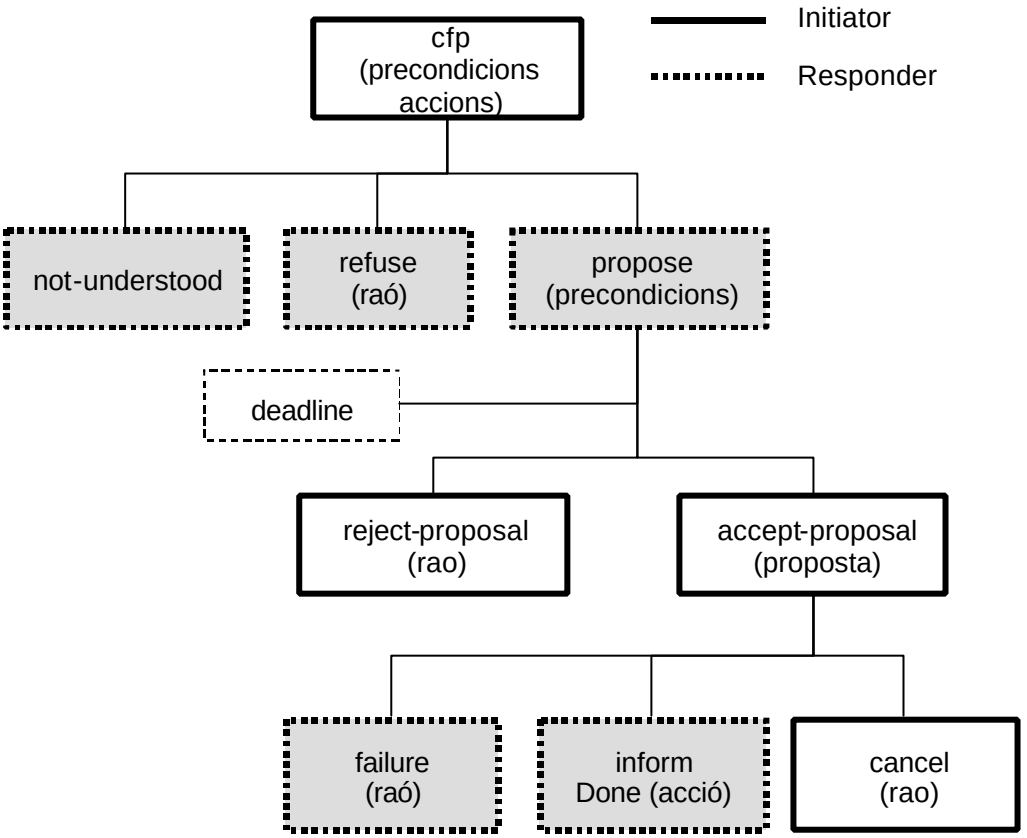

Figura 48 - Protocol FIPA-Contract-Net

Concretament, al paquet **`jade.proto`**, trobem les classes **`ContractNetInitiator`** i **`ContractNetResponder`** que defineixen les dues bandes de la conversa.

A.2.2.1. Adaptació del protocol FIPA-Contract-Net

De la mateixa forma que per a la resta de protocols, a partir de JADE 2.5 les dues classes de què disposàvem antigament per implementar el FIPA-Contract-Net, **`FipaContractNetResponderBehaviour`** i **`FipaContractNetInitiatorBehaviour`**, han estat substituïdes per les noves **`ContractNetInitiator`** i **`ContractNetResponder`**, de manera que tinguin la mateixa estructura que s'ha definit per a la resta de protocols.

INITIATOR

El primer pas per implementar la banda que inicia la conversa és crear la classe que estengui a **`ContractNetInitiator`** (en lloc de l'antiga **`FipaContractNetInitiatorBehaviour`**).

En l'antiga versió, els mètodes que calia implementar per manegar els missatges rebuts eren els següents:

- **`Vector handleProposeMessages (Vector proposals)`**: s'executava un cop rebuts tots els missatges de tipus **Propose** i crear les respectives respostes afirmatives (**Accept**) o negatives (**Reject**).
- **`void handleOtherMessages (ACLMessage msg)`**: s'executava cada cop que es rebia un missatge diferent de **Propose**.
- **`Vector handleFinalMessages (Vector msg):`** s'executava un cop rebuts tots els missatges que informaven dels resultats de les accions (**Inform** o **Failure**).
- **`String createCfpContent(String cfpContent, AID receiver)`**: per a cada receptor del missatge, es permetia definir un contingut específic.

En la nova especificació disposem dels següents mètodes:

- **`void handleAllResponses(Vector responses, Vector results)`**: s'executa un cop rebudes totes les respostes a les proposicions. Per a les que siguin afirmatives (**Propose**) caldrà crear un missatge de resposta que es retornarà al segon paràmetre. Aquest mètode és equivalent als dos primers de l'antiga versió.

Alternativament, també disposem del següent conjunt de mètodes que maneguen els missatges de manera independent:

- **`void handlePropose (ACLMessage propose, Vector acceptances)`**: s'executa cada cop que es rep un missatge de tipus **Propose** i afegeix la seva corresponent resposta al vector del segon paràmetre. Aquest mètode és semblant al **`handleProposeMessages`** de l'antiga especificació, amb la diferència que s'executa un cop per cada missatge rebut.
- **`void handleNotUnderstood (ACLMessage msg)`**: ídem per missatges **Not-understood**.
- **`void handleRefuse (ACLMessage msg)`**: ídem per missatges **Refuse**.

Aquest dos últims mètodes agrupen la funcionalitat del **`handleOtherMessages`** de l'antiga especificació.

Per manegar els resultats de les accions, disposem dels següents mètodes:

- **`void handleAllResultNotifications (Vector result)`**: s'executa un cop rebuts tots els resultats de les accions (**Failure** o **Inform**). Aquest mètode és equivalent al **`handleFinalMessages`** de l'antiga especificació.

Alternativament, disposem d'aquests altres dos mètodes:

- **`void handleInform (ACLMessage msg)`**: s'executa cada cop que es rep un missatge tipus **Inform**.
- **`void handleFailure (ACLMessage msg)`**: ídem per missatge **Failure**.

Per crear el missatge inicial, podem optar per passar-li directament al constructor per paràmetres o ve implementar el mètode:

- **`Vector prepareCfps (ACLMessage msg)`**: rep el missatge passat al constructor i retorna el vector amb tots els missatge que s'enviaran als destins.

Tant en l'antiga versió com en la nova, la definició del constructor i la posterior instanciació es fa passant-li l'Agent que executarà el protocol i el missatge que s'enviarà inicialment a tots els destins.

En general, la forma més convenient de implementar aquest protocol es fa seguint la següent estructura:

```
class CNInitiator extends ContractNetInitiator {

      public CNInitiator (Agent myAgent, ACLMessage CfpMsg) {
            Super (myAgent, CfpMsg);
      }

      protected void handlePropose (ACLMessage propose, Vector
      acceptances) {...}

      protected void handleNotUnderstood (ACLMessage msg) {...}

      protected void handleRefuse (ACLMessage msg) {...}

      protected void handleInform (ACLMessage msg) {...}

      protected void handleFailure (ACLMessage msg) {...}

}
```

RESPONDER

El primer pas és la definició de classe que estendrà a **`ContractNetResponder`** (en lloc de l'antiga **`FipaContractNetResponderBehaviour`**).

Els mètodes de què disposàvem antigament per manegar missatges eren els següents:

- **`ACLMessage handleCfpMessage (ACLMessage cfp)`**: s'executava en rebre un missatge del tipus **CFP** i retornava la resposta.
- **`ACLMessage handleAcceptProposalMessage (ACLMessage msg)`**: s'executava en rebre un missatge d'**AcceptProposal** i retornava el resultat de l'acció (**Inform** o **Failure**).
- **`void handleRejectProposalMessage (ACLMessage msg)`**: avaluava els missatges de tipus **RejectProposal**.
- **`void handleOtherMessages (ACLMessage msg)`**: s'executava quan es rebia algun missatge que no era de cap tipus anterior (**Not-understood**). A la pràctica aquest mètode no s'executava mai.

En la nova especificació, disposem dels següents mètodes equivalent:

- **`ACLMessage prepareResponse (ACLMessage msg)`**: s'executa en rebre un missatge del tipus **CFP** i ha de retornar la resposta (**Not-understood**, **Refuse** o **Propose**). És equivalent al **`handleCfpMessage`** anterior.
- **`ACLMessage prepareResultNotification (ACLMessage cfp, ACLMessage propose, ACLMessage accept)`**: s'executa un cop rebut un missatge de tipus **AcceptProposal** i rep el **CFP** inicial, el missatge de proposta enviat i el d'acceptació rebut. Ha d'executar l'acció i retornar el resultat (**Inform** o **Failure**). És equivalent al **`handleAcceptProposalMessage`** de l'antiga especificació.
- **`void handleRejectProposal (ACLMessage cfp, ACLMessage propose, ACLMessage reject)`**: és equivalent al **`handleRejectProposalMessage`** anterior.

Novament, el constructor no canvia, caldrà indicar-li l'Agent que executa el protocol i el **MessageTemplate** que s'aplicarà als missatges rebuts.

D'aquesta forma, l'esquelet que caldrà implementar per utilitzar aquest protocol serà el següent:

```
class CNResponder extends ContractNetResponder {

      public CNResponder (Agent myAgent, MessageTemplate mt) {
            Super (myAgent, mt);
      }

      protected ACLMessage prepareResponse (ACLMessage msg) {...}

      protected ACLMessage prepareResultNotification (ACLMessage cfp,
      ACLMessage propose, ACLMessage accept) {...}
      protected void handleRejectProposal (ACLMessage cfp, ACLMessage
      propose, ACLMessage reject) {...}

}
```

A.2.3. FIPA-Subscribe

A més d'aquest protocols que ja suportava JADE, a partir de la versió 2.6, es disposa de la possibilitat d'utilitzar el protocol FIPA-Subscribe a través de les classes **`SubscriptionInitiator`** i **`SubscriptionResponder`**.

Tot i això, aquesta implementació encara es troba en una fase inicial (encara no es disposa de documentació detallada) i pot canviar en el futur, doncs la FIPA encara està discutint la seva especificació.

Si es necessiten més detalls sobre les classes es pot consultar la secció de l'API corresponent .

A.3. Ontologies

Des de Jade 2.5, es disposa d'una nova forma de definir ontologies [9] de manera més eficient. Les classes corresponents estan disponibles al paquet **`jade.content`**. Per evitar problemes, es molt recomanable fer servir aquesta sense barrejar-la amb cap tipus de component que estava present a l'antiga (paquet **`jade.onto`**) a la que encara es fa referència des de els exemples i la documentació de Jade.

L'inconvenient més important, és que la migració de la versió vella a la nova requereix rescriure quasi completament la definició de l'ontologia, així com alguns mètodes dins de la pròpia implementació de les classes i la forma en què es creen els missatges que l'utilitzen.

Els elements que formen la ontologia són: el fitxer de definició, les classes d'implementació dels conceptes, predicats i accions i, finalment, la instaciació des de programa i generació de missatges que l'utilitzin.

A.3.1. Generació del fitxer d'especificació

- Importar els paquets corresponents:

```
import jade.content.onto.*;
import jade.content.schema.*;
```

- Inicialitzar la ontologia definint el nom corresponent

```
private static Ontology theInstance = new NOMOntology();
public static Ontology instance()
{
      return theInstance;
}
private NOMOntology()
{
super (NAME, BasicOntology.getInstance());
...
}
```

- Especificar els components de l'ontologia (conceptes, predicats i accions) utilitzant els esquemes proporcionats pel paquet. Per cadascun d'ells, caldrà definir la classes que l'implementa:

```
//PREDICATES
add(new PredicateSchema(NAME), Name.class);

//ACTIONS
add(new AgentActionSchema(NAME), Name.class);

//CONCEPTS
add(new ConceptSchema(NAME), Name.class);
```

- A continuació, caldrà definir els *slots* que composaran cada element de l'ontologia i els seus atributs (tipus, opcionalitat, cardinalitat...):

```
//CONCEPTS
ConceptSchema cs = (ConceptSchema) getSchema (NAME);
cs.add(SLOT_NAME, type, cardinality, optionality);

//PREDICATES
PredicateSchema ps = (PredicateSchema) getSchema (NAME);
cs.add(SLOT_NAME, type, cardinality, optionality);

//ACTIONS
AgentActionSchema cs = (AgentActionSchema) getSchema (NAME);
cs.add(SLOT_NAME, type, cardinality, optionality);
```

El primer paràmetre de la funció **add** és un *string* que indica el nom de l'*slot* (que ha de ser el mateix que la classe que l'implementa).

El segon paràmetre és el tipus d'*slot* que pot ser un dels següents:

- **`BasicOntology.STRING`**: mapejat en un **`java.lang.String`**.
- **`BasicOntology.INTEGER`**: mapejat en un **`int`**, **`long`**, **`java.lang.Integer`** o **`java.lang.Long`**.
- **`BasicOntology.BOOLEAN`**: mapejat en un **`java.lang.Boolean`**.
- **`BasicOntology.FLOAT`**: mapejat en un **`java.lang.Float`** o **`java.lang.Double`**.
- **`BasicOntology.DATE`**: mapejat en un **`java.util.Date`**.
- **`BasicOntology.BYTE_SEQUENCE`**: mapejat en un *array* de **`byte`**.
- **`BasicOntology.AID`**: mapejat en un **`jade.core.aid`**.

Per exemple, podem definir un *slot* de tipus **String** així:

```
cs.add(SLOT, (PrimitiveSchema) getSchema(BasicOntology.STRING));
```

També podem definir que el tipus d'un *slot* sigui un concepte ja definit (NAME):

```
cs.add(SLOT, (ConceptSchema) getSchema(NAME));
```

El tercer paràmetre és opcional i ens permet definir la cardinalitat d'un *slot* indicant el nombre mínim i màxim d'elements, per exemple: **1,n** o **0,n** o **1, `ObjectSchema.UNLIMITED`...**

```
cs.add(SLOT, (PrimitiveSchema) getSchema(BasicOntoloyg.INTEGER),
1, ObjectSchema.UNLIMITED);
```

L'últim paràmetre també es opcional i només pot ser definit si no s'ha indicat la cardinalitat de l'*slot*. Indica si un *slot* pot contenir valors NULL (**OPTIONAL**) o no (**MANDATORY**). L'etiqueta **MANDATORY** és la que s'utilitza per defecte.

```
cs.add(SLOT, (PrimitiveSchema) getSchema(BasicOntology.BOOLEAN),
ObjectSchema.OPTIONAL);
```

A.3.2. Implementació de les classes

El següent pas serà implementar les classes que havíem associat a cada element de l'ontologia (concepte, predicat o acció):

- La definició de la classes ha d'implementar l'interficie corresponent, per exemple, si hem definit un element utilitzant un **ConceptSchema**, llavor la classe ha d'implementar l'interfície **Concept**.

```
public class CONCEPT_NAME implements Concept { . . . }
public class PREDICATE_NAME implements Predicate { . . . }
public class ACTION_NAME implements AgentAction { . . . }
```

- Per a cada *slot* de la classe, hem de definir el tipus adequat d'element (veure les equivalències anteriors) i un parell de mètodes d'accés per llegir i escriure. Per exemple, un *slot* definit com un *Integer* i anomenat **price**, tindrà els següents mètodes:

```
private int price;

public int getPrice()
{
return price;
}

public void setPrice (int price)
{
this.price=price;
}
```

Si l'*slot* ha estat definit amb una cardinalitat més gran que 1, llavors haurem de definir una **`jade.util.leap.List`** d'elements tal com s'indica a continuació:

```
private jade.util.leap.List alias;

public List getAlias()
{
      return alias;
}

public void setAlias (List alias)
{
      this.alias=alias;
}
```

A.3.3. Utilització de l'ontologia

Per tal d'utilitzar la ontologia definida, cal que la registrem al sistema. També serà necessari especificar el llenguatge que serà utilitzat per codificar/descodificar el contingut dels missatges. Jade ofereix el llenguatge SL que es basa en una codificació del contingut mitjançant *Strings* i es fàcilment intel·ligible.

La ontologia i el llenguatge estan continguts al paquet **`jade.content`** i els mètodes de registre estan disponibles mitjançant la classe **ContentManager**.

```
import jade.content.onto.*;
import jade.content.lang.*;
import jade.content.lang.sl.*;


public class myAgent extends Agent {
      private Codec codec = new SLCodec();
      private Ontology ontology = myOntology.intance() ;
      . . .

      protected void setup()
      {
            getContentManager().registerLanguage(codec,
codec.getName());
            getContentManager().registerOntology(ontology,
ontology.NAME);
            . . .
      }
      . . .
}
```

Per crear un missatge només caldrà anar fer servint els mètodes d'escriptura definits per als objectes de l'ontologia:

```
Nom_Concepte cp = new Nom_Concepte();
cp.setAtribut1("Valor_atribut");

Nom_Accio ac = new Nom_Accio();
Ac.setActor(getAID());
Ac.setItem(cp);
```

Per tal de escriure i llegir missatges creats amb la ontologia, cal utilitzar els mètodes **fillContent** i **extractContent** continguts a la classe **ContentManager** que treballen amb **ContentElement** (Accions o predicats):

```
getContentManager().fillContent
(ACLMessage msg, ContentElement Accio_o_predicat);

ContentElement Accio_o_predicat =
getContentManager().extractContent(ACLMessage msg);
```

A.3.4. Altres característiques

A més de les definicions bàsiques d'objectes en l'ontologia, la nova especificació proporciona un conjunt de possibilitats que la fan més versàtil:

- Es pot definir herència entre els diferents conceptes de forma que es comparteixin i s'agrupin els atributs i les seves característiques.

- Es poden combinar diverses ontologies ja definides.

- Es poden fer servir descriptors abstractes. Són útils quan volem definir *slots* però no volem especificar un tipus específic.

- Es poden definir restriccions als valors que pot prendre un *slot* (ex: un *slot* d'enters només pot ser positiu).

- Es poden utilitzar altres llenguatges per a la codificació dels missatges com XML, RDF, AML...

Per obtenir una descripció més extesa així com exemples complets de com definir una ontologia, es molt convenient consultar el tutorial sobre ontologies inclòs en la documentació de Jade.

B. Especificació de l'Ontologia en RDF

Per tal de proporcionar la definició de l'Ontologia en un format estàndard de forma que qualsevol persona la pugui incorporar al seu projecte, l' hem codificat emprant la notació RDF (basada en el llenguatge de marques XML).

La especificació completa en RDFS (que també es troba disponible a través de la nostra pàgina web [31]) és la següent:

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<!DOCTYPE rdf:RDF [
    <!ENTITY rdf 'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
    <!ENTITY patient 'http://protege.stanford.edu/patient#'>
    <!ENTITY a 'http://protege.stanford.edu/system#'>
    <!ENTITY rdfs 'http://www.w3.org/TR/1999/PR-rdf-schema-19990303#'>
]>
<rdf:RDF xmlns:patient="&patient;"
    xmlns:rdf="&rdf;"
    xmlns:a="&a;"
    xmlns:rdfs="&rdfs;">
<rdfs:Class rdf:about="&patient;AID"
    rdfs:label="AID">
   <rdfs:subClassOf rdf:resource="&patient;Concept"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Action"
    rdfs:label="Action">
   <rdfs:subClassOf rdf:resource="&patient;AgentAction"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;AgentAction"
    a:role="abstract"
    rdfs:label="AgentAction">
   <rdfs:subClassOf rdf:resource="&patient;Concept"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;All"
    rdfs:label="All">
   <rdfs:subClassOf rdf:resource="&patient;Predicate"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Booking-medical-center"
    rdfs:label="Booking-medical-center">
   <rdfs:subClassOf rdf:resource="&patient;AgentAction"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Concept"
    a:role="abstract"
    rdfs:label="Concept">
   <rdfs:subClassOf rdf:resource="&rdfs;Resource"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Department"
    rdfs:label="Department">
   <rdfs:subClassOf rdf:resource="&patient;Healthcare-party"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Episode"
    rdfs:label="Episode">
   <rdfs:subClassOf rdf:resource="&patient;Concept"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Equals"
    rdfs:label="Equals">
   <rdfs:subClassOf rdf:resource="&patient;Predicate"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Error-reason"
    rdfs:label="Error-reason">
   <rdfs:subClassOf rdf:resource="&patient;Concept"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Free-time"
    rdfs:label="Free-time">
   <rdfs:subClassOf rdf:resource="&patient;AgentAction"/>
</rdfs:Class>
```

```
<rdfs:Class rdf:about="&patient;Health-domain"
    rdfs:label="Health-domain">
  <rdfs:subClassOf rdf:resource="&patient;Concept"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Healthcare-device"
    rdfs:label="Healthcare-device">
  <rdfs:subClassOf rdf:resource="&patient;Health-domain"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Healthcare-organization"
    rdfs:label="Healthcare-organization">
  <rdfs:subClassOf rdf:resource="&patient;Healthcare-party"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Healthcare-party"
    rdfs:label="Healthcare-party">
  <rdfs:subClassOf rdf:resource="&patient;Health-domain"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Healthcare-person"
    rdfs:label="Healthcare-person">
  <rdfs:subClassOf rdf:resource="&patient;Healthcare-party"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Healthcare-software"
    rdfs:label="Healthcare-software">
  <rdfs:subClassOf rdf:resource="&patient;Health-domain"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Iota"
    rdfs:label="Iota">
  <rdfs:subClassOf rdf:resource="&patient;Predicate"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Is-department-description"
    rdfs:label="Is-department-description">
  <rdfs:subClassOf rdf:resource="&patient;Predicate"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Is-medical-center-description"
    rdfs:label="Is-medical-center-description">
  <rdfs:subClassOf rdf:resource="&patient;Predicate"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Is-patient-description"
    rdfs:label="Is-patient-description">
  <rdfs:subClassOf rdf:resource="&patient;Predicate"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Key-RSA"
    rdfs:label="Key-RSA">
  <rdfs:subClassOf rdf:resource="&patient;AgentAction"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Patient"
    rdfs:label="Patient">
  <rdfs:subClassOf rdf:resource="&patient;Health-domain"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Person-name"
    rdfs:label="Person-name">
  <rdfs:subClassOf rdf:resource="&patient;Concept"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Postal-address"
    rdfs:label="Postal-address">
  <rdfs:subClassOf rdf:resource="&patient;Concept"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Predicate"
    a:role="abstract"
    rdfs:label="Predicate">
  <rdfs:subClassOf rdf:resource="&rdfs;Resource"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Problem-description"
    rdfs:label="Problem-description">
  <rdfs:subClassOf rdf:resource="&patient;Concept"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Quantity"
    rdfs:label="Quantity">
  <rdfs:subClassOf rdf:resource="&patient;Concept"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Register-agent"
    rdfs:label="Register-agent">
  <rdfs:subClassOf rdf:resource="&patient;AgentAction"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Room"
    rdfs:label="Room">
```

```
  <rdfs:subClassOf rdf:resource="&patient;Concept"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Set-medical-record"
    rdfs:label="Set-medical-record">
  <rdfs:subClassOf rdf:resource="&patient;AgentAction"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Telecommunication"
    rdfs:label="Telecommunication">
  <rdfs:subClassOf rdf:resource="&patient;Concept"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Treatment"
    rdfs:label="Treatment">
  <rdfs:subClassOf rdf:resource="&patient;Concept"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Visit-booking"
    rdfs:label="Visit-booking">
  <rdfs:subClassOf rdf:resource="&patient;Concept"/>
</rdfs:Class>
<rdfs:Class rdf:about="&patient;Working-hours"
    rdfs:label="Working-hours">
  <rdfs:subClassOf rdf:resource="&patient;Concept"/>
</rdfs:Class>
<rdf:Property rdf:about="&patient;action-left"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="action-left">
  <rdfs:domain rdf:resource="&patient;Action"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;action-right"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="action-right">
  <rdfs:domain rdf:resource="&patient;Action"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;address"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="cls"
    rdfs:label="address">
  <rdfs:domain rdf:resource="&patient;Healthcare-party"/>
  <a:allowedParents rdf:resource="&patient;Postal-address"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;addresses"
    rdfs:label="addresses">
  <rdfs:domain rdf:resource="&patient;AID"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;aid"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="cls"
    rdfs:label="aid">
  <a:allowedParents rdf:resource="&patient;AID"/>
  <rdfs:domain rdf:resource="&patient;Health-domain"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;all-left"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="all-left">
  <rdfs:domain rdf:resource="&patient;All"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;all-right"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="all-right">
  <rdfs:domain rdf:resource="&patient;All"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;allergies"
    rdfs:label="allergies">
```

```
  <rdfs:domain rdf:resource="&patient;Patient"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;antecedents"
   rdfs:label="antecedents">
  <rdfs:domain rdf:resource="&patient;Patient"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;begin"
   a:maxCardinality="1"
   a:minCardinality="1"
   rdfs:label="begin">
  <rdfs:domain rdf:resource="&patient;Working-hours"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;city"
   a:maxCardinality="1"
   a:minCardinality="1"
   rdfs:label="city">
  <rdfs:domain rdf:resource="&patient;Postal-address"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;country-code"
   a:maxCardinality="1"
   rdfs:label="country-code">
  <rdfs:domain rdf:resource="&patient;Postal-address"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;date-of-birth"
   a:maxCardinality="1"
   a:minCardinality="1"
   rdfs:label="date-of-birth">
  <rdfs:domain rdf:resource="&patient;Patient"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;date-visit"
   a:maxCardinality="1"
   a:minCardinality="1"
   rdfs:label="date-visit">
  <rdfs:domain rdf:resource="&patient;Episode"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;day"
   a:maxCardinality="1"
   a:minCardinality="1"
   a:range="integer"
   rdfs:label="day">
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;day-week"
   a:maxCardinality="1"
   a:minCardinality="1"
   rdfs:label="day-week">
  <rdfs:domain rdf:resource="&patient;Working-hours"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;department-description"
   a:maxCardinality="1"
   a:minCardinality="1"
   a:range="cls"
   rdfs:label="department-description">
  <a:allowedParents rdf:resource="&patient;Department"/>
  <rdfs:domain rdf:resource="&patient;Is-department-description"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;department-name"
   a:maxCardinality="1"
   a:minCardinality="1"
   rdfs:label="department-name">
  <rdfs:domain rdf:resource="&patient;Department"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;departments"
   a:range="cls"
   rdfs:label="departments">
```

```
  <a:allowedParents rdf:resource="&patient;Department"/>
  <rdfs:domain rdf:resource="&patient;Healthcare-organization"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;description"
   a:maxCardinality="1"
   rdfs:label="description">
  <rdfs:domain rdf:resource="&patient;Problem-description"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;device-manufacturer"
   a:maxCardinality="1"
   rdfs:label="device-manufacturer">
  <rdfs:domain rdf:resource="&patient;Healthcare-device"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;device-model"
   a:maxCardinality="1"
   rdfs:label="device-model">
  <rdfs:domain rdf:resource="&patient;Healthcare-device"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;device-type"
   a:maxCardinality="1"
   a:minCardinality="1"
   rdfs:label="device-type">
  <rdfs:domain rdf:resource="&patient;Healthcare-device"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;device-version"
   a:maxCardinality="1"
   rdfs:label="device-version">
  <rdfs:domain rdf:resource="&patient;Healthcare-device"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;disability"
   rdfs:label="disability">
  <rdfs:domain rdf:resource="&patient;Patient"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;dosage-pattern"
   a:maxCardinality="1"
   a:minCardinality="1"
   rdfs:label="dosage-pattern">
  <rdfs:domain rdf:resource="&patient;Treatment"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;dose"
   a:maxCardinality="1"
   a:minCardinality="1"
   a:range="cls"
   rdfs:label="dose">
  <a:allowedParents rdf:resource="&patient;Quantity"/>
  <rdfs:domain rdf:resource="&patient;Treatment"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;drug-name"
   a:maxCardinality="1"
   a:minCardinality="1"
   rdfs:label="drug-name">
  <rdfs:domain rdf:resource="&patient;Treatment"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;drug-type"
   a:maxCardinality="1"
   a:minCardinality="1"
   rdfs:label="drug-type">
  <rdfs:domain rdf:resource="&patient;Treatment"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;duration"
   a:maxCardinality="1"
   a:minCardinality="1"
   a:range="cls"
   rdfs:label="duration">
```

```
  <a:allowedParents rdf:resource="&patient;Quantity"/>
  <rdfs:domain rdf:resource="&patient;Treatment"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;e-mail"
   a:maxCardinality="1"
   rdfs:label="e-mail">
  <rdfs:domain rdf:resource="&patient;Telecommunication"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;end"
   a:maxCardinality="1"
   rdfs:label="end">
  <rdfs:domain rdf:resource="&patient;Working-hours"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;episode-id-department"
   a:maxCardinality="1"
   a:minCardinality="1"
   rdfs:label="episode-id-department">
  <rdfs:domain rdf:resource="&patient;Episode"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;episode-id-doctor"
   a:maxCardinality="1"
   a:minCardinality="1"
   rdfs:label="episode-id-doctor">
  <rdfs:domain rdf:resource="&patient;Episode"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;episode-id-medical-center"
   a:maxCardinality="1"
   a:minCardinality="1"
   rdfs:label="episode-id-medical-center">
  <rdfs:domain rdf:resource="&patient;Episode"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;episode-observations"
   a:maxCardinality="1"
   rdfs:label="episode-observations">
  <rdfs:domain rdf:resource="&patient;Episode"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;episode-problem"
   a:maxCardinality="1"
   a:range="cls"
   rdfs:label="episode-problem">
  <rdfs:domain rdf:resource="&patient;Episode"/>
  <a:allowedParents rdf:resource="&patient;Problem-description"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;equals-left"
   a:maxCardinality="1"
   a:minCardinality="1"
   rdfs:label="equals-left">
  <rdfs:domain rdf:resource="&patient;Equals"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;equals-right"
   rdfs:label="equals-right">
  <rdfs:domain rdf:resource="&patient;Equals"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;fax"
   a:maxCardinality="1"
   rdfs:label="fax">
  <rdfs:domain rdf:resource="&patient;Telecommunication"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;filename"
   a:maxCardinality="1"
   rdfs:label="filename">
  <rdfs:domain rdf:resource="&patient;Healthcare-software"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
```

```
<rdf:Property rdf:about="&patient;first-name"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="first-name">
   <rdfs:domain rdf:resource="&patient;Person-name"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;first-surname"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="first-surname">
   <rdfs:domain rdf:resource="&patient;Person-name"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;free-agent"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="cls"
    rdfs:label="free-agent">
   <a:allowedParents rdf:resource="&patient;AID"/>
   <rdfs:domain rdf:resource="&patient;Free-time"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;free-working-hours"
    a:maxCardinality="1"
    a:range="cls"
    rdfs:label="free-working-hours">
   <rdfs:domain rdf:resource="&patient;Free-time"/>
   <a:allowedParents rdf:resource="&patient;Working-hours"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;history"
    a:range="cls"
    rdfs:label="history">
   <a:allowedParents rdf:resource="&patient;Episode"/>
   <rdfs:domain rdf:resource="&patient;Patient"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;hour"
    a:maxCardinality="1"
    a:range="integer"
    rdfs:label="hour">
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;hour-visit"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="hour-visit">
   <rdfs:domain rdf:resource="&patient;Episode"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;id-card"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="id-card">
   <rdfs:domain rdf:resource="&patient;Patient"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;id-medical"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="id-medical">
   <rdfs:domain rdf:resource="&patient;Patient"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;id-party"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="id-party">
   <rdfs:domain rdf:resource="&patient;Healthcare-party"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;id-room"
    a:maxCardinality="1"
    a:minCardinality="1"
```

```
    rdfs:label="id-room">
   <rdfs:domain rdf:resource="&patient;Room"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;internal-name"
    a:maxCardinality="1"
    rdfs:label="internal-name">
   <rdfs:domain rdf:resource="&patient;Healthcare-software"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;iota-left"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="iota-left">
   <rdfs:domain rdf:resource="&patient;Iota"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;iota-right"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="iota-right">
   <rdfs:domain rdf:resource="&patient;Iota"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;job"
    a:maxCardinality="1"
    rdfs:label="job">
   <rdfs:domain rdf:resource="&patient;Patient"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;key"
    a:maxCardinality="1"
    rdfs:label="key">
   <rdfs:domain rdf:resource="&patient;Key-RSA"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;key-agent"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="cls"
    rdfs:label="key-agent">
   <a:allowedParents rdf:resource="&patient;AID"/>
   <rdfs:domain rdf:resource="&patient;Key-RSA"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;language"
    rdfs:label="language">
   <rdfs:domain rdf:resource="&patient;Healthcare-party"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;medical-attributes"
    a:minCardinality="1"
    rdfs:label="medical-attributes">
   <rdfs:domain rdf:resource="&patient;Is-medical-center-description"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;medical-conditions"
    a:minCardinality="1"
    rdfs:label="medical-conditions">
   <rdfs:domain rdf:resource="&patient;Is-medical-center-description"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;medical-data"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="medical-data">
   <rdfs:domain rdf:resource="&patient;Set-medical-record"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;medical-var"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="medical-var">
   <rdfs:domain rdf:resource="&patient;Is-medical-center-description"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
```

```
</rdf:Property>
<rdf:Property rdf:about="&patient;minute"
    a:maxCardinality="1"
    a:range="integer"
    rdfs:label="minute">
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;month"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="integer"
    rdfs:label="month">
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;name"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="name">
   <rdfs:domain rdf:resource="&patient;AID"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;observations"
    a:maxCardinality="1"
    rdfs:label="observations">
   <rdfs:domain rdf:resource="&patient;Healthcare-party"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;organization-name"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="organization-name">
   <rdfs:domain rdf:resource="&patient;Healthcare-organization"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;organization-type"
    a:maxCardinality="1"
    rdfs:label="organization-type">
   <rdfs:domain rdf:resource="&patient;Healthcare-organization"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;other"
    a:maxCardinality="1"
    rdfs:label="other">
   <rdfs:domain rdf:resource="&patient;Telecommunication"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;patient-address"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="cls"
    rdfs:label="patient-address">
   <rdfs:domain rdf:resource="&patient;Patient"/>
   <a:allowedParents rdf:resource="&patient;Postal-address"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;patient-aid"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="cls"
    rdfs:label="patient-aid">
   <a:allowedParents rdf:resource="&patient;AID"/>
   <rdfs:domain rdf:resource="&patient;Visit-booking"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;patient-attributes"
    a:minCardinality="1"
    rdfs:label="patient-attributes">
   <rdfs:domain rdf:resource="&patient;Is-patient-description"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;patient-conditions"
    rdfs:label="patient-conditions">
   <rdfs:domain rdf:resource="&patient;Is-patient-description"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
```

```
<rdf:Property rdf:about="&patient;patient-id-medical"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="patient-id-medical">
   <rdfs:domain rdf:resource="&patient;Visit-booking"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;patient-name"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="cls"
    rdfs:label="patient-name">
   <rdfs:domain rdf:resource="&patient;Patient"/>
   <a:allowedParents rdf:resource="&patient;Person-name"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;patient-phone"
    a:maxCardinality="1"
    rdfs:label="patient-phone">
   <rdfs:domain rdf:resource="&patient;Patient"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;patient-var"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="patient-var">
   <rdfs:domain rdf:resource="&patient;Is-patient-description"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;person-name"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="cls"
    rdfs:label="person-name">
   <rdfs:domain rdf:resource="&patient;Healthcare-person"/>
   <a:allowedParents rdf:resource="&patient;Person-name"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;person-type"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="person-type">
   <rdfs:domain rdf:resource="&patient;Healthcare-person"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;phone"
    a:maxCardinality="1"
    rdfs:label="phone">
   <rdfs:domain rdf:resource="&patient;Telecommunication"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;position"
    a:maxCardinality="1"
    rdfs:label="position">
   <rdfs:domain rdf:resource="&patient;Healthcare-person"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;postal-address"
    rdfs:label="postal-address">
   <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;problem-treatment"
    a:maxCardinality="1"
    a:range="cls"
    rdfs:label="problem-treatment">
   <rdfs:domain rdf:resource="&patient;Problem-description"/>
   <a:allowedParents rdf:resource="&patient;Treatment"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;problem-type"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="problem-type">
   <rdfs:domain rdf:resource="&patient;Problem-description"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
```

```
</rdf:Property>
<rdf:Property rdf:about="&patient;qualification"
    a:maxCardinality="1"
    rdfs:label="qualification">
   <rdfs:domain rdf:resource="&patient;Healthcare-person"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;reason"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="reason">
   <rdfs:domain rdf:resource="&patient;Error-reason"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;reg-agent"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="cls"
    rdfs:label="reg-agent">
   <a:allowedParents rdf:resource="&patient;AID"/>
   <rdfs:domain rdf:resource="&patient;Register-agent"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;reg-info"
    a:maxCardinality="1"
    rdfs:label="reg-info">
   <rdfs:domain rdf:resource="&patient;Register-agent"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;resolvers"
    a:range="cls"
    rdfs:label="resolvers">
   <a:allowedParents rdf:resource="&patient;AID"/>
   <rdfs:domain rdf:resource="&patient;AID"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;results"
    a:maxCardinality="1"
    rdfs:label="results">
   <rdfs:domain rdf:resource="&patient;Problem-description"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;room-type"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="room-type">
   <rdfs:domain rdf:resource="&patient;Room"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;rooms"
    rdfs:label="rooms">
   <rdfs:domain rdf:resource="&patient;Department"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;route-administration"
    a:maxCardinality="1"
    rdfs:label="route-administration">
   <rdfs:domain rdf:resource="&patient;Treatment"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;second-surname"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="second-surname">
   <rdfs:domain rdf:resource="&patient;Person-name"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;serial-number"
    a:maxCardinality="1"
    rdfs:label="serial-number">
   <rdfs:domain rdf:resource="&patient;Healthcare-device"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;sex"
    a:maxCardinality="1"
```

```
    a:minCardinality="1"
    rdfs:label="sex">
   <rdfs:domain rdf:resource="&patient;Patient"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;software-date"
    a:maxCardinality="1"
    rdfs:label="software-date">
   <rdfs:domain rdf:resource="&patient;Healthcare-software"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;software-manufacturer"
    a:maxCardinality="1"
    rdfs:label="software-manufacturer">
   <rdfs:domain rdf:resource="&patient;Healthcare-software"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;software-name"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="software-name">
   <rdfs:domain rdf:resource="&patient;Healthcare-software"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;software-version"
    a:maxCardinality="1"
    rdfs:label="software-version">
   <rdfs:domain rdf:resource="&patient;Healthcare-software"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;specialty"
    a:minCardinality="1"
    rdfs:label="specialty">
   <rdfs:domain rdf:resource="&patient;Healthcare-party"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;staff"
    a:minCardinality="1"
    a:range="cls"
    rdfs:label="staff">
   <rdfs:domain rdf:resource="&patient;Department"/>
   <a:allowedParents rdf:resource="&patient;Healthcare-person"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;street"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="street">
   <rdfs:domain rdf:resource="&patient;Postal-address"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;street-number"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="street-number">
   <rdfs:domain rdf:resource="&patient;Postal-address"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;telecomm"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="cls"
    rdfs:label="telecomm">
   <rdfs:domain rdf:resource="&patient;Healthcare-party"/>
   <a:allowedParents rdf:resource="&patient;Telecommunication"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;tests"
    rdfs:label="tests">
   <rdfs:domain rdf:resource="&patient;Problem-description"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;timetable"
    a:minCardinality="1"
    a:range="cls"
```

```
    rdfs:label="timetable">
   <rdfs:domain rdf:resource="&patient;Healthcare-party"/>
   <a:allowedParents rdf:resource="&patient;Working-hours"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;title"
    a:maxCardinality="1"
    rdfs:label="title">
   <rdfs:domain rdf:resource="&patient;Person-name"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;unit"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="unit">
   <rdfs:domain rdf:resource="&patient;Quantity"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;urgency"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="integer"
    rdfs:label="urgency">
   <rdfs:domain rdf:resource="&patient;Problem-description"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;value"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="integer"
    rdfs:label="value">
   <rdfs:domain rdf:resource="&patient;Quantity"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;visit"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="cls"
    rdfs:label="visit">
   <rdfs:domain rdf:resource="&patient;Booking-medical-center"/>
   <a:allowedParents rdf:resource="&patient;Visit-booking"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;visit-date"
    a:maxCardinality="1"
    rdfs:label="visit-date">
   <rdfs:domain rdf:resource="&patient;Visit-booking"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;visit-doctor"
    a:maxCardinality="1"
    a:range="cls"
    rdfs:label="visit-doctor">
   <a:allowedParents rdf:resource="&patient;Healthcare-person"/>
   <rdfs:domain rdf:resource="&patient;Visit-booking"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;visit-hour"
    a:maxCardinality="1"
    rdfs:label="visit-hour">
   <rdfs:domain rdf:resource="&patient;Visit-booking"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;visit-place"
    a:maxCardinality="1"
    a:range="cls"
    rdfs:label="visit-place">
   <a:allowedParents rdf:resource="&patient;Healthcare-organization"/>
   <rdfs:domain rdf:resource="&patient;Visit-booking"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;visit-problem"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="cls"
```

```
    rdfs:label="visit-problem">
   <a:allowedParents rdf:resource="&patient;Problem-description"/>
   <rdfs:domain rdf:resource="&patient;Visit-booking"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;web"
    a:maxCardinality="1"
    rdfs:label="web">
   <rdfs:domain rdf:resource="&patient;Telecommunication"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;year"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="integer"
    rdfs:label="year">
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&patient;zip-code"
    a:maxCardinality="1"
    a:minCardinality="1"
    rdfs:label="zip-code">
   <rdfs:domain rdf:resource="&patient;Postal-address"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<a:OverridingProperty rdf:about="&a;AID_name"
    a:maxCardinality="1"
    a:minCardinality="1">
   <a:domain rdf:resource="&patient;AID"/>
   <a:overriddenProperty rdf:resource="&patient;name"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;AID_resolvers"
    a:range="cls">
   <a:allowedParents rdf:resource="&patient;AID"/>
   <a:domain rdf:resource="&patient;AID"/>
   <a:overriddenProperty rdf:resource="&patient;resolvers"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Booking-medical-center_visit"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="cls">
   <a:domain rdf:resource="&patient;Booking-medical-center"/>
   <a:allowedParents rdf:resource="&patient;Visit-booking"/>
   <a:overriddenProperty rdf:resource="&patient;visit"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Department_department-name"
    a:maxCardinality="1"
    a:minCardinality="1">
   <a:domain rdf:resource="&patient;Department"/>
   <a:overriddenProperty rdf:resource="&patient;department-name"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Department_staff"
    a:minCardinality="1"
    a:range="cls">
   <a:domain rdf:resource="&patient;Department"/>
   <a:allowedParents rdf:resource="&patient;Healthcare-person"/>
   <a:overriddenProperty rdf:resource="&patient;staff"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Episode_episode-id-department"
    a:maxCardinality="1"
    a:minCardinality="1">
   <a:domain rdf:resource="&patient;Episode"/>
   <a:overriddenProperty rdf:resource="&patient;episode-id-department"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Episode_episode-id-doctor"
    a:maxCardinality="1"
    a:minCardinality="1">
   <a:domain rdf:resource="&patient;Episode"/>
   <a:overriddenProperty rdf:resource="&patient;episode-id-doctor"/>
```

```
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Episode_episode-id-medical-center"
   a:maxCardinality="1"
   a:minCardinality="1">
  <a:domain rdf:resource="&patient;Episode"/>
  <a:overriddenProperty rdf:resource="&patient;episode-id-medical-center"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Episode_episode-observations"
   a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Episode"/>
  <a:overriddenProperty rdf:resource="&patient;episode-observations"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Episode_episode-problem"
   a:maxCardinality="1"
   a:range="cls">
  <a:domain rdf:resource="&patient;Episode"/>
  <a:allowedParents rdf:resource="&patient;Problem-description"/>
  <a:overriddenProperty rdf:resource="&patient;episode-problem"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Error-reason_reason"
   a:maxCardinality="1"
   a:minCardinality="1">
  <a:domain rdf:resource="&patient;Error-reason"/>
  <a:overriddenProperty rdf:resource="&patient;reason"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Free-time_free-agent"
   a:maxCardinality="1"
   a:minCardinality="1"
   a:range="cls">
  <a:allowedParents rdf:resource="&patient;AID"/>
  <a:domain rdf:resource="&patient;Free-time"/>
  <a:overriddenProperty rdf:resource="&patient;free-agent"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Free-time_free-working-hours"
   a:maxCardinality="1"
   a:range="cls">
  <a:domain rdf:resource="&patient;Free-time"/>
  <a:allowedParents rdf:resource="&patient;Working-hours"/>
  <a:overriddenProperty rdf:resource="&patient;free-working-hours"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Health-domain_aid"
   a:maxCardinality="1"
   a:minCardinality="1"
   a:range="cls">
  <a:allowedParents rdf:resource="&patient;AID"/>
  <a:domain rdf:resource="&patient;Health-domain"/>
  <a:overriddenProperty rdf:resource="&patient;aid"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-device_device-manufacturer"
   a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Healthcare-device"/>
  <a:overriddenProperty rdf:resource="&patient;device-manufacturer"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-device_device-model"
   a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Healthcare-device"/>
  <a:overriddenProperty rdf:resource="&patient;device-model"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-device_device-type"
   a:maxCardinality="1"
   a:minCardinality="1">
  <a:domain rdf:resource="&patient;Healthcare-device"/>
  <a:overriddenProperty rdf:resource="&patient;device-type"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
```

```
<a:OverridingProperty rdf:about="&a;Healthcare-device_device-version"
    a:maxCardinality="1">
   <a:domain rdf:resource="&patient;Healthcare-device"/>
   <a:overriddenProperty rdf:resource="&patient;device-version"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-device_serial-number"
    a:maxCardinality="1">
   <a:domain rdf:resource="&patient;Healthcare-device"/>
   <a:overriddenProperty rdf:resource="&patient;serial-number"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-organization_departments"
    a:range="cls">
   <a:allowedParents rdf:resource="&patient;Department"/>
   <a:domain rdf:resource="&patient;Healthcare-organization"/>
   <a:overriddenProperty rdf:resource="&patient;departments"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-organization_organization-name"
    a:maxCardinality="1"
    a:minCardinality="1">
   <a:domain rdf:resource="&patient;Healthcare-organization"/>
   <a:overriddenProperty rdf:resource="&patient;organization-name"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-organization_organization-type"
    a:maxCardinality="1">
   <a:domain rdf:resource="&patient;Healthcare-organization"/>
   <a:overriddenProperty rdf:resource="&patient;organization-type"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-party_address"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="cls">
   <a:domain rdf:resource="&patient;Healthcare-party"/>
   <a:allowedParents rdf:resource="&patient;Postal-address"/>
   <a:overriddenProperty rdf:resource="&patient;address"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-party_id-party"
    a:maxCardinality="1"
    a:minCardinality="1">
   <a:domain rdf:resource="&patient;Healthcare-party"/>
   <a:overriddenProperty rdf:resource="&patient;id-party"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-party_observations"
    a:maxCardinality="1">
   <a:domain rdf:resource="&patient;Healthcare-party"/>
   <a:overriddenProperty rdf:resource="&patient;observations"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-party_specialty"
    a:minCardinality="1">
   <a:domain rdf:resource="&patient;Healthcare-party"/>
   <a:overriddenProperty rdf:resource="&patient;specialty"/>
   <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-party_telecomm"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="cls">
   <a:domain rdf:resource="&patient;Healthcare-party"/>
   <a:allowedParents rdf:resource="&patient;Telecommunication"/>
   <a:overriddenProperty rdf:resource="&patient;telecomm"/>
   <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-party_timetable"
    a:minCardinality="1"
    a:range="cls">
   <a:domain rdf:resource="&patient;Healthcare-party"/>
   <a:allowedParents rdf:resource="&patient;Working-hours"/>
   <a:overriddenProperty rdf:resource="&patient;timetable"/>
```

```
  <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-person_person-name"
   a:maxCardinality="1"
   a:minCardinality="1"
   a:range="cls">
  <a:domain rdf:resource="&patient;Healthcare-person"/>
  <a:allowedParents rdf:resource="&patient;Person-name"/>
  <a:overriddenProperty rdf:resource="&patient;person-name"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-person_person-type"
   a:maxCardinality="1"
   a:minCardinality="1">
  <a:domain rdf:resource="&patient;Healthcare-person"/>
  <a:overriddenProperty rdf:resource="&patient;person-type"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-person_position"
   a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Healthcare-person"/>
  <a:overriddenProperty rdf:resource="&patient;position"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-person_qualification"
   a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Healthcare-person"/>
  <a:overriddenProperty rdf:resource="&patient;qualification"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-software_filename"
   a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Healthcare-software"/>
  <a:overriddenProperty rdf:resource="&patient;filename"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-software_internal-name"
   a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Healthcare-software"/>
  <a:overriddenProperty rdf:resource="&patient;internal-name"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-software_software-manufacturer"
   a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Healthcare-software"/>
  <a:overriddenProperty rdf:resource="&patient;software-manufacturer"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-software_software-name"
   a:maxCardinality="1"
   a:minCardinality="1">
  <a:domain rdf:resource="&patient;Healthcare-software"/>
  <a:overriddenProperty rdf:resource="&patient;software-name"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Healthcare-software_software-version"
   a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Healthcare-software"/>
  <a:overriddenProperty rdf:resource="&patient;software-version"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Is-department-description_department-description"
   a:maxCardinality="1"
   a:minCardinality="1"
   a:range="cls">
  <a:allowedParents rdf:resource="&patient;Department"/>
  <a:domain rdf:resource="&patient;Is-department-description"/>
  <a:overriddenProperty rdf:resource="&patient;department-description"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Is-medical-center-description_medical-var"
   a:maxCardinality="1"
   a:minCardinality="1">
  <a:domain rdf:resource="&patient;Is-medical-center-description"/>
  <a:overriddenProperty rdf:resource="&patient;medical-var"/>
```

```
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Is-patient-description_patient-var"
   a:maxCardinality="1"
   a:minCardinality="1">
  <a:domain rdf:resource="&patient;Is-patient-description"/>
  <a:overriddenProperty rdf:resource="&patient;patient-var"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Key-RSA_key"
   a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Key-RSA"/>
  <a:overriddenProperty rdf:resource="&patient;key"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Key-RSA_key-agent"
   a:maxCardinality="1"
   a:minCardinality="1"
   a:range="cls">
  <a:allowedParents rdf:resource="&patient;AID"/>
  <a:domain rdf:resource="&patient;Key-RSA"/>
  <a:overriddenProperty rdf:resource="&patient;key-agent"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Patient_history"
   a:range="cls">
  <a:allowedParents rdf:resource="&patient;Episode"/>
  <a:domain rdf:resource="&patient;Patient"/>
  <a:overriddenProperty rdf:resource="&patient;history"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Patient_id-card"
   a:maxCardinality="1"
   a:minCardinality="1">
  <a:domain rdf:resource="&patient;Patient"/>
  <a:overriddenProperty rdf:resource="&patient;id-card"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Patient_id-medical"
   a:maxCardinality="1"
   a:minCardinality="1">
  <a:domain rdf:resource="&patient;Patient"/>
  <a:overriddenProperty rdf:resource="&patient;id-medical"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Patient_job"
   a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Patient"/>
  <a:overriddenProperty rdf:resource="&patient;job"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Patient_patient-address"
   a:maxCardinality="1"
   a:minCardinality="1"
   a:range="cls">
  <a:domain rdf:resource="&patient;Patient"/>
  <a:allowedParents rdf:resource="&patient;Postal-address"/>
  <a:overriddenProperty rdf:resource="&patient;patient-address"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Patient_patient-name"
   a:maxCardinality="1"
   a:minCardinality="1"
   a:range="cls">
  <a:domain rdf:resource="&patient;Patient"/>
  <a:allowedParents rdf:resource="&patient;Person-name"/>
  <a:overriddenProperty rdf:resource="&patient;patient-name"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Patient_patient-phone"
   a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Patient"/>
  <a:overriddenProperty rdf:resource="&patient;patient-phone"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
```

```
<a:OverridingProperty rdf:about="&a;Person-name_first-name"
    a:maxCardinality="1"
    a:minCardinality="1">
  <a:domain rdf:resource="&patient;Person-name"/>
  <a:overriddenProperty rdf:resource="&patient;first-name"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Person-name_first-surname"
    a:maxCardinality="1"
    a:minCardinality="1">
  <a:domain rdf:resource="&patient;Person-name"/>
  <a:overriddenProperty rdf:resource="&patient;first-surname"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Person-name_second-surname"
    a:maxCardinality="1"
    a:minCardinality="1">
  <a:domain rdf:resource="&patient;Person-name"/>
  <a:overriddenProperty rdf:resource="&patient;second-surname"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Person-name_title"
    a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Person-name"/>
  <a:overriddenProperty rdf:resource="&patient;title"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Postal-address_city"
    a:maxCardinality="1"
    a:minCardinality="1">
  <a:domain rdf:resource="&patient;Postal-address"/>
  <a:overriddenProperty rdf:resource="&patient;city"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Postal-address_country-code"
    a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Postal-address"/>
  <a:overriddenProperty rdf:resource="&patient;country-code"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Postal-address_street"
    a:maxCardinality="1"
    a:minCardinality="1">
  <a:domain rdf:resource="&patient;Postal-address"/>
  <a:overriddenProperty rdf:resource="&patient;street"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Postal-address_street-number"
    a:maxCardinality="1"
    a:minCardinality="1">
  <a:domain rdf:resource="&patient;Postal-address"/>
  <a:overriddenProperty rdf:resource="&patient;street-number"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Postal-address_zip-code"
    a:maxCardinality="1"
    a:minCardinality="1">
  <a:domain rdf:resource="&patient;Postal-address"/>
  <a:overriddenProperty rdf:resource="&patient;zip-code"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Problem-description_description"
    a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Problem-description"/>
  <a:overriddenProperty rdf:resource="&patient;description"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Problem-description_problem-treatment"
    a:maxCardinality="1"
    a:range="cls">
  <a:domain rdf:resource="&patient;Problem-description"/>
  <a:allowedParents rdf:resource="&patient;Treatment"/>
  <a:overriddenProperty rdf:resource="&patient;problem-treatment"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
```

```
<a:OverridingProperty rdf:about="&a;Problem-description_problem-type"
    a:maxCardinality="1"
    a:minCardinality="1">
  <a:domain rdf:resource="&patient;Problem-description"/>
  <a:overriddenProperty rdf:resource="&patient;problem-type"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Problem-description_results"
    a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Problem-description"/>
  <a:overriddenProperty rdf:resource="&patient;results"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Problem-description_urgency"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="integer">
  <a:domain rdf:resource="&patient;Problem-description"/>
  <a:overriddenProperty rdf:resource="&patient;urgency"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Quantity_unit"
    a:maxCardinality="1"
    a:minCardinality="1">
  <a:domain rdf:resource="&patient;Quantity"/>
  <a:overriddenProperty rdf:resource="&patient;unit"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Quantity_value"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="integer">
  <a:domain rdf:resource="&patient;Quantity"/>
  <a:overriddenProperty rdf:resource="&patient;value"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Register-agent_reg-agent"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="cls">
  <a:allowedParents rdf:resource="&patient;AID"/>
  <a:domain rdf:resource="&patient;Register-agent"/>
  <a:overriddenProperty rdf:resource="&patient;reg-agent"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Telecommunication_e-mail"
    a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Telecommunication"/>
  <a:overriddenProperty rdf:resource="&patient;e-mail"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Telecommunication_fax"
    a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Telecommunication"/>
  <a:overriddenProperty rdf:resource="&patient;fax"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Telecommunication_other"
    a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Telecommunication"/>
  <a:overriddenProperty rdf:resource="&patient;other"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Telecommunication_phone"
    a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Telecommunication"/>
  <a:overriddenProperty rdf:resource="&patient;phone"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Telecommunication_web"
    a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Telecommunication"/>
  <a:overriddenProperty rdf:resource="&patient;web"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
```

```
<a:OverridingProperty rdf:about="&a;Treatment_dosage-pattern"
    a:maxCardinality="1"
    a:minCardinality="1">
  <a:domain rdf:resource="&patient;Treatment"/>
  <a:overriddenProperty rdf:resource="&patient;dosage-pattern"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Treatment_dose"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="cls">
  <a:allowedParents rdf:resource="&patient;Quantity"/>
  <a:domain rdf:resource="&patient;Treatment"/>
  <a:overriddenProperty rdf:resource="&patient;dose"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Treatment_drug-name"
    a:maxCardinality="1"
    a:minCardinality="1">
  <a:domain rdf:resource="&patient;Treatment"/>
  <a:overriddenProperty rdf:resource="&patient;drug-name"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Treatment_drug-type"
    a:maxCardinality="1"
    a:minCardinality="1">
  <a:domain rdf:resource="&patient;Treatment"/>
  <a:overriddenProperty rdf:resource="&patient;drug-type"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Treatment_duration"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="cls">
  <a:allowedParents rdf:resource="&patient;Quantity"/>
  <a:domain rdf:resource="&patient;Treatment"/>
  <a:overriddenProperty rdf:resource="&patient;duration"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Treatment_route-administration"
    a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Treatment"/>
  <a:overriddenProperty rdf:resource="&patient;route-administration"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Visit-booking_patient-aid"
    a:maxCardinality="1"
    a:minCardinality="1"
    a:range="cls">
  <a:allowedParents rdf:resource="&patient;AID"/>
  <a:domain rdf:resource="&patient;Visit-booking"/>
  <a:overriddenProperty rdf:resource="&patient;patient-aid"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Visit-booking_patient-id-medical"
    a:maxCardinality="1"
    a:minCardinality="1">
  <a:domain rdf:resource="&patient;Visit-booking"/>
  <a:overriddenProperty rdf:resource="&patient;patient-id-medical"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Visit-booking_visit-doctor"
    a:maxCardinality="1"
    a:range="cls">
  <a:allowedParents rdf:resource="&patient;Healthcare-person"/>
  <a:domain rdf:resource="&patient;Visit-booking"/>
  <a:overriddenProperty rdf:resource="&patient;visit-doctor"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Visit-booking_visit-place"
    a:maxCardinality="1"
    a:range="cls">
  <a:allowedParents rdf:resource="&patient;Healthcare-organization"/>
  <a:domain rdf:resource="&patient;Visit-booking"/>
  <a:overriddenProperty rdf:resource="&patient;visit-place"/>
```

```
  <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Visit-booking_visit-problem"
   a:maxCardinality="1"
   a:minCardinality="1"
   a:range="cls">
  <a:allowedParents rdf:resource="&patient;Problem-description"/>
  <a:domain rdf:resource="&patient;Visit-booking"/>
  <a:overriddenProperty rdf:resource="&patient;visit-problem"/>
  <rdfs:range rdf:resource="&rdfs;Class"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Working-hours_begin"
   a:maxCardinality="1"
   a:minCardinality="1">
  <a:domain rdf:resource="&patient;Working-hours"/>
  <a:overriddenProperty rdf:resource="&patient;begin"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Working-hours_day-week"
   a:maxCardinality="1"
   a:minCardinality="1">
  <a:domain rdf:resource="&patient;Working-hours"/>
  <a:overriddenProperty rdf:resource="&patient;day-week"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
<a:OverridingProperty rdf:about="&a;Working-hours_end"
   a:maxCardinality="1">
  <a:domain rdf:resource="&patient;Working-hours"/>
  <a:overriddenProperty rdf:resource="&patient;end"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</a:OverridingProperty>
</rdf:RDF>
```

C. Fitxer de configuració

Per tal que cada agent del nostre sistema (*Personal*, *Doctor*, *Department* i *Medical Centre*) presenti unes característiques diferenciades al de la resta d'agents del seu mateix tipus (ex: l'agent *Personal* s'adapti al perfil de l'usuari), cadascun tindrà a la seva disposició un fitxer de configuració del que llegirà les seves dades d'inicialització.

Així, els agents *Doctors*, *Departments* i *Medical Centres* que podran obtenir quines seran les característiques pròpies que definiran el seu comportament (ex: nom, horari de treball, centre al que pertany...) a través d'un fitxer definit a priori. El *Personal*, per la seva banda, heretarà les característiques que l'indiqui l'usuari a través del fitxer descriptor obtingut a través del programa de configuració.

Els fitxers tindran el mateix format que el contingut dels missatges que s'envien entre els agents (per fer-los fàcilment integrables, permetre l'interoperabilitat i agilitar el tractament).

Un exemple d'aquests fitxers corresponent al cas d'un *Personal Agent* és el que s'inclou a continuació.

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/TR/1999/PR-rdf-schema-19990303#" xmlns:FIPA-
RDF="http://www.fipa.org/schemas/FIPA-RDF#">
  <FIPA-RDF:Object>
    <FIPA-RDF:type>Equals</FIPA-RDF:type>
    <FIPA-RDF:attribute>
      <FIPA-RDF:AttributeDescription>
        <FIPA-RDF:type>equals_right</FIPA-RDF:type>
        <FIPA-RDF:attributeValue>
          <FIPA-RDF:Aggregate>
            <FIPA-RDF:type>set</FIPA-RDF:type>
            <FIPA-RDF:aggregateElement>
              <FIPA-RDF:Primitive>
                <FIPA-RDF:type>String</FIPA-RDF:type>
                <FIPA-RDF:primitiveValue><![CDATA[true]]></FIPA-RDF:primitiveValue>
              </FIPA-RDF:Primitive>
            </FIPA-RDF:aggregateElement>
          </FIPA-RDF:Aggregate>
        </FIPA-RDF:attributeValue>
      </FIPA-RDF:AttributeDescription>
    </FIPA-RDF:attribute>
    <FIPA-RDF:attribute>
      <FIPA-RDF:AttributeDescription>
        <FIPA-RDF:type>equals_left</FIPA-RDF:type>
        <FIPA-RDF:attributeValue>
          <FIPA-RDF:Object>
            <FIPA-RDF:type>Patient</FIPA-RDF:type>
            <FIPA-RDF:attribute>
              <FIPA-RDF:AttributeDescription>
                <FIPA-RDF:type>antecedents</FIPA-RDF:type>
                <FIPA-RDF:attributeValue>
                  <FIPA-RDF:Aggregate>
                    <FIPA-RDF:type>set</FIPA-RDF:type>
                    <FIPA-RDF:aggregateElement>
                      <FIPA-RDF:Primitive>
                        <FIPA-RDF:type>String</FIPA-RDF:type>
                      <FIPA-RDF:primitiveValue><![CDATA[nothing]]></FIPA-
RDF:primitiveValue>
                      </FIPA-RDF:Primitive>
                    </FIPA-RDF:aggregateElement>
                  </FIPA-RDF:Aggregate>
```

```
                </FIPA-RDF:attributeValue>
              </FIPA-RDF:AttributeDescription>
            </FIPA-RDF:attribute>
            <FIPA-RDF:attribute>
              <FIPA-RDF:AttributeDescription>
                <FIPA-RDF:type>allergies</FIPA-RDF:type>
                <FIPA-RDF:attributeValue>
                  <FIPA-RDF:Aggregate>
                    <FIPA-RDF:type>set</FIPA-RDF:type>
                    <FIPA-RDF:aggregateElement>
                      <FIPA-RDF:Primitive>
                        <FIPA-RDF:type>String</FIPA-RDF:type>
                      <FIPA-RDF:primitiveValue><![CDATA[nothing]]></FIPA-
RDF:primitiveValue>
                      </FIPA-RDF:Primitive>
                    </FIPA-RDF:aggregateElement>
                  </FIPA-RDF:Aggregate>
                </FIPA-RDF:attributeValue>
              </FIPA-RDF:AttributeDescription>
            </FIPA-RDF:attribute>
            <FIPA-RDF:attribute>
              <FIPA-RDF:AttributeDescription>
                <FIPA-RDF:type>job</FIPA-RDF:type>
                <FIPA-RDF:attributeValue>
                  <FIPA-RDF:Primitive>
                    <FIPA-RDF:type>String</FIPA-RDF:type>
                   <FIPA-RDF:primitiveValue><![CDATA[Student]]></FIPA-
RDF:primitiveValue>
                  </FIPA-RDF:Primitive>
                </FIPA-RDF:attributeValue>
              </FIPA-RDF:AttributeDescription>
            </FIPA-RDF:attribute>
            <FIPA-RDF:attribute>
              <FIPA-RDF:AttributeDescription>
                <FIPA-RDF:type>aid</FIPA-RDF:type>
                <FIPA-RDF:attributeValue>
                  <FIPA-RDF:Object>
                    <FIPA-RDF:type>agent-identifier</FIPA-RDF:type>
                    <FIPA-RDF:attribute>
                      <FIPA-RDF:AttributeDescription>
                        <FIPA-RDF:type>name</FIPA-RDF:type>
                        <FIPA-RDF:attributeValue>
                          <FIPA-RDF:Primitive>
                            <FIPA-RDF:type>String</FIPA-RDF:type>
                           <FIPA-RDF:primitiveValue><![CDATA[empty]]></FIPA-
RDF:primitiveValue>
                          </FIPA-RDF:Primitive>
                        </FIPA-RDF:attributeValue>
                      </FIPA-RDF:AttributeDescription>
                    </FIPA-RDF:attribute>
                  </FIPA-RDF:Object>
                </FIPA-RDF:attributeValue>
              </FIPA-RDF:AttributeDescription>
            </FIPA-RDF:attribute>
            <FIPA-RDF:attribute>
              <FIPA-RDF:AttributeDescription>
                <FIPA-RDF:type>id_medical</FIPA-RDF:type>
                <FIPA-RDF:attributeValue>
                  <FIPA-RDF:Primitive>
                    <FIPA-RDF:type>String</FIPA-RDF:type>
                   <FIPA-RDF:primitiveValue><![CDATA[2347263]]></FIPA-
RDF:primitiveValue>
                  </FIPA-RDF:Primitive>
                </FIPA-RDF:attributeValue>
              </FIPA-RDF:AttributeDescription>
            </FIPA-RDF:attribute>
            <FIPA-RDF:attribute>
              <FIPA-RDF:AttributeDescription>
                <FIPA-RDF:type>disability</FIPA-RDF:type>
                <FIPA-RDF:attributeValue>
                  <FIPA-RDF:Aggregate>
                    <FIPA-RDF:type>set</FIPA-RDF:type>
                    <FIPA-RDF:aggregateElement>
                      <FIPA-RDF:Primitive>
                        <FIPA-RDF:type>String</FIPA-RDF:type>
```

```
                      <FIPA-RDF:primitiveValue><![CDATA[nothing]]></FIPA-
RDF:primitiveValue>
                      </FIPA-RDF:Primitive>
                    </FIPA-RDF:aggregateElement>
                  </FIPA-RDF:Aggregate>
                </FIPA-RDF:attributeValue>
              </FIPA-RDF:AttributeDescription>
            </FIPA-RDF:attribute>
            <FIPA-RDF:attribute>
              <FIPA-RDF:AttributeDescription>
                <FIPA-RDF:type>id_card</FIPA-RDF:type>
                <FIPA-RDF:attributeValue>
                  <FIPA-RDF:Primitive>
                    <FIPA-RDF:type>String</FIPA-RDF:type>
                   <FIPA-RDF:primitiveValue><![CDATA[47763566]]></FIPA-
RDF:primitiveValue>
                  </FIPA-RDF:Primitive>
                </FIPA-RDF:attributeValue>
              </FIPA-RDF:AttributeDescription>
            </FIPA-RDF:attribute>
            <FIPA-RDF:attribute>
              <FIPA-RDF:AttributeDescription>
                <FIPA-RDF:type>sex</FIPA-RDF:type>
                <FIPA-RDF:attributeValue>
                  <FIPA-RDF:Primitive>
                    <FIPA-RDF:type>String</FIPA-RDF:type>
                    <FIPA-RDF:primitiveValue><![CDATA[male]]></FIPA-RDF:primitiveValue>
                  </FIPA-RDF:Primitive>
                </FIPA-RDF:attributeValue>
              </FIPA-RDF:AttributeDescription>
            </FIPA-RDF:attribute>
            <FIPA-RDF:attribute>
              <FIPA-RDF:AttributeDescription>
                <FIPA-RDF:type>patient_name</FIPA-RDF:type>
                <FIPA-RDF:attributeValue>
                  <FIPA-RDF:Object>
                    <FIPA-RDF:type>Person_name</FIPA-RDF:type>
                    <FIPA-RDF:attribute>
                      <FIPA-RDF:AttributeDescription>
                        <FIPA-RDF:type>first_name</FIPA-RDF:type>
                        <FIPA-RDF:attributeValue>
                          <FIPA-RDF:Primitive>
                            <FIPA-RDF:type>String</FIPA-RDF:type>
                           <FIPA-RDF:primitiveValue><![CDATA[David]]></FIPA-
RDF:primitiveValue>
                          </FIPA-RDF:Primitive>
                        </FIPA-RDF:attributeValue>
                      </FIPA-RDF:AttributeDescription>
                    </FIPA-RDF:attribute>
                    <FIPA-RDF:attribute>
                      <FIPA-RDF:AttributeDescription>
                        <FIPA-RDF:type>second_surname</FIPA-RDF:type>
                        <FIPA-RDF:attributeValue>
                          <FIPA-RDF:Primitive>
                            <FIPA-RDF:type>String</FIPA-RDF:type>
                           <FIPA-RDF:primitiveValue><![CDATA[Ruenes]]></FIPA-
RDF:primitiveValue>
                          </FIPA-RDF:Primitive>
                        </FIPA-RDF:attributeValue>
                      </FIPA-RDF:AttributeDescription>
                    </FIPA-RDF:attribute>
                    <FIPA-RDF:attribute>
                      <FIPA-RDF:AttributeDescription>
                        <FIPA-RDF:type>title</FIPA-RDF:type>
                        <FIPA-RDF:attributeValue>
                          <FIPA-RDF:Primitive>
                            <FIPA-RDF:type>String</FIPA-RDF:type>
                           <FIPA-RDF:primitiveValue><![CDATA[empty]]></FIPA-
RDF:primitiveValue>
                          </FIPA-RDF:Primitive>
                        </FIPA-RDF:attributeValue>
                      </FIPA-RDF:AttributeDescription>
                    </FIPA-RDF:attribute>
                    <FIPA-RDF:attribute>
                      <FIPA-RDF:AttributeDescription>
```

```
                        <FIPA-RDF:type>first_surname</FIPA-RDF:type>
                        <FIPA-RDF:attributeValue>
                          <FIPA-RDF:Primitive>
                            <FIPA-RDF:type>String</FIPA-RDF:type>
                           <FIPA-RDF:primitiveValue><![CDATA[Sanchez]]></FIPA-
RDF:primitiveValue>
                          </FIPA-RDF:Primitive>
                        </FIPA-RDF:attributeValue>
                      </FIPA-RDF:AttributeDescription>
                    </FIPA-RDF:attribute>
                  </FIPA-RDF:Object>
                </FIPA-RDF:attributeValue>
              </FIPA-RDF:AttributeDescription>
            </FIPA-RDF:attribute>
            <FIPA-RDF:attribute>
              <FIPA-RDF:AttributeDescription>
                <FIPA-RDF:type>patient_phone</FIPA-RDF:type>
                <FIPA-RDF:attributeValue>
                  <FIPA-RDF:Primitive>
                    <FIPA-RDF:type>String</FIPA-RDF:type>
                   <FIPA-RDF:primitiveValue><![CDATA[977373340]]></FIPA-
RDF:primitiveValue>
                  </FIPA-RDF:Primitive>
                </FIPA-RDF:attributeValue>
              </FIPA-RDF:AttributeDescription>
            </FIPA-RDF:attribute>
            <FIPA-RDF:attribute>
              <FIPA-RDF:AttributeDescription>
                <FIPA-RDF:type>patient_address</FIPA-RDF:type>
                <FIPA-RDF:attributeValue>
                  <FIPA-RDF:Object>
                    <FIPA-RDF:type>Postal_address</FIPA-RDF:type>
                    <FIPA-RDF:attribute>
                      <FIPA-RDF:AttributeDescription>
                        <FIPA-RDF:type>zip_code</FIPA-RDF:type>
                        <FIPA-RDF:attributeValue>
                          <FIPA-RDF:Primitive>
                            <FIPA-RDF:type>String</FIPA-RDF:type>
                           <FIPA-RDF:primitiveValue><![CDATA[43840]]></FIPA-
RDF:primitiveValue>
                          </FIPA-RDF:Primitive>
                        </FIPA-RDF:attributeValue>
                      </FIPA-RDF:AttributeDescription>
                    </FIPA-RDF:attribute>
                    <FIPA-RDF:attribute>
                      <FIPA-RDF:AttributeDescription>
                        <FIPA-RDF:type>country_code</FIPA-RDF:type>
                        <FIPA-RDF:attributeValue>
                          <FIPA-RDF:Primitive>
                            <FIPA-RDF:type>String</FIPA-RDF:type>
                           <FIPA-RDF:primitiveValue><![CDATA[SP]]></FIPA-
RDF:primitiveValue>
                          </FIPA-RDF:Primitive>
                        </FIPA-RDF:attributeValue>
                      </FIPA-RDF:AttributeDescription>
                    </FIPA-RDF:attribute>
                    <FIPA-RDF:attribute>
                      <FIPA-RDF:AttributeDescription>
                        <FIPA-RDF:type>street_number</FIPA-RDF:type>
                        <FIPA-RDF:attributeValue>
                          <FIPA-RDF:Primitive>
                            <FIPA-RDF:type>String</FIPA-RDF:type>
                           <FIPA-RDF:primitiveValue><![CDATA[58]]></FIPA-
RDF:primitiveValue>
                          </FIPA-RDF:Primitive>
                        </FIPA-RDF:attributeValue>
                      </FIPA-RDF:AttributeDescription>
                    </FIPA-RDF:attribute>
                    <FIPA-RDF:attribute>
                      <FIPA-RDF:AttributeDescription>
                        <FIPA-RDF:type>city</FIPA-RDF:type>
                        <FIPA-RDF:attributeValue>
                          <FIPA-RDF:Primitive>
                            <FIPA-RDF:type>String</FIPA-RDF:type>
```

```
                          <FIPA-RDF:primitiveValue><![CDATA[Salou]]></FIPA-
RDF:primitiveValue>
                          </FIPA-RDF:Primitive>
                        </FIPA-RDF:attributeValue>
                      </FIPA-RDF:AttributeDescription>
                    </FIPA-RDF:attribute>
                    <FIPA-RDF:attribute>
                      <FIPA-RDF:AttributeDescription>
                        <FIPA-RDF:type>street</FIPA-RDF:type>
                        <FIPA-RDF:attributeValue>
                          <FIPA-RDF:Primitive>
                            <FIPA-RDF:type>String</FIPA-RDF:type>
                          <FIPA-RDF:primitiveValue><![CDATA[Faro]]></FIPA-
RDF:primitiveValue>
                          </FIPA-RDF:Primitive>
                        </FIPA-RDF:attributeValue>
                      </FIPA-RDF:AttributeDescription>
                    </FIPA-RDF:attribute>
                  </FIPA-RDF:Object>
                </FIPA-RDF:attributeValue>
              </FIPA-RDF:AttributeDescription>
            </FIPA-RDF:attribute>
            <FIPA-RDF:attribute>
              <FIPA-RDF:AttributeDescription>
                <FIPA-RDF:type>date_of_birth</FIPA-RDF:type>
                <FIPA-RDF:attributeValue>
                  <FIPA-RDF:Primitive>
                    <FIPA-RDF:type>String</FIPA-RDF:type>
                   <FIPA-RDF:primitiveValue><![CDATA[1980/09/23]]></FIPA-
RDF:primitiveValue>
                  </FIPA-RDF:Primitive>
                </FIPA-RDF:attributeValue>
              </FIPA-RDF:AttributeDescription>
            </FIPA-RDF:attribute>
          </FIPA-RDF:Object>
        </FIPA-RDF:attributeValue>
      </FIPA-RDF:AttributeDescription>
    </FIPA-RDF:attribute>
  </FIPA-RDF:Object>
</rdf:RDF>
```

D. Contingut de la Base de Dades

La base de dades guarda les dades personals del pacient, el seu historial mèdic i les seves claus pública/privada.

Per aquest motiu s'han dissenyat dues taules:

- PERSONALDATA: guarda la informació personal del pacient juntament amb les seves claus, utilitzant com a clau primària el seu identificador personal (DNI) i l'identificador mèdic (número de la SS).

- HISTORY: en aquesta taula es guarden les dades referents a un episodi mèdic, per tant, un usuari podrà tenir-ne vàries entrades segons el nombre de visites realitzades. Aquests enregistraments es relacionen amb el pacient a través del seu identificador mèdic (clau forana). Com a identificador de la taula, utilitzarem el seu identificador mèdic, la data i l'hora de la visita i el metge (d'aquesta forma es podrà visitar diversos cops en un dia amb diversos metges o fins i tot amb el mateix).

Els camps de cada taula i els tipus de dades i atributs associats a cadascun són els següents:

```
str =
        "CREATE TABLE PERSONALDATA " +
        "(id_medical CHAR(10) NOT NULL, " +
        "id_card CHAR(9) NOT NULL, "+
        "first_name CHAR(10), "+
        "first_surname CHAR(10), "+
        "second_surname CHAR(10), "+
        "title CHAR(5), "+
        "sex CHAR(10), "+
        "date_of_birth CHAR(10), "+
        "allergies CHAR(50), " +
        "street CHAR(50), "+
        "street_number CHAR(3), "+
        "zip_code CHAR(5), "+
        "country_code CHAR(5), "+
        "city CHAR(30), "+
        "observations CHAR(100), " +
        "job CHAR(10), "+
        "disability CHAR(50), " +
        "patient_phone CHAR(15), " +
        "antecedents CHAR(50), " +
        "private_key BLOB, "+
        "public_key CHAR(255), "+
        "PRIMARY KEY (id_medical,id_card))";
      bd.Update(str);
      System.err.println("Created PERSONALDATA table");


      str =
        "CREATE TABLE HISTORY " +
```

```
  "(idmedical CHAR(10) NOT NULL, " +
  "datevisit CHAR(10) NOT NULL, "+
  "hourvisit CHAR(8) NOT NULL, " +
  "doctor CHAR(30) NOT NULL, "+
  "center CHAR(30), "+
  "department CHAR(30), "+
  "typeproblem CHAR(20), " +
  "urgency CHAR(10), " +
  "description CHAR(100), "+
  "observations CHAR(100), "+
  "results CHAR(100), "+
  "tests CHAR(100), " +
  "drug_name CHAR(20), "+
  "drug_type CHAR(10), " +
  "route_administration CHAR (10), "+
  "dose_value CHAR (10), "+
  "dose_unit CHAR(10), "+
  "dosage_pattern CHAR(30), "+
  "duration_value CHAR(10), "+
  "duration_unit CHAR(10), "+
  "PRIMARY KEY (idmedical,datevisit,hourvisit,doctor))";
bd.Update(str);
System.err.println("Created HISTORY table.");
```

Per la comunicació amb la BD física s'ha utilitzat la llibreria JDBC sobre una MySql [42] (BD de lliure distribució força versàtil i fàcil d'utilitzar que suporta fins a 100 clients i funciona sota Windows i Linux, sent una de les més utilitzades a Internet).

REFERÈNCIES

[1] Agentcities, Agentcities Agent Technology Comptetition, plana web: http://www.agentcities.org/EUNET/Competition

[2] AgentCities, plana web: http://www.agentcities.org

[3] Alvedra, O., Domingo Ferrer, J., Herrera, J., Rius, M. A., Sebe, F., Soriano, M., *Comerç electrònic*, EdiUOC 2003.

[4] Apache XML Project, Xerces Java parser, plana web : http://xml.apache.org/xerces-j/index.html

[5] ASTM E31, plana web: http://www.astm.org/COMMIT/COMMITTEE/E31.htm

[6] Banzai, Grup de recerca en Intel·ligència Artificial, DEIM, Universitat Rovira i Virgili, plana web: http://www.etse.urv.es/recerca/banzai

[7] Bellifine, F., Caire, G., [et. al.] *JADE Programer's Guide*. CSET S.p.A. and Univeristy of Parma, Setembre 2002 (v.2.61). http://sharon.cselt.it/projects/jade/

[8] Bellifine, F., Caire, G., [et. al.]. *JADE Administrator'g guide*, CSELT S.p.A. and University of Parma, Setembre 2002 (v 2.61), http://sharon.cselt.it/projects/jade/

[9] Bellifine, F., Caire, G., [et. al.]. *JADE Application-defined content language and ontologies*, CSELT S.p.A. and University of Parma, Febrer 2002 (v 2.5), http://sharon.cselt.it/projects/jade/

[10] Bellifine, F., Caire, G., [et. al.]. *JADE Security Administrator Guide*, CSELT S.p.A. and University of Parma, Setembre 2002 (v 2.61), http://sharon.cselt.it/projects/jade/

[11] Brenner, W., Zarnekow, R., Wittig, H., *Intelligent Software Agents. Foundations and Applications*. Springer Verlag, 1998.

[12] CEN/TC251 (European Committee for Standardization), plana web: http://www.centc251.org

[13] Cryptonite. Package of Java logi.crypto, plana web: http://logi.org/logi.crypto.

[14] Domingo Ferrer, J., Herrera Joancomartí, J., *Criptografia per als serveis telemàtics i el comerç electrònic*, EdiUOC, 1999.

[15] Foundation for Intelligent Physical Agents (1998), *Agent Communication Language*, FIPA

[16] Foundation for Intelligent Physical Agents (1998), *Agent Management*, FIPA

[17] Foundation for Intelligent Physical Agents (1998), *FIPA97 Developers Guide*, FIPA

[18] Foundation for Intelligent Physical Agents (2000), *FIPA ACL Message Structure*, FIPA

[19] Foundation for Intelligent Physical Agents (2000), *FIPA Agent Management Specification*, FIPA

[20] Foundation for Intelligent Physical Agents (2000), *FIPA Agent Message Transport Envelope Representation in XML Specification*, FIPA

[21] Foundation for Intelligent Physical Agents (2000), *FIPA Content Languages Specification*, FIPA

[22] Foundation for Intelligent Physical Agents (2000), *FIPA Protocol Library Specification*, FIPA

[23] Foundation for Intelligent Physical Agents (2000), *FIPA RDF Content Language Specification*, FIPA

[24] Foundation for Intelligent Physical Agents (2000), *FIPA SL Content Language Specification*, FIPA

[25] Foundation for Intelligent Physical Agents (2001), *FIPA Device Ontology*, FIPA

[26] Foundation for Intelligent Physical Agents (2001), *FIPA MAS Security white paper*, FIPA

[27] Foundation for Intelligent Physical Agents, plana web: http://www.fipa.org

[28] Foundation fro Intelligent Physical Agents (2002) *FIPA MAS Security white paper*, FIPA

[29] Generalitat de Catalunya. *Llei sobre els drets d'informació concernent la salut i l'autonomia del pacient, i la documentació clínica*. Llei 21/2000, 29 Decembre 2000, Departament de Salut i Seguretat Social, Generalitat de Catalunya. Quaderns de legislació, 31. ISBN 84-393-5459-2.

[30] GruSMA, Grup de Sistemes Multi-Agent, DEIM, Universitat Rovira i Virgili, plana web: http://www.etse.urv.es/recerca/banzai/toni/MAS

[31] GruSMA, GruSMA1, plana web: http://grusma.etse.urv.es/~agentcities

[32] Guia de la Salut 2002-03, CatSalut, Generalitat de Catalunya

[33] HL7 (Health Level Seven), plana web: http://hl7.org

[34] Institut Català de la Salut, plana web : http://www.gencat.es/ics/

[35] Isern, D., (2001), *Disseny i implementació d'un Sistema Multiagent per proporcionar serveis mèdics dins del projecte AgentCities*, Projecte Final de Carrera, Dept. Enginyeria Informàtica i Matemàtiques, Universitat Rovira i Virgili

[36] Isern, D., Moreno, A., Valls, A. (2001), *Desenvolupament de sistemes Multi-Agent en JADE*, Report de Recerca DEIM-RR-01-003, Dept. Enginyeria Informàtica i Matemàtiques, Universitat Rovira i Virgili

[37] Java Ontology BeanGenerator for JADE 2.6, plana web: http://gaper.swi.psy.uva.nl/beangenerator/content/main.php

[38] Kermanog, Knowledge and Language products, plana web: http://www.kermanog.com/

[39] Lassila, O., Swick, R. R., *Resource description framework (RDF) model and syntax specification*. Technical report, W3c. W3c Recommendacion, 1999.

[40] Moreno, A., Isern, D., Sánchez, D., *Provision of agent-based health care services*, Special issue of AI-Communications devoted to Applications of Intelligent Agents in Health Care (Ed. A. Moreno).

[41] Moreno, A., Valls, A., Isern, D., Sánchez, D., *GruSMA1: Experience on the Deployment of Agent-Based Health Care Services*, Computer Science & Mathematics Departmet, Universitat Rovira i Virgili

[42] MySQL database 3.23, plana web: http://mysql.com

[43] MySQL driver for Java, plana web: http://mmmysql.sourceforge.net/

[44] Nealon, J., Moreno, A., *The application of agent technology to health care*. Proceedings of the workshop *AgentCities: research in large scale open agents environments*, within the *First International Join Conference on Autonomous Agents and Multi-Agent Systems*, AAMAS 2002, ACM Press, pp. 169-173. Bolonia, Italy, July 2002.

[45] Netbeans IDE for Java, plana web: http://www.netbeans.org/

[46] Netscape Communication Corp, *Introduction to Public-key Cryptography*, http://developer.nescape.com/docs/manuals/security/pkin/contents.htm

[47] Netscape Communication Corp, *Introduction to SSL*, http://developer.netscape.com/docs/manuals/security/sslin/contents.htm

[48] Netscape Communication Corp, *Undertanding Public Key Infraestructure (PKI)*, http://verisign.netscape.com/security/pki/understanding.html

[49] OntoEdit, plana web: http://ontoprise.de/ontoedit.htm

[50] OpenGALEN, plana web: http://www.opengalen.org

[51] Paola T., (University of Parma), RDF parser http://sharon.cselt.it/projects/jade/doc/tutorials/RDFCodec.html

[52] Perreault, S., Wiederhold, F., *Medical Informatics, Computer Applications in Health Care and Biomedicine*, Ed. Springer

[53] Poggi, A., Rimassa, G., Tomaiuolo, M., *Multi-User and Security Support for Multi-Agent Systems*, DII – Universitat de Parma, Proceedings of WOA 2001, Modena, Sep 2001.

[54] Protègè Project, Protègè 2000 1.7, plana web: http://www.protege.stanford.edu

[55] Servei Català de la Salut, CatSalut, plana web: http://www.gentcat.es/scs

[56] SNOMED, plana web: http://www.snomed.org

[57] Staab, S., Erdmann, M., Maedche, A. *Ontologies in RDF(S)*. Linköping Electronic Articles in Computer and Information Systems, Vol. 6 (2001).

[58] Sun Microsystems, *Authentication and Authorization Service (JAAS)*, http://java.sun.com/products/jass/index-14.html

[59] Sun Microsystems, *Java Default Policy Implementation and Policy File Syntax*, http://java.sun.com/j2se/1.4/docs/guide/security/PolicyFiles.html

[60] Sun Microsystems, Java Development Kit 1.4, http://java.sun.com

[61] Sun Microsystems, *Java Secure Socket Extension (JSSE) Reference Guide*, http://java.sun.com/j2se/1.4/docs/guide7security/jsse/SEERefGuide.html

[62] Sun Microsystems, *Java Security*, http://java.sun.com/security

[63] TILab, S.p.A., Jade 2.6.1, plana web: http://sharon.cslet.it/project/jade

[64] TILab, S.p.A., Jade-S security add-on, plana web: http://sharon.cslet.it/project/jade

[65] TILab, S.p.A., RDF Jade add-on, plana web: http://shareon.cslet.it/project./jade

[66] UMLS, plana web: http://www.nlm.nih.gov/research/umls

[67] University of Parma, RDF parser, plana web: http://www-db.stanford.edu/~melnik/rdf/api.html

[68] US Department of Health and Human Services, *Standards for Privacy and Individually Identifiable Health Information*, Federal Register, volume 65. December 28th, 2000.

[69] Verisign, plana web: http://www.verisign.com/

[70] W3C, Extensible Markup Language (XML), plana web: http://www.w3.org/XML/

[71] W3C, Resource Description Framework (RDF), plana web: http://www.w3.org/RDF

[72] W3C, World Wide Web Consortium, plana web: http://www.w3.org/

[73] WebODE, plana web: http://delicias.dia.fi.upm.es/webODE

[74] Weiss, G., (1999) (Ed.) *Multiagent Systems. A modern approach to distributed artificial intelligence*, MIT Press.

[75] Wong, H., Sycara, K., *Adding security and trust to multi-agent systems*, In Proceedings of Autonomous Agents'99, Workshop on deception, fraud and trust in agent societies, pp. 149-161. Seattle, Washington, May 1999.

[76] Wooldridge, M., *An introduction to multiagent systems*, John Wiley Ed., 2002. ISBN 0-471-49691-X.