

Departament d'Enginyeria



Informàtica i
Matemàtiques



UNIVERSITAT
ROVIRA I VIRGILI

PROJECTE FINAL DE CARRERA

Visualitzador de Guies de Pràctica Clínica

Autor: Francesc Alcolea Llaveria
[fal.ei@estudiants.urv.cat]

Enginyeria Informàtica (EI)

Directors del projecte: Dr. Antonio Moreno Ribas
[amoreno@urv.cat]
David Isern Alarcón
[disern@urv.cat]

Juliol de 2006

ÍNDEX

1.	INTRODUCCIÓ.....	1
1.1.	Antecedents	2
1.2.	Objectius.....	3
2.	GUIES DE PRÀCTICA CLÍNICA	5
2.1.	Descripció del llenguatge PROforma.....	5
2.1.1.	Model PROforma	5
2.2.	Descripció del sistema PROforma.....	7
2.2.1.	La sintaxi PROforma	7
2.2.2.	Els components de PROforma.....	7
2.2.3.	Estat d'una tasca PROforma.....	8
2.3.	Exemple d'una GPC en PROforma.....	9
2.3.1.	Representació de la GPC	9
3.	EINES GRÀFIQUES	15
3.1.	JGraph.....	15
3.1.1.	Arquitectura JGraph	16
3.1.1.1.	Model JGraph (MVC)	17
3.1.1.2.	Vista JGraph (MVC)	18
3.1.1.3.	Control JGraph (MVC).....	19
3.1.2.	Conclusions	19
3.2.	GraphViz	20
3.2.1.	Introducció.....	20
3.2.2.	Conclusió	22
3.3.	TouchGraph	23
3.3.1.	Descripció	23
3.3.2.	KAON OlModeller	25
3.3.3.	GraphLayout.....	26
3.3.3.1.	Diagrama de Paquets	27
3.3.3.2.	Diagrama de Classes.....	28
3.3.4.	WikiBrowser.....	29
3.3.5.	LinkBrowser	30
3.4.	JOpenChart	31
3.4.1.	Descripció	31
3.4.2.	Conclusió	32
3.5.	JFreeChart.....	33
3.5.1.	Descripció	33
3.5.2.	Conclusió	34
3.6.	JGo: GoDiagram for Java	35
3.6.1.	Descripció	35
3.6.2.	Conclusió	36
3.7.	ILOG JViews Diagrammer.....	37
3.7.1.	Descripció	37
3.7.2.	Conclusió	39
3.8.	Elecció de l'eina gràfica	40
3.8.1.	Taula comparativa	40
3.8.2.	TouchGraph en el PFC	41
4.	TECNOLOGIA WEB I VALIDACIÓ D'USUARIS WEB.....	43

4.1.	Pàgines dinàmiques	43
4.1.1.	HTML	43
4.1.2.	JSP: JavaServer Page	44
4.1.3.	JavaScript	45
4.1.4.	Applets	47
4.1.5.	Sessions	48
4.2.	Validació d' usuaris via web	49
4.2.1.	Autenticació i autorització	49
4.2.2.	Integritat de les dades	51
4.2.3.	Confidencialitat	51
4.2.4.	JAAS Realm	51
4.2.4.1.	Descripció de les classes més importants	53
4.2.4.2.	Configuració	57
4.2.5.	LCF: fitxer de configuració	59
4.2.6.	Justificació de JAAS en el PFC	60
5.	JOC DE PROVES	61
5.1.	Guia pràctica clínica: AHDH Model	61
5.2.	Guia pràctica clínica: Cancer Diagnosis Model	62
5.3.	Guia pràctica clínica: Dvt Model	64
5.4.	Guia pràctica clínica: Hypertension Model	65
5.5.	Guia pràctica clínica: Weigh loss Model	66
5.6.	Guia pràctica clínica: Cough Model	67
6.	CONCLUSIONS	69
7.	BIBLIOGRAFIA I RECURSOS UTILITZATS	71
7.1.	RECURSOS UTILITZATS	72
7.1.1.	Software	72
7.1.1.1.	Llibreries	72
7.1.1.2.	Programes	73
ANNEX		74
	Gpc Hypertensio en PROforma	74
	Manual d'usuari	81

1. INTRODUCCIÓ

Com en qualsevol enginyeria i en particular en la d'informàtica, l'estudi, disseny i creació d'una aplicació informàtica té com a principal objectiu la necessitat de crear una solució a un determinat problema que d'altra manera seria una feina complexa o fins i tot costosa de desenvolupar. Per tant, s'han de proporcionar les eines adequades per tal que la gestió de l'aplicació informàtica no sigui un entrellat i sigui fàcil de manegar, en aquest cas per l'usuari final.

La tasca que es vol desenvolupar està ubicada en el camp de la medicina, i es desitja presentar una aplicació visual que permeti ajudar a la presa de decisions.

En termes generals, la gestió dels serveis mèdics és un tema complex de tractar pel nombre de camps que inclou i la gran quantitat de volum d'informació que manega, com poden ser gestió de centres de salut, horaris, personal mèdic, documentació, etc.; per tant, es crea la necessitat de tenir sistemes informàtics que ajudin a la gestió de tota aquesta informació.

El grup d'Intel·ligència artificial BANZAI, un grup de recerca format per professors i estudiants de doctorats, ubicat a la Universitat Rovira i Virgili, fa temps que treballa en el camp de la gestió de serveis mèdics, fent recerca i donant solucions per aquest tema. Prova d'això és el desenvolupament d'un programa informàtic, anomenat HeCaSe [1], que proporciona un conjunt de serveis a l'usuari, com poden ser gestió de visites, consultes d'expedients, etc., i també ofereix serveis al metge, com gestió de proves mèdiques, gestió de pacients i gestió de guies de pràctica clínica, entre altres.

El present PFC té com a objectiu principal la millora del sistema de guies de pràctica clínica de manera independent, per tal d'aportar un sistema alternatiu que ajudi, per exemple, un metge a prendre una decisió el més òptima possible, és a dir, que disposi de tota la informació necessària per prendre un diagnòstic el més precís possible.

Si analitzem el concepte de guies de pràctica clínica, a partir d'ara GPC, tenim que és un tema bastant recent, però podem definir una GPC com un conjunt de pautes (proves, resultats, estats...) a seguir i realitzar per al tractament d'una malaltia.

Actualment hi ha centres de recerca que treballen conjuntament amb centres i institucions mèdiques per l'elaboració de noves GPC. Aquestes GPC finalment són publicades per tal que altres institucions mèdiques les puguin avaluar i fins i tot adoptar. Avui en dia podem trobar diverses Webs que donen informació actualitzada de GPC de nova creació.

Però com qualsevol cosa que es realitza de forma descentralitzada (per exemple hi ha institucions privades que desenvolupen les seves pròpies GPC), pot arribar a ser un autèntic caos si no s'estableixen unes regles de creació per a les GPC, sobretot des del punt de vista computacional. La idea de crear un estàndard de representació és peça clau per tal que les aplicacions siguin homogènies i compatibles entre elles, ja que finalment es pretén tenir una solució global per a tothom. La hipertensió és una malaltia igual aquí que a la Xina, per tant un programa ha de ser capaç de tractar una GPC de qualsevol país. D'aquí sorgeix la idea de crear un llenguatge per modelar i mostrar tot el coneixement que representa la informació en una GPC. En el cas del present PFC es treballarà amb el llenguatge **PROforma**.

1.1. Antecedents

El present projecte sorgeix com a complement a una part del projecte “Disseny i implementació d’un SMA (Sistema Multi-Agent) per proporcionar serveis mèdics” que desenvolupà David Isern [2] dins del grup de recerca *GRUSMA*.

En principi, el projecte proporcionava serveis mèdics a persones, i també podíem trobar altres entitats que interactuaven amb el sistema, com podien ser doctors, expedients mèdics, etc.; era un sistema complet per al qual es va crear una estructura jeràrquica d’agents.

A partir del projecte, juntament amb aportacions d’altres projectes, es va desenvolupar l’eina *HeCaSe* (Health Care Services), una aplicació basada en agents que ofereix serveis mèdics. Incorpora agents (entitats autònomes de software) que contenen informació mèdica i implementen comportaments sobre un conjunt d’entitats, que són: *centres mèdics, departaments i doctors*. Aquests agents tenen capacitat de coordinació entre ells per tal de simular la gestió organitzadora en un centre mèdic.

Una de les característiques que oferia el sistema era referent a la manipulació de guies de pràctica clínica que utilitzaven els agents doctors per tal de fer el seguiment d’un determinat pacient. En aquest apartat el sistema realitzava una simulació de l’ús d’aquestes GPC’s, tot i que estaven presents en la Base de Dades. Per tant, era necessària la creació d’algun tipus d’eina que ajudés el “doctor” de forma bastant visual a representar una GPC.

A partir d’aquí, es concreta la idea de crear un projecte que, partint d’un entorn Web, ens permeti la visualització d’una GPC en forma de graf, per tal que de forma còmoda i visual s’identifiquin els objectes principals de la GPC, així com establir una navegació pels diferents nivells de la guia i les relacions que s’estableixen entre ells. També s’incorpora un mecanisme de validació d’usuaris per la manipulació de les GPC, per tal de permetre en un futur afegir-hi noves funcionalitats d’accés, per exemple, establiments de rols per accedir a determinades GPC, etc.

En el capítol 6 d’aquest document, s’hi troben algunes propostes que complementarien el sistema.

1.2. Objectius

El principal objectiu d'aquest projecte és la creació d'una eina que permeti la manipulació de guies de pràctica clínica, per exemple, per un doctor o altres entitats, com pot ser un agent.

La manipulació de la Gpc, per exemple per part d'un doctor ha de ser fàcil, que es pugui trobar ràpidament la informació, que no li suposi un treball extra comprendre la guia, que s'identifiquin ràpidament les tasques (com, per exemple, realitzar una radiografia) a partir de colors, formes, etc., que les precondicions i poscondicions de cada estat d'una tasca siguin visibles en tot moment, així com la informació pròpia de la tasca.

Les tasques a realitzar per arribar a l'objectiu final són les següents:

- ✚ *Anàlisi de les diferents eines gràfiques disponibles:* aquest apartat tractat en el capítol 3, ha estat un dels punts més laboriosos del projecte, ja que a part d'escollir una eina on poder representar un graf dirigit, s'ha realitzat un estudi de les eines que actualment trobem per a la manipulació de grafs en particular, o qualsevol estructura en general com arbres, organigrames, etc. S'havien d'estudiar alguns paràmetres importants, per exemple, si l'eina gràfica oferia prou documentació per realitzar el projecte; també era important si permetia interacció (per exemple, amb l'ús d'events) amb l'usuari, ja que moltes eines gràfiques només pinten el graf; també eren interessants paràmetres d'aprenentatge, escalabilitat, opcions per defecte incorporades (per exemple, zooms, rotacions, ...), si era gratuït, llenguatge de programació utilitzat, etc.
- ✚ Al tractar-se d'una aplicació web s'havia de crear una interfície web que donés resposta a les necessitats de l'usuari; per tant, vaig escollir un disseny simple que, a la vegada, sigués útil i pràctic, sense entrar en un disseny web molt carregat i massa vistós.
- ✚ Creació d'un sistema de validació per part de l'usuari, a partir d'una clau d'accés (login) i una contrasenya, per tal d'accedir al visualitzador de GPC. Aquest punt s'ha incorporat pensant en el futur, ja que interessarà més endavant poder identificar usuaris i establir diferents tipus d'accés sobre el mateix contingut de forma fàcil. En aquest apartat també es va fer un estudi rigorós, ja que no volia un sistema típic basat en login i contrasenya; es va decidir finalment per l'ús de les llibreries JAAS que ofereixen serveis interessants, com s'explica en el capítol 4.2.
- ✚ Un altre objectiu que es proposava era la interacció que hi havia entre el JSP i l'Applet, és a dir, parametritzar l'Applet a partir d'uns valors que li passava el JSP de forma dinàmica. La tasca de relacionar JSP i Applet no va ser fàcil.
- ✚ Disseny i creació de la Base de Dades per a la manipulació dels usuaris i guies de pràctica clínica.
- ✚ Sistema de navegació en el graf que representa la GPC. D'aquesta manera es permet a l'usuari generar noves vistes (sub-nivells) d'una part de la GPC; tot això a partir d'una barra de navegació i les operacions associades als nodes.

2. GUIES DE PRÀCTICA CLÍNICA

Una guia de pràctica clínica és un conjunt d'instruccions, directrius, afirmacions o recomanacions, desenvolupades de manera sistemàtica, amb el propòsit d'ajudar a metges i pacients a prendre decisions sobre la modalitat d'assistència sanitària apropiada per a unes circumstàncies clíniques específiques.

2.1. Descripció del llenguatge PROforma.

PROforma és un llenguatge que permet modelar aplicacions dintre de l'àmbit dels sistemes de suport de decisions, guies i altres aplicacions clíniques, d'aquesta manera podem arribar a representar guies i protocols clínics en un format executable per a un ordinador convencional. Actualment es fa recerca sobre la representació de guies mèdiques i és un camp en contínua evolució. Sistemes de representació com és 'Arden Syntax' ens permeten trobar regles de condició-acció molt utilitzades en el camp de la indústria actual.

Molts són els llenguatges que modelen aquestes guies a partir de regles lògiques i estructures complexes per representar tasques en forma de possibles decisions, plans o accions a realitzar. Alguns exemples de llenguatges són Asbru, EON, GLIF, GUIDE, PRODIGY i PROforma.

La primera implementació software per a la creació, visualització i representació de guies en PROforma va ser **Arezzo**. Actualment és un producte comercial de l'empresa *InferMed* a Londres. Arezzo està dissenyat per a entorns Windows.

Una altra implementació destacada és **Tallis**, realitzada en el llenguatge de programació Java per al centre de recerca del càncer en UK. Tallis està basada en l'última versió de PROforma i permet la creació, edició i visualització gràfica de les guies. També permeten ser representades perquè altres aplicacions la utilitzin. Tallis possibilita la interacció amb serveis Web i és una aplicació multiplataforma.

2.1.1. Model PROforma

En el model PROforma, una guia es modela a partir d'un conjunt de **tasques i dades**; les tasques s'organitzen jeràrquicament formant **plans**.

El model PROforma divideix les tasques en quatre classes, detallades a continuació i representades en la figura 1:

- *Accions (Actions)*: són procediments que s'executen en un entorn extern, per exemple, l'administració d'un fàrmac, o l'actualització d'una base de dades.
- *Preguntes (Enquiry)*: representen parts de la guia amb informació necessària per una persona o una entitat externa.

- *Decisions (Decisions)*: representen parts de la guia en què s'ha de prendre una elecció entre varies opcions.
- *Plans (Plans)*: són una col·lecció de tasques agrupades per alguna raó, per exemple, perquè comparteixen un mateix objectiu, o bé utilitzen un mateix recurs, etc.

En PROforma, el procés es representa a partir d'un diagrama d'un graf dirigit on cada node representa una tasca i cada aresta representa una restricció associada als nodes que intervenen. En aquest projecte s'ha escollit el mateix sistema de representació de cada ítem, per tant es representen els plans mitjançant rectangles arrodonits, les decisions com a cercles, les preguntes com a rombes i les accions com a quadrats.

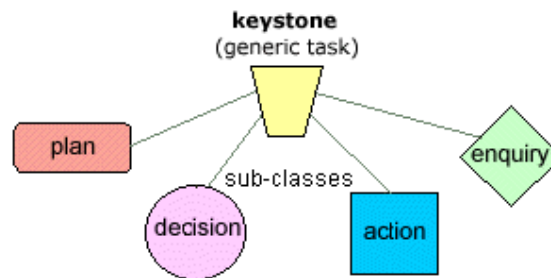


Figura 1: Jerarquia de classes en PROforma

Cada tasca conté un conjunt de propietats que identifiquen quina és la interpretació que té aquesta tasca. El valor d'aquestes propietats pot ser per exemple un valor escalar, i també es pot tractar d'una expressió o, fins i tot, un objecte que a la vegada conté diverses propietats.

A continuació es detallen les propietats més importants, comunes a qualsevol tasca:

- **Caption i descripció**: aquesta propietat esdevé obligatòria per cada tasca i descriu el propòsit de la tasca, o simplement, a nivell de comentari.
- **Precondició**: defineix una expressió lògica que ha de ser certa quan la tasca s'inicia. (Per exemple, $\text{Temperatura} < 35^{\circ}\text{C}$)
- **Restriccions (constraints)**: defineix restriccions lògiques que prevenen que una tasca s'iniciï abans que una altra, o un conjunt d'elles, fins que aquesta no estigui del tot completada.

Tot seguit es detallen algunes de les propietats que conté una tasca "decisió":

- **Candidats (Candidates)**: són objectes que representen opcions a ser considerades quan s'ha pres la decisió. Aquests objectes a la vegada contenen propietats anomenades *regles de recomanació*.
- **Arguments**: consisteix en una expressió lògica associada a cada candidat.

I algunes propietats de la tasca “pla” són:

- Condicions de finalització o cancel·lació: són expressions lògiques que representen condicions suficients (però no necessàries) per finalitzar la tasca “pla” de manera correcta i continuar amb les següents tasques, també per cancel·lar la tasca “pla” i les tasques successores al pla.

2.2. Descripció del sistema PROforma

2.2.1. La sintaxi PROforma

La sintaxi de PROforma segueix el model BNF (**B**ackus **N**aur **F**orm) que es divideix en dues parts. La sintaxi de les expressions en PROforma, que defineix les condicions lògiques i expressions matemàtiques que s'utilitzen, i per l'altra banda, com les definicions de les tasques i altres components de la guia són unides i separades. En l'especificació de PROforma hi ha algorismes que infereixen en les expressions i imposen propietats amb valors de restricció, per exemple per a una *precondició*.

2.2.2. Els components de PROforma

En la semàntica de PROforma es tracten les guies com a un conjunt d'objectes; aquests tipus d'objectes se'ls anomena els components de PROforma. Les classes que implementen són utilitzades per la creació d'aquests objectes que formen una estructura jeràrquica com es mostra a la figura 2.

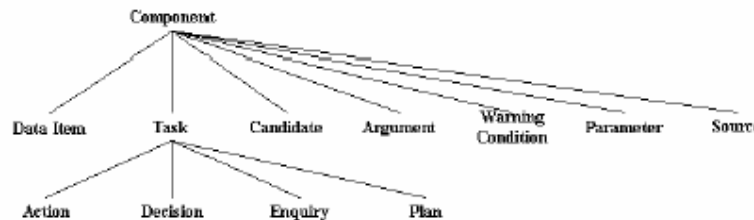


Figura 2: Components PROforma

Com s'ha comentat anteriorment, cada classe conté diferents tipus de propietats, i cada classe hereta totes les propietats de la superclasse. Una instància d'un objecte conté

valors per cada una de les propietats de la classe. Per exemple, quan afegim una instància de la classe *Task* a la guia, s'han d'especificar els valors de cada una de les seves propietats de la tasca, però també per les propietats que hereta de la classe *Component*, com poden ser la propietat “Nom” i la propietat “Caption”. Cal comentar que cada instància d'un objecte *Component* té un únic identificador.

A continuació es detallen alguns dels valors que pot tenir una propietat en PROforma:

- Un valor numèric. (enter o real)
- Una cadena de text. (string)
- Una expressió en PROforma.
- Un identificador d'un component.
- Constants definides en PROforma, com poden ser: *dormant*, *in_progress*, *discarded*, i *completed*.
- Seqüències de valors entre claus, per exemple: {1,2,3,4,5}

2.2.3. Estat d'una tasca PROforma

Les tasques en PROforma tenen una propietat que indica l'estat en què es troba la tasca en qüestió. Els valors que pot prendre aquesta propietat són: *dormant*, *in_progress*, *discarded* i *completed*. La semàntica de PROforma no imposa una interpretació per cadascun dels estats d'una tasca, però normalment el significat de cada estat és:

- ✓ *Dormant*: la tasca no s'ha iniciat, i tampoc és possible saber quan s'iniciarà.
- ✓ *In_progress*: la tasca ja ha començat.
- ✓ *Discarded*: es dona quan la lògica de la guia indica que la tasca no s'iniciarà o perquè la tasca no s'ha completat.
- ✓ *Completed*: la tasca s'ha realitzat.

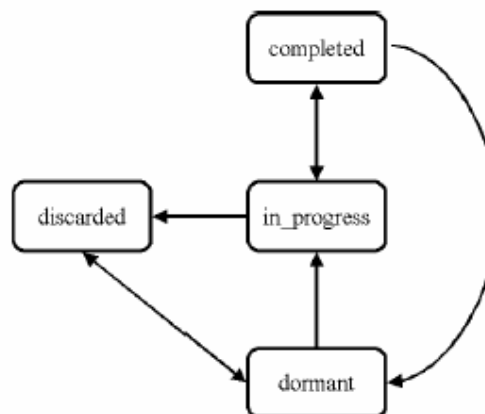


Figura 3: Transicions en l'estat d'una tasca

En la figura 3 es mostra l'evolució que pot seguir una tasca durant la seva execució. El fet que hi hagin transicions de sortida des dels estats *completed* i *discarded*, es produeixen perquè durant l'evolució dintre d'un pla, aquests estats es poden repetir de manera cíclica varies vegades.

2.3. Exemple d'una GPC en PROforma

2.3.1. Representació de la GPC

A continuació es representarà gràficament la guia mèdica referent a la **Hipertensió**. En l'annex d'aquest pfc es pot veure el fitxer de text que representa la guia. Mitjançant l'ús del visualitzador d'aquest PFC es mostra la guia resultant en la Figura 4, que ens mostra el pla general que forma la guia. La descripció de cada tasca es detalla a continuació:

- + *Comorbidities*: es tracta d'un estat de pregunta.
- + *Current therapy (Teràpia actual)*: estat de pregunta.
- + *Target BP decision*: es tracta d'una decisió a prendre.
- + *Additional drug choice (Escollir un medicament addicional)*: es tracta d'una decisió a prendre.
- + *Prescribe additional drug (Prescriure el medicament addicional)*: es tracta de realitzar l'acció.
- + *Assess Response (Determinar una resposta)*: es tracta d'un nou pla. Arribat en aquest punt, els següents passos descriuen el subpla associat.(Figura 5)
 - *Await adequate response (Espera una resposta adequada)*: es tracta de realitzar l'acció associada.
 - *Monitor BP*: es tracta d'un estat de pregunta.

El major requeriment d'aquesta guia està en modelar la decisió d'afegir un medicament alternatiu (Estat de decisió). En el model PROforma, un estat de decisió té els següents components:

- Un nombre variable de *Candidats*.
- Un nombre variable d'*arguments* relacionat amb cada *candidat*.
- Un *mode de suport*, que descriu com s'han d'agregar els arguments.
- Una regla de recomanació per cada Candidat, que descriu sota quines circumstàncies el candidat es considera recomanable.

La decisió utilitza un mode de suport simbòlic, per tant, cada argument pot prendre els següents valors: *for*, *against*, *confirming*, *excluding*, afectant, per tant, al candidat. En la *taula 1* es descriuen els arguments i els candidats.

- Si es compleixen les indicacions de caràcter obligat, tenim un nivell de suport a “**confirming**”.
- Si es compleixen les contraindicacions, tenim un nivell “**excluding**”.
- Si es compleixen les indicacions i contraindicacions relatives, tenim un nivell de suport per a “**for**” i “**against**”.
- Si es compleixen combinacions que s’havien d’evitar, tenim un nivell de suport a “**excluding**”.

Per al mode de suport simbòlic que s'utilitza, per cada candidat els valors amb nivell “for” sumen, i els valors amb nivell “against” resten. A la vegada, els valors amb nivell “confirming” són acceptats notablement, sempre i quan no hi hagi un valor de nivell “excluding”, d’aquesta manera s’obté un valor numèric que ajuda a la presa d’una decisió òptima.

Per altra banda, PROforma defineix uns objectius (*goals*) associats als nodes pla. Quan un pla és troba en un estat de “in progress” (en procés) llavors es produeix que el valor del “goal” (que és una expressió lògica) és certa, i la tasca finalment passarà a un estat de “performed”. En aquest cas, les subtasques associades al pla hauran d’estar finalitzades.

En aquest model tenim el pla “Assess Response” amb el següent objectiu:

$$\text{DBP} < \text{targetDBP and SBP} < \text{targetSBP}$$

El pla segueix un model cíclic, és a dir, es repeteix contínuament la tasca “Await adequate response” i “Monitor BP” fins que **s’aconsegueix l’objectiu**.

Taula 1:

Candidate	Argument	Support level
ACEI		
	Current_Rx = ACEI	Excluding
	Pregnancy = Yes	Excluding
	Type_1_Diabetes = Yes and Proteinuria = Yes	Confirming
	Heart_failure = Yes	Confirming
	MI = Yes and Systolic_dysfunction = Yes	Confirming
	Current_Rx = K_sparing_diuretic	Excluding
ISA_beta_blocker		
	Asthma = Yes or COPD = Yes	Excluding
	Heart_block = Yes	Excluding
	Type_1_Diabetes = Yes and Proteinuria = Yes	Against
	Current_Rx = Ca_blocker_non_DHP	Excluding
	Current_Rx = Thiazide_diuretic	For
	Current_Rx = Ca_blocker_DHP_long or Current_Rx = Ca_blocker_DHP_short	For
	Current_Rx = Beta_blocker_ISA or Current_Rx = Beta_blocker_non_ISA	Excluding
Non_ISA_beta_blocker		
	Asthma = Yes or COPD = Yes	Excluding
	Heart_block = Yes	Excluding
	MI = Yes	Confirming
	Type_1_Diabetes = Yes and Proteinuria = Yes	Against
	Current_Rx = Ca_blocker_non_DHP	Excluding
	Current_Rx = Thiazide_diuretic	For
	Current_Rx = Ca_blocker_DHP_long or Current_Rx = Ca_blocker_DHP_short	For
	Current_Rx = Beta_blocker_non_ISA or Current_Rx = Beta_blocker_ISA	For
Thiazide_diuretic		
	Isolated_Sys_hypertension = Yes and Age_group = Older	Confirming
	Current_Rx = Beta_blocker_ISA or Current_Rx = Beta_blocker_non_ISA	For
	Current_Rx = Thiazide_diuretic or Current_Rx = K_sparing_diuretic or Current_Rx = Loop_diuretic	Excluding
Loop_diuretic		
	Heart_failure = Yes	Confirming
	Isolated_Sys_hypertension = Yes and Age_group = Older	Confirming
	Current_Rx = Thiazide_diuretic or Current_Rx = K_sparing_diuretic or Current_Rx = Loop_diuretic	Excluding
K_sparing_diuretic		
	Isolated_Sys_hypertension = Yes and Age_group = Older	Confirming
	Current_Rx = ACEI	Excluding
	Current_Rx = Thiazide_diuretic or Current_Rx = K_sparing_diuretic or Current_Rx = Loop_diuretic	Excluding

Candidate	Argument	Support level
Ca_channel_dihydro_short		
	Isolated_Sys_hypertension = Yes and Age_group = Older	Confirming
	Type_1_Diabetes = Yes and Proteinuria = Yes	For
	Current_Rx = Ca_blocker_non_DHP	Excluding
	Current_Rx = Beta_blocker_ISA or Current_Rx = Beta_blocker_non_ISA	For
	Current_Rx = Ca_blocker_DHP_long or Current_Rx = Ca_blocker_DHP_short or Current_Rx = Ca_blocker_non_DHP	Excluding
Ca_channel_dihydro_long		
	Isolated_Sys_hypertension = Yes and Age_group = Older	confirming
	Type_1_Diabetes = Yes and Proteinuria = Yes	For
	Current_Rx = Ca_blocker_non_DHP	Excluding
	Current_Rx = Beta_blocker_ISA or Current_Rx = Beta_blocker_non_ISA	For
	Current_Rx = Ca_blocker_DHP_long or Current_Rx = Ca_blocker_DHP_short or Current_Rx = Ca_blocker_non_DHP	Excluding
Ca_channel_non_dihydro		
	Heart_block = Yes	Excluding
	Current_Rx = Beta_blocker_ISA or Current_Rx = Beta_blocker_non_ISA	Excluding
	Current_Rx = Ca_blocker_DHP_long or Current_Rx = Ca_blocker_DHP_short	Excluding
	Type_1_Diabetes = Yes and Proteinuria = Yes	For
	Current_Rx = Ca_blocker_DHP_long or Current_Rx = Ca_blocker_DHP_short or Current_Rx = Ca_blocker_non_DHP	Excluding

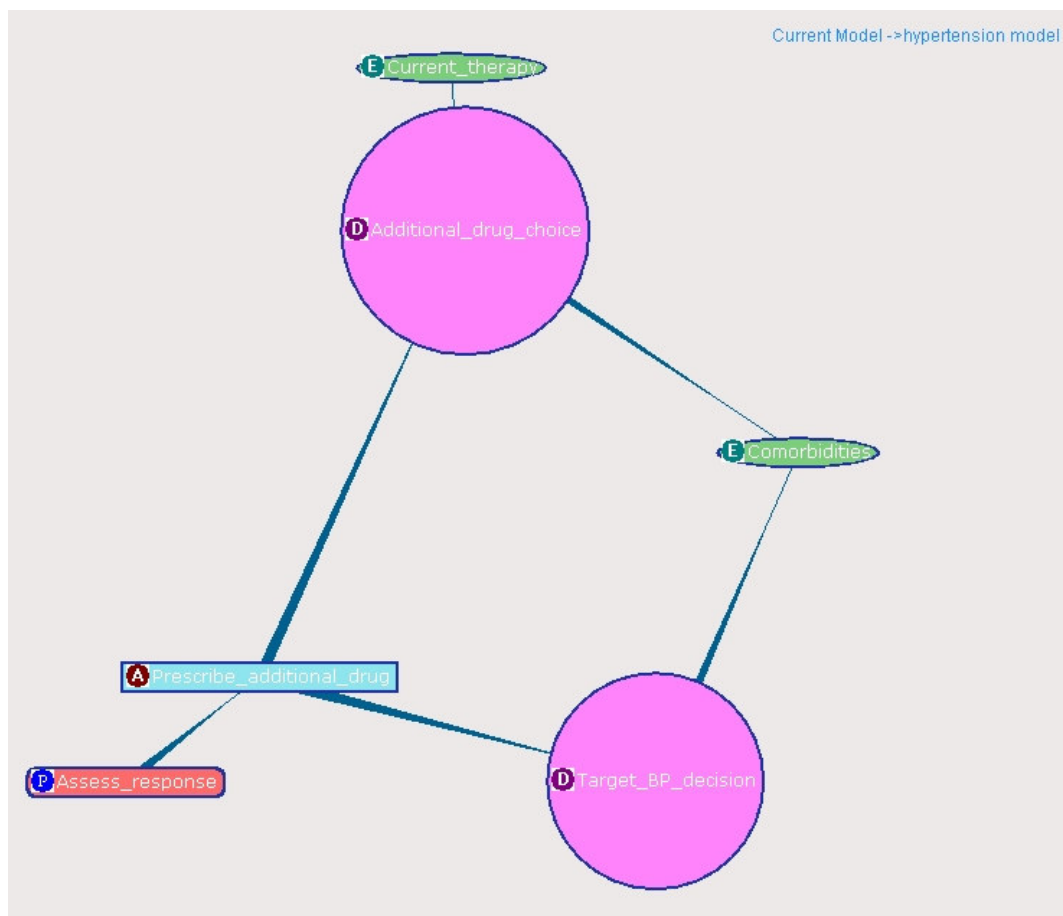


Figura 4: Representació de la GPC Hipertensió

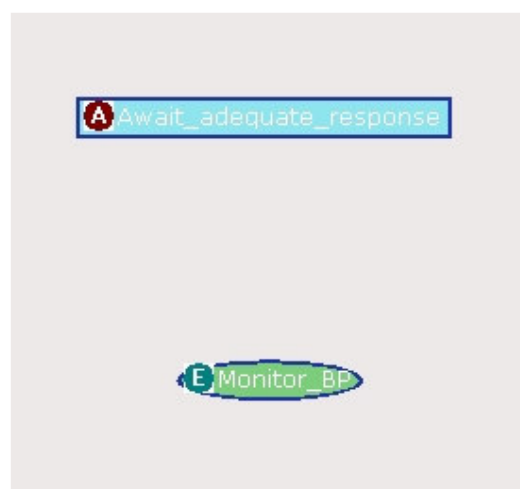


Figura 5: Expansió del Pla: Assess_response

3. EINES GRÀFIQUES

En la implementació del present pfc, la part gràfica juga un paper destacat , ja que serà utilitzada per la representació de la guia de pràctica clínica, per tant trobar l'eina apropiada per a desenvolupar aquesta tasca serà molt important. Per tant es realitzarà un estudi de les eines actuals que permeten la creació i manipulació d'estructures en graf de forma fàcil i eficient.

L'eina gràfica finalment adoptada haurà de poder generar grafs dirigits , permetre la interacció entre l'usuari i el graf de forma fàcil (per exemple amb l'ús d'events), dibuixar formes i colors necessàries per a la bona representació del graf, així com una bona distribució del graf en l'entorn de dibuix i operacions sobre el graf com per exemple ampliar el graf, rotar-lo , etc.

Per tal d'analitzar el bon ús de l'eina gràfica sobre el projecte, s'avaluaran un conjunt de paràmetres que s'han de satisfer per tal d'arribar al objectiu final de representació de la guia. Per tant, s'estudiaran paràmetres referents al cost de l'eina gràfica, l'arquitectura que segueix la llibreria, si permet interacció amb l'usuari, afegir noves funcionalitats (extensibilitat), etc... aquests paràmetres juntament amb els resultats obtinguts es representen en la taula comparativa present en la secció 3.8.1. Les eines comentades en aquesta secció les poden classificar en un primer paràmetre segons siguin:

Eines no comercials:

- *JGraph*
- *GraphViz*
- *TouchGraph*
- *JOpenChart*
- *JFreeChart*

Eines comercials:

- *JGo: GoDiagram for Java*
- *ILOG JViews Diagrammer*

A continuació es detallen cadascuna d'elles.

3.1. JGraph

Actualment les eines gràfiques a Internet són molt importants, des del punt de vista del que l'usuari percep. La facilitat que donen les llibreries per crear diferents formes gràfiques fa imprescindible l'ús en qualsevol aplicació Web.

Les llibreries gràfiques ofereixen components per poder treballar sobre diferents objectes, com poden ser llistes, taules, arbres, però la representació de grafs és encara un sistema en desenvolupament.

JGraph es presenta com un component sobre la llibreria gràfica Java Swing que permet la visualització i edició de grafs. La llibreria està basada en la teoria de Grafs i la llibreria gràfica Swing de Java, que defineix l'arquitectura de JGraph.

Els principis que defineixen JGraph són els següents:

- Comptabilitat amb Swing.
- Disseny clar i eficient.
- Desenvolupament en Java.

En la següent figura es mostren uns exemples creats en JGraph. Representen les diferents tipologies de graf que es poden representar amb aquesta eina, des de xarxes de nodes a arbres n-aris passant per diagrames de decisió.

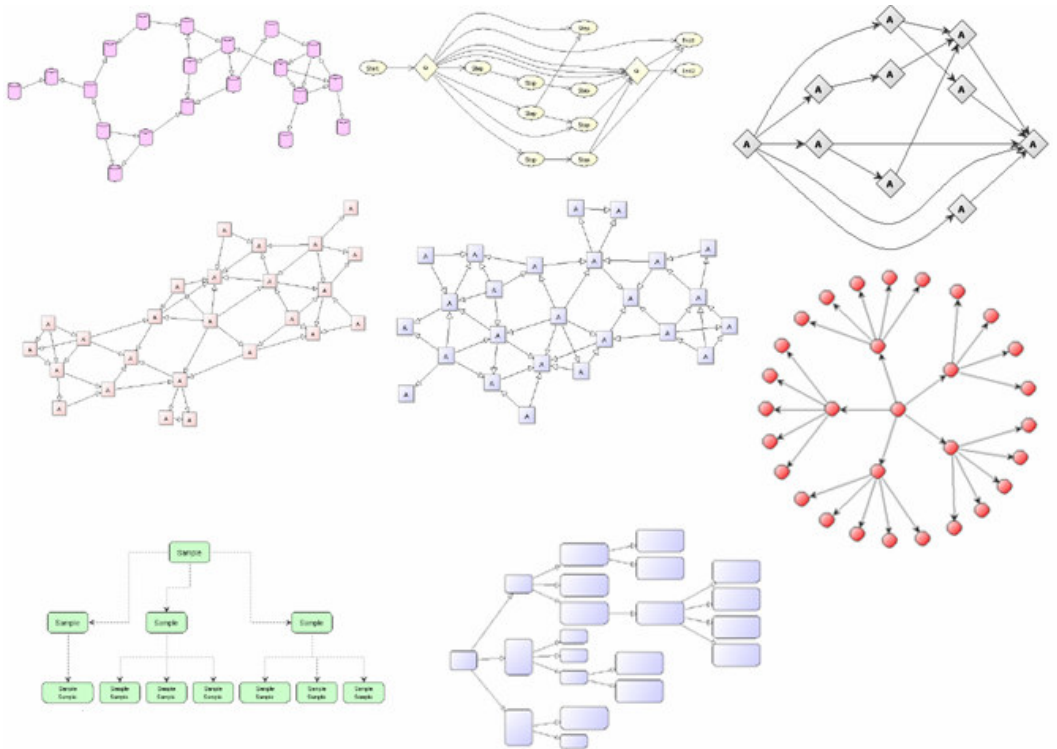


Figura 6 : Exemples amb JGraph.

3.1.1. Arquitectura JGraph

El nucli que forma la llibreria JGraph està basat en el model **MVC** (Model-Vista-Control) que implementa. El *model* representa les dades i els objectes JGraph donen la *vista* de les dades (Figura 7).

El patró de disseny MVC de Swing s'utilitza per separar el model de la vista i establir relacions entre ells; per exemple, una vista pot estar relacionada amb el mateix model i diverses vistes es poden modificar automàticament quan es produeixin canvis en el model.

JGraph compleix amb l'estàndard de Swing, accessibilitat, internacionalitat, transmissió de dades, serialització, etc., i per tal de realitzar millores, com, per exemple, sistemes d'impressió, suport *XML*, ..., s'han utilitzat altres estàndards.

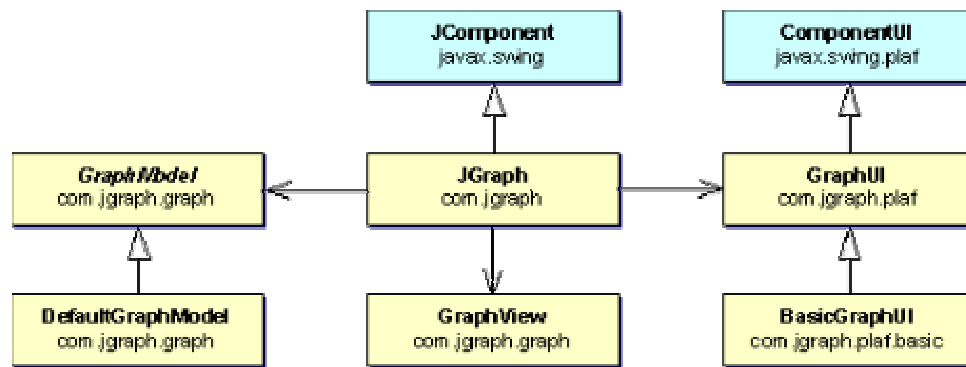


Figura 7 : Esquema MVC de JGraph.

3.1.1.1. Model JGraph (MVC)

El model representa les dades del graf, que consisteix en un conjunt de cel·les que estan formades per vèrtexs, arestes i informació relacionada amb la connectivitat entre cel·les.

Les **cel·les** (*Graph cells*, Figura 8) són la clau principal en JGraph; aquesta presenta per defecte implementacions en forma de vèrtexs (implementació per defecte), arestes i ports.

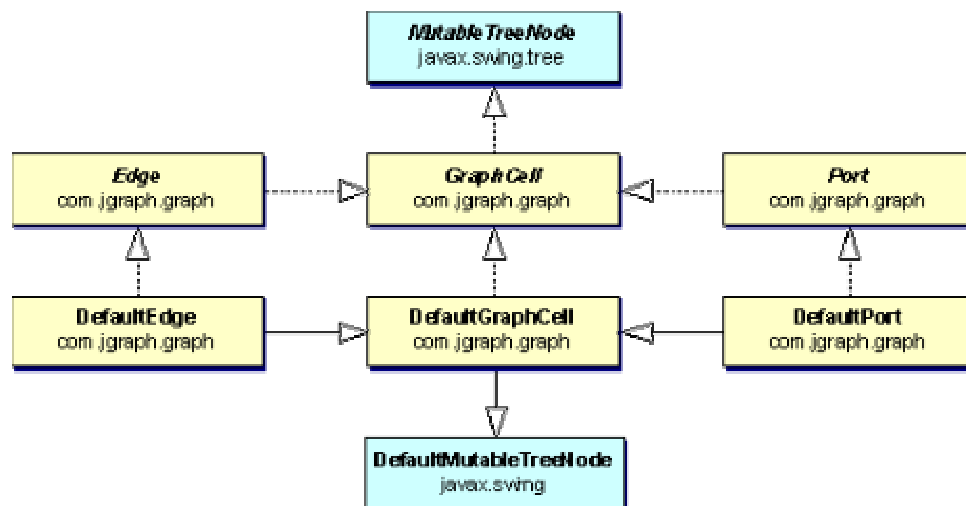


Figura 8 : Jerarquia de l'interfície GraphCell

JGraph ha modificat part del model per tal de millorar el seu ús i crear noves funcionalitats. Per exemple, alguns canvis que presenta el model són els següents: insercions (*Inserts*), eliminacions (*Remove*) i modificacions (*Change*), exemple Figura 9.

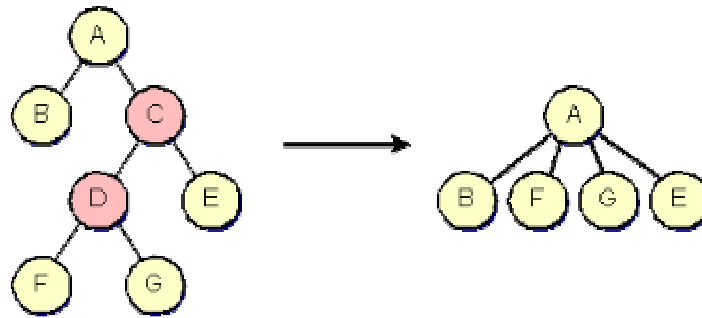


Figura 9 : Exemple d'eliminació del node C i D

3.1.1.2. Vista JGraph (MVC)

La vista en JGraph és sensiblement diferent a la vista que implementen alguns objectes de la llibreria Swing. La diferència principal és que en JGraph la vista manté l'estat (figura 10), és a dir, manté informació continguda en la pròpia vista. Per tal de realitzar aquest procés s'utilitzen les instàncies de classe **GraphLayoutCache** i **CellView**.

L'objecte *GraphLayoutCache* manté les vistes de les cel·les; aquestes vistes es guarden en dues estructures.

- ✚ 1 array amb vistes de ports.
- ✚ 1 llista que manté vistes de les cel·les arrel.

Per tal de mapejar les cel·les amb les seves corresponents vistes, JGraph implementa taules de Hash.

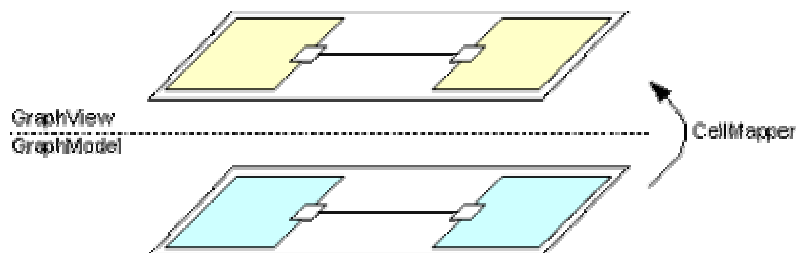


Figura 10 : Separació entre capes del model i la vista

3.1.1.3. Control JGraph (MVC)

El control defineix la relació que hi ha entre els events que genera l'usuari i els mètodes del graf o seleccions sobre el model o vista. Per aconseguir-ho JGraph utilitza el **UI-Delegate**.

La classe abstracta *GraphUI* i la seva implementació per defecte *BasicGraphUI* formen el que s'anomena UI-Delegate. Les seves funcions principals són la de pintar el graf, modificar la finestra quan el graf canvia, etc. S'ha de tenir en compte que quan el graf es pinta, el UI-delegate recorre les vistes i pinta les que correspongui.

En la següent taula es mostren alguns dels events per defecte generats amb el teclat.

Double-click / F2	Starts editing current cell
Shift-Click	Forces marquee selection
Shift-Select	Extends selection
Control-Select	Toggles selection
Control-Drag	Clones selection
Shift-Drag	Constrained drag

3.1.2. Conclusions

JGraph és una llibreria gràfica molt completa implementada totalment en Java, a la vegada que implica que sigui independent del S.O. utilitzat. JGraph es troba sota llicència GPL i, per tant, és codi obert per a la comunitat de programadors, fet que ressalta encara més la fiabilitat que ofereix.

També cal destacar l'àmplia documentació que forma part de la suite JGraph, des de manuals de disseny i implementació, fins a manuals d'usuari final.

En resum, JGraph i TouchGraph són dues possibles candidates per a la implementació final d'aquest PFC. En l'apartat 3.8.2 es comenta quina opció ha estat finalment l'escollida.

La pàgina oficial de JGraph: <http://www.jgraph.org>.

3.2. GraphViz

3.2.1. Introducció

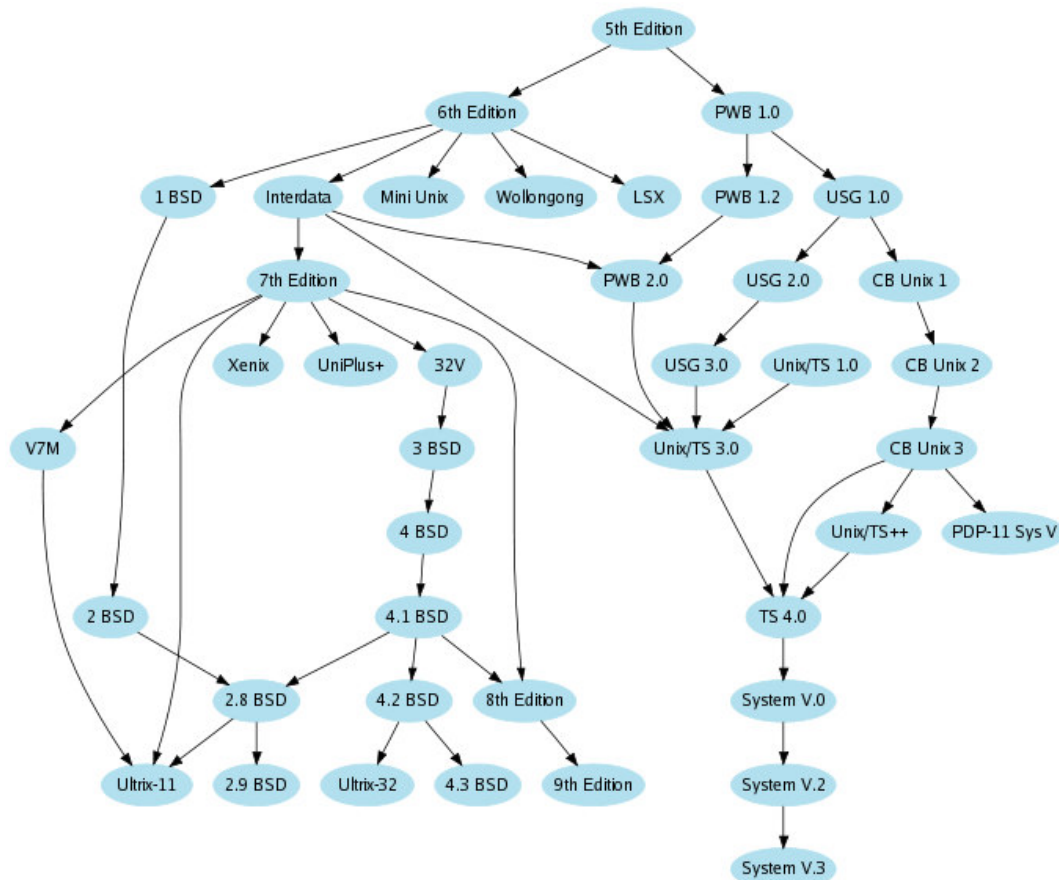
GraphViz (Graph Visualization) és una eina per a la representació estructurada d'informació, tal com diagrames, grafs abstractes i xarxes, i permet automatitzar la representació gràfica per a aplicacions del tipus d'enginyeria del software, base de dades i disseny web, xarxes informàtiques i interfícies visuals en altres dominis.

Graphviz és codi obert (Open Source, sota llicència CPL), i al voltant d'aquest projecte s'han creat moltes altres petites aplicacions complementàries. Els programes GraphViz representen la descripció d'un graf en un simple llenguatge de text i poden generar diagrames en diferents formats, com per exemple, imatges, Postscripts per incloure en pdfs, etc.; GraphViz suporta GXL (un dialecte del XML).

Alguns exemples d'aplicacions són els següents:

- Roadmaps: dot, neato and fdp, twopi, circo, ...
- Visors (Viewers): Grappa, ZGRViewer, etc. ...
- Filtres: gvpr.

En la figura 11 es mostren alguns exemples de grafs elaborats en GraphViz.



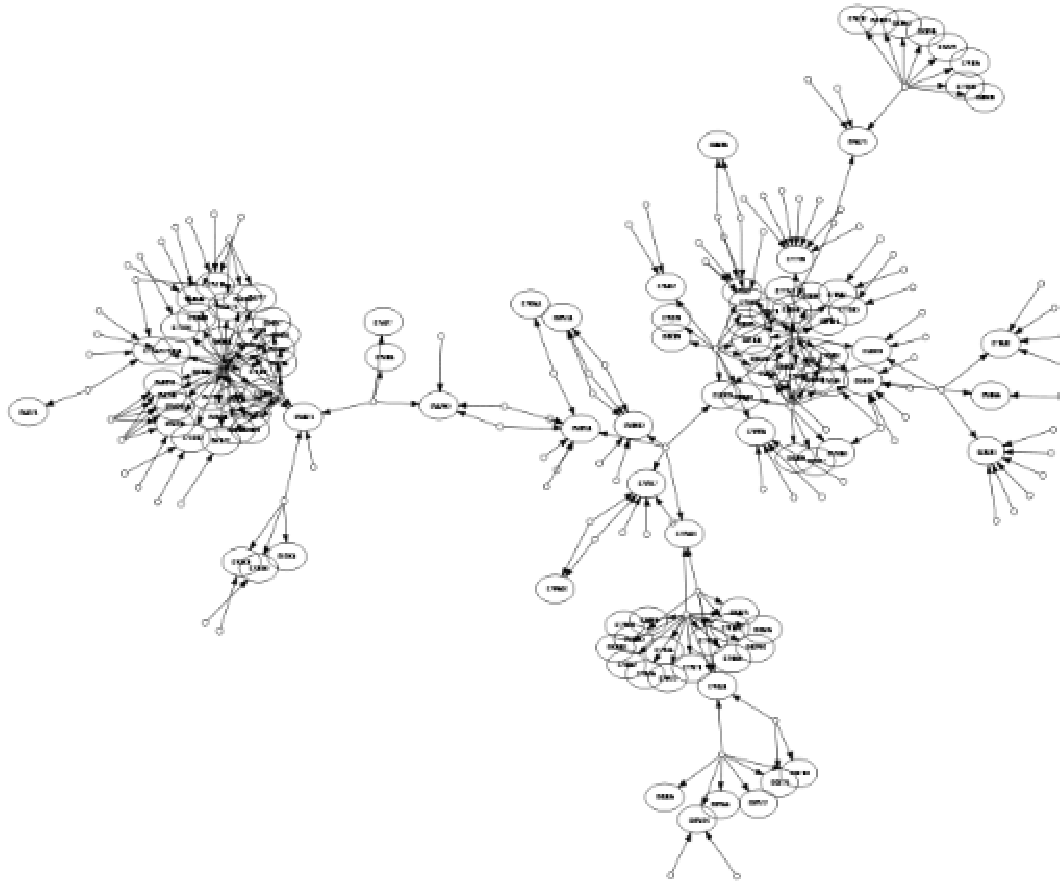


Figura 11 : Exemples en GraphViz

Per la generació d'un graf, s'ha de generar un fitxer de text que conté l'estructura del graf; a continuació es mostra un exemple:

```
digraph finite_state_machine {
    rankdir=LR;
    size="8,5"
    node [shape = doublecircle]; LR_0 LR_3 LR_4 LR_8;
    node [shape = circle];
    LR_0 -> LR_2 [ label = "SS(B)" ];
    LR_0 -> LR_1 [ label = "SS(S)" ];
    LR_1 -> LR_3 [ label = "S($end)" ];
    LR_2 -> LR_6 [ label = "SS(b)" ];
    LR_2 -> LR_5 [ label = "SS(a)" ];
    LR_2 -> LR_4 [ label = "S(A)" ];
    LR_5 -> LR_7 [ label = "S(b)" ];
    LR_5 -> LR_5 [ label = "S(a)" ];
    LR_6 -> LR_6 [ label = "S(b)" ];
    LR_6 -> LR_5 [ label = "S(a)" ];
    LR_7 -> LR_8 [ label = "S(b)" ];
    LR_7 -> LR_5 [ label = "S(a)" ];
    LR_8 -> LR_6 [ label = "S(b)" ];
    LR_8 -> LR_5 [ label = "S(a)" ];
}
```

3.2.2. Conclusió

GraphViz és una eina bastant potent per a la generació i representació de qualsevol estructura en graf. Des de la pàgina web podem descarregar molts exemples i algun manual o tutorial d'iniciació.

Des del punt de vista d'aquest projecte, el GraphViz no era una solució òptima simplement pel fet que haguem de definir un fitxer de text per la representació d'un graf; en el nostre cas els grafs evolucionen ràpidament i no ens interessa un graf estàtic. També GraphViz no és gaire clar en la interacció d'events amb el usuari, per tant va ser una opció descartada per al PFC.

3.3. TouchGraph

3.3.1. Descripció

TouchGraph és un conjunt d'interfícies per al desenvolupament d'entorns de visualització de grafs. De manera nativa TouchGraph utilitza 'spring-layout' i tècniques de focus+context per a la interacció entre l'usuari i el graf. Actualment s'hi poden trobar aplicacions que inclouen utilitats per a l'organització d'enllaços web, un entorn visual per al Wiki Browser, i un navegador gràfic per a Google que utilitza la pròpia API de Google.

TouchGraph ens permet, de manera ràpida i eficient, la representació en forma de graf d'una xarxa qualsevol i les seves interconnexions. La informació es presenta correlacionada i l'usuari pot interactuar amb el graf per tal de realitzar les transformacions que desitgi. Per exemple, l'usuari pot navegar a través del graf mostrant les relacions d'un node en concret, visualitzant la informació del node, etc.

En la figura 12 es pot veure la visualització en TouchGraph dels enllaços web de la Universitat Rovira i Virgili (www.urv.cat).

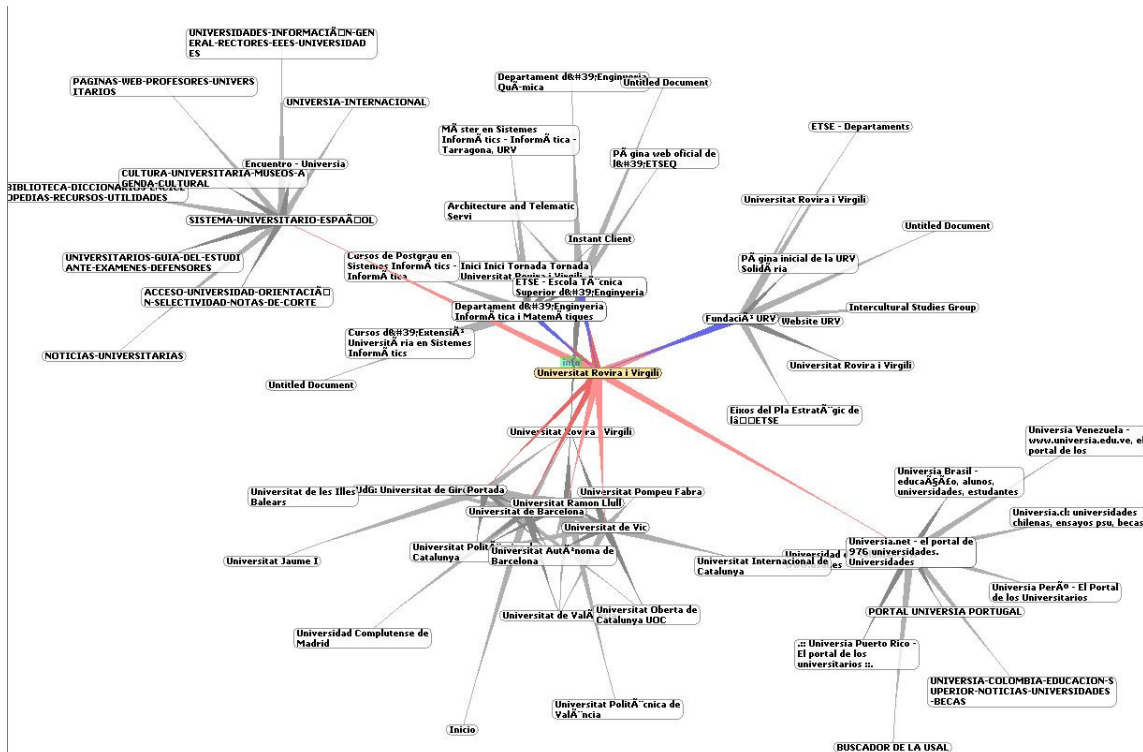


Figura 12: *Enllaços Web relacionats amb la URL www.urv.cat*

TouchGraph es va iniciar com un projecte de codi obert (Open Source) per tal que la comunitat de programadors desenvolupés el codi, i així aconseguir una òptima implementació del programa. D'aquesta manera s'aconsegueix la lliure distribució del programa i una millor depuració del codi, així com freqüents millores sobre la implementació final.

Actualment ens podem baixar les llibreries des de les següents url's:

- **Touchgraph.com**: pàgina oficial del Touchgraph, amb exemples, codi, etc.
- **Touchgraph.sourceforge.net**: pàgina per a la comunitat de desenvolupadors.

Tot i ser el projecte codi obert, trobem alguns exemples d'ús comercial de la llibreria TouchGraph, com poden ser: *InXight*, *TheBrain* i *ThinkMap*.

A partir de TouchGraph com a base inicial, s'han creat noves eines per a la implementació de noves aplicacions. Aquestes són:

- *KAON OlModeller*.
- *GraphLayout*.
- *WikiBrowser*.
- *LinkBrowser*.

3.3.2. KAON OIModeler

KAON OIModeler és una eina d'edició i manipulació d'Ontologies, creada per *Boris Motik*. Aquest nou model està basat en la llibreria GraphLayout (comentada en l'apartat 3.4.3), de manera que se li ha incorporat una arquitectura de MVC (Model Vista Control), proporcionant noves característiques i facilitant la creació de noves aplicacions.

KAON OIModeler utilitza codi de la llibreria TouchGraph però no es considera un projecte inclòs dintre dels diferents projectes que té creats TouchGraph, tot i que algunes de les noves funcionalitats que ofereix KAON han estat incorporades al codi base del TouchGraph (*Avantatges del codi obert*). En la figura 13 es pot veure un exemple de Kaon.

A continuació es detallen alguns dels nous elements incorporats.

- Desconnexió de nodes sense que es repel·lin entre ells.
- Arrossegat un node més enllà de l'àrea provoca un scrolling automàtic.
- Els nodes poden tenir múltiples botons amb accions associades, com per exemple expandir el graf.
- Realitzar múltiples seleccions.

Per a més informació, es pot consultar la web de l'autor : <http://kaon.semanticweb.org>

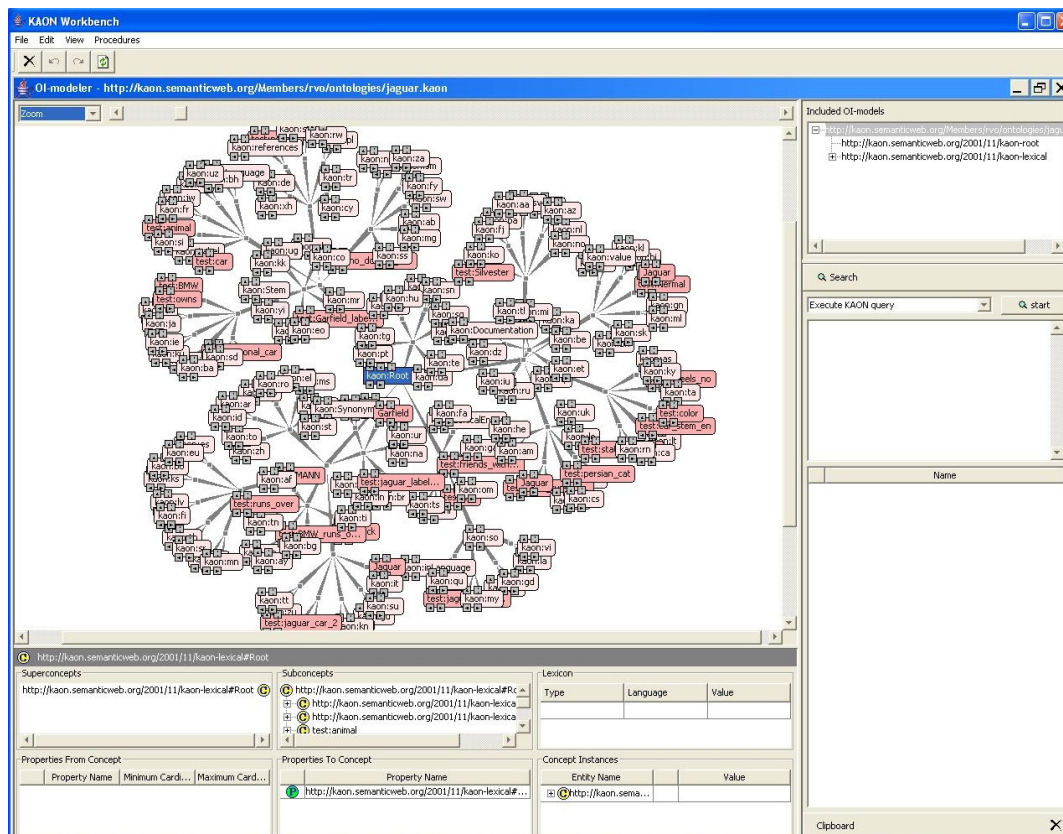


Figura 13: Ontologia sobre el jaguar i altres animals.

3.3.3. GraphLayout

La llibreria GraphLayout és el paquet bàsic de TouchGraph, i conté el codi necessari per a visualitzar, navegar i altres interaccions sobre un graf. Aquest paquet no té cap propòsit per si sol, però sí és molt útil en el cas que es vulguin implementar nous projectes que utilitzin com a context qualsevol manipulació de grafs, exemple de visualització figura 14.

Per tant, és aquest el tema principal del present projecte i és el paquet GraphLayout l'utilitza't per a la representació de guies mèdiques en aquest PFC.

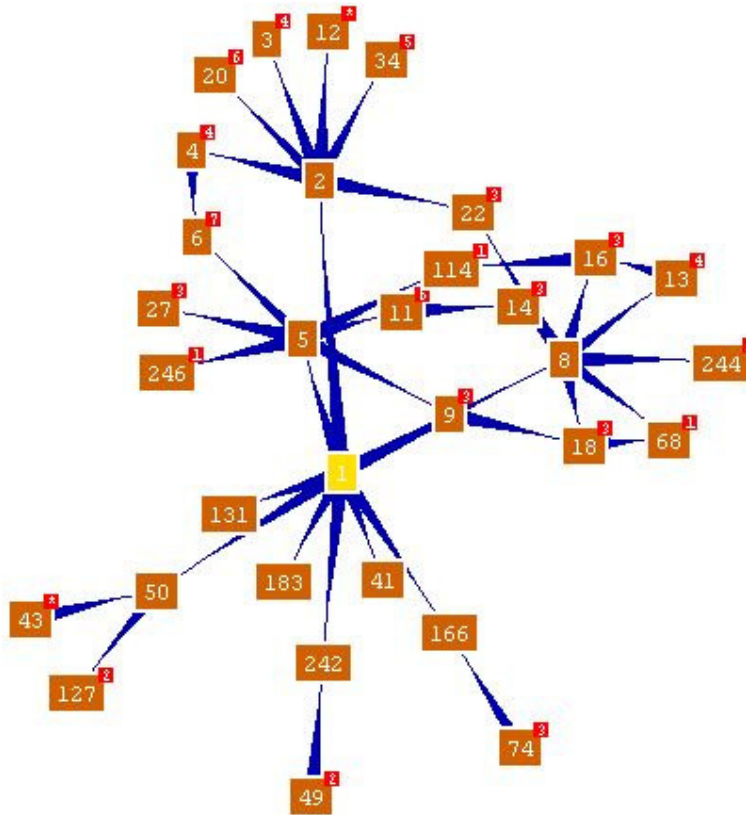


Figura 14: *Esquema general d'un graf*

Al tractar-se d'una part fonamental en el PFC, a continuació es detalla el paquet **com.touchgraph.graphlayout**.

3.3.3.1. Diagrama de Paquets

A continuació es detallen els paquets que s'han utilitzat en el present PFC.

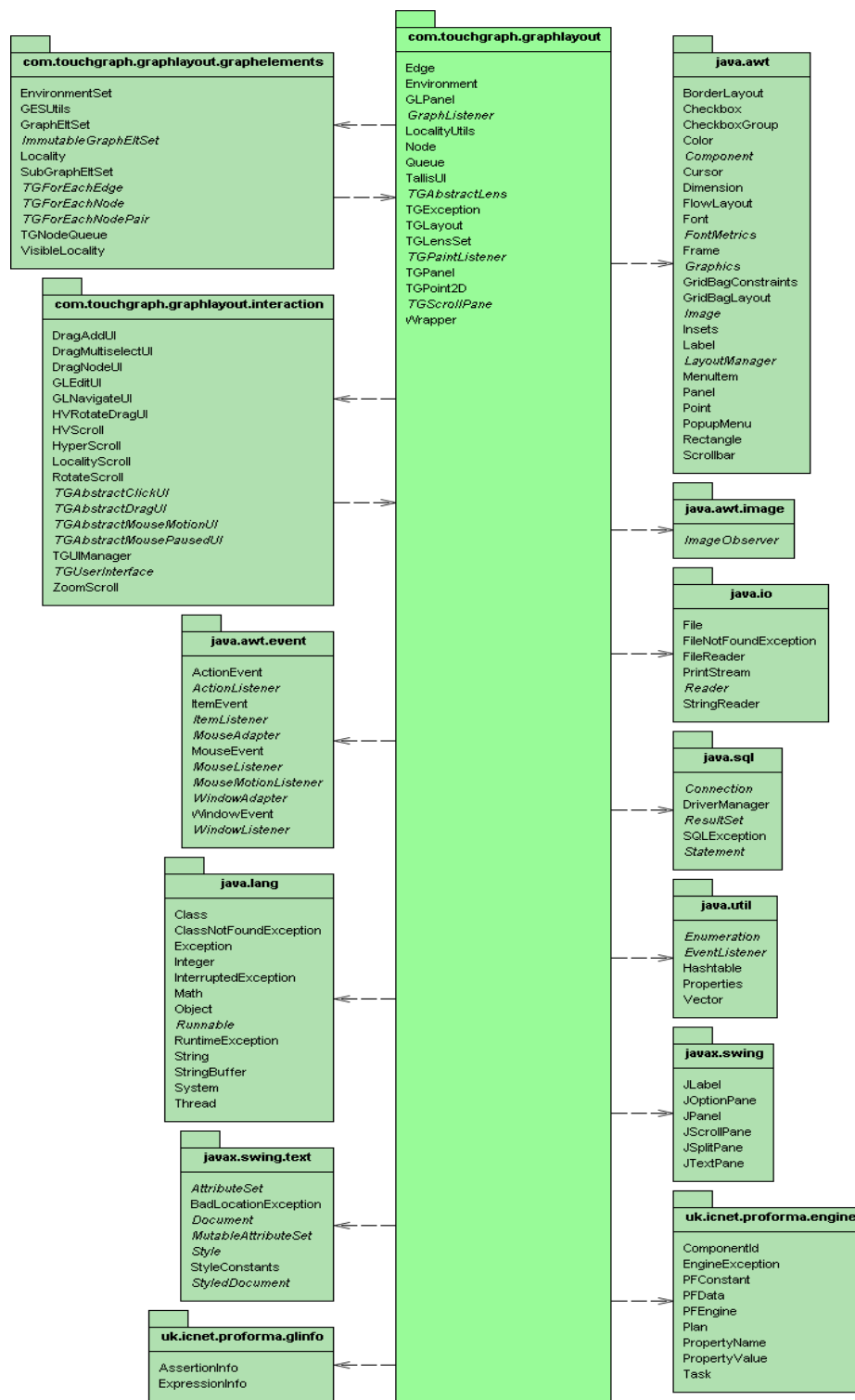


Figura 15: Diagrama de paquets en el PFC

3.3.3.2. Diagrama de Classes

Per la seva extensibilitat, a continuació es detalla una part del diagrama de classes del pfc, a la part central es troba la classe important del projecte i les seves relacions.

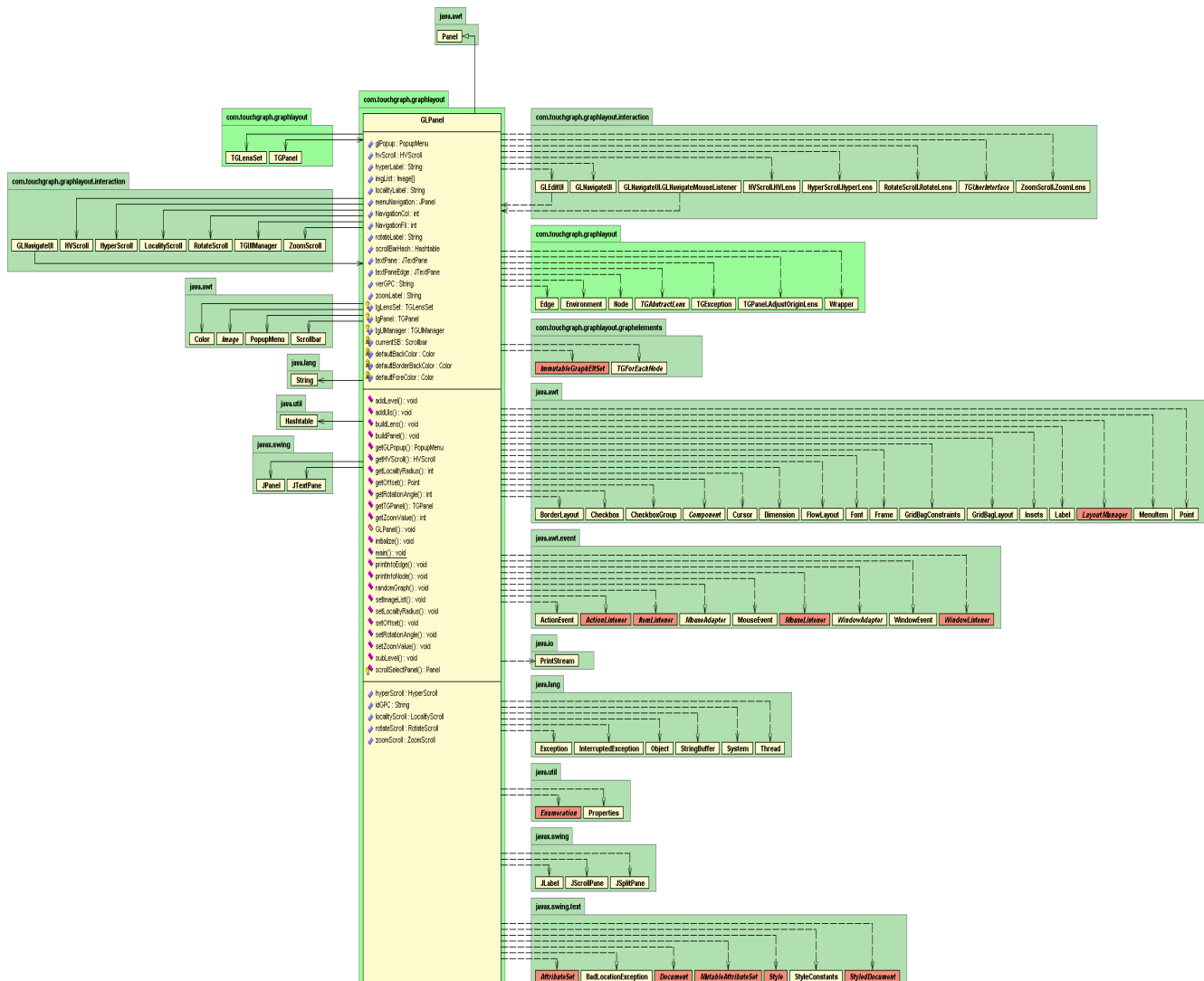


Figura 16: *Diagrama de classes en el PFC*

3.3.4. WikiBrowser

El terme Wiki s'utilitza per a nombrar una col·lecció de pàgines web d'hipertext, cada una de les quals pot ser visitada i editada per qualsevol persona. Una versió web d'un Wiki també se l'anomena *WikiWikiWeb*.

Per tant, en termes generals, podem parlar d'una aplicació informàtica col·laborativa que permet crear col·lectivament documents web utilitzant un simple esquema d'etiquetes i marques, sense que la revisió del seu contingut sigui necessària abans de la seva acceptació per a ser publicada en el lloc web d'Internet.

El WikiBrowser és una aplicació en TouchGraph que permet establir una relació entre les pàgines web que formen un Wiki determinat, i per tant facilita la navegació i edició de les pàgines, ja que cada node representa la web. Exemple present en la figura 17.

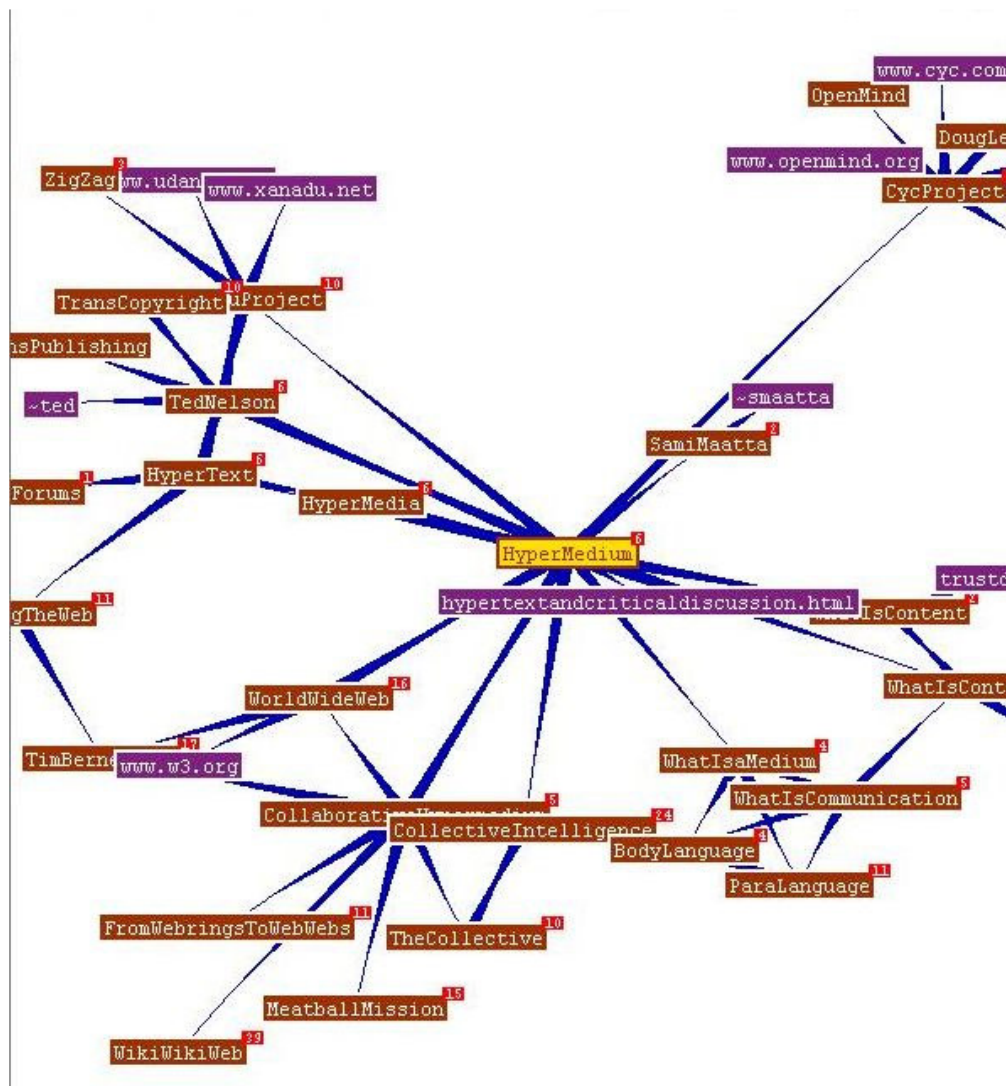


Figura 17: *Exemple d'un WikiBrowser*

3.3.5. LinkBrowser

Aquesta llibreria hereta de la llibreria GraphLayout per crear una eina que permeti presentar la relació dels diferents links que conté un lloc d'Internet, així com també descripcions associades.

Cada node representa un enllaç a Internet i, fent doble clic sobre ell, ens mostra les pàgines associades a aquesta, o també presenta una nova finestra amb un subgraf associat. Una altra funcionalitat permet veure informació descriptiva sobre el 'site' simplement passant el punter del mouse pel damunt del node.

Actualment la darrera versió és la 1.02, que inclou un nou mecanisme que permet carregar el graf a partir de la descripció d'un fitxer XML. *NanoXML* és una eina molt útil per parsejar el contingut d'aquest fitxer. Exemple present en la figura 18.

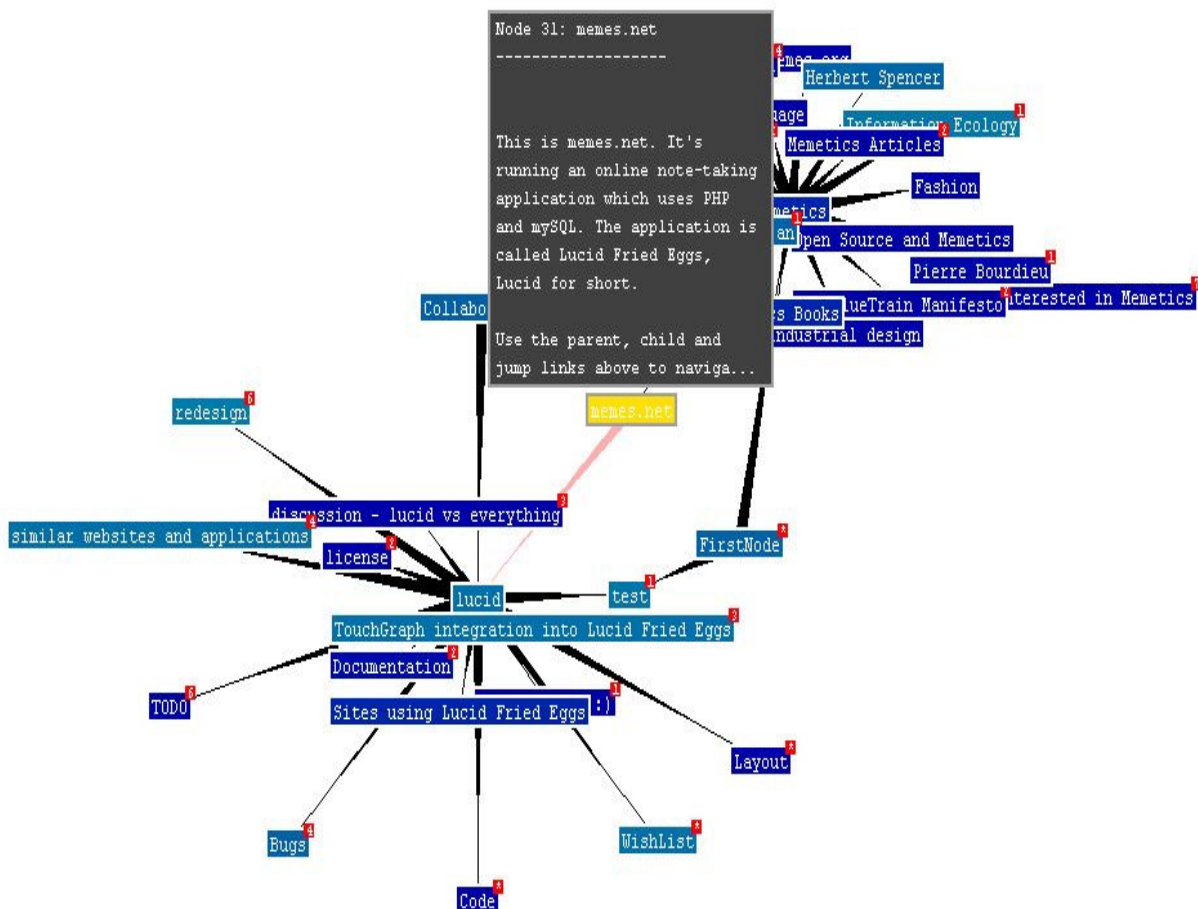


Figura 18: LinkBrowser de *www.memes.net*

3.4. JOpenChart

3.4.1. Descripció

JOpenChart és una llibreria de codi obert implementada en Java per incorporar diagrames en qualsevol tipus d'aplicació, tal com aplicacions de servidor, aplicacions d'escriptori o aplicacions web.

Actualment la llibreria permet la representació de qualsevol diagrama, per exemple diagrames de línies, diagrames de barres, etc. I el seu disseny permet encapsular conjunts de dades relacionades amb cada una de les classes, per exemple el sistema de coordenades, els eixos, la llegenda, etc.

Segons l'autor, el codi segueix un bon disseny mitjançant l'ús d'estructures lògiques, l'ús d'interfícies i classes abstractes. Així, per exemple, s'aconsegueix extensibilitat de la llibreria per tal que el propi programador implementi noves funcionalitats. Prova d'això és que el render (dibuixat) dels diagrames s'aconsegueix amb unes 150 línies de codi. En la figura 19 es presenta un exemple de la llibreria.

Algunes de les funcionalitats que ofereix JOpenChart són:

- ❖ 'Rendering' per als diagrames de línies, barres i plotejats.
- ❖ Fàcil extensibilitat utilitzant l'API *interface Renderer*.
- ❖ Eixos lineals i logarítmics.
- ❖ Diferents models de dades per a la implementació de conjunts numèrics, conjunts no numèrics per als eixos, conjunts dinàmics de dades per a la representació de valors de mesura, representació a partir de consultes SQL.
- ❖ Interpolació polinòmica de les dades, i regressions lineals per a la mesura de dades.
- ❖ Associar dades a una o varies coordenades amb diferents tipus d'escala.
- ❖ Representació de funcions matemàtiques a partir d'uns paràmetres d'entrada.
- ❖ Exportació dels diagrames com un fitxer d'imatge PNG o JPEG.
- ❖ '*Servlet Toolkit*' per incloure fàcilment diagrames en les aplicacions Web.
- ❖ Components Swing per integrar els diagrames amb aplicacions Swing.

La darrera versió de JOpenChart és la 0.94, i a continuació es detallen algunes de les millores aportades:

- Nous diagrames de radar i barres apilades.
- Els marges s'adapten automàticament, no es tallen les etiquetes.
- Nous models de dades.
- *Zooming* dels diagrames.
- Implementació de classes per a establir restriccions sobre el model.
- Empaquetat en format *WAR* suportat per servidors Tomcat.
- Etc.

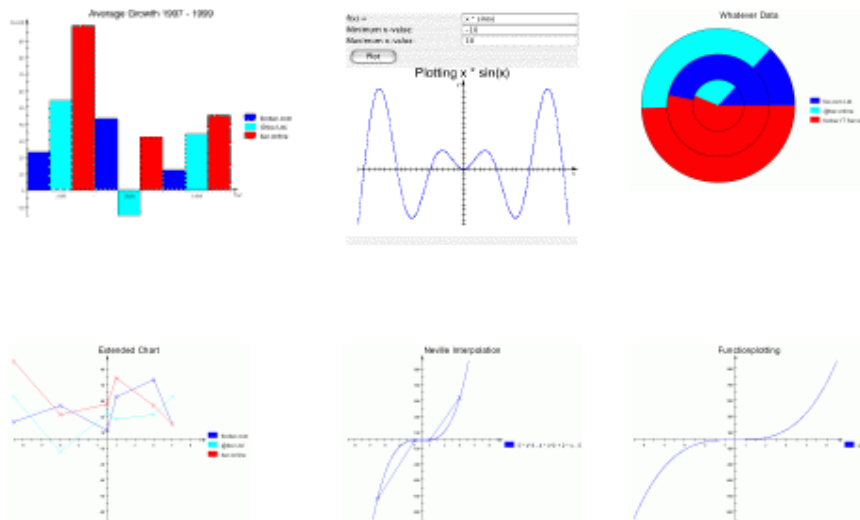


Figura 19: Exemples en JOpenChart

3.4.2. Conclusió

JOpenChart es presenta com una bona eina per al desenvolupament de diagrames de representació de dades, al igual que la llibreria JFreeChart presentada en l'apartat 3.6, que té el mateix propòsit, i són de codi obert.

Des del punt de vista de rapidesa en el processament de la imatge, JOpenChart *renderitza* millor que JFreeChart, però en qualsevol cas no segueix el propòsit del PFC i, per tant, no és la millor opció.

De totes maneres, es presenten aquestes eines per mostrar el ampli ventall de llibreries que s'ofereixen sota el llenguatge de programació Java, i la possibilitat d'implementar qualsevol tipus de diagrama. Per tant, es tracta d'un software que competeix fortament amb altres eines de caràcter privatiu que es detallen en aquest capítol.

JOpenChart el podem descarregar des de la plataforma de software lliure més popular a Internet, la direcció és <http://jopenchart.sourceforge.net/download.html>, en la pàgina Web de la llibreria podem trobar multitud de recursos disponibles per al programador, com poden ser fòrums, llistes de correu, informació de *bugs* recents, documentació, etc.

3.5. JFreeChart

3.5.1. Descripció

JFreeChart és una llibreria per a la representació de diagrames sota la plataforma Java, dissenyada especialment per a aplicacions, applets, servlets i JSP. JFreeChart es distribueix sota la llicència GNU LGPL.

Amb JFreeChart es poden generar diagrames lineals, diagrames de barres (normals o apilats i amb l'opció en 3D), diagrames de dispersió, diagrames en sèries, diagrames de Gantt, diagrames de símbols, etc.

Algunes característiques addicionals són:

- Les dades són accessibles en qualsevol implementació de les interfícies definides.
- Exportació a imatges PNG i JPEG.
- Exportació a qualsevol format que inclogui una implementació Graphics2D.
 - PDF via iText (<http://www.lowagie.com/iText>)
 - SVG via Batik (<http://xml.apache.org/batik>)
- Tool tips.
- Zoom interactiu.
- Sistema d'anotacions.
- Generació d'imatges HTML.

JFreeChart està escrita íntegrament en Java, per tant, pot ser executada en qualsevol plataforma que inclogui la plataforma JAVA. La pàgina oficial es: <http://www.jfree.org>
En la següent figura 20 es mostren exemples en JFreeChart.

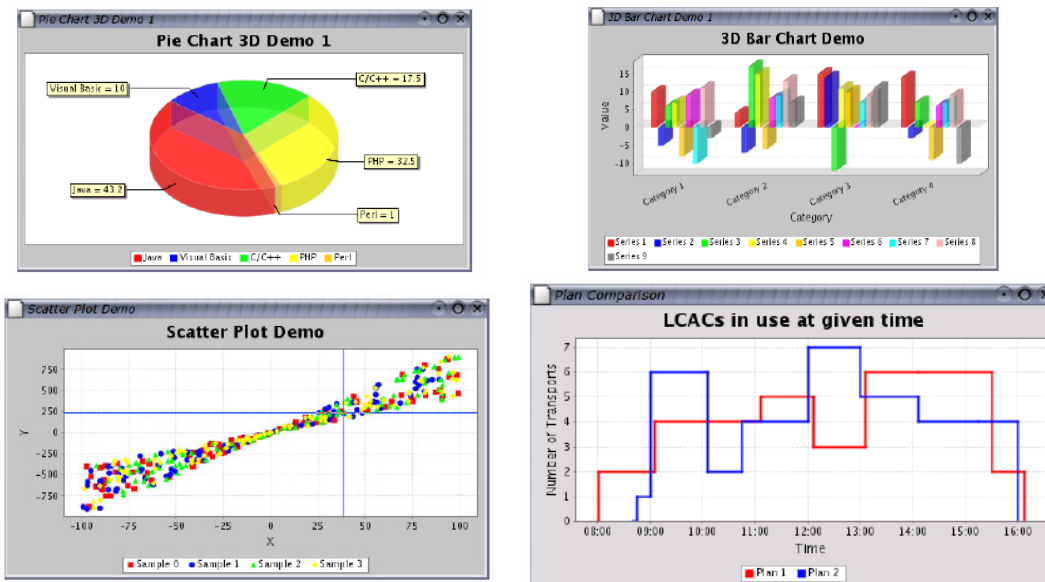


Figura 20: Exemples en JFreeChart

3.5.2. Conclusió

Durant la fase d'estudi de l'eina gràfica que es faria servir per al desenvolupament del projecte, es van analitzar diferents programes que oferien la possibilitat de pintar diagrames. JFreeChart és una bona opció per a la representació de diagrames sobre un conjunt de dades que seran analitzades. En el cas del present PFC, la idea era la representació de Grafts, per tant, no era en absolut útil per al propòsit de la nostra tasca.

En qualsevol cas, l'eina ofereix una molt bona impressió, des del punt de vista de la funcionalitat i la facilitat amb què es pot generar un diagrama. En la pàgina web del programa podem trobar diferents manuals i exemples de com utilitzar la llibreria.

3.6. JGo: GoDiagram for Java

3.6.1. Descripció

JGo és una llibreria gràfica que permet realitzar diagrames personals, editors per a xarxes o *workflows*, connexió de grafs, organitzadors o planificadors d'organigrames, mapes, *flowcharts*, i disseny d'eines de software.

És una suite molt completa que suporta gran quantitat de figures, textos, imatges, connectors, enllaços interactius, drag-and-drop, selectors, etc., a més a més dóna suport a la llibreria Swing de Java, com es mostra en la figura 21.

Es tracta doncs d'una eina bastant potent que es presenta en diferents plataformes. Des de la pàgina Web de l'empresa creadora de JGo (www.nwoods.com) donen informació relacionada amb els tres models que ofereixen. Aquests són:

- **GoDiagram for .NET**: implementació de la llibreria gràfica sota la plataforma .NET de Microsoft.
- **GoDiagram for Java (JGo)**: implementació de la llibreria gràfica en Java.
- **GoDiagram for MFC (Go++)**: implementació de la llibreria gràfica sota l'api MFC (Microsoft Foundation Classes) de Windows.

Cal destacar que JGo suporta Java 2, Java 2D i Swing. També es permet des de la pàgina web descarregar un *kit* d'avaluació, de manera gratuïta, que inclou:

- Codi font d'exemples.
- Exemples d'aplicacions en *JGo.jar* i *JGoLayout.jar*.
- Permet crear un programa d'exemple.
- Documentació.

Finalment podem concloure que es tracta d'una eina gràfica molt completa i que ofereix moltes opcions per a l'usuari; també és important destacar les implementacions en les tres plataformes de programació, i la documentació i suport *online* que ofereix l'empresa.

Cal afegir a més que es tracta d'un software de pagament d'uns 1.245 dòlars amb un any de suport i actualitzacions.

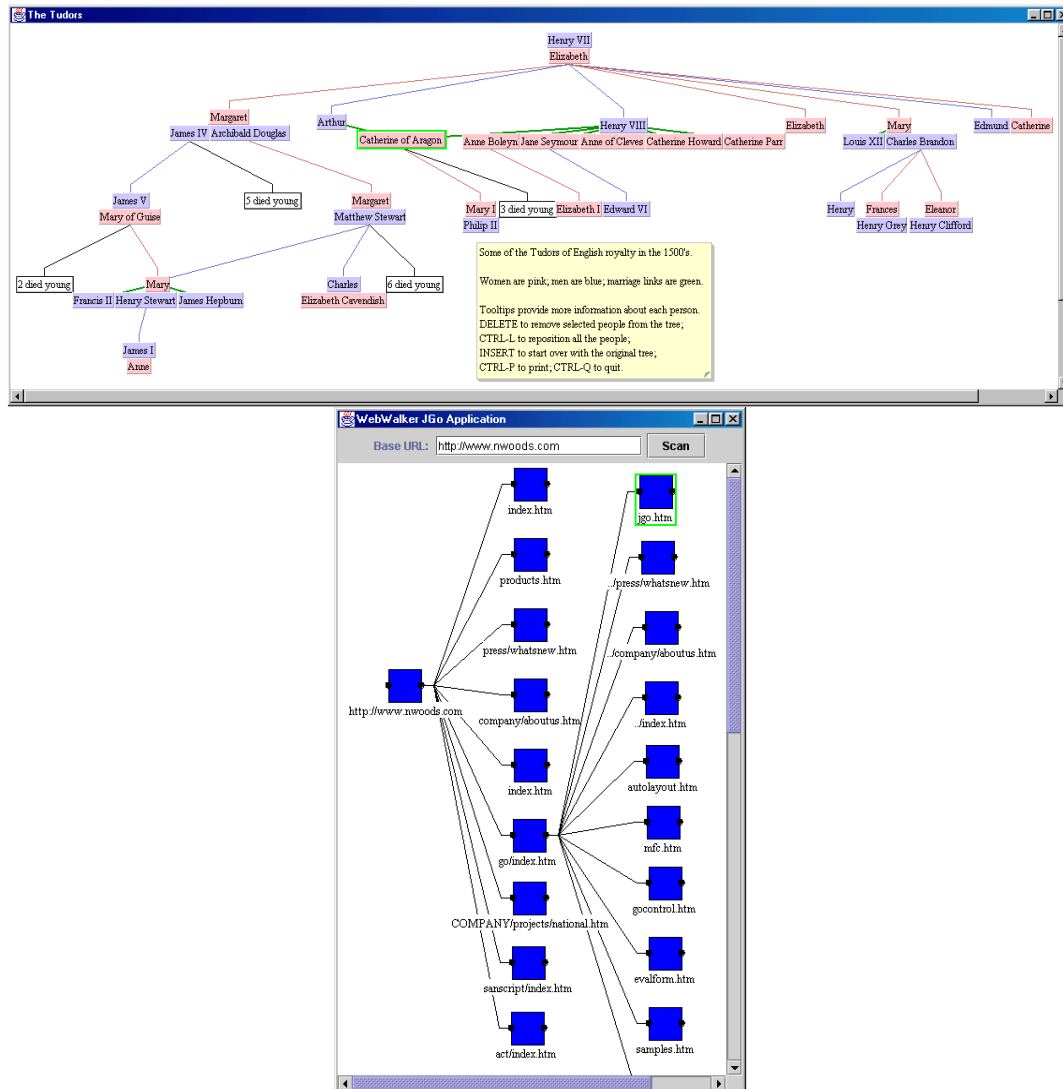


Figura 21 : Exemples amb JGo

3.6.2. Conclusió

Des de la pàgina web, la informació que dona són òbviament avantatges d'utilitat, escalabilitat, rapidesa, eficiència, etc. Com a opinió personal crec que es tracta d'un producte molt ben finalitzat i molt ben documentat, que permet fer quasi qualsevol diagrama, i per les imatges que tenen a la pàgina web és bastant potent.

Per al PFC, podria ser una opció tan vàlida com una altra ja que compleix els criteris necessaris per crear un graf dirigit, però en aquest cas el fet que sigui un software privatiu i el seu cost molt elevat implica l'estudi d'altres eines disponibles.

3.7. ILOG JViews Diagrammer

3.7.1. Descripció

ILOG JViews Diagrammer és un producte desenvolupat per l'empresa ILOG i està destinat a la representació d'un ampli nombre de grafs. Des de la pàgina oficial de ILOG es presenta el producte com a revolucionari, mostrant els avantatges que proporciona el software, figura 22.

Aquest programa inclou diferents tipus d'algorismes que tenen associats diferents dominis on s'hi pot associar un diagrama. Alguns exemples d'aquests algorismes inclouen els següents dominis:

- ✚ *Jeràrquic*: processament per a diagrames de procés, diagrames UML, flowcharts i diagrames de disseny.
- ✚ *Arbres*: processament per a organigrames, arbres de decisió, etc.
- ✚ *Arbres en Radi*: inclou processament per a navegació en website, visualització d'enllaços web, i xarxes semàntiques.
- ✚ *Malla Topològica*: per aplicacions d'enginyeria.
- ✚ *Circulars (estrella o anell) i lineals*: processament de xarxes LAN i sistemes de manteniment.
- ✚ *Sistemes en Grid*: processament per a xarxes WAN, xarxes socials, etc.

També existeix interoperabilitat amb altres productes que presenta l'empresa, com són: *ILOG JViews Maps, Gantt, Charts*, etc., que formen una suite molt completa per manejar qualsevol tipus de representació gràfica.

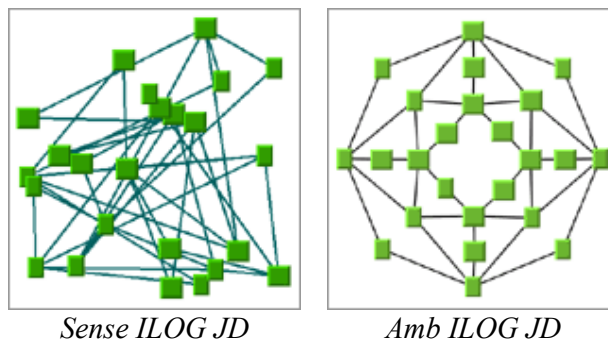


Figura 22: Reclam publicitari

A continuació es detallen les principals característiques de ILOG JViews Diagrammer:

- **Manegament de SubGrafs:** ILOG JD dona suport a la manipulació de grafs que a la vegada contenen altres grafs amb nous elements. El diagrama utilitza el concepte de 'Meta Node' per a expandir un nou graf associat a l'anterior. D'aquesta manera l'usuari pot representar i modelar informació més precisa i real. ILOG JD manté de forma intel·ligent els diferents links i manté la visibilitat de forma ordenada i sense solapament. Un exemple es mostra a la figura 23.

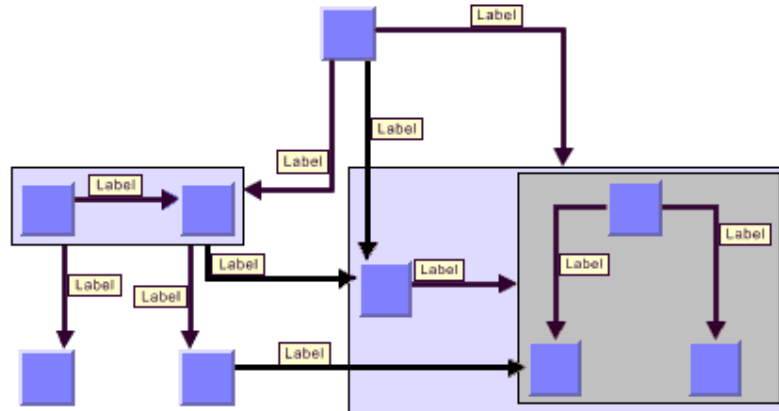
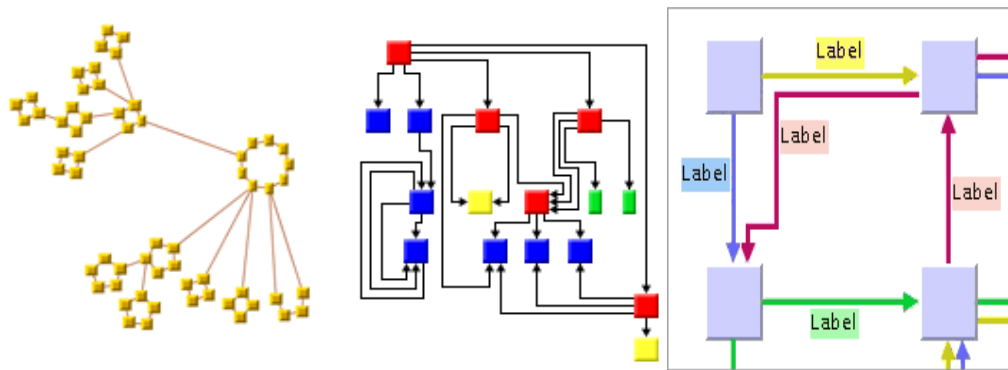


Figura 23: Exemple de SubGraf

- **Establiment de paràmetres per a un millor control del diagrama:** cada algorisme pot rebre paràmetres per tal d'especificar si es vol un layout de dreta a esquerra o a la inversa, els espais entre nodes, etiquetes, enllaços, etc.
- **Interactivitat per a l'edició del graf:** creació i modificació de la informació associada a un node o als seus enllaços, fer visible o no un graf, aplicar efectes de zoom, drag-and-drop, còpia i pega, animacions, etc.
- **API per la extensibilitat:** a partir de l'api especificada podem crear les nostres pròpies classes i millorar-les o adequar-les al nostre objectiu. L'API està implementada en Java.
- **Fàcil integració i instal·lació:** permet la integració amb Java 2D i amb components Swing; també permet l'ús en servidors web, ja que utilitza DHTML per a la visualització d'un graf.

A continuació es presenten algunes gràfiques que mostren l'ús de ILOG JD:



3.7.2. Conclusió

Tot i que en el present document no s'ha entrat molt en detall en la funcionalitat que ofereix ILOG JD, podem dir que és un dels millors programes per a la representació de gràfics.

Es tracta doncs d'un programa privatiu i la seva documentació no és accessible a no ser que es compri el producte.

3.8. Elecció de l'eina gràfica

3.8.1. Taula comparativa

En la següent taula es mostren de forma resumida cadascuna de les eines gràfiques avaluades durant l'etapa d'anàlisi. Tot seguit es descriuen els significats dels paràmetres avaluats, ordenats segons la seva importància en el present pfc.

- *Gratuït* : Òbviament és el més important, en l'ús acadèmic del projecte necessitem una eina de cost gratuït o bé, de codi obert.

- *Documentació* : Una eina ben documentada és molt important per finalment escollir la millor opció, a partir d'una bona documentació es poden treure els valors dels altres paràmetres.

- *Interacció amb l'usuari i Extensibilitat* : Aquests dos paràmetres van fortament lligats, en el nostre cas volem una llibreria gràfica que ens permeti treballar amb events generats per l'usuari (per exemple events del ratolí, etc.) així com ampliar la funcionalitat del propi codi amb l'ús per exemple d'herència (extensibilitat).

- *Arquitectura*: Les arquitectures més usades estan basades en patrons de disseny com és el cas del model MVC (Model-Vista-Control) que utilitzen la majoria.

- *Llenguatge i Organització* són paràmetres propis de l'eina gràfica i per tant no són del tot rellevants en el nostre cas.

Taula Comparativa de Software Gràfic

	Llenguatge	Organització	Gratuït	Documentació	Arquitectura	Interacció Usuari	Extensibilitat
Ilog JD	Java	ILOG	No	No	-	Sí	Sí
Jgraph**	Java	-	Sí*	Sí (molta)	MVC	Sí	Sí
JOpenChart**	Java	-	Sí	Sí (poca)	-	No	Sí
OpenJGraph**	Java	J. Salvo	Sí	-	-	No	Sí
JFreeChart	Java	-	Sí	Sí (molta)	-	No	No
GraphViz	Java	-	Sí		-	-	No
TouchGraph**	Java	TouchGraph Team	Sí	No	MVC	Sí	Sí
JGo	Java	Nwoods	No	No	-	-	No
Java 3D	Java	Sun	Sí	Sí	-	Sí	Sí
Giny	Java	I.S. Biology	Sí	API	Piccolo, Colt,...	No	-
JGraphT**	Java	-	Sí	API	JGraph	Sí	Sí
GVF**	Java	Royere	Sí	Sí	Framework	Sí	No
JUNG	Java	Fisher, Nelson	Sí	Sí	Colt, Xerces	Sí	-
HyperGraph**	Java	-	Sí	Sí	-	-	Sí
Jdigraph**	Java	D. Walend	Sí (BSD)	Sí	-	Sí	-
VrmlGraph	Java	-	Sí	Sí	VRML	No	No
WilmaScope**	Java	T. Duyer	Sí	Sí	Java 3D	No	No

* Existeix una versió Comercial anomenada Jgraph Pro

3.8.2. TouchGraph en el PFC

La representació gràfica que es plantejava en el projecte era clara i estava ben definida, es tractava de la representació bàsicament d'un graf dirigit. Sobre aquest graf no s'havien de fer operacions complexes d'esborrat de nodes o difícils operacions d'afegir nodes o suprimir arestes, només es volia de forma ràpida i eficient la representació d'un graf que simbolitzés una guia de pràctica clínica. Sí que es pretenia, en canvi, que el *renderitzat* del graf fos ràpid i eficient. Des del punt de vista d'interacció amb usuari es tenia clar que volíem que la informació fos en tot moment visible i que bàsicament les operacions per l'usuari fossin expandir o contraure el graf. També es volia que el graf fos amigable des del punt de vista de distribució dels nodes en l'espai de dibuix, els colors i formes, els events del ratolí i la mobilitat del graf per l'espai, així com fer zooming sobre el graf, etc.

Per tant, a l'hora d'analitzar cada un dels programes informàtics per a la representació de grafs es van considerar totes aquestes funcionalitats, no es volia en cap cas una eina molt complexa que permetés fer grans dibuixos en 3D, o complexos entorns d'execució. Finalment el ventall de possibilitats es va reduir a tres opcions, que eren: **GraphViz**, **JGraph** i **TouchGraph**.

Les tres llibreries complien en gran mesura part de les especificacions fetes en un principi, però hi havia trets singulars que feien que no fossin del tot acceptades. Per exemple en el cas de la llibreria GraphViz es va veure que la càrrega d'un graf per a la seva posterior representació era de forma estàtica, per exemple, amb un fitxer de vèrtexs i arestes, i a més a més el control d'events que ofería al programador no era del tot satisfactori. Finalment el GraphViz va ser descartat.

La decisió final estava doncs entre el JGraph i el TouchGraph. Durant varies setmanes va ser difícil apostar per un o l'altre, ja que tots dos eren bastant complets.

En principi, JGraph és una llibreria més potent que TouchGraph, per tant, vaig iniciar la fase "d'aprenentatge" amb aquesta llibreria; a més a més disposava d'una bona documentació *online*. A mesura que utilitzava JGraph vaig adonar-me que al ser una llibreria més completa i amb més capacitat per fer qualsevol cosa, la manipulació del graf s'havia de fer des de l'inici, és a dir, era una feina més costosa, ja que tenia que implementar un graf des de molt baix nivell. En aquest punt, em vaig decidir per provar de forma paral·lela la llibreria TouchGraph. El primer que em vaig trobar a TouchGraph va ser la falta de documentació; en certa manera no hi havia cap documentació, per tant, vaig pensar en reiniciar el procés amb JGraph, però tot i no haver documentació a TouchGraph vaig aconseguir trobar-hi diversos exemples desenvolupats que fins i tot, es podien executar i veure les seves prestacions (Figura 24). Va ser llavors quan em va agradar la manipulació i ús que feia d'un simple graf implementat amb GraphLayout de TouchGraph. A partir d'aquí, vaig començar a crear els meus propis grafs i vaig veure que era molt més fàcil que fer-ho amb JGraph. L'únic inconvenient va ser el fet "d'estudiar" l'api de TouchGraph a partir del codi font.

Quan vaig realitzar el primer graf amb TouchGraph, em vaig adonar que era molt fàcil la interacció amb l'usuari, que fins i tot per defecte havien botons de zoom, rotacions, ..., que era precisament el que necessitava pel projecte.

El fet que el codi estigués molt ben comentat i que la programació del codi font fos el més oberta possible em va ser molt útil perquè vaig incorporar les meves funcions de forma molt fàcil.

De manera que amb TouchGraph vaig tenir moltes de les funcionalitats que necessitava creades per defecte i era fàcil d'utilitzar. En aquest moment vaig decidir *finalment utilitzar les llibreries de TouchGraph per a la implementació final en el PFC*.

Durant la fase de disseny de l'aplicació vaig poder utilitzar alguna estructura de dades definida pel propi TouchGraph, de manera que vaig integrar les noves classes amb les de la llibreria i em va resultar bastant fàcil.

Finalment, s'ha de concloure que, tot i que JGraph es presentava com la millor opció per integrar en el PFC gràcies a la seva extensa documentació, a vegades és molt millor un exemple visual d'allò que pot arribar a fer un programa, que no pas una documentació massa exhaustiva que amagui realment la funcionalitat del programa.

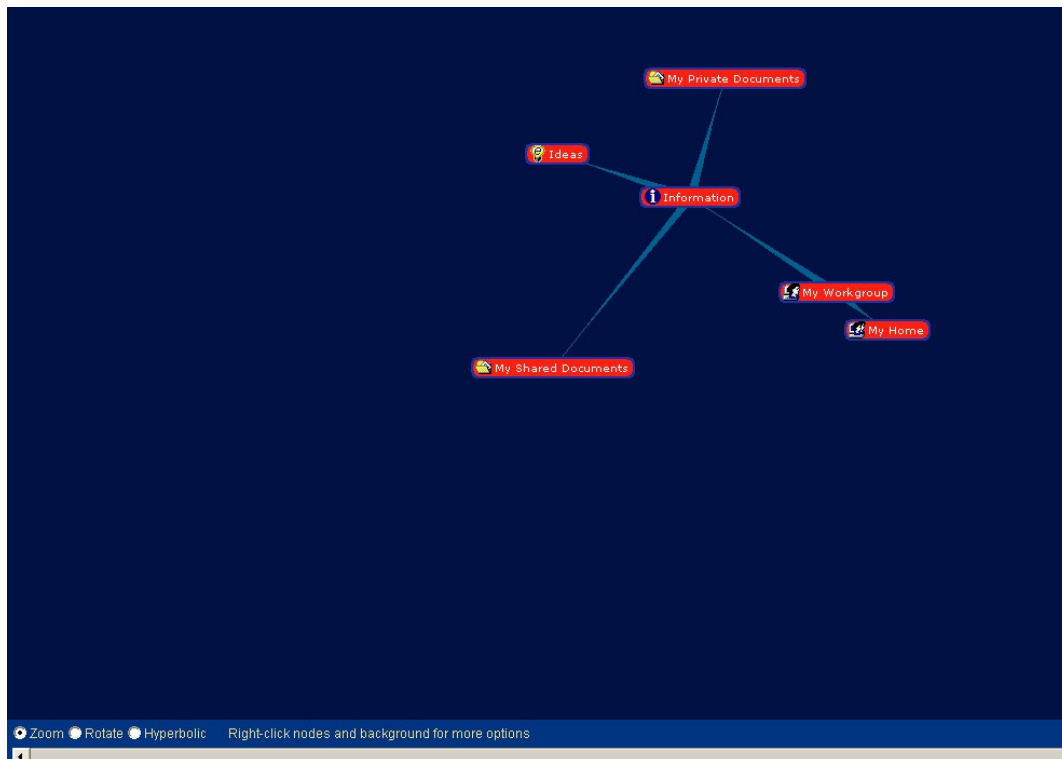


Figura 24: Exemple base inicial generat en TouchGraf

4. TECNOLOGIA WEB I VALIDACIÓ D'USUARIS WEB

Per al desenvolupament d'aquest PFC s'han utilitzat les següents tecnologies web: Html, Jsp, Applets i Javascript que són detallades en les pròximes seccions.

4.1. Pàgines dinàmiques

Les pàgines dinàmiques són pàgines HTML generades a partir de llenguatges de programació (scripts), executades en el servidor web. A diferència d'altres scripts, com el Javascript (explicat en la secció 4.1.3), que s'executen en el propi navegador de l'usuari, els "Server Side" scripts generen codi HTML des del propi servidor Web.

4.1.1. HTML

El HTML va ser desenvolupat originalment per Tim Bernes-Lee al CERN, i va ser popularitzat pel navegador Mosaic creat per la NCSA. Els documents que utilitzin HTML han de funcionar en diferents plataformes i navegadors. Assegurant la interoperabilitat entre ambdós s'assegura un menor cost per als proveïdors de continguts en la web, ja que només s'han de preocupar per publicar una versió dels documents. Si no s'estandarditzés el HTML, llavors la web seria un món ple d'estàndards propietaris, amb formats incompatibles entre ells, reduint així el potencial comercial per als usuaris. Cada versió del HTML intenta reflectir un major consens entre tots els membres de la indústria, de forma que la inversió dels proveïdors de continguts no es converteixi en inservible per no poder llegir els documents.

HTML (HyperText Markup Language) és un llenguatge fàcil d'utilitzar que permet descriure hipertext, és a dir, text presentat de forma estructurada i agradable, amb enllaços (hyperlinks) que condueixen en altres documents o fonts d'informació relacionada, a més a més, amb insercions multimèdia (gràfics, so, etc.).

La descripció es basa en especificar el text de l'estructura lògica dels continguts (títols, paràgrafs de text normal, enumeracions, definicions, etc.), així com els diferents efectes que es vulguin donar (parts de text en cursiva, negreta, o un gràfic) i deixar que la presentació final de l'hipertext la realitzi un programa especial, anomenat navegador web (Mozilla, Internet Explorer, Opera, Firefox, ...).

Un document HTML comença amb l'etiqueta <html> i acaba amb </html>. Dins el document (entre les etiquetes de principi i fi de html), es troben dues zones ben diferenciades: la *capçalera* delimitada per <head> i </head> que s'utilitza per definir valors vàlids en tot el document; i el *cos* delimitat per <body> i </body> on resideix la informació del document.

El cos del document HTML conté text que, amb la presentació i efectes que es decideixin, es presentarà amb l'*hyperlector*.

En resum, l'estructura bàsica d'un document HTML té la següent forma:

```
<html>  
  <head>
```

```

        <title>Título</title>
    </head>
    <body>
        Exemple de HTML!.
    </body>
</html>

```

En l'actualitat, trobem una nova especificació del HTML, amb l'aparició de l'estàndard XHTML 2.0 que engloba el potencial del HTML (darrera versió 4.0) amb la incorporació del XML (llenguatge d'etiquetes per representar dades). Per a més informació cal adreçar-se a la plana Web del W3C en <http://www.w3.org/MarkUp> que és un consorci format per diverses institucions i empreses que generen els estàndards.

4.1.2. JSP: JavaServer Page

JSP és una especificació de *Sun Microsystems* per a la creació i gestió de pàgines web dinàmiques. Per tant, permet barrejar en una pàgina codi HTML per a generar la part estàtica, amb contingut dinàmic generat a partir de les marques especials `<% ... %>` que s'utilitzen per a JSP. El codi entre les claus que conté els JSP és codi Java, i el seu principal objectiu és separar l'interfície (presentació visual) de la implementació (lògica d'execució). La pàgina JSP es transforma finalment amb un Servlet, i la conversió la realitza la màquina servidora o contenidor JSP la primera vegada que es sol·licita la pàgina JSP, com es mostra en la figura 25.



Figura 25: Transformació de JSP a Servlet

Com s'ha comentat anteriorment, quan es crida per primera vegada el fitxer JSP es genera un Servlet, que té un cicle de vida amb les etapes següents:

- ✚ `jspInit()`: Inicia el Servlet generat i només es crida en la primera petició.
- ✚ `jspService` (petició, resposta): Manega les peticions, s'invoca en cada petició incloent-hi la primera vegada.
- ✚ `jspDestroy()`: S'invoca pel motor JSP per eliminar el Servlet.

Tant els JSP com el Servlet necessiten un programa que els contingui, i sigui el que envii efectivament pàgines web al servidor, i rebi les peticions, les distribueixi entre els servlets, i faci tota la gestió pròpia d'un servidor web. Servidors com l'Apache estan especialment pensats per a pàgines web estàtiques CGI, i programes executats per

servidors, com el PHP. Hi ha altres servidors específics per a Servlets i JSPs anomenats contenidors de servlets (servlets containers). Els principals són:

- ✚ Resin, de Caucho Technologies, un motor especialment per al servei de pàgines XML, amb una llicència lliure per a programadors. Inclou suport per a JavaScript i Java. Inclou també un llenguatge de plantilles anomenat XTP.
- ✚ BEA Weblogic és un servidor d'aplicacions d'alt nivell. Està desenvolupat en Java i es combina amb altres productes, com per exemple, Tuxedo, un servidor de base de dades per a XML.
- ✚ JRun, de Macromedia, és un servidor d'aplicacions Java. Es pot descarregar una versió d'avaluació des de la pàgina web oficial.
- ✚ Lutris Enhydra, servidor gratuït i de codi obert. Enfocat a servir XML i per a plataformes de mòbils.
- ✚ Jakarta Apache: del consorci Apache, és un contenidor de servlets amb molts programes addicionals, per exemple, Cocoon per a servir pàgines XML.

4.1.3. JavaScript

JavaScript permet crear aplicacions específicament orientades al seu funcionament en la xarxa d'Internet. A partir de Javascript es poden crear pàgines HTML dinàmiques que processin l'entrada de l'usuari i que permetin gestionar dades persistents amb objectes especials, fitxers i bases de dades relacionals.

Amb JavaScript es poden construir aplicacions que varien des de la gestió d'informació corporativa interna i la seva publicació en Intranets, fins a la gestió massiva de transaccions de comerç electrònic.

Les aplicacions client i servidor en JavaScript tenen el mateix nucli del llenguatge. Aquest nucli correspon a l'ECMA-262 [4], el llenguatge de script estàndard de l'oficina d'estàndards de la Unió Europea. Els components que formen JavaScript són els següents:

- ✚ Nucli de JavaScript (Core JavaScript).
- ✚ JavaScript pel Client.
- ✚ JavaScript pel Servidor.

Les principals característiques de JavaScript s'enumeren a continuació:

- ✚ JavaScript és un llenguatge interpretat, és a dir, no requereix d'una compilació. El navegador de l'usuari s'encarrega d'interpretar les sentències JavaScript en una pàgina HTML i executar-les adequadament.
- ✚ JavaScript és un llenguatge orientat a events. Quan un usuari fa un clic sobre un enllaç o mou el punter sobre una imatge es produeix un event. Amb JavaScript es poden crear scripts que executin accions en resposta a aquests events.

- JavaScript és un llenguatge orientat a objectes. El model d'objectes (figura 26) de JavaScript és reduït i simplificat, però inclou els elements necessaris perquè els scripts puguin accedir a informació d'una pàgina i puguin actuar sobre d'interfície del navegador.

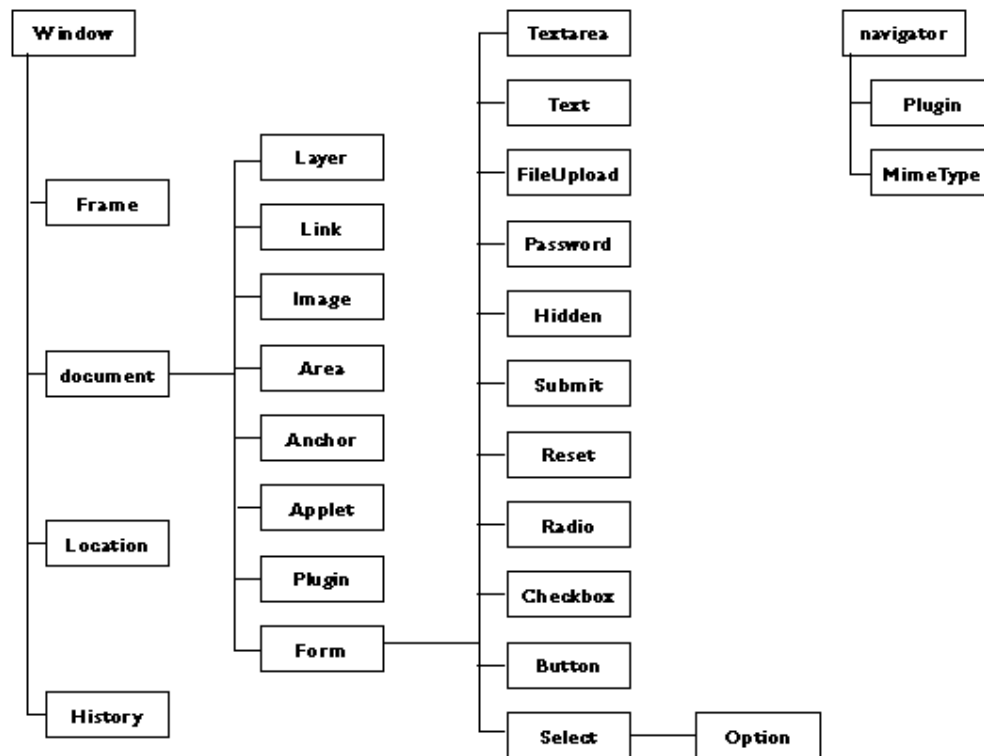


Figura 26: Jerarquia dels principals objectes en JavaScript

Cal recordar que JavaScript i Java són llenguatges totalment diferents tal com es mostra en la següent taula:

JAVASCRIPT	JAVA
Interpretat pel client	Compilat (bytecodes). Es descarrega del servidor i s'executa en el client
Orientat Objectes	Basat en Classes
El codi s'integra i incrusta en documents HTML	S'utilitzen APPLETs. S'accedeix a ells des de documents.
Els tipus de dades de les variables no es declara	És necessari definir els tipus de dades de les variables
No es pot escriure automàticament en el disc dur	No pot escriure automàticament en el disc dur

4.1.4. Applets

Un applet és un programa escrit en Java que està contingut normalment en una pàgina Web i s'executa en un navegador Web. L'applet inicia la seva execució quan es carrega la pàgina web que el conté. Per visualitzar un applet és necessari un navegador web o bé la comanda `appletviewer` del JDK. El navegador web detecta un applet quan es processen les etiquetes `<APPLET>` contingudes en el codi HTML. En la figura 27, podem veure el cicle de vida que segueix un applet, aquests mètodes s'hauran de sobreescrivre per part del programador.

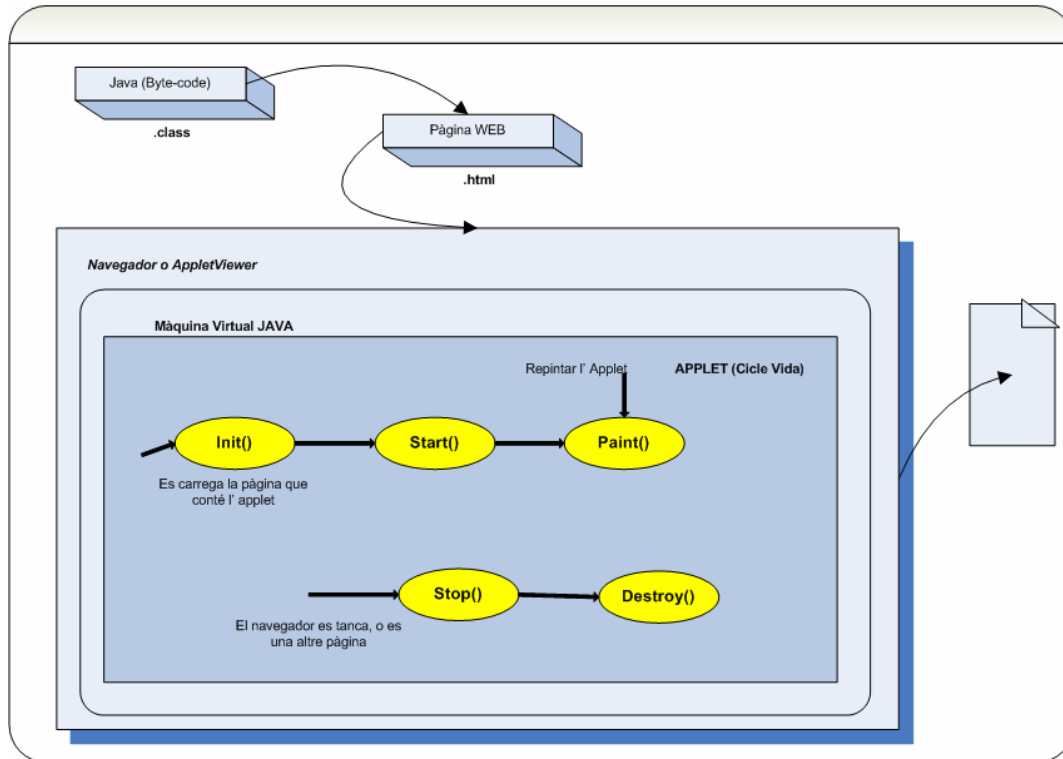


Figura 27: Esquema general d'un Applet.

Per herència de la classe `java.applet.Applet` es defineix la següent estructura per funcionar amb el Navegador Web.

- ```

public class MyApplet extends java.applet.Applet {
 public MyApplet() { //creació del applet
 System.out.println("In ctor");
 }
 public void init() { // inicialitzar el applet
 System.out.println("In init");
 }
 public void start() { // visualitzar
 System.out.println("In start");
 }
 public void stop() { // sortir de la pàgina
 System.out.println("In stop");
 }
 public void destroy() { // alliberar recursos

```



```
 System.out.println("In destroy");
 }
}
```

En el codi d'una pàgina HTML, la definició d'un applet té la següent estructura:

```
<HTML>
 <HEAD>
 <TITLE>
 Applet
 </TITLE>
 </HEAD>
 <BODY>
 <APPLET
 ARCHIVE=fichero.jar
 CODE=ejemplo.class
 CODEBASE=http://...
 WIDTH=400
 HEIGHT=400
 ALIGN=MIDDLE>
 <PARAM NAME="FICHERO" VALUE="c:/tmp/prueba.txt"/>
 </APPLET>
 </BODY>
</HTML>
```

- **ARCHIVE**: especifica el fitxer .jar on es troben les classes i altres recursos que utilitzi l'applet.
- **CODE**: especifica la classe que hereta d'Applet.
- **CODEBASE**: estableix la URL on trobar la classe o fitxer .jar.
- **WIDTH**: especifica l'ample de l'applet.
- **HEIGHT**: especifica l'altura de l'applet.
- **ALIGN**: estableix la posició on s'ubicarà l'applet. Els valors poden ser *left*, *right*, *top*, *bottom*, *middle*.
- **PARAM**: s'utilitza per passar paràmetres a l'applet.

#### 4.1.5. Sessions

Les sessions són objectes que emmagatzemen variables i paràmetres que el programador decideix utilitzar en la implementació de l'aplicació. Les principals característiques són:

- S'emmagatzemen en el servidor, no en el client.
- Guarden qualsevol tipus de dada.
- No té límit de mida.

Objecte **session**: es poden guardar objectes Java assignant-los un nom per després recuperar-los. Els principals mètodes són:

- `session.setAttribute(nom,objecte)`
- `session.getAttribute(nom)`

Els objectes guardats en la sessió es mantenen mentre l'usuari navegui per les pàgines web. En el cas del present PFC les variables que s'han guardat en la sessió de l'usuari fan referència a l'estat de la seva validació.

## 4.2. Validació d' usuaris via web

Actualment, les necessitats de seguretat en les aplicacions Web venen cobertes per diferents àrees. Aquestes descriuen els diferents requeriments de l'aplicació Web, així com les dades que seran utilitzades per l'aplicació.

Una seguretat des del punt de vista estricte ha de donar resposta a les següents preguntes:

- A qui se li permet accés a l'aplicació Web?
- Com identificarem els usuaris?
- Existeixen restriccions d'accés a diferents parts del sistema?
- Existeixen dades privades?
- Qui pot afegir/modificar/esborrar dades?

Aquestes preguntes aporten la necessitat de crear un sistema de seguretat propi i independent que formarà part dels mòduls de l'aplicació web. Un error típic a l'hora de crear un sistema Web és establir la seguretat una vegada implementat el programa. Això pot dur a futurs errors o incoherències que debilitin la seguretat del sistema en front de possibles atacs.

És important considerar que els requeriments de seguretat són una part destacada dintre del procés de requeriments funcionals, dins del cicle de vida de qualsevol aplicació informàtica.

Les principals àrees relacionades amb els requeriments de seguretat són les següents:

-  Autenticació
-  Autorització
-  Integritat de les dades.
-  Confidencialitat.

### 4.2.1. Autenticació i autorització

L'autenticació simplement és la comprovació de la identitat. El sistema més utilitzat i més extens en les aplicacions web és l'ús d'un parell de claus: el nom de l'usuari (*login*) i la contrasenya (*password*).

L'autenticació és utilitzada freqüentment en les sessions HTTP entre client i servidor per tal de permetre o no accés a diferents parts de l'aplicació; un exemple clar és l'accés a Bancs Electrònics o el Correu Electrònic a Internet.

Els mètodes d'autenticació en les aplicacions Web són:

- Autenticació HTTP bàsica:

És el mètode més simple en el procés d'autenticació. El principal avantatge que ofereix és l'ús mínim de recursos necessaris per a la validació. A partir del nom de l'usuari i la contrasenya només es vol assegurar l'accés a un determinat recurs. Normalment la informació enviada entre navegador i servidor té una pobre protecció i la codificació de les dades es fa amb BASE-64. El navegador és qui controla el procés d'autenticació i, per exemple, pot mancar informació relacionada amb que l'usuari abandona la sessió simplement tancat el navegador. En resum, aquest tipus de solució aconsegueix autenticació per part de l'usuari però pot ser fàcilment burlada i no garanteix un model de seguretat idoni per una aplicació WEB.

- Autenticació HTTP resum:

Aquest model és molt semblant a l'autenticació bàsica, la única diferència està en què en el procés de validació entre Navegador (IEx, Mozilla, Firefox, Opera, ...) i servidor Web, la contrasenya no és enviada a través de la xarxa, sinó que utilitza un **resum** (*cadena de text*) més uns nombres generats aleatòriament que formaran la pseudo-contrasenya. El principal desavantatge que presenta és la incompatibilitat amb els navegadors Web.

- Autenticació basada en formulari:

És un sistema molt semblant a l'autenticació bàsica. En aquest cas s'utilitza el model **HTML.Form** que permet l'enviament del nom d'usuari i contrasenya dintre de la sessió HTML.

- Autenticació SSL (Socket Secure Layer):

Aquest model integra una infraestructura de certificats, i utilitza comunicació xifrada entre client i servidor Web. El protocol SSL és un sistema dissenyat i proposat per Netscape. Proporciona serveis de seguretat xifrant les dades intercanviades entre el servidor i el client amb un algorisme de xifratge simètric, típicament el RC4 o IDEA, i xifrant la clau de sessió amb RSA. També comentar que s'utilitza MD5 com algorisme de hash i així mantenir la integritat de les dades.[3]

**L'autorització** defineix quin és l'accés que es té sobre un determinat recurs. Aquest afer es duu a terme a partir de la definició d'uns permisos d'accés per cada usuari.

En Java, dintre de l'especificació *Servlet* es defineixen aquests permisos a partir d'allò que s'anomena "rols d'accés". A cada usuari se li assigna un o més rols, i els recursos són accedits a partir d'usuaris autenticats amb un rol específic. El grup d'usuaris i rols definits per una determinada política d'autenticació es coneix amb el terme en anglès **Realm**, utilitzat en aquest projecte com s'explicarà més endavant.

Cal tenir en compte que la implementació dels mecanismes d'autorització per als contenidors *J2EE* o *JSP/Servlet* no està definida, per tant, aquesta és específica de cada

contenedor. En el nostre cas, com treballem amb el contenidor d'aplicacions Web **Tomcat**, ens ofereix multitud de possibilitats per als processos d'autorització (*Authorization Realms*), com poden ser: Fitxers descriptors en XML, JAAS, Base de Dades Relacionals, connectors LDAP, etc.

A l'hora d'implementar, ho podrem fer de manera declarativa, és a dir, a partir de fitxers declaratius en el servidor, com el *web.xml*, o de manera programada dintre de la pròpia aplicació.

#### 4.2.2. Integritat de les dades

En el camp de la seguretat informàtica aquesta propietat és moltes vegades oblidada i és una de les més importants. Es tracta doncs que les dades que es transmeten entre navegador i servidor (en el cas d'una aplicació Web) no estiguin afectades de modificacions o supressions. Per dur a terme aquesta propietat, els servidors Web es valen de l'ús de connexions SSL que garanteixen la transmissió segura de les dades. El protocol SSL és un sistema dissenyat i proposat per Netscape Communications Corporation, i es troba en la pila OSI entre els nivells TCP/IP i els protocols HTTP,FTP,SMTP entre altres. Proporciona serveis de seguretat xifrant les dades intercanviades entre el servidor i el client mitjançant un algorisme de xifratge simètric, típicament RC4 o IDEA, i xifrant la clau de sessió de RC4 o IDEA amb un algorisme de xifratge de clau pública, típicament RSA.

La clau de sessió s'utilitza per xifrar les dades que s'intercanvien servidor web i client, i aquesta és diferent en cada transacció produïda.

#### 4.2.3. Confidencialitat

La confidencialitat generalment és la única part que molta gent considera quan pensa en seguretat informàtica. La confidencialitat es basa en la privacitat de les dades durant la transmissió i és implementada a partir de la **Criptografia**.

El servidors web també ofereixen aquest servei a partir de comunicacions utilitzant SSL.

#### 4.2.4. JAAS Realm

JAAS són les sigles de **Java Authentication and Authorization Service** que va ser introduïda com una extensió al llenguatge de programació Java 2 SDK, edició estàndard (J2SDK). Actualment JAAS està en la versió 1.3 i està integrada de manera nativa en el J2SDK versió 1.4.

La implementació de JAAS té dos propòsits bàsics:

1. **Autenticació dels usuaris:** determinar de manera fiable i segura qui està executant el codi Java, sense tenir en compte si es tracta d'una aplicació, un applet java, un bean de java o un servlet.

2. **Autorització dels usuaris**, per assegurar que tenen permisos que són requerits per dur a terme diverses accions dins de l'aplicació.

JAAS internament és una implementació en Java de l'estàndard PAM (*Pluggable Authentication Module*), figura 28. Bàsicament PAM és un estàndard que defineix un conjunt de mòduls que tenen com a propòsit final fer que la implementació de seguretat sigui *independent* de l'aplicació que es vulgui realitzar. D'aquesta manera es té el codi modular, la qual cosa permet una major eficiència i depuració d'errors per part del programador.

Cal destacar que PAM utilitza un format de fitxer estàndard per tal de configurar el procés d'autenticació. En la implementació particular que fa JAAS de PAM podem veure que també utilitza un fitxer descriptiu que serà explicat en més profunditat en els apartats següents.

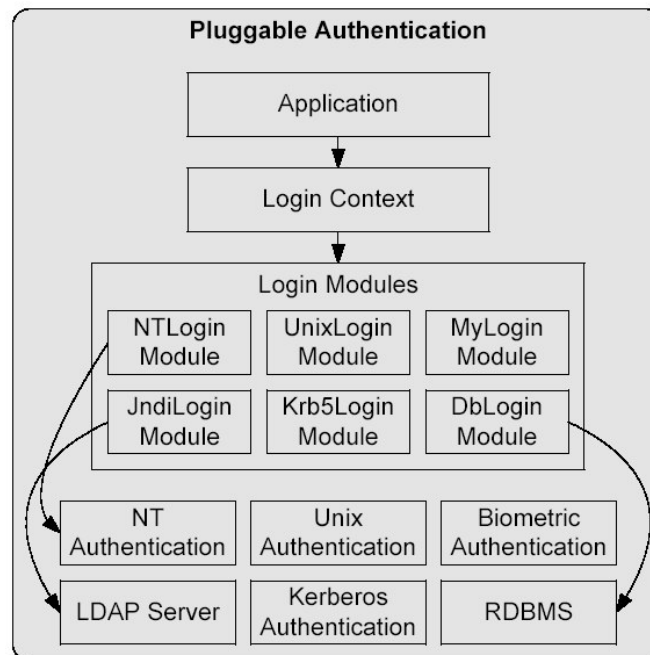


**Figura 28 :** Estàndard PAM

Les aplicacions poden activar el procés d'autenticació instanciant les classes següents:

**LoginContext:** determina quina tecnologia es vol fer servir per al procés d'autenticació. Aquest paràmetre s'estableix a partir del fitxer de configuració.

**LoginModule:** aquest mòdul és l'encarregat de realitzar pròpiament la validació de l'usuari a partir del seu nom d'usuari i contrasenya. Altres sistemes, per exemple, permeten realitzar la validació a partir de sistemes de veu.

Quan un usuari o servei executa un codi que està autenticat, el component d'autorització de JAAS treballa en conjunció amb el nucli de Java 2 per tal de protegir l'accés als diferents recursos.

#### 4.2.4.1. Descripció de les classes més importants

El nucli de JAAS està format per una sèrie de classes i interfícies que es poden agrupar en tres categories:

- **Classes comunes:** estan formades per les classes Subject, Principal, Credential.
- **Classes per a l'autenticació:** hi pertanyen les classes LoginContext, LoginModule, CallbackHandler, Callback.
- **Classes per a l'autorització:** Són les classes Policy, AuthPermission, PrivateCredentialPermission.

A continuació es detallen cada una d'elles.

##### ➤ Common Classes (Classes comunes)

Aquestes classes contenen la informació necessària utilitzada i compartida per les classes d'autenticació i autorització, i representen bàsicament a l'entitat que vol fer el procés de validació.

La classe principal s'anomena *javax.security.auth.Subject* i representa informació relacionada sobre una entitat, normalment una persona.

##### *javax.security.auth.Subject*

Per autoritzar l'accés a un recurs, l'aplicació, primerament, necessita autenticar qui realitza aquesta petició. JAAS defineix el terme subjecte per representar qui fa realment aquesta petició, per exemple, una persona o un servei.

Una vegada el subjecte és autenticat, llavors es relaciona aquest amb el que s'anomena identitats associades o **Principals**. Un subjecte pot tenir diversos Principals. Per exemple, una persona com a '*subject*' pot tenir diferents '*principals*', com, per exemple, *principal-nom*, *principal-dni*, etc., que el diferencien de la resta de '*subjectes*'.

Per altra banda un '*subjecte*' també té relacionats uns atributs de seguretat, que reben el nom de *credentials*. Aquesta informació té una protecció especial; per tant, es val de la criptografia (certificats, claus, etc.) per tal d'assegurar el seu contingut, fins i tot, l'accés per tal de modificar aquesta informació (mitjançant '*setters*' i '*getters*') que es fa a través de l'establiment de diferents permisos. Els tipus de credentials poden ser de tipus públic o privat.

Les signatures (noms dels constructors) d'aquesta classe són:

```
public Subject();
public Subject (boolean readOnly, Set principals, Set pubCredentials, Set
privCredentials)
```

### **javax.securityPrincipal**

Com anteriorment s'ha comentat, la classe principal va associada a la classe subject. Si el procés d'autenticació és correcte, llavors aquesta classe representa la identitat del subjecte.

### **javax.security.Credentials**

La classe 'credentials' pública i privada no és part de la llibreria JAAS, per tant qualsevol classe pot representar un credencial. Així doncs, és el programador qui decideix fer els seus propis credencials, sempre i quan implementin les interfícies de credencials que són: *Refreshable* i *Destroyable*.

### ➤ **Classes per a l'autenticació**

Per autenticar qualsevol subjecte (usuari o servei) es realitzen els següents passos:

1. L'aplicació instancia un objecte del tipus *LoginContext*.
2. L'objecte *LoginContext* consulta un fitxer de configuració per carregar tots els *LoginModule(s)* configurats per l'aplicació.
3. L'aplicació invoca el mètode *login* en el objecte *LoginContext*.
4. El mètode *login* invoca a tots els *LoginModule*, on a la vegada cada un d'ells autenticar el subjecte. Si el procés d'autenticació ha estat correcte, llavors cada *LoginModule(s)* associa al subjecte els seus principals i credencials.
5. Finalment l'objecte *LoginContext* retorna l'estat de l'autenticació a l'aplicació.
6. Si l'autenticació ha estat satisfactòria el *subjecte* és vàlid dintre del seu context (*LoginContext*).

### **javax.security.auth.login.LoginContext**

Aquesta classe dona els mètodes bàsics per l'autenticació del subjecte, i a més a més permet la **independència** entre l'aplicació i la tecnologia d'autenticació utilitzada.

La classe *LoginContext* consulta el fitxer de configuració de JAAS (comentat al següent apartat del capítol), per tal de realitzar la configuració correcta dels *LoginModule(s)*. D'aquesta manera permet al programador tenir diferents *LoginModule(s)* per tal de fer l'autenticació en la seva aplicació sense necessitat de realitzar cap modificació sobre el codi de l'aplicació, aconseguint un sistema modular.

La signatura de '*LoginContext*' és la següent:

```
Public LoginContext (String name) throws LoginException;
Public LoginContext (String name, Subject subject) throws LoginException;
```

```

Public LoginContext (String name, CallbackHandler callbackHandler) throws
LoginException;
Public LoginContext (String name, Subject subject, CallbackHandler callbackHandler)
throws LoginException;

```

És important destacar el funcionament del *CallbackHandler*. Aquest és necessari dintre del mètode *Login* de l'objecte *LoginContext*, i s'utilitza principalment perquè aquest pugui fer la crida als nostres objectes del tipus *LoginModule(s)*. Si l'autenticació finalment és correcta, es podrà accedir al subject a partir del mètode:

```
Public Subject getSubject();
```

Finalment per acabar el procés d'autenticació s'invocarà el mètode:

```
Public void logout() throws LoginException;
```

A continuació es mostra un petit codi en Java que permet veure les crides necessàries per tal d'autenticar un subject (Usuari):

```

LoginContext lc = new LoginContext("entryUser");
try {
 // authenticate the Subject
 lc.login();
 System.out.println("authentication successful");
 // get the authenticated Subject
 Subject subject = lc.getSubject();
 ...
 // all finished -- logout
 lc.logout();
} catch (LoginException le) {
 System.err.println("authentication unsuccessful: " + le.getMessage());
}

```

### **javax.security.auth.spi.LoginModule (interface)**

La interfície *LoginModule* dona als programadors l'opció d'implementar diferents tecnologies d'autenticació que poden ser inserides com un mòdul dintre de l'aplicació que es desenvolupa. Per exemple, un tipus de *LoginModule* podria ser un sistema usuari/contrasenya basat en autenticació de formulari (based-form authentication), o un altre cas seria una interfície sobre un dispositiu hardware, per exemple, targetes intel·ligents o dispositius biomètrics.

### **javax.security.auth.callback.CallbackHandler (interface)**

Moltes vegades és necessari que l'objecte *LoginModule* es pugui comunicar amb l'usuari per obtenir la informació necessària per l'autenticació. Per tant, l'objecte *LoginModule* utilitza el *CallbackHandler* per aquest propòsit.

En aquest cas el programador implementa la interfície *CallbackHandler* i li passa la informació al *LoginContext* (comentat a l'apartat anterior); per tant l'objecte *LoginModule* utilitzarà la interfície per demanar les dades d'autenticació, per exemple, la contrasenya del usuari o el PIN d'una targeta magnètica

La interfície *CallbackHandler* només té un mètode que es detalla a continuació:

```
Void handle(Callback[] callbacks) throws java.io.IOException,
UnsupportedCallbackException;
```

### ➤ Classes per l'autorització

Per tal que JAAS pugui garantir accés controlat basat en permisos, cal que es compleixin els següents punts:

- L'usuari ha d'estar autenticat.
- Una vegada l'usuari ha estat autenticat, se li assignarà un *Subject* associat a un context.
- S'han de definir unes polítiques de seguretat; per tal de realitzar aquest procés s'utilitzen les classes *policy*, *Authpermission* i *PrivateCredentialPermission*.

#### *javax.security.Policy*

Es tracta d'una classe abstracta que representa les polítiques d'accés a un determinat recurs. Des de la versió del J2SDK v1.4 la classe *policy* dona suport a la classe *Principal* del procés d'autenticació comentat anteriorment.

#### *javax.security.auth.AuthPermission*

Aquesta classe s'encarrega d'encapsular els permisos bàsics que requereix la llibreria JAAS, i és usada per establir l'accés als objectes *Policy*, *Subject* i *LoginContext*.

La signatura de la classe és:

```
Public AuthPermission (String name);
Public AuthPermission (String name, String actions);
```

#### *javax.security.auth.PrivateCredentialPermission*

Aquesta classe protegeix l'accés als credencials privats de l'objecte *Subject*. El seu únic constructor és el següent:

```
Public PrivateCredentialPermission (String name, String actions);
```

#### 4.2.4.2. Configuració

El sistema d'autenticació de JAAS es modela com un sistema independent de l'aplicació, per tant aquesta és independent de la tecnologia que es faci servir per tal de dur a terme el procés de validació.

La manera de configurar el procés d'autenticació es defineix en la classe *javax.security.auth.login.Configuration*, per defecte aquesta informació s'obté a partir d'un fitxer de text ( LCF ). A continuació es detalla el fitxer de configuració.

El LCF consisteix en un fitxer que conté una o vàries entrades, cadascuna d'elles representa la tecnologia d'autenticació que es fa servir per una o vàries aplicacions.

L'estructura de cada entrada és la següent:

```
<name used by application to refer to this entry> {
 <LoginModule> <flag> <LoginModule options>;
 <optional additional LoginModules, flags and options>;
};
```

Cada entrada en el fitxer de configuració consisteix en un nom seguit per una o vàries entrades de l'etiqueta <LoginModule>. A continuació s'exposa un exemple on es veu de manera més clara:

```
Sample {
 sample.module.SampleLoginModule required debug=true;
};
```

En l'exemple, l'etiqueta d'entrada rep el nom de 'sample'; aquesta especifica quin serà el LoginModule que es farà servir per autenticar l'usuari. En aquest cas serà la classe 'SampleLoginModule' qui definitivament acceptarà o no com a vàlid un usuari.

El nom de l'entrada (en el cas de l'exemple 'sample') en el LCF, és el nom que l'aplicació farà servir quan s'instancii la classe *LoginContext*.

A continuació es detallen els continguts de l'etiqueta <LoginModule>:

- *LoginModule* :

Especifica la classe que implementa la tecnologia d'autenticació. En especial, es tracta d'una subclasse de LoginModule inclosa en el paquet *javax.security.auth.spi*.

Típicament aquesta classe verifica el procés mitjançant l'ús d'un nom d'usuari i una contrasenya associada, tal com s'ha implementat en aquest projecte. Però, podem trobar diferents tipus de validacions com la que ens ofereix Sun Microsystems en el JRE (Java

Rootkit Environment), totes elles les podem trobar en el paquet **com.sun.security.auth** i es detallen a continuació:

*O JndiLoginModule*  
*O KeyStoreLoginModule*  
*O Krb5LoginModule*  
*O NTLoginModule*  
*O UnixLoginModule*

- *Flag.*

Indica per al *LoginModule* on s'ha definit el tag *flag*, si la validació és de caràcter : “required”, “requisite”, “sufficient”, “optional”.

- *Altres opcions.*

En el cas que la implementació de la classe *LoginModule* necessiti obtenir paràmetres, aquests es poden passar a partir d'una llista de valors separats per comes. Per exemple, si el *LoginModule* necessita opcions per al suport de depuració o test. La manera correcta d'especificar els paràmetres és fer-ho a partir del parell clau-valor. Per exemple: *debug=true*.

El fitxer de configuració es pot especificar de dues formes:

### **1. A partir de la línia de comanda.**

Ex. Java -Djava.security.auth.login.config==sample\_jaas.config Sample.java

### **2. En el fitxer de propietats de seguretat del Java.**

Consisteix en establir la propietat ‘login.config.url.*n*’ en el fitxer de propietats de Java. Aquest fitxer es localitza en el directori lib/security del J2RE.

El paràmetre ‘*n*’ indica un número enter consecutiu començant per 1, sempre que es vulgui especificar més d'un fitxer de configuració. Ex. Login.config.url.1, etc.

Ex. Login.config.url.1 = file:c:\sample\_jaas.config

#### 4.2.5. LCF: fitxer de configuració

La configuració del fitxer de configuracions per al procés de validació s'ha realitzat a partir del **manager** (figura 29) que ofereix el servidor web Apache Tomcat 5. En aquest cas en l'apartat de propietats de l'Apache Tomcat s'ha establert una opció en els paràmetres de java que indica el camí (*path*) al fitxer de configuracions.

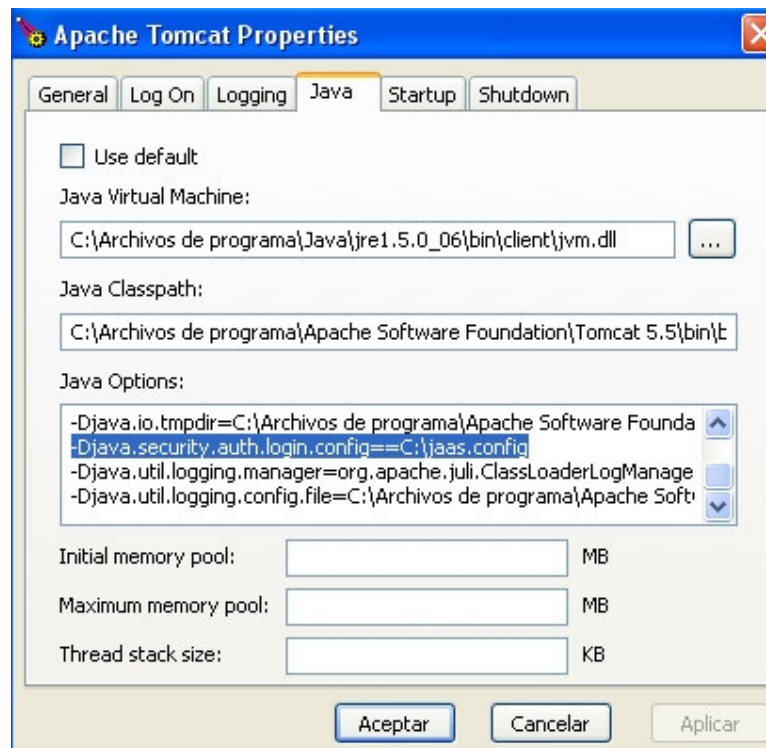


**Figura 29 :** Configurador en Apache Tomcat

El contingut del fitxer jaas.config es detalla a continuació:

```
Secure_cfg {
 mod_secure.RdbmsLoginModule required debug="true"
 url="jdbc:mysql://localhost/jaasdb?user=root&password=admin"
 driver="org.gjt.mm.mysql.Driver";
};
```

#### **4.2.6. Justificació de JAAS en el PFC**

El procés de validació d'usuaris en el pfc era un tema important i calia donar-li un aspecte de robustesa i fiabilitat en l'aplicació. Per realitzar aquesta feina es podia haver realitzat un sistema típic basat en formularis HTML, però vaig creure que l'aplicació en un futur podria necessitar més funcionalitats que les que es van trobar en un principi. Per tant, vaig decidir la implementació utilitzant les llibreries JAAS, que proporcionen un ventall ampli de mecanismes per realitzar validacions d'usuaris. D'aquesta manera s'aconsegueix un sistema obert a canvis o modificacions necessàries per noves aplicacions d'una manera fàcil i eficient. A més a més, em donava l'oportunitat de treballar en aquesta especificació que desconeixia i poder aprendre noves eines.

## 5. JOC DE PROVES

Per tal de realitzar el joc de proves, no s'imprimirà cadascun dels possibles subplans que pot tenir una guia, perquè sinó el document seria molt extens. Per poder veure la correcta representació de la guia de pràctica clínica, es presentaran a continuació cadascuna de les guies en **un sol nivell**.

### 5.1. Guia pràctica clínica: AHDH Model

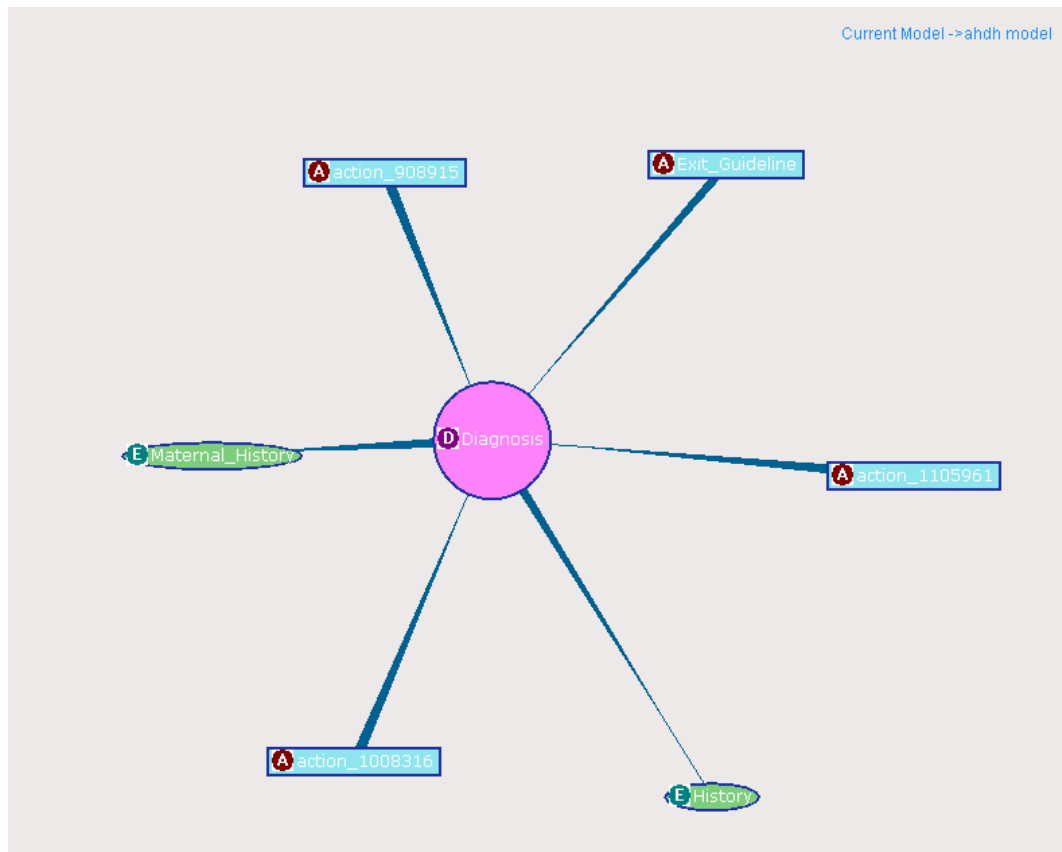


**Figura 30:** Representació de la Gpc “AHDH Model”

Aquest model de guia està relacionada amb el tractament d'un desordre en la hiperactivitat. El pla principal està format pels següents components:

- I. Les tasques de decisió contenen la realització d'un diagnòstic.
- II. Les tasques de pregunta contenen informació dels historials.
- III. Les tasques d'acció inclouen abandonar la guia.
- IV. S'hi presenta una tasca pla, aquesta inclou tres accions :
  - acció\_1008316
  - acció\_908915
  - acció\_1105961

## 5.2. Guia pràctica clínica: Cancer Diagnosis Model

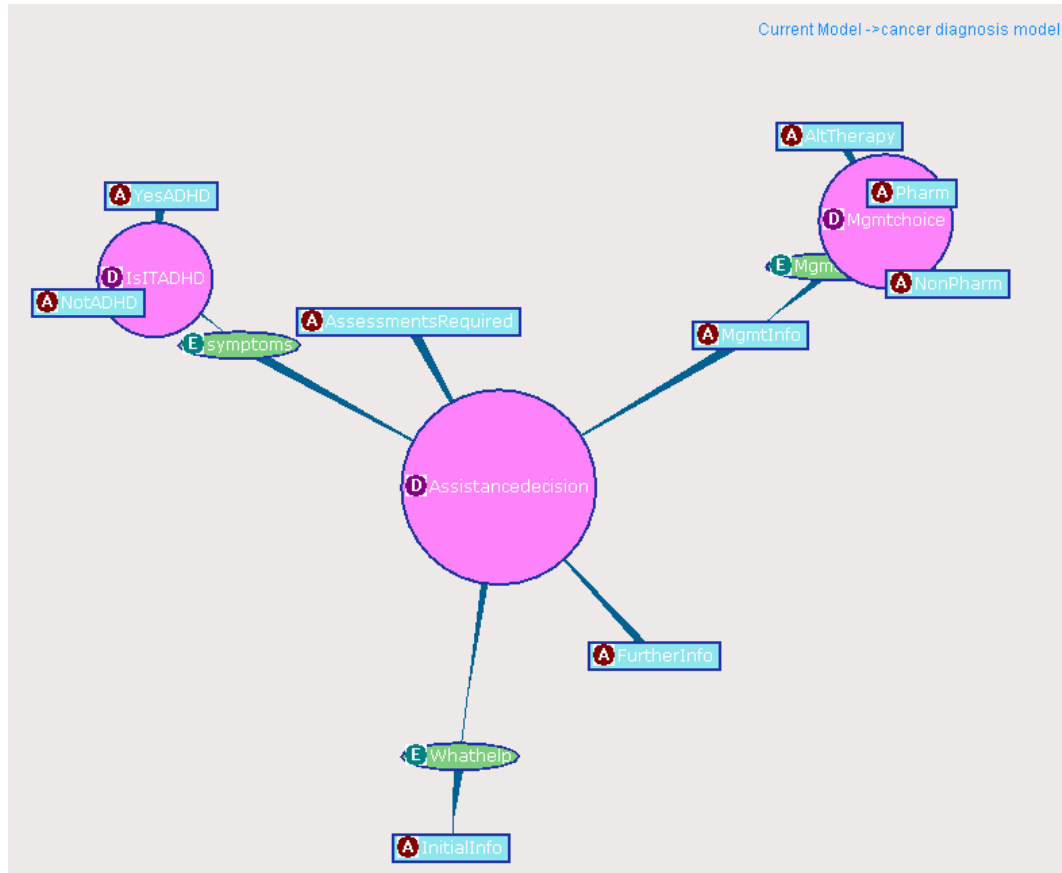


**Figura 31:** Representació de la Gpc “Cancer Diagnosis Model”

Aquesta guia representa el model del diagnòstic del càncer. En el seu pla principal trobem les següents tasques:

- I. Tasca de decisió : d’assistència
- II. Tasca de pregunta : Ajuda (aquesta et porta a una tasca d’acció “Informació inicial”)
- III. Hi ha quatre tasques de subplans, aquestes són:
  - Management Pla: aquest inclou un subpla que es detalla a continuació:
    - Representació relacionada amb informació MGMT.
    - Una tasca de pregunta, possibles opcions ?
    - Una tasca decisió per escollir entre les tres possibles accions a fer: realitzar medicació o no, o bé, realitzar una teràpia alternativa.
  - Sources Plan: Conté únicament una acció “Further Info”
  - DiagnosisPlan: Aquest inclou un subpla que es detalla a continuació:

- Una tasca decisió, preguntant és AHDH? (gpc de la hiperactivitat).
- Dues accions que defineixin Sí o No.
- Una tasca pregunta referent als símptomes.
- AssessmentPlan: Inclou una tasca acció “Assessment Required”

### 5.3. Guia pràctica clínica: Dvt Model

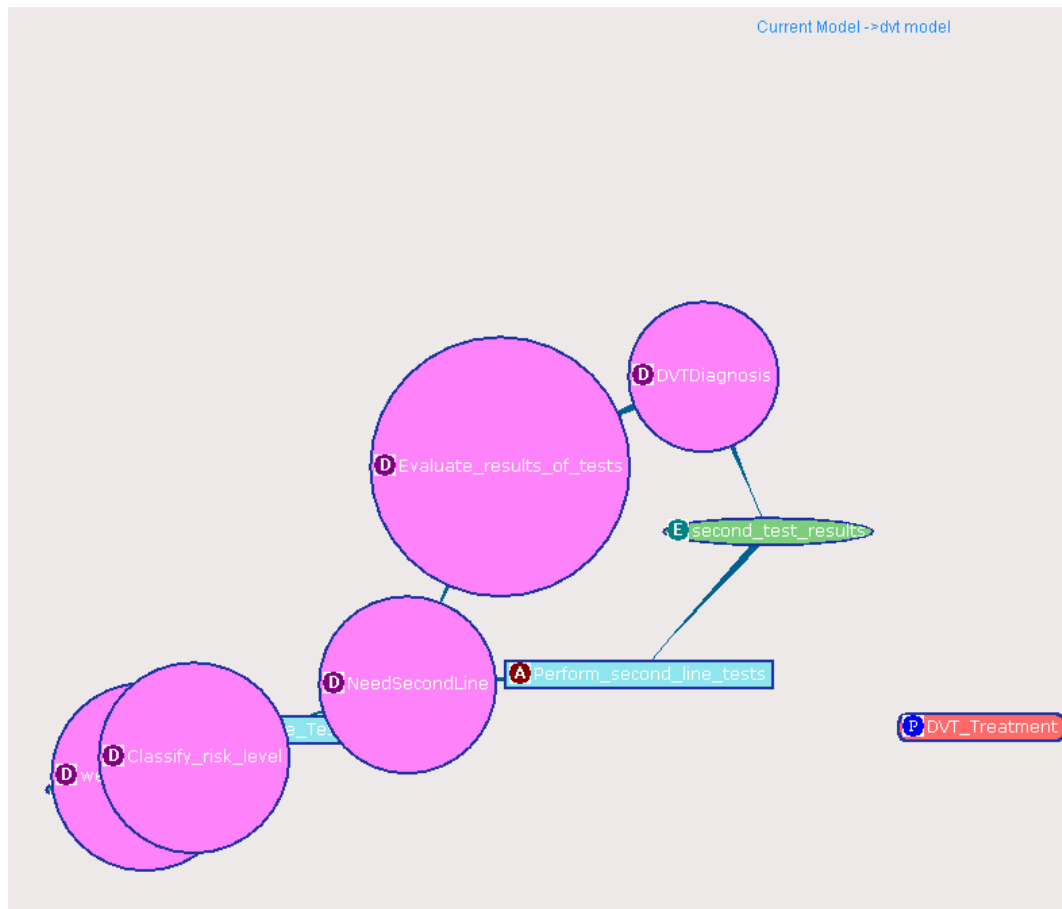


**Figura 32:** Representació de la Gpc “Dvt Model”

La guia DVTModel en el seu pla principal conté dos plans:

- I. DVT Diagnosis: conté un subpla amb la informació següent:
  - Una tasca decisió referent al diagnòstic.
  - Un tasca pla referent a un test per el DVT, incloent-hi.
    - Una tasca acció que representa una primera línia de test.
    - Arribant a una tasca decisió sobre una segona línia de test.
    - A partir de la informació anterior, trobem avaluar els resultats dels test o en altre cas efectuar la segona línia de test i avaluar els resultats.
  - Una tasca amb el risc del DVT. Incloent-hi.
    - Obtenció del historial, com a tasca de pregunta.
    - Prendre la decisió d’analitzar els criteris de risc.
    - Seguidament, prendre la decisió de classificar el risc.
- II. DVT Tractament: És buit.

### 5.4. Guia pràctica clínica: Hypertension Model

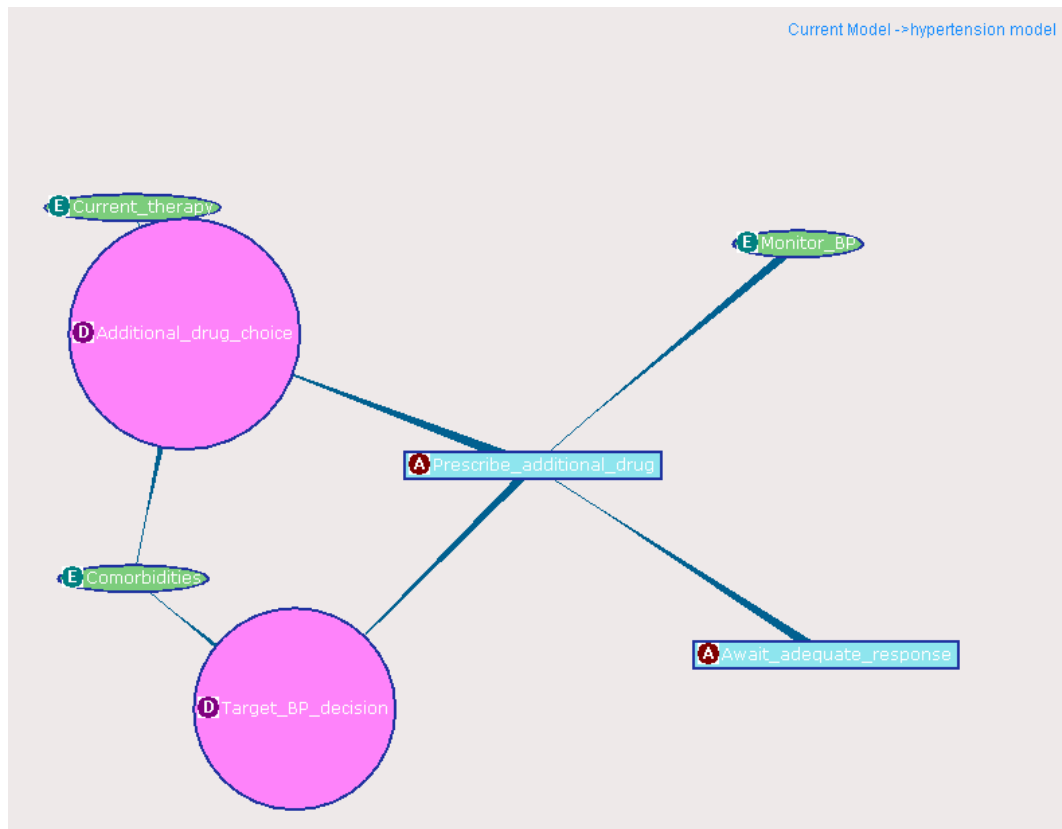


**Figura 33:** Representació de la Gpc “Hypertension Model”

Aquesta guia modela la malaltia de la hipertensió. En el seu pla principal tenim la següent informació:

- I. Dues tasques que representen decisió, referent a subministrar un medicament addicional o bé, sobre “Target BP decision”.
- II. Una tasca d’acció referent a subministrar un medicament addicional.
- III. Dues tasques de pregunta que representen la teràpia actual i “Comorbidities” respectivament.
- IV. Un subpla anomenat “Assess Response” que conté:
  - Esperar una resposta adequada com a tasca d’acció.
  - “Monitor\_BP” com a tasca de pregunta.

### 5.5. Guia pràctica clínica: Weigth loss Model

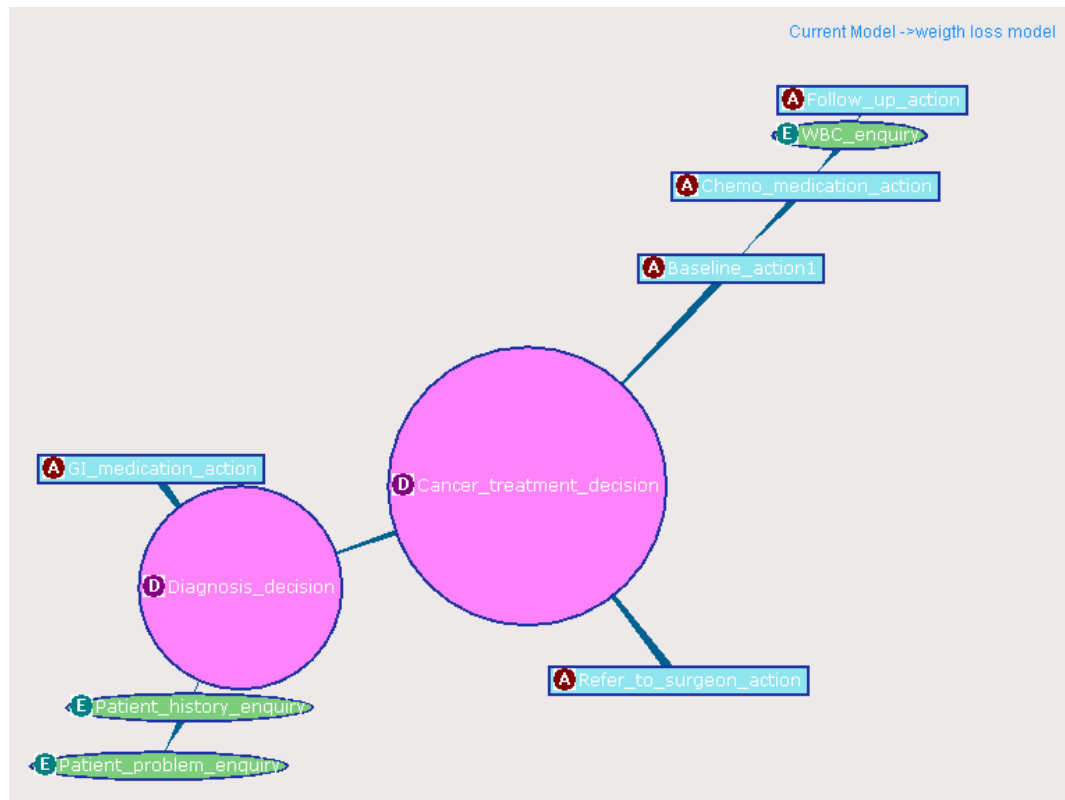


**Figura 34:** Representació de la Gpc “Weigth loss Model”

Aquesta guia modela la pèrdua de pes. La informació que presenta el pla inicial és la següent:

- I. Dues tasques preguntant referent als problemes i historials del pacient respectivament.
- II. Dues decisions que afecten a la realització d'un tractament per al càncer o bé, realitzar un diagnòstic.
- III. S'inclouen tres subplans que es detallen a continuació:
  - Tractament gastrointestinal : Conté únicament una acció anomenada “GI medicació”
  - Pla de cirurgia: Inclou una acció referent al cirurgia.
  - Chemotherapy Plan.

## 5.6. Guia pràctica clínica: Cough Model

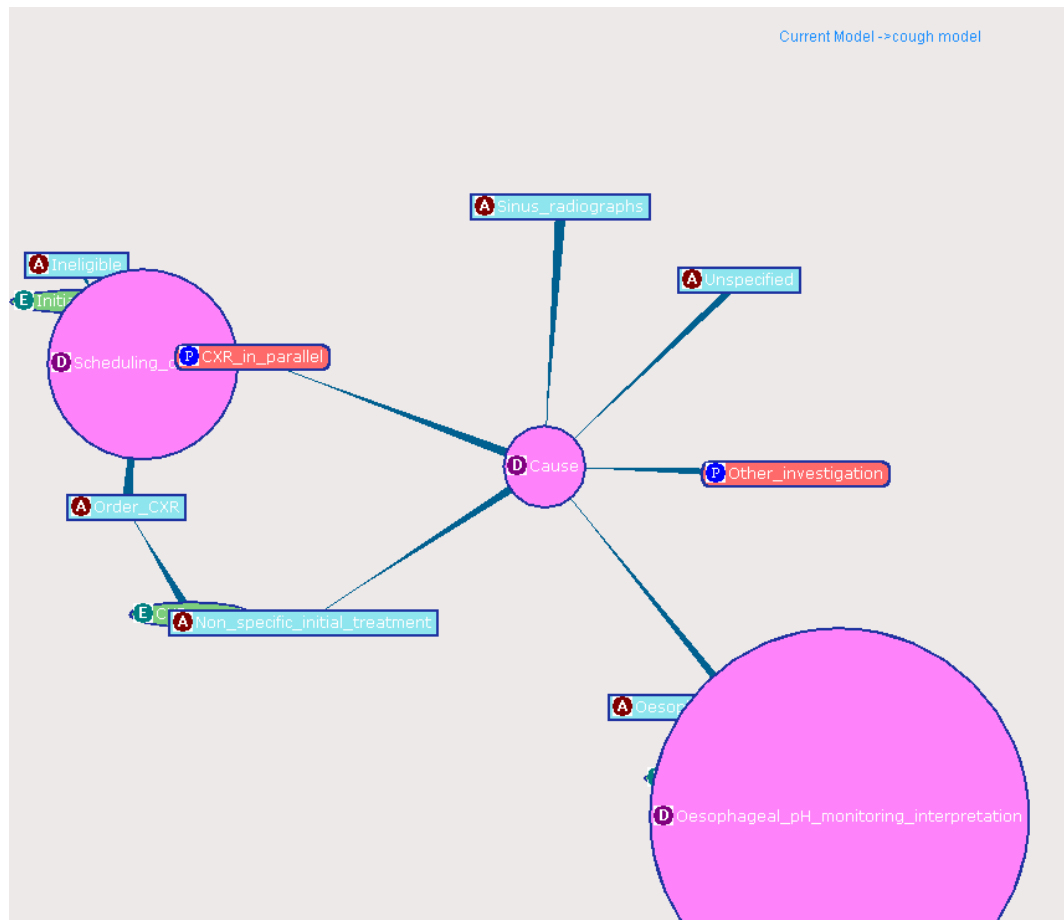


**Figura 35:** Representació de la Gpc “Cough Model”

Aquesta guia modela la tos crònica. Aquesta guia inclou en el seu pla principal les següents tasques:

- I. Una tasca acció anomenada “Ineligible”
- II. Una tasca pregunta referent a un primer anàlisi.
- III. S’inclouen dos subplans detallats a continuació:
  - Proves CXR i RX
  - Investigacions



## 6. CONCLUSIONS

Durant l'elaboració del projecte, hi hagut moments difícils que s'han anat resolent de la millor manera, amb moltes hores d'estudi i implementació. Crec que en el desenvolupament del projecte l'etapa d'estudi i anàlisi de com fer les coses ha estat molt més gran que la implementació en sí. En comparació amb el projecte que vaig desenvolupar en l'enginyeria tècnica, que va ser un projecte molt bonic però bastant mecànic, en aquest he comprovat que hi ha moltes maneres de fer una cosa, i que, a vegades, la millor solució no és la més complexa sinó la més fàcil. És a dir, durant l'etapa d'estudi vaig comprendre moltes eines que, per exemple, es podien utilitzar per fer una validació d'usuari, totes elles complien el propòsit de validació, però unes tenien uns avantatges que altres no oferien.

En l'elecció de la llibreria gràfica més o menys va ser el mateix, el número d'eines disponibles és bastant gran (com es mostra en la taula comparativa de l'apartat 3.8.1) i l'elecció d'una o altra depèn en gran part de la necessitat del projecte. En certa manera, la decisió d'elecció no és fàcil i representa en gran mesura el treball principal d'un cap de projecte en un grup de programadors. També cal dir que, gràcies a la immensa base de dades d'informació que és Internet, pots arribar a prendre una decisió, en la majoria dels casos, bastant encertada.

També vull destacar que, durant l'elaboració del projecte, he pogut treballar amb noves eines de les quals fins ara ni tan sols havia sentit a parlar; la llibreria JAAS per a la implementació de la validació d'usuari, o la llibreria gràfica TouchGraph, en són alguns exemples. També les tecnologies utilitzades, que en algun cas tenia "rovellades" per manca d'utilització, m'han permès tornar a utilitzar-les i ha estat molt bé, com, per exemple, els JSP, Java. Altres, m'ha tocat aprendre-les o profunditzar en elles, com poden ser els Applets de Java, o llenguatges de script com Javascript.

Pel que fa als objectius que es van definir en un inici per al projecte, crec que s'han complert i s'han finalitzat correctament. Tot i que, sempre sembla que s'haguessin pogut afegir noves funcionalitats, aquestes queden pendents per a treballs futurs i algunes d'elles s'han intentat mig definir, deixant part del codi implementat perquè sigui fàcil incorporar-les en un futur, per exemple, la interpretació d'informació associada a les relacions del graf (arestes) amb colors i text.

Finalment, vull donar les gràcies al *GRUSMA* per donar-me l'oportunitat de realitzar un projecte dins de l'àmbit de les seves línies d'investigació, així com al tutor que he tingut durant l'elaboració del projecte, en David Isern, que, gràcies als seus consells, aclariments i a la seva tenacitat, m'ha ajudat a la finalització d'aquest PFC.

L'objectiu del present projecte ha estat la representació de guies de pràctica clínica a través d'una interfície Web i un sistema de validació d'usuari també via Web. A continuació es numeren algunes noves funcionalitats que es podrien realitzar en treballs futurs.

- Representació en el graf de l'estat actual on es troba per exemple un pacient en una gpc personalitzada. Des d'aquest punt de vista seria interessant veure si el

pacient ha fet tal prova o tal altre, així com gràficament veure un sub-graf associat.

- Interactuar amb el graf per tal de modificar dades de la GPC. En aquest cas seria interessant realitzar canvis visualment sobre la gpc que es representessin en la informació de la gpc en la Base de Dades.
- Utilitzar el visualitzador de gpc's com a eina de creació de guies de manera visual, tot i que seria un projecte complex, es podrien afegir nodes i arestes per tal de crear la gpc i emmagatzemar-la a la Base de Dades.

## 7. BIBLIOGRAFIA I RECURSOS UTILITZATS

[1] Projecte HeCaSe (Health Care Services)  
Multi Agent System Group, Universitat Rovira i Virgili  
<http://grusma.etse.urv.es/hecase/index.htm>

[2] David Isern Alarcón  
Projecte Final de Carrera, Escola Tècnica Superior Enginyeria (ETSE)  
Universitat Rovira i Virgili (URV 2001)  
*Disseny i implementació d'un SMA per proporcionar serveis mèdics dins el projecte AgentCities.*

[3] Josep Domingo i Ferrer, Jordi Herrera i Joancomartí  
Editorial UOC, 1999. ISBN 84-8429-007-7  
Collection Manuals nº 31, Barcelona  
*Criptografia per als Serveis Telemàtics i el Comerç Electrònic*

[4] ECMA International Standards  
<http://www.ecma-international.org>

A continuació es detallen alguns dels llibres utilitzats en el pfc.

- Steven Holzner  
Anaya Multimedia  
ISBN 84-415-1037-7 , 922 pàgines.  
*Java 2*

- Danny Goodman  
Hungry Minds, 2001  
ISBN 0-7645-4718-6, 2177 pages  
*JavaScript Bible, Gold Edition*

- Justin Gehtland, Bruce A. Tate  
O'Reilly , June 2004  
ISBN 0596006764, 250 pages  
*Better, Faster, Lighter Java*

- Vivek Chopra, Jon Eaves, Rupert Jones, Sing Li, John T. Bell  
Publisher by Wrox  
*Beginning JavaServer Pages*

- David M. Geary  
Publisher: Prentice Hall PTR, First Edition May 01, 2001  
ISBN 0-13-030704-1, 508 pages  
*Advanced JavaServer Pages*

- Vivek Chopra, Amit Bakore, Jon Eaves, Ben Galbraith, Sing Li, Chanoch Wiggers  
Publisher by Wrox  
*Professional Apache Tomcat 5*

- David R. Sutton, John Fox  
Advanced Computation Laboratory, Cancer Research UK  
*The Syntax and Semantics of the PROforma guideline modelling language*

## 7.1. RECURSOS UTILITZATS

### 7.1.1. Software

A continuació es detallen molt breument els diferents materials que s'han utilitzat per al present projecte, es proporciona una descripció i una referència web.

#### 7.1.1.1. Llibreries

- ✓ *JAAS*: llibreria per a la validació d'usuaris. Va ser creada per Sun Microsystems, i en les darreres versions del JDK s'inclou per defecte. Presenta una API força completa i bastants exemples d'utilització. Per a més informació consultar la web: <http://java.sun.com/products/jaas/>



- ✓ *TouchGraph*: llibreria que permet la creació de grafs i la interacció del seu contingut a partir d'events. Actualment hi ha poca informació de la llibreria, però, en canvi, l'API que proporciona i el codi bastant entenedor són suficients. Per a més informació la web oficial de TouchGraph és: <http://www.touchgraph.com/>  
La pàgina web del projecte en SourceForge : <http://touchgraph.sourceforge.net/>



- ✓ *Tallis*: llibreria escrita en Java que permet carregar una guia de pràctica clínica definida en PROforma. A partir d'aquesta es pot treure tota la informació de la GPC.



- ✓ *Driver Mysql*: llibreria en Java que implementa la connexió JDBC amb la base de dades Mysql.



- ✓ *Llibreries Java*: inclou totes les llibreries necessàries per la implementació, Swing, Awt principalment. Per a més informació la web oficial de Sun Microsystems: <http://java.sun.com/>

### 7.1.1.2. Programes



- ✓ *MySQL BD:* base de dades relacional, en aquest cas se n'ha utilitzat la darrera versió 5.0.



- ✓ *IDE Eclipse:* entorn de desenvolupament per aplicacions Java i altres. És un projecte de codi obert i molt utilitzat per la comunitat Java, a més a més hi ha un gran nombre de plugins que s'hi poden incorporar.



- ✓ *JBUILDER:* eina molt completa per a crear qualsevol tipus d'aplicació Java, conté moltes extensions, diagrames, etc. JBUILDER és un software de la companyia Borland.  
Per a més informació : <http://www.borland.com/us/products/jbuilder/index.html>



- ✓ *Apache+Tomcat:* es tracta d'un servidor d'aplicacions web, que inclou suport per a l'especificació Servlet 2.0, per tant, inclou JSP. Pertany al projecte de software lliure Apache. Per a més informació : <http://www.apache.org/>



- ✓ *ANT:* llibreries per a la gestió de projectes creats en Java, permet compilar, executar, generar documentació, etc. També forma part del projecte Apache. Per a més informació: <http://ant.apache.org/>



- ✓ *Poseidon for UML:* programa per a la creació de diagrames UML, entre ells, diagrama de classes, de paquets, ..., en Java. Per a més informació la web de l'empresa: <http://gentleware.com/index.php>



- ✓ *EMS Manager for MySQL:* programa que permet gestionar i administrar una base de dades MySQL de forma visual.  
Per a més informació: <http://www.sqlmanager.net/>

## ANNEX

### Gpc Hypertensio en PROforma

```

/** PROforma (plain text) version 1.3.43 */
plan :: 'Hypertension' ;
 caption :: "Hypertension";
 component :: 'Current_therapy' ;
 caption :: "Current_therapy";
 task_definition :: 'Current_therapy' ;
 number_of_cycles :: 1;
 ltwh :: 33,100,42,36;
 component :: 'Comorbidities' ;
 caption :: "Comorbidities";
 task_definition :: 'Comorbidities' ;
 number_of_cycles :: 1;
 ltwh :: 203,168,42,36;
 component :: 'Additional_drug_choice' ;
 caption :: "Additional_drug_choice";
 task_definition :: 'Additional_drug_choice' ;
 schedule_constraint :: completed('Comorbidities') ;
 schedule_constraint :: completed('Current_therapy') ;
 number_of_cycles :: 1;
 ltwh :: 182,56,36,36;
 component :: 'Target_BP_decision' ;
 caption :: "Target_BP_decision";
 task_definition :: 'Target_BP_decision' ;
 autonomous :: yes ;
 schedule_constraint :: completed('Comorbidities') ;
 number_of_cycles :: 1;
 ltwh :: 358,176,36,36;
 component :: 'Prescribe_additional_drug' ;
 caption :: "Prescribe_additional_drug";
 task_definition :: 'Prescribe_additional_drug' ;
 schedule_constraint :: completed('Additional_drug_choice') ;
 schedule_constraint :: completed('Target_BP_decision') ;
 number_of_cycles :: 1;
 ltwh :: 478,64,36,36;
 component :: 'Assess_response' ;
 caption :: "Assess_response";
 task_definition :: 'Assess_response' ;
 schedule_constraint :: completed('Prescribe_additional_drug') ;
 number_of_cycles :: -1 ;
 ltwh :: 576,136,48,36;
end plan.

decision :: 'Additional_drug_choice' ;
 caption :: "Additional drug choice";
 candidate :: 'ACEI' ;
 argument :: confirming,(Type_1_Diabetes = Yes and Proteinuria != None) attributes
 argument_name :: ' (Type_1_Diabetes = Yes and Proteinuria <> None) ' ;
 end attributes
 ;
 argument :: confirming,(Heart_failure = Yes) attributes
 argument_name :: ' (Heart_failure = Yes) ' ;
 end attributes
 ;
 argument :: confirming,(MI = Yes and Systolic_dysfunction = Yes) attributes
 argument_name :: ' (MI = Yes and Systolic_dysfunction = Yes) ' ;
 end attributes
 ;
 argument :: excluding,(Pregnancy = Yes) attributes
 argument_name :: ' (Pregnancy = Yes) ' ;
 end attributes
 ;
 argument :: excluding,(Current_Rx = K_sparing_diuretic) attributes

```

```

 argument_name :: ' (Current_Rx = K_sparing_diuretic) ' ;
 end attributes
;
 argument :: excluding,(Current_Rx = ACEI) attributes
 argument_name :: ' (Current_Rx = ACEI) ' ;
 end attributes
;
 recommendation :: Netsupport(Additional_drug_choice, ACEI) >= 1 ;
candidate :: 'Loop_diuretic' ;
 argument :: confirming,(Heart_failure = Yes) attributes
 argument_name :: ' (Heart_failure = Yes) ' ;
 end attributes
;
 argument :: confirming,(Isolated_Sys_hypertension = Yes and Age_group = Older) attributes
 argument_name :: ' (Isolated_Sys_hypertension = Yes and Age_group = Older) ' ;
 end attributes
;
 argument :: excluding,(Current_Rx = Thiazide_diuretic or Current_Rx = K_sparing_diuretic or Current_Rx =
Loop_diuretic) attributes
 argument_name :: ' (Current_Rx = Thiazide_diuretic or Current_Rx = K_sparing_diuretic or Current_Rx =
Loop_diuretic) ' ;
 end attributes
;
 recommendation :: Netsupport(Additional_drug_choice, Loop_diuretic) >= 1 ;
candidate :: 'Ca_channel_dihydro_long' ;
 argument :: for,(Type_1_Diabetes = Yes and Proteinuria != None) attributes
 argument_name :: ' (Type_1_Diabetes = Yes and Proteinuria <> None) ' ;
 end attributes
;
 argument :: excluding,(Current_Rx = Ca_blocker_non_DHP) attributes
 argument_name :: ' (Current_Rx = Ca_blocker_non_DHP) ' ;
 end attributes
;
 argument :: for,(Current_Rx = Beta_blocker_ISA or Current_Rx = Beta_blocker_non_ISA) attributes
 argument_name :: ' (Current_Rx = Beta_blocker_ISA or Current_Rx = Beta_blocker_non_ISA) ' ;
 end attributes
;
 argument :: confirming,(Isolated_Sys_hypertension = Yes and Age_group = Older) attributes
 argument_name :: ' (Isolated_Sys_hypertension = Yes and Age_group = Older) ' ;
 end attributes
;
 argument :: excluding,(Current_Rx = Ca_blocker_DHP_long or Current_Rx = Ca_blocker_DHP_short or
Current_Rx = Ca_blocker_non_DHP) attributes
 argument_name :: ' (Current_Rx = Ca_blocker_DHP_long or Current_Rx = Ca_blocker_DHP_short or
Current_Rx = Ca_blocker_non_DHP) ' ;
 end attributes
;
 recommendation :: Netsupport(Additional_drug_choice, Ca_channel_dihydro_long) >= 1 ;
candidate :: 'Ca_channel_dihydro_short' ;
 argument :: for,(Type_1_Diabetes = Yes and Proteinuria != None) attributes
 argument_name :: ' (Type_1_Diabetes = Yes and Proteinuria <> None) ' ;
 end attributes
;
 argument :: excluding,(Current_Rx = Ca_blocker_non_DHP) attributes
 argument_name :: ' (Current_Rx = Ca_blocker_non_DHP) ' ;
 end attributes
;
 argument :: for,(Current_Rx = Beta_blocker_ISA or Current_Rx = Beta_blocker_non_ISA) attributes
 argument_name :: ' (Current_Rx = Beta_blocker_ISA or Current_Rx = Beta_blocker_non_ISA) ' ;
 end attributes
;
 argument :: confirming,(Isolated_Sys_hypertension = Yes and Age_group = Older) attributes
 argument_name :: ' (Isolated_Sys_hypertension = Yes and Age_group = Older) ' ;
 end attributes
;
 argument :: excluding,(Current_Rx = Ca_blocker_DHP_long or Current_Rx = Ca_blocker_DHP_short or
Current_Rx = Ca_blocker_non_DHP) attributes
 argument_name :: ' (Current_Rx = Ca_blocker_DHP_long or Current_Rx = Ca_blocker_DHP_short or
Current_Rx = Ca_blocker_non_DHP) ' ;
 end attributes
;
 recommendation :: Netsupport(Additional_drug_choice, Ca_channel_dihydro_short) >= 1 ;

```

```

candidate :: 'Non_ISA_beta_blocker' ;
 argument :: excluding,(STD_Heart_block = Yes) attributes
 argument_name :: ' (STD_Heart_block = Yes) ' ;
 end attributes
;
argument :: against,(Type_1_Diabetes = Yes and Proteinuria != None) attributes
 argument_name :: ' (Type_1_Diabetes = Yes and Proteinuria <> None) ' ;
end attributes
;
argument :: confirming,(MI = Yes) attributes
 argument_name :: ' (MI = Yes) ' ;
end attributes
;
argument :: excluding,(Current_Rx = Ca_blocker_non_DHP) attributes
 argument_name :: ' (Current_Rx = Ca_blocker_non_DHP) ' ;
end attributes
;
argument :: for,(Current_Rx = Ca_blocker_DHP_long or Current_Rx = Ca_blocker_DHP_short) attributes
 argument_name :: ' (Current_Rx = Ca_blocker_DHP_long or Current_Rx = Ca_blocker_DHP_short) ' ;
end attributes
;
argument :: excluding,(Current_Rx = Beta_blocker_non_ISA or Current_Rx = Beta_blocker_ISA) attributes
 argument_name :: ' (Current_Rx = Beta_blocker_non_ISA or Current_Rx = Beta_blocker_ISA) ' ;
end attributes
;
argument :: excluding,(Asthma = Yes or COPD = Yes) attributes
 argument_name :: ' (Asthma = Yes or COPD = Yes) ' ;
end attributes
;
argument :: for,(Current_Rx = Thiazide_diuretic) attributes
 argument_name :: ' (Current_Rx = Thiazide_diuretic) ' ;
end attributes
;
recommendation :: Netsupport(Additional_drug_choice, Non_ISA_beta_blocker) >= 1 ;
candidate :: 'K_sparing_diuretic' ;
 argument :: confirming,(Isolated_Sys_hypertension = Yes and Age_group = Older) attributes
 argument_name :: ' (Isolated_Sys_hypertension = Yes and Age_group = Older) ' ;
 end attributes
;
 argument :: excluding,(Current_Rx = Thiazide_diuretic or Current_Rx = K_sparing_diuretic or Current_Rx =
Loop_diuretic) attributes
 argument_name :: ' (Current_Rx = Thiazide_diuretic or Current_Rx = K_sparing_diuretic or Current_Rx =
Loop_diuretic) ' ;
 end attributes
;
 argument :: excluding,(Current_Rx = ACEI) attributes
 argument_name :: ' (Current_Rx = ACEI) ' ;
 end attributes
;
recommendation :: Netsupport(Additional_drug_choice, K_sparing_diuretic) >= 1 ;
candidate :: 'ISA_beta_blocker' ;
 argument :: excluding,(STD_Heart_block = Yes) attributes
 argument_name :: ' (STD_Heart_block = Yes) ' ;
 end attributes
;
 argument :: against,(Type_1_Diabetes = Yes and Proteinuria != None) attributes
 argument_name :: ' (Type_1_Diabetes = Yes and Proteinuria <> None) ' ;
 end attributes
;
 argument :: excluding,(Current_Rx = Ca_blocker_non_DHP) attributes
 argument_name :: ' (Current_Rx = Ca_blocker_non_DHP) ' ;
 end attributes
;
 argument :: excluding,(Current_Rx = Beta_blocker_ISA or Current_Rx = Beta_blocker_non_ISA) attributes
 argument_name :: ' (Current_Rx = Beta_blocker_ISA or Current_Rx = Beta_blocker_non_ISA) ' ;
 end attributes
;
 argument :: for,(Current_Rx = Ca_blocker_DHP_long or Current_Rx = Ca_blocker_DHP_short) attributes
 argument_name :: ' (Current_Rx = Ca_blocker_DHP_long or Current_Rx = Ca_blocker_DHP_short) ' ;
 end attributes
;
 argument :: excluding,(Asthma = Yes or COPD = Yes) attributes

```

```

 argument_name :: ' (Asthma = Yes or COPD = Yes) ' ;
 end attributes
 ;
 argument :: for,(Current_Rx = Thiazide_diuretic) attributes
 argument_name :: ' (Current_Rx = Thiazide_diuretic) ' ;
 end attributes
 ;
 recommendation :: Netsupport(Additional_drug_choice, ISA_beta_blocker) >= 1 ;
 candidate :: 'Thiazide_diuretic' ;
 argument :: for,(Current_Rx = Beta_blocker_ISA or Current_Rx = Beta_blocker_non_ISA) attributes
 argument_name :: ' (Current_Rx = Beta_blocker_ISA or Current_Rx = Beta_blocker_non_ISA) ' ;
 end attributes
 ;
 argument :: confirming,(Isolated_Sys_hypertension = Yes and Age_group = Older) attributes
 argument_name :: ' (Isolated_Sys_hypertension = Yes and Age_group = Older) ' ;
 end attributes
 ;
 argument :: excluding,(Current_Rx = Thiazide_diuretic or Current_Rx = K_sparing_diuretic or Current_Rx =
Loop_diuretic) attributes
 argument_name :: ' (Current_Rx = Thiazide_diuretic or Current_Rx = K_sparing_diuretic or Current_Rx =
Loop_diuretic) ' ;
 end attributes
 ;
 recommendation :: Netsupport(Additional_drug_choice, Thiazide_diuretic) >= 1 ;
 candidate :: 'Ca_channel_non_dihydro' ;
 argument :: excluding,(STD_Heart_block = Yes) attributes
 argument_name :: ' (STD_Heart_block = Yes) ' ;
 end attributes
 ;
 argument :: for,(Type_1_Diabetes = Yes and Proteinuria != None) attributes
 argument_name :: ' (Type_1_Diabetes = Yes and Proteinuria <> None) ' ;
 end attributes
 ;
 argument :: excluding,(Current_Rx = Beta_blocker_ISA or Current_Rx = Beta_blocker_non_ISA) attributes
 argument_name :: ' (Current_Rx = Beta_blocker_ISA or Current_Rx = Beta_blocker_non_ISA) ' ;
 end attributes
 ;
 argument :: excluding,(Current_Rx = Ca_blocker_DHP_long or Current_Rx = Ca_blocker_DHP_short) attributes
 argument_name :: ' (Current_Rx = Ca_blocker_DHP_long or Current_Rx = Ca_blocker_DHP_short) ' ;
 end attributes
 ;
 argument :: excluding,(Current_Rx = Ca_blocker_DHP_long or Current_Rx = Ca_blocker_DHP_short or
Current_Rx = Ca_blocker_non_DHP) attributes
 argument_name :: ' (Current_Rx = Ca_blocker_DHP_long or Current_Rx = Ca_blocker_DHP_short or
Current_Rx = Ca_blocker_non_DHP) ' ;
 end attributes
 ;
 recommendation :: Netsupport(Additional_drug_choice, Ca_channel_non_dihydro) >= 1 ;
end decision.

action :: 'Prescribe_additional_drug' ;
 caption :: ""Prescribe additional drug"";
 procedure :: 'Prescribe additional drug as chosen.';
end action.

action :: 'Await_adequate_response' ;
 caption :: ""Await adequate response"";
 procedure :: 'Continue therapy until target BPs reached';
end action.

enquiry :: 'Current_therapy' ;
 caption :: ""Current therapy"";
 source :: 'Current_Rx' ;
 data_definition :: 'Current_Rx' ;
end enquiry.

decision :: 'Target_BP_decision' ;
 caption :: ""Target BP decision"";
 postcondition :: TargetSBP = if(result_of(Target_BP_decision) = Standard, 140, if(result_of(Target_BP_decision) =
Medium, 130, 125)) and TargetDBP = if(result_of(Target_BP_decision) = Standard, 90, if(result_of(Target_BP_decision)
= Medium, 85, 75));
 candidate :: 'Medium' ;

```

```

 argument :: for,(Type_1_Diabetes = Yes or Proteinuria = '<1 g / 24 hrs') attributes
 argument_name :: ' (Type_1_Diabetes = Yes or Proteinuria = '<1 g / 24 hrs') ' ;
 end attributes
 ;
 recommendation :: Netsupport(Target_BP_decision, Medium) >= 1 ;
candidate :: 'Low' ;
 argument :: confirming,(Proteinuria = '>1 g / 24 hrs') attributes
 argument_name :: ' (Proteinuria = '>1 g / 24 hrs') ' ;
 end attributes
 ;
 recommendation :: Netsupport(Target_BP_decision, Low) >= 1 ;
candidate :: 'Standard' ;
 argument :: for,(Proteinuria = None and Type_1_Diabetes = No) attributes
 argument_name :: ' (Proteinuria = None and Type_1_Diabetes = No) ' ;
 end attributes
 ;
 recommendation :: Netsupport(Target_BP_decision, Standard) >= 1 ;
end decision.

enquiry :: 'Monitor_BP' ;
caption :: "Monitor BP";
source :: 'DBP' ;
 data_definition :: 'DBP' ;
source :: 'SBP' ;
 data_definition :: 'SBP' ;
end enquiry.

plan :: 'Assess_response' ;
caption :: "Assess response";
goal :: "DBP < targetDBP and SBP < targetSBP";
component :: 'Monitor_BP' ;
 caption :: "Monitor_BP";
 task_definition :: 'Monitor_BP' ;
 number_of_cycles :: 1;
 ltwh :: 33,100,42,36;
component :: 'Await_adequate_response' ;
 caption :: "Await_adequate_response";
 task_definition :: 'Await_adequate_response' ;
 number_of_cycles :: 1;
 ltwh :: 88,100,36,36;
end plan.

enquiry :: 'Comorbidities' ;
caption :: "Comorbidities";
source :: 'Age_group' ;
 data_definition :: 'Age_group' ;
source :: 'Asthma' ;
 data_definition :: 'Asthma' ;
source :: 'COPD' ;
 data_definition :: 'COPD' ;
source :: 'Heart_failure' ;
 data_definition :: 'Heart_failure' ;
source :: 'Isolated_Sys_hypertension' ;
 data_definition :: 'Isolated_Sys_hypertension' ;
source :: 'MI' ;
 data_definition :: 'MI' ;
source :: 'Pregnancy' ;
 data_definition :: 'Pregnancy' ;
source :: 'Proteinuria' ;
 data_definition :: 'Proteinuria' ;
source :: 'STD_Heart_block' ;
 data_definition :: 'STD_Heart_block' ;
 mandatory :: no ;
source :: 'Systolic_dysfunction' ;
 data_definition :: 'Systolic_dysfunction' ;
source :: 'Type_1_Diabetes' ;
 data_definition :: 'Type_1_Diabetes' ;
 mandatory :: no ;
end enquiry.

data :: 'Age_group' ;
type :: text ;

```

```
caption :: "Patients age group";
range :: "Younger", "Older";
end data.

data :: 'Asthma' ;
type :: boolean ;
caption :: "Does the patient have asthma?";
true_value :: 'Yes' ;
false_value :: 'No' ;
end data.

data :: 'COPD' ;
type :: boolean ;
caption :: "Does that patient have COPD?";
true_value :: 'Yes' ;
false_value :: 'No' ;
end data.

data :: 'Current_Rx' ;
type :: text ;
caption :: "Patients current therapy";
range
:: "ACEI", "Beta_blocker_ISA", "Beta_blocker_non_ISA", "Ca_blocker_DHP_long", "Ca_blocker_DHP_short", "Ca_blocker_no
n_DHP", "Thiazide_diuretic", "Loop_diuretic", "K_sparing_diuretic";
end data.

data :: 'DBP' ;
type :: integer ;
caption :: "Diastolic BP";
end data.

data :: 'Heart_failure' ;
type :: boolean ;
caption :: "Does the patient have heart failure?";
true_value :: 'Yes' ;
false_value :: 'No' ;
end data.

data :: 'Isolated_Sys_hypertension' ;
type :: boolean ;
caption :: "Isolated systolic hypertension?";
true_value :: 'Yes' ;
false_value :: 'No' ;
end data.

data :: 'MI' ;
type :: boolean ;
caption :: "Previous MI?";
true_value :: 'Yes' ;
false_value :: 'No' ;
end data.

data :: 'Pregnancy' ;
type :: boolean ;
caption :: "Is the patient pregnant?";
true_value :: 'Yes' ;
false_value :: 'No' ;
end data.

data :: 'Proteinuria' ;
type :: text ;
caption :: "Proteinuria?";
range :: "None", "<1 g / 24 hrs", ">1 g / 24 hrs";
end data.

data :: 'SBP' ;
type :: integer ;
caption :: "Systolic BP";
end data.

data :: 'STD_Heart_block' ;
type :: boolean ;
```

```
caption :: "Second or third degree heart block";
true_value :: 'Yes' ;
false_value :: 'No' ;
end data.
```

```
data :: 'Systolic_dysfunction' ;
type :: boolean ;
caption :: "Systolic dysfunction";
true_value :: 'Yes' ;
false_value :: 'No' ;
end data.
```

```
data :: 'Type_1_Diabetes' ;
type :: boolean ;
caption :: "Type 1 diabetes?";
true_value :: 'Yes' ;
false_value :: 'No' ;
end data.
```

## Manual d'usuari

A continuació es detalla gràficament el funcionament de l'aplicació.

- L'usuari, en el seu navegador Web, escriurà la següent URL: [http://localhost:8080/sample/ini\\_service.jsp](http://localhost:8080/sample/ini_service.jsp). La pantalla que li apareixerà serà la de validació de l'usuari.

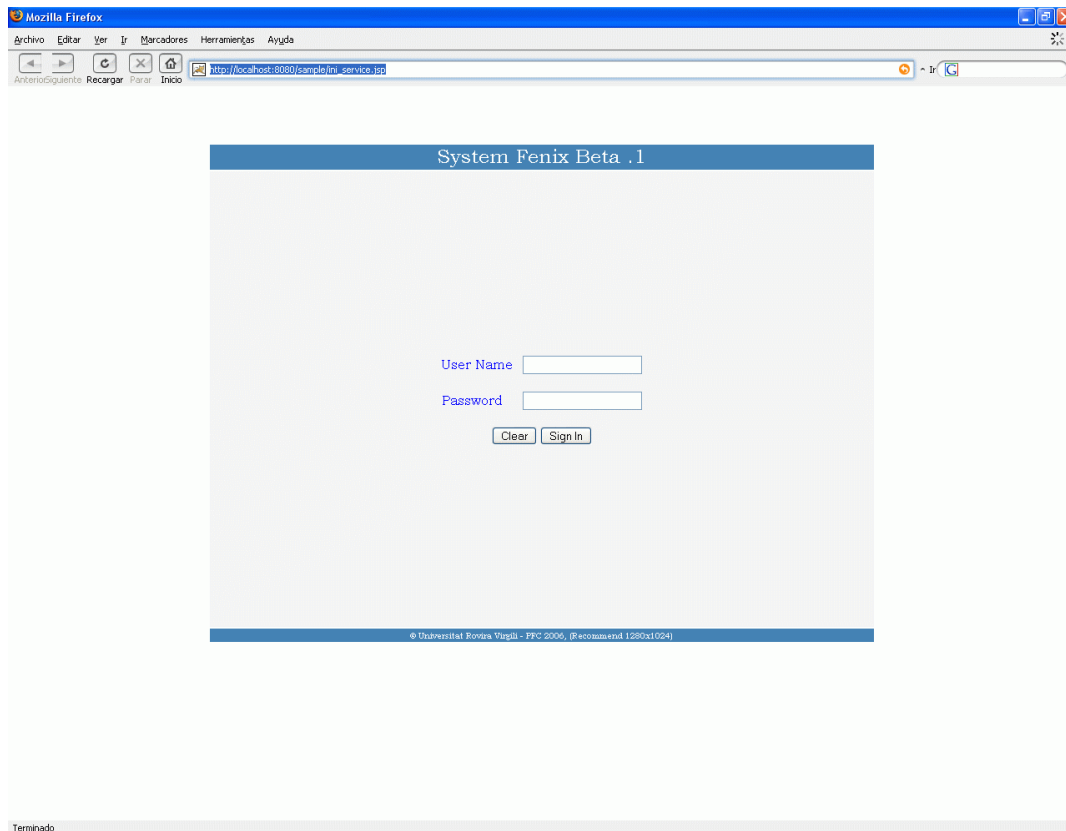


**Figura 36:** Pantalla Inicial: L'usuari s'ha d'identificar

- En el cas que l'usuari s'identifiqui incorrectament.

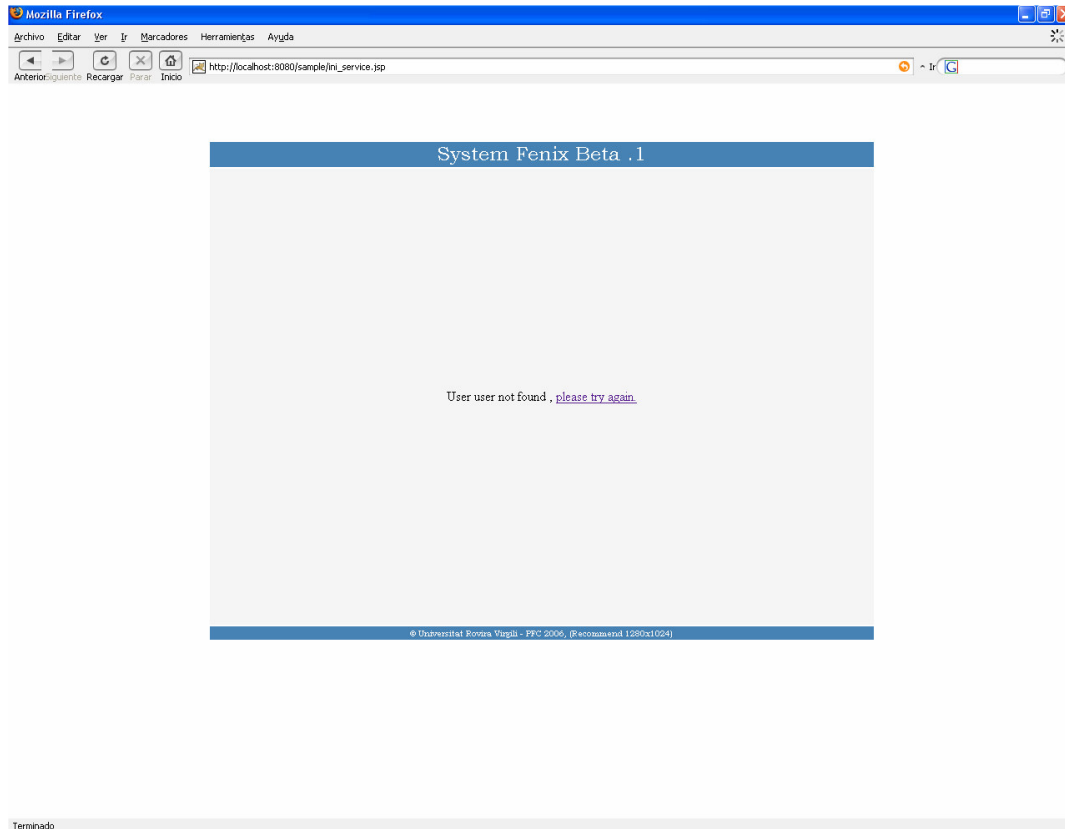


**Figura 37:** Pantalla Validació Incorrecta: L'usuari no s'ha identificat

- Una vegada l'usuari ha introduït el nom d'usuari (login) i contrasenya correctament, se li presenta la pantalla principal per a visualitzar GPC's.

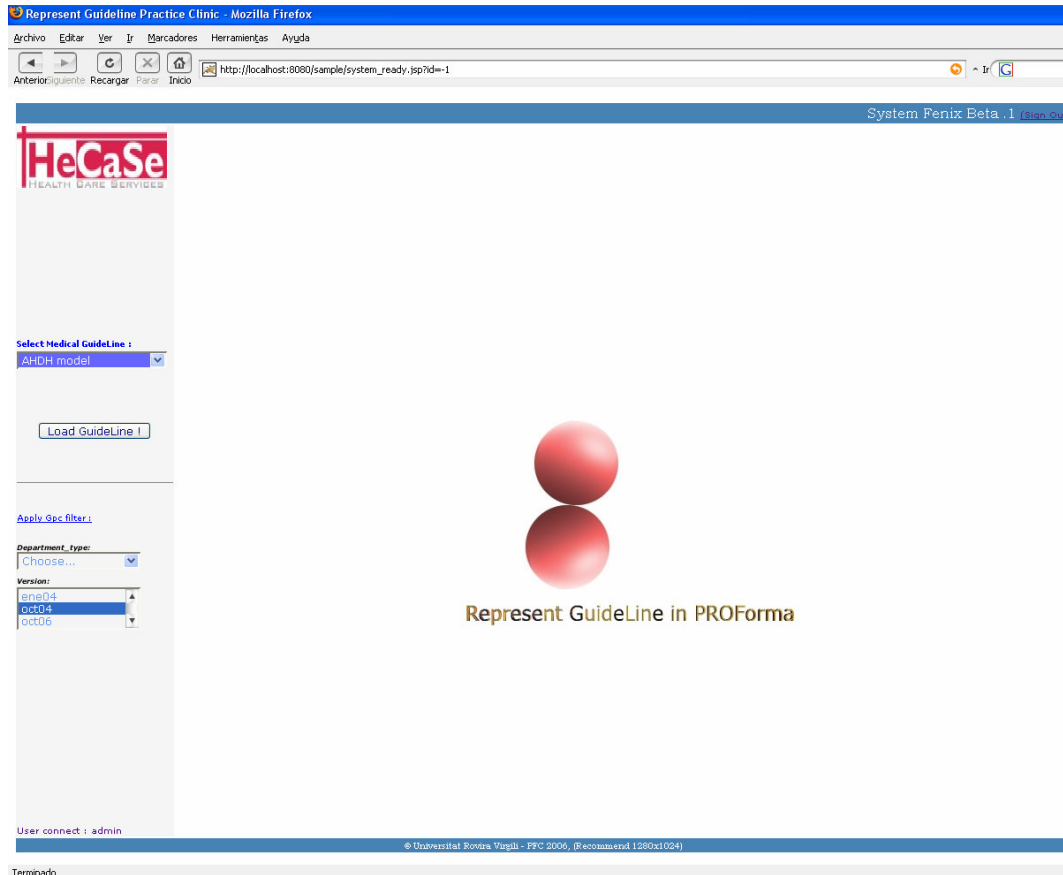


**Figura 38:** Pantalla per visualitzar les GPC's

En aquesta pantalla se li presenta la informació següent:

- Llistat de les guies de pràctica clínica disponibles en la base de dades.
- Nom de l'usuari que ha iniciat la sessió. (en el cas de l'exemple **Admin**)
- Etiqueta “Sign Out” per abandonar l'aplicació.
- Camps per a filtrar el llistat de les guies de pràctica clínica disponibles.

- L'usuari decideix carregar la GPC "càncer diagnosis model". En aquest moment se li presenta el primer nivell de la guia.

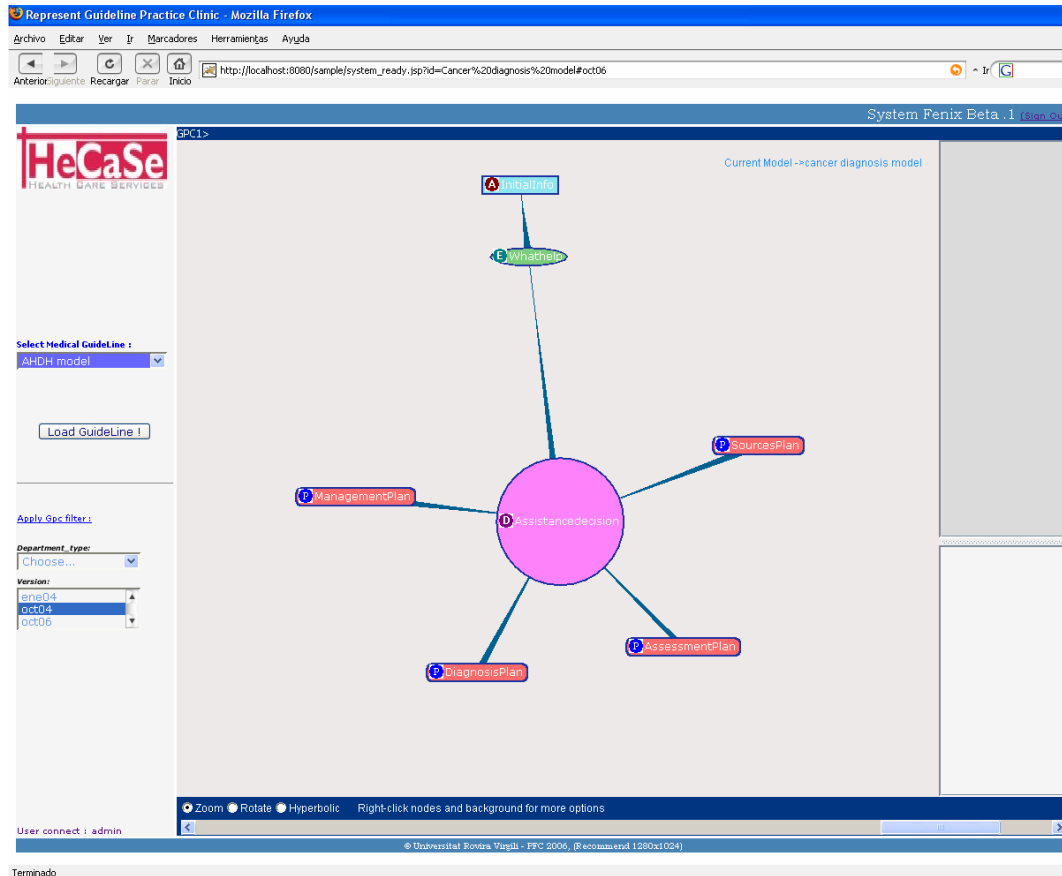


**Figura 39:** Gpc "càncer diagnosis model" nivell inicial

Cal notar en la interfície:

- La GPC està formada per 4 plans, una decisió, una enquiry (o pregunta) i una acció.
- A la part superior apareix un sistema de navegació pels diferents nivells de la GPC.
- A la part inferior se li apareixen les opcions de rotar, zoom sobre la GPC.
- A la part dreta superior, s'hi troba el quadre que visualitzarà la informació referent a un node.
- A la part dreta inferior, s'hi troba el quadre que visualitzarà la informació referent a les relacions entre nodes.

- A partir d'aquí l'usuari pot 'navegar' pels nivells de la GPC. Les opcions que es poden fer sobre un node són les següents (**botó dret sobre el node**):
  - **Center Node**: centra el graf amb el node seleccionat al mig.
  - **View All**: es genera una vista amb els subnodes continguts en el node seleccionat; per tant podem arribar a veure tota la guia en un únic nivell.
  - **View Alone**: es genera una vista únicament amb els subnodes que es troben dins el node seleccionat, d'aquesta manera només ens centrem amb una part de la GPC (nivell alternatiu).
  - **View Info**: presenta la informació més rellevant del node, per exemple, el nom, els antecedents, els objectius, ...

Imaginem que realitzem un "View Alone" sobre el pla "ManagementPla", de la següent forma:

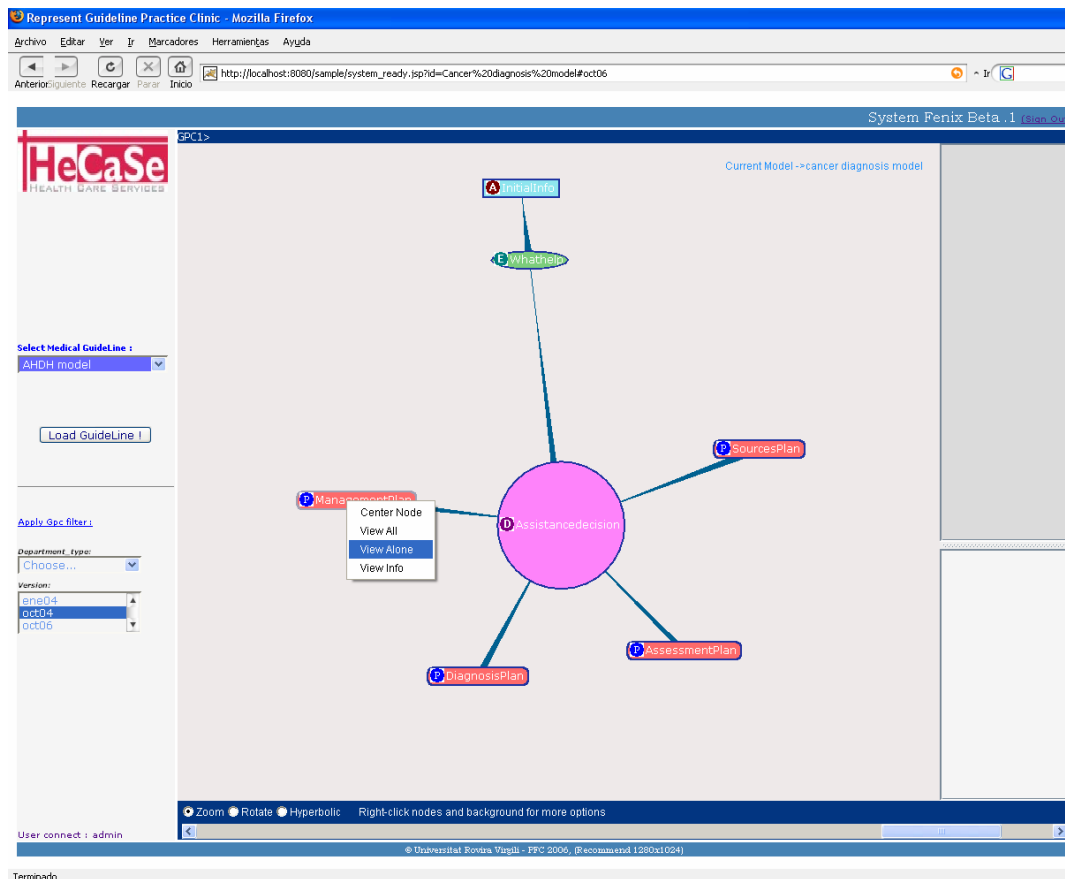


**Figura 40:** "View Alone" sobre el node "ManagementPla"

➤ El resultat és el següent :

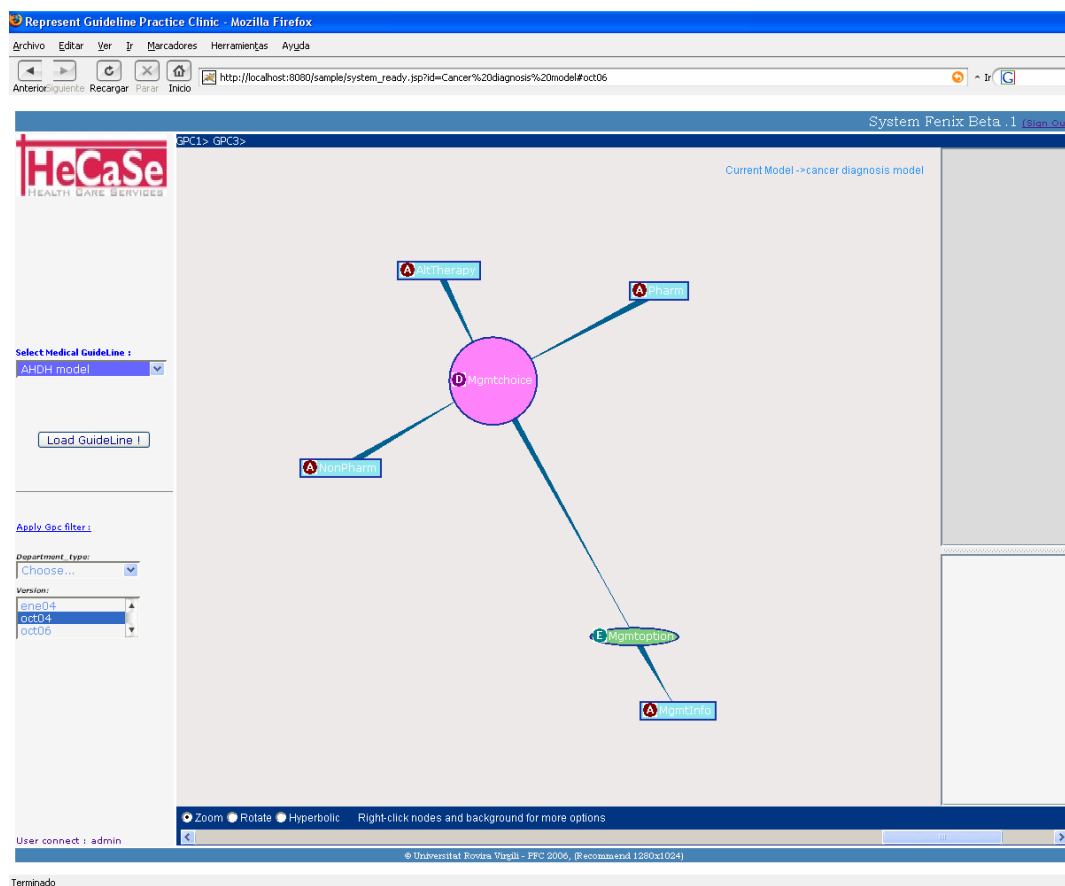


**Figura 41:** “Sub-pla” del node “ManagementPla”

- A continuació realitzem un “View Info” sobre el node “mgmtchoice” que és un node decisió. El resultat és el següent:

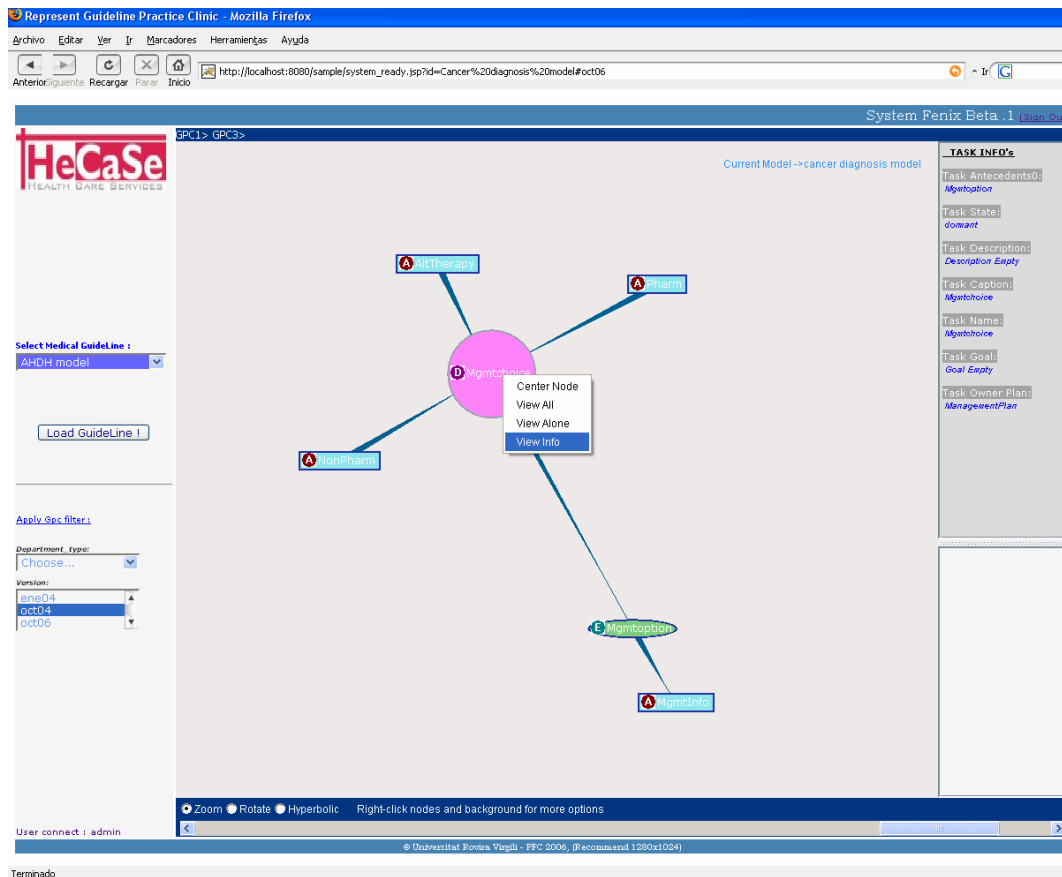


**Figura 42:** Visualització de la informació del node decisió “mgmtchoice”

- A continuació realitzem un “View Info Edge” sobre la relació entre el node “mgmtchoice” i el node “mgmtoption”. El resultat és el següent:

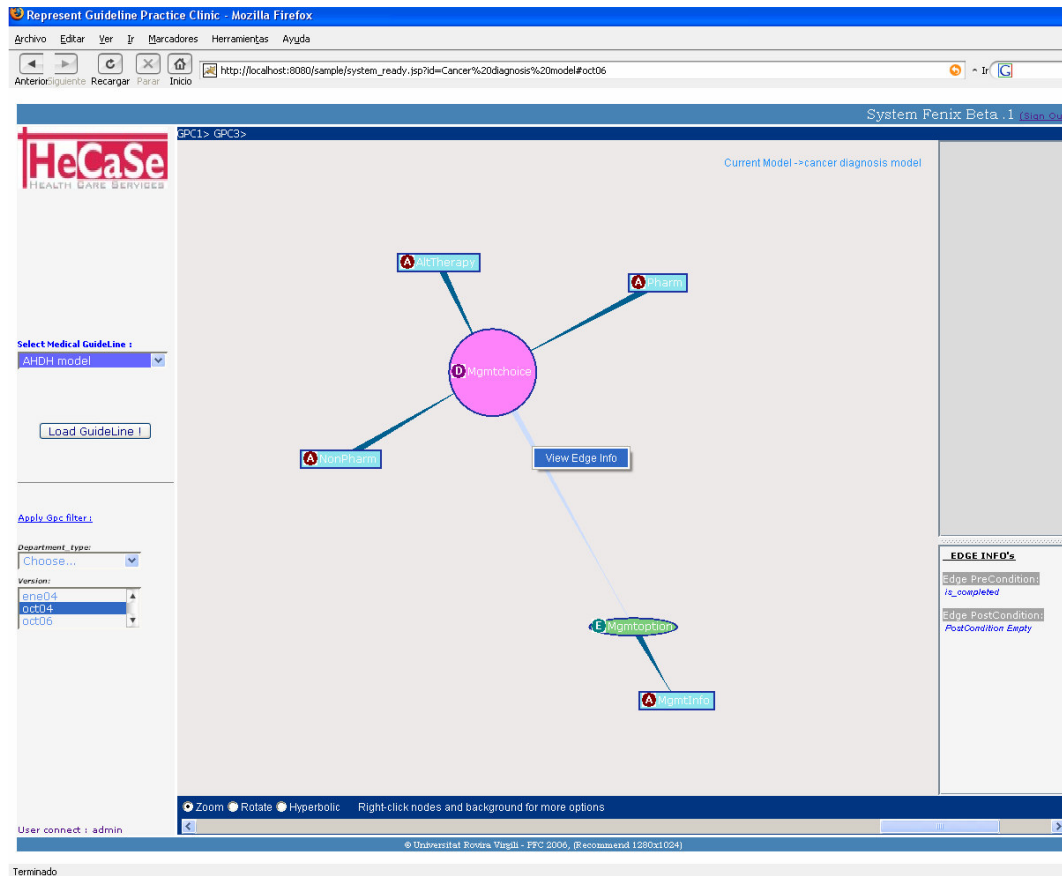


**Figura 43:** Visualització de la informació de la relació “mgmtchoice” i “mgmtoption”

- A continuació, amb la barra de navegació, tornem a l'estat anterior. El resultat és el següent:

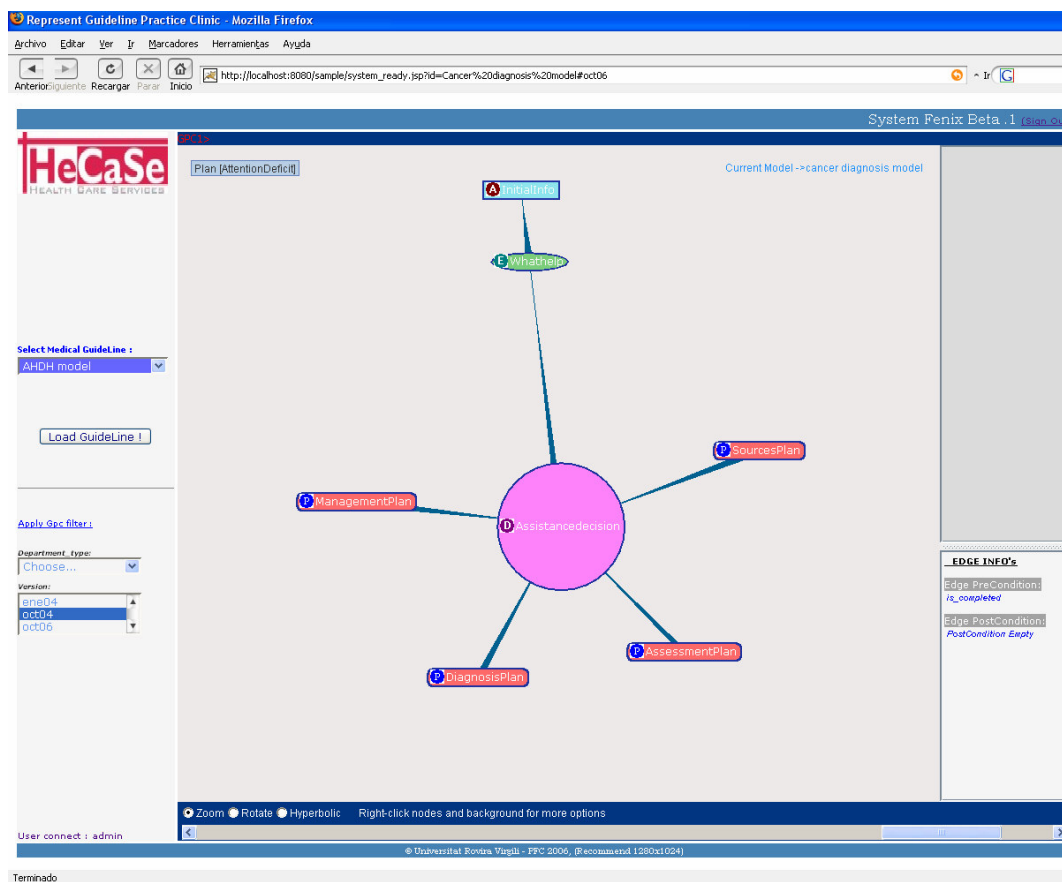


**Figura 44:** *Nivell anterior*

- Finalment utilitzarem la funció de “View All” en cadascun dels nodes plans, d’aquesta manera veurem la GPC “*cancer diagnosis model*” en tota la seva extensió. El resultat és el següent:

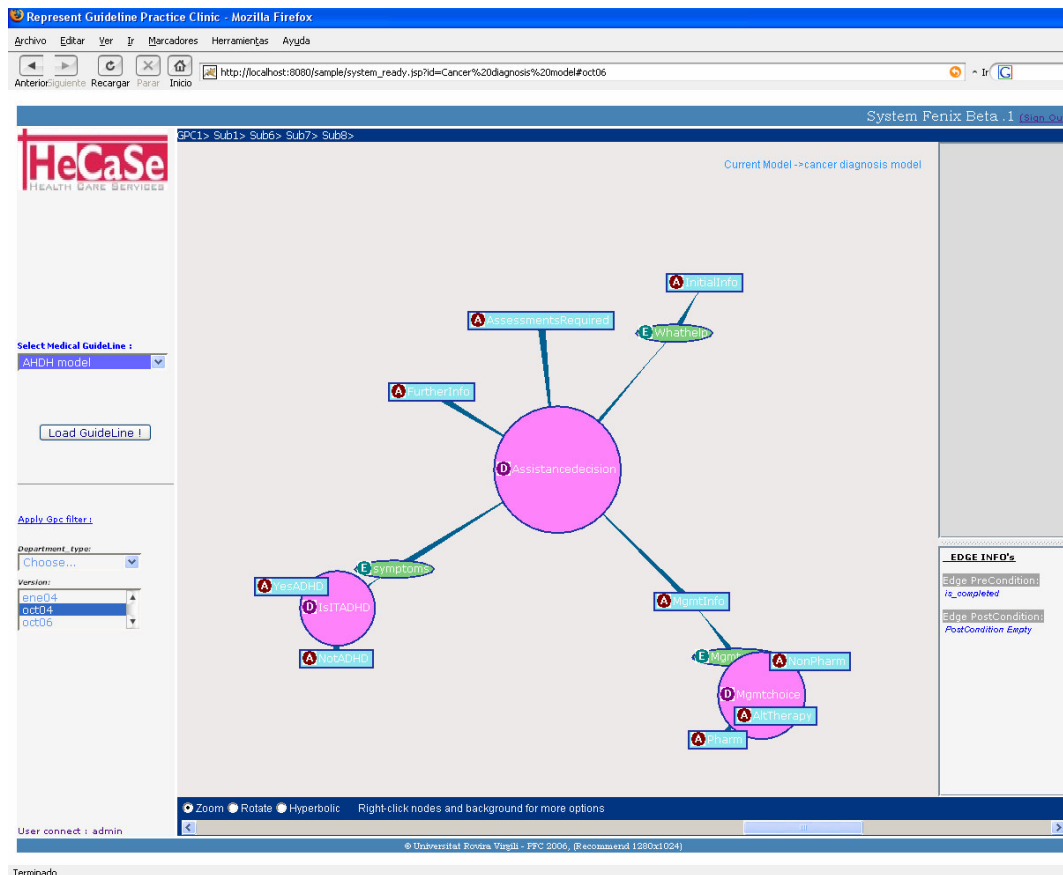


**Figura 45:** Representació total de la GPC “*cancer diagnosis model*”