Índex

1. INTRODUCCIÓ

En qualsevol projecte complexe en el que es veuen implicades diverses persones treballant conjuntament s'intenta escollir sempre el millor equip possible per a la seva realització. Aquesta feina no és precisament fàcil de realitzar donat l'enorme quantitat de factors que s'hi veuen involucrats i que hi intervenen. Entre d'ells la experiència de cada persona en el món laboral i el seu grau de sociabilitat són factors bastant importants.

1.1 Plantejament del Problema

Per tant, l'encarregat d'escollir quin serà el grup per a realitzar el projecte, ha de perdre molt de temps abans de decidir quin seria el grup de persones que millor funcionarien entre sí. Ja que, a banda dels factors abans esmentats, també n'ha de tenir en compte molts d'altres. O sigui, ha de pendre les decisions basant-se en el següent:

- ***Les característiques de cada persona individual:*** S'ha de tenir en compte les característiques de cada individu ja que aquestes poden influir decisivament en el resultat final:

    1. *Nivell Professional:* Es tracta del factor a tenir en compte que més relevància pren a l'hora d'assignar les tasques als treballadors. El nivell Professional d'un individu serà el que delimitarà quines tasques pot realitzar i quines no, i en les que pot realitzar quanta estona trigarà. Per exemple, un Enginyer Tècnic no pot realitzar segons quines tasques destinades a un Enginyer Superior. En canvi, un Enginyer Superior sí que pot realitzar la feina destinada a un Tècnic i, segurament, molt més ràpidament que no ho faria aquest.

    2. *Nivell d'Experiència:* És el segon factor més important a tenir en compte. Ja que, per molt que algú tingui un nivell professional elevat, el fet de no posseir experiència delimita la seva capacitat ja que, molt segurament,

tindrà més possibilitats de retrassar-se o d'errar-la mentres realitza una tasca completament nova per a ell. O sigui, si a un treballador sense experiència li diuen que arregli una màquina, és molt possible que trigui molt més que un altre que ho porta fent fa anys i s'ha trobat en molts casos en situacions semblants a aquella.

3. *Grau de Sociabilitat:* Potser es tracta d'un factor de menor importància que els anteriorment esmentats. Però no per això deixa de tenir-ne. El grau de sociabilitat pot influir de forma positiva si una persona és oberta. Això és degut a que si es troba davant d'una situació complexe és més probable que demani ajuda a un tercer, alleugerint així el procés. En canvi, una persona de caràcter tancat preferirà abans fer-ho tot sol que demanar ajuda, tardant així més que una d'oberta.

- **Altres aspectes generals a tenir en compte:** No ens podem contentar en escollir al grup de gent per al projecte basant-nos només en les característiques personals de cadascú. Hi ha d'altres factors igualment importants que poden influir de forma directa fent-nos descartar opcions que en un principi considerariem encertades. Les més importants són dues que, curiosament, estan lligades de forma inversa. O sigui, quan una "decreix" l'altra "augmenta". Entenent com decréixer i augmentar el fet de beneficiar i perjudicar respectivament els nostres propòsits. Aquestes són el cost i el temps.

  1. *El cost:* En un principi podria resultar interessant la elecció d'un grup de persones amb un alt nivell professional que acabessin ràpida i eficientment les feines que els hi encomanessim, tanmateix això presenta un important inconvenient: el cost. No podem pretendre sobrepassar-nos de certs límits únicament perquè així acabarem abans. A vegades es disposa d'un pressupost limitat i, conseqüentment, d'un marge de beneficis limitat. I donat que són aquests últims els que interessen, no ens podem sobrepassar amb el gast monetari.

  2. *El temps*: Es troba en contraposició directa amb el cost. Ja que, normalment, a menor cost, més llarg es torna el període d'execució del projecte. I, igualment que no ens podiem sobrepassar amb el cost, tampoc podem fer-ho amb el temps. En tots els projectes existeix un límit de temps (deadline) que no es pot rebassar si es volen obtenir beneficis.

Tal i com s'ha vist, la feina del que ha de decidir és realment complexe. Per un grup de persones reduit es pot portar a terme amb una certa facilitat i eficiència. Sobretot si han treballat conjuntament algun cop. Però a mida que augmenta el número de persones a col·laborar i el possible comportament entre aquestes és desconegut per l'encarregat d'assignar les tasques, tot es complica enormement. És per això que a vegades podrien ser necessàries eines que ajudessin a pendre aquesta decisió.
Aquest projecte pretén ser una eina per a ajudar a aquesta tasca i reduir així la seva complexitat.

1.2 Un Simulador de Grups de Treball:

Hem pogut veure que no és fàcil escollir un grup de treball. Tot seria molt més fàcil si poguessim saber d'avantmà com actuaran les diferents possibles combinacions de persones. D'aquesta manera ens evitariem sorpreses desagradables i no hauriem de pensar gaire per a escollir el grup.

Aquesta idea de saber com actuaran un grup de persones és una mica agosarada, més que agosarada impracticable. No es pot saber fins que no es fa. Però tanmateix, sí que podem intentar preveure aquest comportament. I aquesta és la funció del projecte que ens ocupa, l'intentar preveure els possibles comportaments d'un grup de persones per a unes determinades tasques i intentar esbrinar quines són les combinacions amb més probabilitats d'èxit i quines és millor no provar.

Aquest simulador de grups de treball intenta, mitjançant l'ús de tècniques d'Intel·ligència Artificial, més concretament emprant un Sistema Multiagent (SMA), modelar el comportament d'uns quants grups de treball i donar una idea de com és possible que funcionin les coses amb les diferentes combinacions. El nostre propòsit és el de modelar els treballadors, és per això que poden ser útils els Agents, aquests són com programes independents amb una certa autonomia capaços de comunicar-se amb d'altres agents per a poder portar a terme un objectiu comú. L'autonomia que poseeixen els agents els permet tenir un coneixement propi on podrem guardar les característiques de cada treballador.

La idea és la de tenir diversos agents que actuin com si fóssin els diferents possibles membres que tenim a escollir. Cada un d'ells tindrà les seves característiques personals:

- **El nom:** Servirà per a identificar-los, a més de fer més amable la interacció amb el simulador.

- **El nivell professional:** Cada treballador tindrà un nivell professional que delimitarà el tipus de tasques que podrà realitzar. Aquest podrà pendre tres possibles valors, de més a menys grau aquests són: ADP* (Agent Director de Projecte), AEQ (Agent Enginyer Químic) i AT (Agent Tècnic). Aquesta característica també ens servirà a l'hora de mesurar el temps que trigarà un agent (treballador) a realitzar una tasca.

- **El grau d'experiència:** Aquest ens servirà per a calcular la possibilitat de fallida en la realització del treball encomanat. Aquesta podrà pendre tres valors: Alt, Mitja i Baix. Cadascun d'ells pot variar el grau en el que afecti a un agent, això dependrà del seu grau de professionalitat.

- **Caràcter:** Aquesta és la quarta i última característica que té un treballador. És amb aquesta amb la que es calcula el fet de que busqui o no un ajudant per a realitzar la tasca encomanada.. Un treballador pot ser obert o tancat.

* Nota: Aquest simulador està basat en una investigació per a l'utilització d'un sistema multiagent per a ajudar en el disseny de processos químics. Per això el tipus d'especialitat escollida

Un cop hem vist com són els treballadors, ens toca veure com són els possibles ajudants: els Obrers. Aquests veuen reduides el seu nombre de característiques a dos donat que la seva funció no és la mateixa que la dels seus superiors (els treballadors). Un obrer només té el nom i el grau d'experiència com a característiques. No necessita el nivell professional ni el caràcter ja que, el seu nivell professional sempre serà el mateix, el d'obrer i el caràcter no influirà en les seves decisions, no és ell qui ha de decidir si demana ajuda o no. Està per a que li demanin ajuda.

La feina a realitzar la dividirem en un graf de tasques. Cada tasca podrà ser assignada a un agent. Un graf de tasques podria tenir l'aspecte de la Figura 1.1 En ella hi podem apreciar que poden existir paral·lelismes entre les tasques.

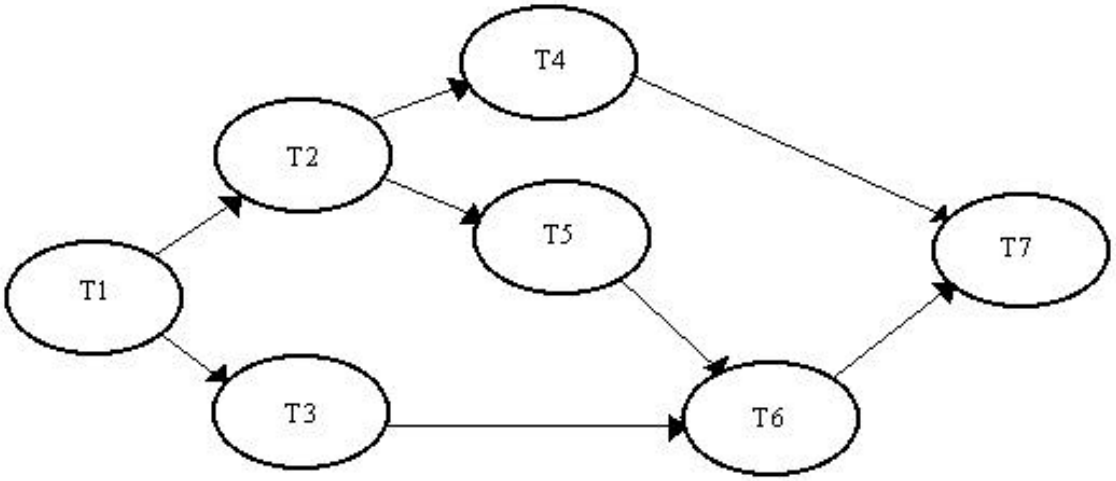

Figura 1.1 Graf de tasques

Per a assignar les tasques corresponents als treballadors usarem l'agent assignador. A cada treballador se li assignarà una tasca depenent del seu grau de professionalitat. O sigui, un treballador per una tasca. Un treballador, però, podrà tenir alhora més d'una tasca assignada, sempre i quant aquestes no es realitzin en paral·lel. Cada treballador podrà optar a un ajudant per a cada tasca diferent que li hagi estat assignada. Depenent del seu caràcter i grau de professionalitat aquest decidirà si demana o no ajuda a un obrer.

En el cas que decideixi demanar ajuda a un obrer, preguntarà a tots els Agents Obrers si estan disponibles i quines condicions temporals li ofereixen. L'Agent Treballador escollirà la millor d'entre totes les possibles ofertes.

Hem dit que serà l'agent assignador qui decidirà quina tasca ha de realitzar cada agent. Aquest, el que en realitat farà, serà realitzar diverses assignacions (tantes com li hagi demanat l'usuari) i per a cada una d'elles li demanarà a l'Agent Simulador que les "executi". Aquest últim serà l'encarregat d'anar informat als respectius Agents Treballadors de les tasques que els hi toca realitzar i anirà recollint els resultats que aquests generin: el temps que han trigat, si han demanat l'ajuda d'un Agent Obrer o bé si han fallat. En aquest últim cas considerarem la simulació com una fallada.

L'agent simulador realitzarà diverses simulacions per a cada una de les assignacions que li passi l'assignació. El número de simulacions a realitzar per a cada assignació també serà un paràmetre que li passarà l'usuari. Per a cada simulació, l'Agent Simulador haurà de tenir en compte l'ordre en el que s'han d'executar les tasques i les dependències entre elles. O sigui, no pot dir-li a un agent que simuli la execució d'una tasca si aquesta necessita d'una altra que encara no hagi acabat.

Un cop realitzada una simulació, l'Agent Simulador passarà els resultats obtinguts a l'Agent Estadístic. Aquest serà l'encarregat de transformar i analitzar les dades per a informar posteriorment a l'usuari de com han anat les coses. Per a tal feina constarà d'una sèrie de consultes tals com l'assignació que en la mitja de temps de les simulacions realitzades hagi obtingut un temps menor. O pel contrari, la que hagi obtingut un temps superior a totes les demés. Per cada consulta podem obtenir com a resultat més d'una assignació.

L'estadístic també disposarà d'un panell en el que podrem veure les gràfiques de la evolució dels costos i temps en el total de la simulació.

En resum, el Simulador de Grups de treball és un Sistema Multiagent en el que hi ha un conjunt de treballadors que interactuen entre ells simulant el comportament d'uns treballadors parells a ells en el món real. A més, consta de tres agents extres, un Agent Assignador, un Agent Simulador i un Agent Estadístic que se n'encarregaran respectivament de fer les assignacions, simular les diverses assignacions interactuant amb els Agent Treballadors i analitzar els resultats.

Un esquema del nostre sistema seria el mostrat en la Figura1.2 En ella podem apreciar l'agent assignador, el simulador, l'estadístic, els agents treballadors (AT) i els obrers (AO).

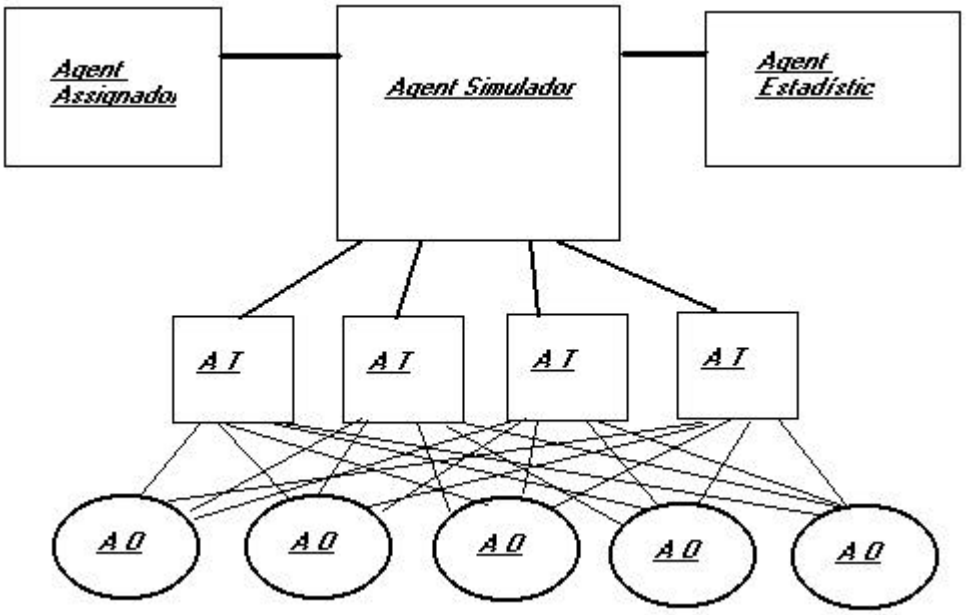

Figura 1.2 Esquema del Simulador de Grups de Treball

A continuació, en el capítol 2 definirem millor els conceptes bàsics d'agent i SMA i detallarem les seves característiques. En el capítol 3 es descriurà l'eina utilitzada per a implementar el Sistema Multi-Agent, el Jade. Finalment, en el capítol 4 descriurem amb detall el nostre Sistema Multi-Agent. Començarem comentant a fons l'arquitectura d'aquest, explicarem el seu funcionament i finalitzarem amb el joc de proves.

2. AGENT I SISTEMES MULTI-AGENT (SMA)

En aquest capítol parlarem sobre els Sistemes Multi-Agent i els Agents que els formen. S'ha de tenir en compte, però, que avui en dia encara no existeixen unes definicions consensuades sobre aquests.

En primer lloc donarem una sèrie de definicions que diferents científics i investigadors han anat donant sobre Agents. Seguidament veurem com es classifiquen i coneixerem més a fons com és i està format un SMA, finalment, tractarem el tema de la comunicació basant-nos en les directives proporcionades per la FIPA(Foundation for Intelligent Physical Agents).

2.1 Definició d'Agent

Com acabem de comentar, no existeix una definició clara del que és un Agent [17]. Tot i això, el terme agent ha esdevingut molt emprat al llarg dels últims anys i s'ha utilitzat per a referir-se a qualsevol software que tingués atributs d'intel·ligència, autonomia, percepció o interacció amb l'usuari.

Conseqüentmennt, les definicions que la comunitat científica ha proposat són ben variades. A continuació en llistem unes quantes:

Nicholas Negroponte i Alan Kay:

a) Un agent pot ser considerat un programa assistent (personal software assistant) amb autoritat delegada per l'usuari, per fer tasques que a l'usuari li són difícils o impossibles de realitzar. Aquests programes simulen una relació hunana per a fer alguna cosa que cap altra persona podria fer per tu.

Hyacinth S.Nwana de BT i, Michael Wooldridge de Queen Mary College:

b) Un agent és un component de softare i/o hardware que és capaç d'actuar per complir tasques en representació d'un usuari. Quan ha de realitzar alguna tasca, cal donar-li aquesta informació en expressions, meta-expressions, o classes. Aquesta informació cobreix diferents tipus d'agents i cal triar de quin tipus es tracta.

Pattie Maes de l'MIT[18]:

c) Un agent és un sistema computacional que habita en un entorn complex i dinàmic. L'agent pot sentir i actuar autònomament en el seu entorn, i té un conjunt d'objectius o motivacions que ell intenta d'aconseguir a través de les seves accions.

Foundation For Intelligent Physical Agents (FIPA[1]):

d) Un agent és una entitat que resideix en entorns, on interpreta dades de sensors reflectint events d'aquest entorn, i executa comandes que produeixen certs efectes en l'entorn. Un agent pot ser purament software o hardware. En el darrer cas, es necessita gran quantitat de software per a fer del hardware un agent.

2.1.1 Atributs dels Agents

Hem vist que no existeix cap definició global sobre què és un agent. Això ha portat a que el mètode més usat a l'hora de definir agents consisteixi en donar una llista de propietats o atributs que un programa hauria de tenir per a ser considerat un agent (les propietats bàsiques són solament la 1,2,3 i 9, la resta són opcionals:

1. **Autonomia:** Els agents han d'operar sense la intervenció directa dels humans o altres, i tenir algun tipus de control sobre les pròpies accions i l'estat intern. Això requereix aspectes com accions periòdiques, execucions espontànies i iniciativa, i que cada agent ha de poder prendre accions preventives o independents que beneficiïn l'usuari.

2. **Sociabilitat**: Els agents interactuen amb altres agents i/o humans a través d'algun tipus de llenguatge de comunicació. La interacció entre l'usuari i l'agent pot ser descrita com un contracte de col·laboració on l'usuari dóna especificacions que hauran de ser portades a terme per l'agent, el qual controla la fabricació d'aquest producte. La interacció entre agents és un mecanisme pel qual els agents intercanvien el seu coneixement, les seves creences i els seus plans per treballar junts i solucionar grans problemes que es troben més enllà de les seves possibilitats individuals.

3. **Reactivitat:** Els agents perceben el seu entorn i responen en un temps raonable als canvis detectats. Les accions es realitzen quan es compleixen les precondicions de les regles, que donen lloc a l'execució d'aquestes, o per l'execució de plans ja predissenyats.

4. **Pro-activitat o capacitat de prendre iniciativa:** Els agents no només han de respondre al seu entorn, sinó que han d'exhibir algun tipus de comportament encaminat a assolir algun tipus d'objectiu prenent la iniciativa.

5. **Mobilitat:** Els agent poden moure's a altres entorns a través d'una xarxa electrònica. Poden traslladar dades amb instruccions que poden ser executades remotament.

6. **Continuïtat temporal:** Els agents estan contínuament executant processos, no només un procés que s'acaba.

7. **Veracitat:** És l'assumpció que un agent no comunicarà informació falsa premeditadament.

8. **Benevolència:** És l'assumpció que els agents no tenen objectius conflictius, i que cada agent sempre intentarà fer allò pel qual és requerit.

9. **Racionalitat:** És l'assumpció que un agent actuarà per tal d'aconseguir el seu objectiu. No realitzarà una acció si les condicions per a realitzar-la no es compleixen.

2.1.2 Tipus d'Agents

A continuació hi ha una possible classificació de diferents tipus d'agents. Normalment es tracta de grans grups que poden compartir més d'una característica entre ells. Citarem les seves característiques fonamentals i els avantatges i inconvenients de cadascun.[9]

1. **Agents d'informació/ Internet:** Es tracta d'agents que tenen accés a les fonts d'informació (Internet) i són capaços d'organitzar i manipular la informació obtinguda d'aquests medis, de tal forma que d'aquesta manera poden respondre als requeriments imposats per l'usuari i/o altres agents. Els agents d'Internet són identificats majoritariament com agents que poden vagar per Internet recollint informació que se'ls hagi demanat o que puguin considerar útil.

2. **Agents d'interfície:** Es tracta d'un sistema en el que els agents col·laboren amb l'usuari per a resoldre un problema. L'agent observa i monitoriza les accions que pren l'usuari, apren d'aquestes i intenta suggerir millors maneres de dur a terme certes tasques. L'aprenentatge es pot fer de quatre formes diferents:

    a) Obsevant i imitant l'usuari
    b) A través de rebre confirmacions positives o negatives de les accions
    c) Rebent instruccions explícites de l'usuari
    d) Demanant consell a altres agents

3. **Agents de col·laboració:** Les característiques principals d'aquests agents són la comunicació i cooperació amb altres agents per a resoldre un problema comú.
   Aquests agents han de negociar per tal de poder establir una bona col·laboració. S'utilitzen tècniques d'IA distribuïda per a resoldre els problemes que puguin sorgir en un grup d'agents. La base fonamental de totes les tasques que han de dur a terme els agents col·laboratius és la comunicació. Aquesta és un punt clau en el desenvolupament d'entorns multiagent perquè es fa necessària una metodologia que permeti que s'entenguin entre ells (llenguatge de comunicació -ACL(Agent Communication Language)-) i un protocol de transmissió de dades (p.ex. TCP/IP).

4. **Agents Mòbils:** La principal característica d'aquests agents és la de poder moure's per una xarxa electrònica com pot ser per exemple la WWW. Per a poder fer-ho l'agent interactua amb altres hosts per tal de recollir informació o realitzar serveis que han estat demanats per l'usuari.

5. **Agents Reactius:** Aquest tipus d'agents, a diferència de la majoria, no treballen sobre un model simbòlic de la realitat. La seva forma d'actuar es basa en respondre als estímuls seguint un patró de l'estat en el que es troben en aquell moment. Tot això fa que aquest tipus d'agents siguin realment simples, fins i tot amb les interaccions amb d'altres agents.

   Tot el que s'ha formulat sobre aquest tipus d'agents (tècniques, teories i arquitectures), s'ha desenvolupat ja que aquesta simplicitat que remarcavem, molts cops no resulta òbvia. Sobretot quan parlem de sistemes grans amb una gran quantitat d'agents. Un exemple és l'arquitectura modularitzada de Brooks. Aquest ens proposa una arquitectura on cada mòdul és independent (reactiu), però alhora, tots ells estan pensats per a que cadascun estigui controlat per un altre formant una jerarquia.
   Les característiques dels agents en són tres de principals:

   a) La capacitat del grup es veu augmentada a la d'un element sol.
   b) Podem descomposar un problema en tasques que poden ser resoltes pels diferents mòduls independentment.
   c) Els Agents reactius treballen directament sobre les dades dels sensors, i no treballen sobre alts nivells d'abstracció com seria si ho fessin sobre símbols.

6. **Hibrids:** La pretensió d'aquests tipus d'agents és la d'ajuntar diverses característiques que poden tenir els diferents típus d'agents per tal de minimitzar els desavantatges que poseeix,per separat, cadascun d'ells.
   La imatge que es té d'un agent híbrid és la d'un Agent que està format per una part deliberativa i una altra de reactiva que interactúen entre elles. D'aquesta forma, els processos que hagin de ser ràpids i previstos a priori seran tractats per la part reactiva. La resta, les accions que comportin una planificació, es realitzaran a la part deliberativa.

2.1.3 Implementació d'Agents

Els agents poden ser implementats des de zero o bé a partir d'un programa ja existent. Per a poder ser un software heredat, necessiten primer ser convertits en un agent de software. A aquest procés se li denomina agentificació i el podem realizar de les tres següents formes:

a) Traductor: Se n'encarrega de passar el llenguage de comunicació de l'agent al protocol d'entrada del programa i a l'inrevés, de passar la sortida del programa al llenguatge de comunicació de l'agent. Tot i presentar l'avantatge de no haver de modificar ni el programa ni l'agent, amb aquest mètode es perd eficiència.
b) Embolicar (Wrapper): Consisteix en modificar el codi del programa el necessari per a poder llegir/escriure missatges en el llenguatge de comunicació de l'agent. Pot resultar una solució més eficient. Tot i així presenta l'inconvenient d'haver de modificar tots els programes.
c) Reescriure: És potser la solució més dràstica de les tres i també la més costosa. Consisteix en reescriure de nou tot el codi del programa. Amb aquesta solució es poden optimitzar al màxim les comunicacions però es perd molt de temps a l'haver de fer tota la feina un altre cop.

2.2 Sistemes Multi-Agent (SMA)

Podriem definir un Sistema Multi-Agent com un sistema compost per varis agents que interactuen entre sí per tal d'assumir un fi comú. La existència de Sistemes Multi-Agent és necessària degut a que un agent sol no seria capaç de resoldre el problema, ja que per si sols no tenen prous recursos, a més, necessitarien el coneixement de diferents dominis i haurien de manejar un volum de treball molt gran. És per això que col·laboren amb d'altres i resolen el problema entre tots, creant així un SMA. A continuació veurem com està estructurat un SMA. El procés de comunicació dels agents el resoldrem en l'apartat 2.3 on tractarem tot el relacionat amb la comunicació i els missatges basant-nos en l'especificació proposada per la FIPA.

2.2.1 Arquitectura d'un Sistema Multi-Agent

Ens podriem trobar amb més d'una especificació per a l'arquitectura d'un SMA, però donat a que és la que hem utilitzat en aquest projecte i és la més extesa actualment, explicarem la que ens defineix la FIPA[4]. A la Figura 2.1 podem veure un esquema d'aquesta arquitectura.

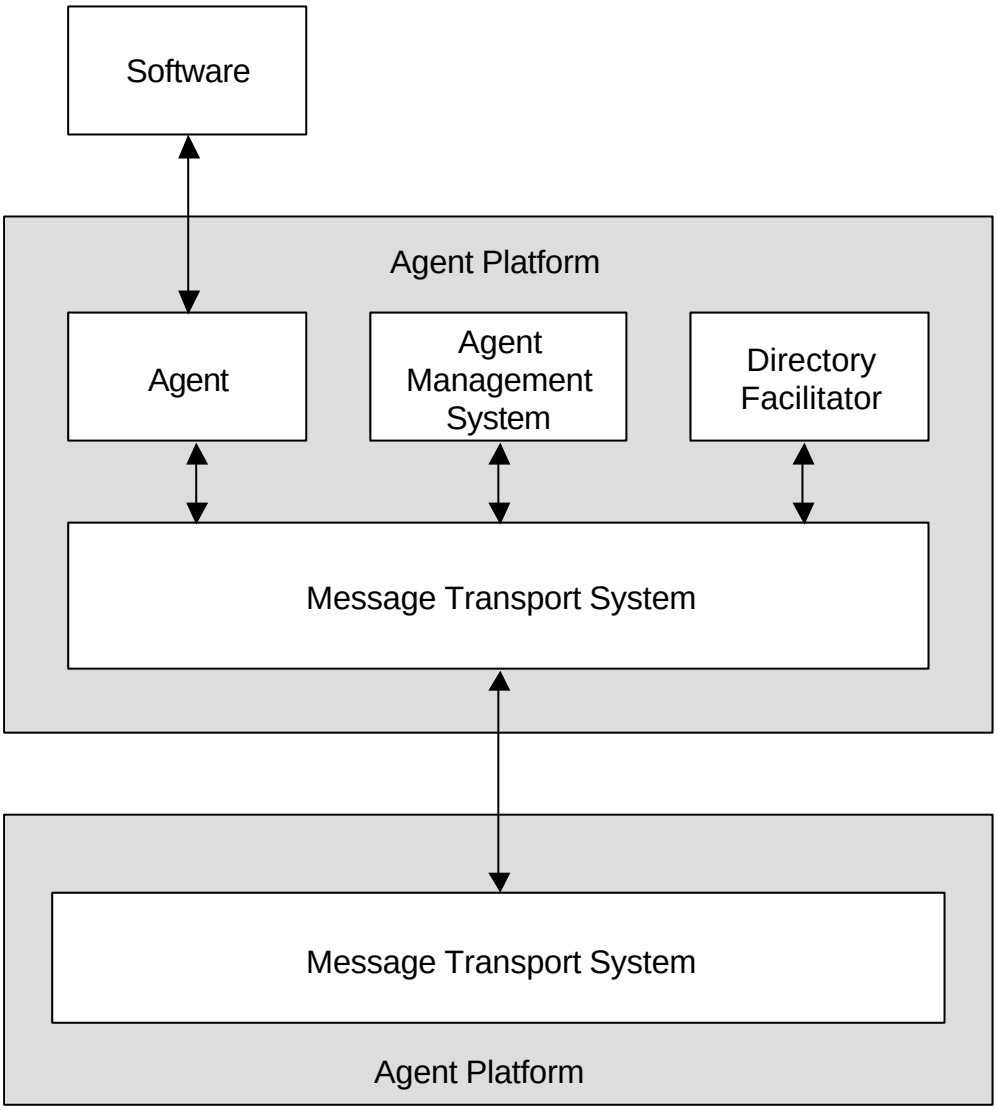

Figura2. 1 Aspecte d'un Sistema Multi-Agent basat en les especificacions de la FIPA.

En la Figura 2 podem apreciar diversos components tal com els agents o el Agent Management System (AMS). A continuació els expliquem un per un:

a) Agent: Com hem dit abans, es tracta de l'element principal d'un SMA. És un programa que ofereix una sèrie de serveis i que presenta la capacitat de poder-se comunicar amb la resta.

b) Agent Plataform (AP): Proporciona la infraestructura bàsica en la qual poden crear-se i executar-se els agents.

c) Agent Management System (AMS): Es tracta d'un agent especialitzat la funció del qual és controlar l'accés i l'ús de la plataforma propocionant un servei de Pàgines Blanques.

d) Directory Facilitator (DF): A l'igual que l'AMS, es tracta d'un agent especialitzat que sempre ha de constar en la plataforma per a que aquesta pugui funcionar correctament. Aquest ens ofereix un servei de pàgines Grogues. És a dir, coneix tots els serveis que poden oferir els agents del sistema. Per a que un agent consti amb els seus serveis, s'ha de registrar.

e) Software: Descriu qualsevol programa accessible des d'un agent.

2.2.2 Avantatges d'utilitzar un Sistema Multi-Agent

Ara que ja hem explicat com és un SMA i les parts que l'integren, se'ns planteja una nova pregunta, perquè crear un SMA. La seva estructura no és precisament senzilla. Potser amb un agent acabariem abans. Però la realitat és que un SMA presenta uns aventatges que no tindriem amb un sol agent. Els avantatges més importants són els següents:

1) Modularitat: Ens permet realitzar la feina distribuint-la en tasques que poden ser executades per diferents agents. D'aquesta manera reduim la complexitat i se'ns permet una programació més estructurada.

2) Eficiència: Al distribuir les feines entre els agents, podem aconseguir paral·lelisme si tenim agents treballant en diferents màquines.

3) Fiabilitat: Al tenir el sistema composat per un alt nombre d'elements podem suposar que el fet que un element del sistema deixi de funcionar no impedeix que la resta continuin fent-ho. També podem aconseguir augmentar la seguretat duplicant sereis esencials.

4) Flexibilitat: Podem afegir i eliminar agents de forma dinàmica.

2.3 Comunicació entre Agents

Per a que els agents del SMA pugin col·laborar cal que siguin capaços d'intercanviar informació. Per això cal definir estàndards per a poder realitzar SMA que pugin ser desenvolupats per diferents grups i es pugin comunicar entre ells.

La comunicació és la part més importat d'un agent. Si tenim més d'un agent, la seva existència és imprescindible per a que els agents pugin treballar conjuntament i formar així un sistema multiagent. La especificació que explicarem a continuació és la que ha realitzat la FIPA (Foundation for Intelligent Physical Agents)[3].

A banda de la de la FIPA, també podem trobar-ne d'altres tals com les especificacions realitzades per el KQML (Knowledge Query Modeling Language), aquestes són força semblants a les proposades per la FIPA. Aquestes dues especificacions són genèriques, o sigui, la seva utilitat s'exten a un ampli grup d'aplicacions. A banda d'aquests, però, també ens podem trobar amb llenguatges definits especialment per una aplicació particular. A banda dels indubtables aventatges de que d'aquesta forma es poden centrar més en el problema particular, també se'ns presenta l'inconvenient de que no poden tenir compatibilitat amb d'altres aplicacions.

2.3.1 Missatges FIPA

En la comunicació, l'element més important és el missatge. La FIPA ens descriu la forma que han de tenir els seus llenguatges i com s'han d'utilitzar.

Els missatges estan formats per una estructura formada per diversos camps o 'slots'. La utilització de cada un d'ells ve donada per el que nosaltres necessitem i volem realitzar. Aquests parametres es troben classificats per la funció que realitzen. A continució podem veure aquests camps i la categoria de la que són. Més tard els analitzarem un per un.

Nom del camp	**Categoria a la que pertanyen**
performative	Definició tipus missatge
sender	Participant de la comunicació
receiver	Participant de la comunicació
reply-to	Participant de la comunicació
content	Contingut del missatge
language	Descripció del contingut
encoding	Descripció del contingut
ontology	Descripció del contingut
protocol	Control de conversa
conversation-id	Control de conversa
reply-with	Control de conversa
in-reply-to	Control de conversa
reply-by	Control de conversa

a) Definició del tipus de missatge que s'envia

Com hem vist, en aquesta categoria només hi ha un element: performative. És a on s'indica quin és el tipus d'acte de comunicació que s'està realitzant i que volem fer constar en el missatge ACL. Els difernts tipus d'actes de comunicació que podem realizar són els següents:

Performative	**Descripció**
accept-proposal	Es realitza l'acceptació d'una proposta per a una acció
agree	Ens diu que s'està d'acord en realitzar l'acció
cancel	Cancela alguna petició anterior quan encara no ha acbat
cfp	(Call For Proposal) Demana propostes a un conjunt no buit d'agents per a realitzar alguna acció
confirm	Es confirma al receptor que el que ha preguntat és cert. Aquest no n'estava segur.
disconfirm	S'envia per a desmentir una proposició. És l'inversa del confirm
failure	S'indica la fallada d'una petició feta amb anterioritat
inform	S'informa d'algun fet al receptor
inform-if	S'informa si una proposició és certa o no al receptor
inform-ref	S'informa al receptor si un objecte al qual es fa referència, és cert o no.
not-understood	Missatge no entès
propagate	Un missatge i un conjunt d'AIDS als qui es renviarà el missatge és enviat. El primer agent tracta el missatge i el torna a enviar. Per tal de tallar els reenviaments es té una restricció com, per exemple, un timeout.
propose	Es fa una proposta a un cfp rebut
proxy	S'espera que el receptor identifiqui els agents que poden satisfer una descripció donada i els retorni
query-if	Serveix per a formular una pregunta d'una proposició
query-ref	El mateix però sobre un objecte
refuse	Es refusa una petició
reject-proposal	És l'oposat de l'accept-proposal
request	Es demana a un agent que realitzi una acció
request-when	El mateix que l'anterior però sota unes determinades condicions
request-whenever	Es demana a un agent que realitzi una acció tan bon punt es donin les condicions especificades
subscribe	Serveix per a subscriure's a un agent per a un determinat servei.

b) Participant de la comunicació

Un participant de la comunicació és un identificador d'un agent. Aquest pot ser en tres dels slots d'un missatge. El sender que és qui envia el missatge, el receiver que és qui el rep i el reply-to que és a qui ha d'anar destinat el següent missatge de la conversa.

c) Contingut del missatge

Igual que en la primera categoria, en aquesta només es té en compte un paràmetre: content. El contingut del missatge pot variar el format depenent de les necessitats que tinguem. Seguint les especificacions de la FIPA [5]ens podem trobar amb quatre tipus de continguts:

- El llenguatge FIPA-CCL (FIPA Constraint Choice Language). Aquest defineix una semàntica que permet l'especificació de predicats amb satisfació de restriccions (CSP). Suporta la representació de problemes , recolecció d'informació, tècniques de fusió i aplicació de tècniques de resolució de problemes.

- El llenguatge FIPA-SL (FIPA Semantic Language). Aquest ens permet de formar expresions lògiques, intercanviar de forma òptima coneixement entre agents, i expressar accions a realitzar.

- El llenguatge FIPA-KIF (FIPA Knowledge Interchange Format). Ens permet d'expressar objectes com a termes i proposicions com a sentències. No permet l'expressió d'accions.

- Finalment, el llenguatge FIPA-RDF (FIPA Resource Description Framework). Aquest és molt semblant al FIPA-SL i permet representar objectes, interaccions i accions entre ells.

d) Descripció del contingut

Per a que un agent entengui el contingut necessita saber:

a) language: És on es defineix el llenguatge que s'ha emprat per a escriure el camp de content. Els possibles valors que pot pendre aquest camp són els següents: FIPA-SL, FIPA-KIF, FIPA_RDF, FIPA-CCL, plain, Prolog, Lisp, Java,etc. Quan un agent rep un missatge mira aquest camp per a veure si podrà contestar.

b) encoding: EL conjunt del missatge pot ser codificat de diverses maneres, tot i que normalment es transmeten cadenes de caràcters es pot donar el cas de que es codifiqui. Aquest camp , opcional, es té en compte la codificació emprada.

c) Ontology: En aquest camp , força important, és on se'ns defineix la ontologia que utilitzarem per a donar significat al conjunt del missatge. Més endavant, en la secció 3.5 podrem veure com es crea i s'estructura una ontologia. Però per a fer-nos una idea, en la Ontologia és on es defineix el vocabulari i les estructures que ens podrem trobar en el contingut del missatge

e) Control de conversa

Els agents mantenen converses entre si. Aquests slots estan destinats a identificar quin tipus de converses es mantenen i quines són.

- protocol: Identifica el tipus de protocol que es fa servir en el missatge. En la següent secció s'expliquen els protocols i el seu funcionament
- conversation-id: És l'identificador de la conversa sencera
- reply-with: Identificador que s'empra per a seguir els diferents passos de la conversa per a que poguem saber en quin punt ens trobem
- in-reply-to: Identifica una acció passada de la qual el missatge és la resposta
- reply-by: Aquest camp és una marca de timeout. S'especifica una data i/u hora que indica la data de caducitat del missatge. Més enllà, qui ha enviat el missatge pot o no fer-ne cas, dependrà de com s'implementi.

2.2.1 Protocols de Comunicació

Com hem dit anteriorment, a l'hora d'enviar un missatge s'ha d'especificar el protocol que s'utilitzarà. Els protocols de comunicació defineixen els passos que s'han de seguir per a realitzar una determinada conversa. Hem de tenir en compte que n'hi ha de diferents tipus depenent de la finalitat buscada. Ara explicarem una mica els protocols proposats per la FIPA[8].

- FIPA Request: Aquest protocol ens permet de demanar la realització d'accions i retornar els resultats

- FIPA Query: Protocol emprat per a fer preguntes. Es poden fer dos tipus de preguntes.: query-if i query-ref. El primer ens pregunta si una proposició és certa o no. El segon demana alguna cosa descrita per alguna expressió.

- FIPA Contract Net: és un protocol de negociació en el que un agent fa una proposta a un grup d'agents, aquests li retornen propostes amb les condicions que realitzaran l'acció. El demandant n'escolleix un.

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

- FIPA Brokering: Apareix en els sistemes que disposen d'un agent broker que té coneixements dels serveis que ofereixen un conjunt d'agents. Aquest protocol serveix per a gestionar la comunicació amb un agent d'aquest tipus.

- FIPA Recruiting: Es tracta d'una variant del protocol anterior en la qual hi intervé un tercer agent.

- FIPA Subscribe: Aquest protocol permet la notificació de fets importants. Un agent demanarà a un altre que quan succeeixi un fet aquest li sigui notificat.

- FIPA Propose: Es tracta d'una simplificació del Contract Net.Aquest és especialment útil en les ocasions en les que el Contract Net sigui massa general.

3. IMPLEMENTACIO D'UN SMA EN JADE

JADE (Java Agent Development Enviroment) [12] és un suport per a desenvolupar Sistemes multiagent. Està format per un conjunt de lllibreries escrites en Java que ens permeten el poder desenvolupar Sistemes Multiagent més fàcilment.

El que ens permet fer JADE és desenvolupar un SMA sense que ens haguem de preocupar de com resolem alguns detalls de baix nivell. A més, JADE també ens ofereix un entorn d'execució per als nostres agents.

A continuació descriurem les parts de JADE, com s'implementes agents en aquest entorn i veurem per sobre les eines gràfiques que ens dóna.

3.1 Plataforma d'Agents de JADE

Per JADE, els agents resideixen en un entorn predefinit anomenat plataforma. Cada plataforma està dividida en contenidors i cadascun d'ells és on es trobaran continguts els agents[14]. Els contenidors poden entendre's com dominis d'agents.

La plataforma s'engega en una màquina determinada. En canvi, els agents podran trobar-se executant en d'altres estacions o en la mateixa on s'ha engegat la plataforma. Aquesta plataforma permet als agents comunicar-se emprant el FIPA-ACL.

Tal com es defineix a la FIPA , en cada plataforma hi ha d'haver dos agents que tenen unes funcions imprescindibles per al funcionament del sistema:

- *Domain Facilitator* (DF): Normalment se'l descriu com un servei de "pàgines grogues" on els diferents agents que es trobin en el sistema podran registrar-se juntament amb els serveis que ofereixin per a que així la resta de agents que els requereixin els puguin localitzar.

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

Aquests dos agents són d'existència obligatòria. Quan ens trobem en el cas de que un agent A desitja enviar un missatge a un agent B el qual desenvolupa la tasca T, l'agent A preguntarà al DF quin o quins agents poder fer-se càrrec de la tasca T. Llavors el DF li retornarà l'adreça de l'agent B. Quan es vol conèixer l'adreça d'un altre agent, la pregunta es fa al AMS.

3.2 Implementació d'Agents

Un agent es crea mitjançant l'extensió de la classe Agent. Un cop hem extès la classe ens cal implementar els mètodes obligatoris de la classe . Aquests són el setup() i el takedown().

- setup: S'executa el primer cop que l'agent entra en el sistema. S'acostuma a usar per a realitzar les inicialitzacions pertinents i posar en funcionament algun comportament per a iniciar les tasques de l'agent.

- takedown: Aquest és el mètode complementari de l'anterior. S'executa quan l'agent surt del sistema per algun motiu.

Després de crear l'esquelet i implementar les dues funcions anteriors adequadament, cal implementar els serveis de l'agent. Per a fer-ho utilitzarem els *behaviours* (o comportaments). És obligatori implementar com a mínim un *behaviour* per agent. Aquests *behaviours* són els que controlen les seves accions. Un *scheduler* intern serà l'encarregat d'anar controlant automàticament l'execució de tots els comportaments.

JADE ens proporciona una col·lecció bàsica de behaviours que ens permeten implementar diferents protocols de comunicació. Tenim dos grans grups, els simples i els complexos. Els comportaments imples són els que no tindran fills.

- SIMPLES

  - *SimpleBehaviour*: representa un comportament atòmic que s'executa sense interrupcions. Aquest pot ser de tipus OneShotBehaviour (aquest s'executarà un sol cop), o CyclicBehaviour (executarà una acció cíclicament).

  - *ReceiverBehaviour:* Aquest behaviour permet engegar un comportament que espera la recepció d'un missatge.

  - *SenderBehaviour*: Aquest és equivalent a l'anterior però des del punt de vista del qui envia el missatge.

- COMPLEXES

  - *SequencialBehaviour:* Aquest comportament permet la creació de fills els quals executarà de manera seqüencial. Aquest acabarà quan tots els mètodes finalitzen correctament.

  - *NonDeterministicBehaviour:* A l'igual que l'anterior permet la creació de fills, però els executa segons una seqüència no determinista.

3.3 Creació de missatges en JADE

JADE, en el format dels missatges segueix les especificacions donades per la FIPA. Així, ens trobem que un missatge estarà dividit en slots que ens donen una informació (llenguatge, protocol, contingut...). JADE ens proporciona uns mètodes setxxx i getxxx que ens permet escriure i llegir el contingut dels corresponents missatges[15].

Els mètodes bàsics que ens proporciona per al manegament dels missatges entre agents són els següents:

- *Send (ACLMessage):* Amb aquest mètode se'ns permet enviar un missatge un cop hagim omplerts tots els paràmetres requerits.

- *ACLMessage receive():* Amb aquest mètode, l'agent rep un missatge que es troba en la seva cua. Podem especificar el tipus de missatges que volem rebre especificant uns patrons de cerca que ens proporciona JADE. Aquests patrons són els anoments Message Template.

L'estructura del missatge en JADE és la que defineix la FIPA (veure secció 2.3.1). En el nostre cas, el llenguatge per escriure el contingut del missatge serà el FIPA-SL. La nostra elecció ha estat aquesta ja que és el que disposa de més possibilitats. Amb ell podrem utilitzar expressions lògiques, intercanviarem informació entre agents i podrem expressar les accions a realitzar.

3.4 Ontologia definida en JADE

A l'hora de crear un missatge un dels camps que haurem de determinar és l'Ontologia que és on definim el vocabulari utilitzat pels nostres agents per a escriure el contingut dels missatges que seran capaços d'entendre.
Definir una ontologia adaptada al problema que volem tractar, és un dels elements més importants i que cal haver decidit completament abans d'implementar els agents. El manegament d'un missatge FIPA-SL[2], si es té present el següent no és difícil:

- Els missatges s'estructuren en frames i cada frame en slots.
- Els slots poden contenir dades simples o bé complexes, aquests últims podran contenir frames.
- Per a accedir a un slot es farà mitjançant el nom o el tipus, això es podrà fer ja que estàn etiquetats.

A l'hora de definir la ontologia hem de crear un mòdul en el que definirem tots els frames. Un frame pot ser un objecte o un predicat depenent de la utilitat final a la que es destini. La única diferència a l'hora de la implementació és que en el cas dels predicats no ens farà falta donar un identificador pels camps (slots) que el formen.

A banda d'aquest mòdul, també haurem d'implementar una sèrie de classes en Java on implementarem els mètodes d'accés de cada slot. Haurem de crear tantes classes com frames tinguem.

3.5 Implementació de protocols de comunicació

Després de definir l'ontologia haurem de decidir quin protocol de comunicació volem escollir per a cada cas intentant trobar el que més s'adapti a les nostres necesitats. Un cop decidit quin volem, haurem d'implementar-lo. És en aquest punt on JADE intenta facilitar-nos les coses. JADE ens deixa implementar diferents protocols de comunicació

definits per la FIPA amb certa facilitat ja que disposa dels **behaviours** i **performatives** corresponents, lamentablement no tots els protocols de la FIPA estan disponibles, en JADE només tenim:

Protocol FIPA **Query**
Protocol FIPA **Request**
Protocol FIPA **Contract Net**

Per exemple, el protocol *subscribe*, tot i que els agents es registren al DF i aquest disposa en tot moment de quins agents hi ha actius al sistema, els agents no es poden subscriure a cap servei.

Els protocols consten de dues parts bàsiques, el que fa la crida del protocol (el que inicia la conersa, el *initiator*) i el que manega la resposta d'aquest (*responder*). Per a implementar cadascuna de les dues parts s'empra una classe abstracta que ens proporciona JADE. Aquestes classes són la xxxxInitiatorBehaviour i xxxxResponderBehaiour on xxxx es refereix al protocol que volem implementar. A l'hora d'utilitzar-les el progrmador solament caldrà que sobreescrigui els mètodes que manegaran els diferents estats pels quals es passa en una conversa.

Per posar un exemple, en aquest projecte s'han hagut d'implementar els protocols FIPA **Request** i FIPA **Contract-Net**. Per el primer s'han utilitzat les classes *FIPARequestInitiatorBehaviour* i *FIPARequestResponderBehaviour*. En el cas del segon les clases emprades han estat la *FIPAContractNetInitiator* i la *FIPAContractNetResponder*.

A continuació passem a descriure amb més detall els tres protocols que es poden implementar en JADE[17]:

3.5.1 FIPA Query Protocol

Aquest protocol s'utilitza per a demanar informació a un altre agent (Figura 3.1). Al llarg de la seva execució es van seguint una sèrie de passos:

- Es realitza l'enviament d'una pregunta per part de l'iniciator. Aquesta pot ser un query o un query-ref. depenent del que desitgem.

- A continuació l'agent responder enviarà la informació demanada o podrà refusar la petició. També ens podem trobar en un cas en el que es produeixi un error, en aquest cas es rebrà una fallada. En el cas de que no hagi entès el missatge (això pot venir donat per diversos motius), s'enviarà un not-understood.

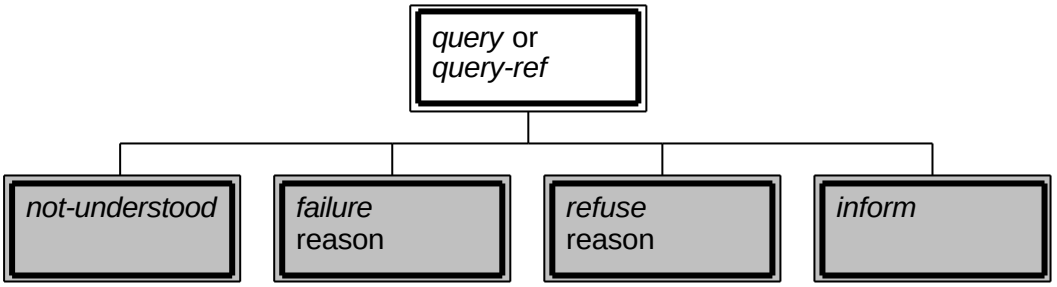

Figura3.1. Estructura del FIPA Query Protocol

3.5.2 FIPA Request Protocol

És segurament el protocol més utilitzat. Ens permet demanar la realització d'accions i rebre un informe de com han acabat (Figura 3.2). El seu funcionament el podriem resumir amb els següents passos:

- Un agent iniciator envia el missatge amb la peticició de la realització de l'acció corresponent. Aquest és un missatge de tipus request.

- L'agent receptor (el receiver), al rebre el misatge i esbrinar quina acció li han demanat, sospesarà si la pot realitzar o no. En el cas afirmatiu respondrà amb un agree i iniciarà el procés. En canvi, si no la pot dur a terme, rebutjarà la petició amb un refuse en el qual pot informar del motiu si és necessari. Si per algun motiu no hagués entès el missatge, respondria amb un not-understood.

- Si l'iniciador rep un agree s'esperarà a que l'altre agent acabi de reaitzar l'acció i li envii un inform. Aquest inform pot ser de dos tipus. O bé un inform-ref en el que et retorna el resultat o bé un inform-done conforme ha realitzat l'acció

- Quan el receiver finalitzi enviarà l'inform corresponent. En el cas de que es produís algun error s'enviaria un failure indicant el motiu.

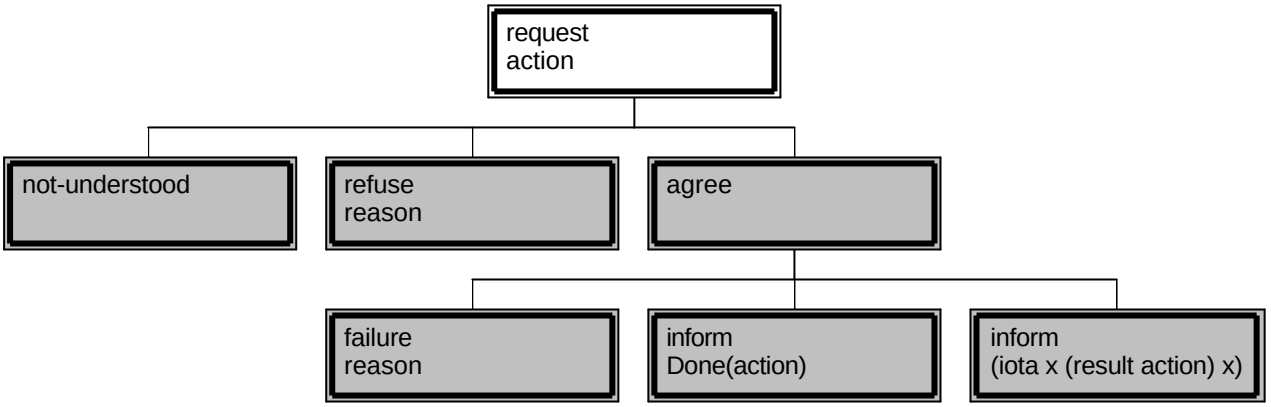

Figura 3.2 Estructura del FIPA Request Protocol

3.5.3 FIPA Contract Net Protocol

Aquest és potser el protocol més complexe de tots. La seva utilització està limitada a quan es vol realitzar una negociació d'accións amb varis agents, buscant el que et dóni millors condicions (Figura 3.3). Els passos a seguir en aquest cas són els següents.

- L'agent iniciador envia una petició per a realitzar una acció, passant unes precondicions a complir a un grup d'agents.

- Els agents que la rebin analitzaran el que se'ls hi demana i o bé podràn rebutjar-la o bé acceptar-la. Si la accepten s'enviarà una proposta d'acció amb unes condicions determinades.

- Quan l'agent inicial sàpiga que ha acabat de rebre totes les propostes (Ho sabrà mitjançant l'utilització d'un timeout). Mirarà totes les propostes que li hagin arribat i escollirà la que millors condicions presenti. A continuació enviarà un missatge d'acceptació de proposta a l'agent corresponent, i un altre de rebuig a la resta d'agents.

- Per a finalitzar, l'agent seleccionat començarà a realitzar l'acció, quan finalitzi enviarà un missatge d'inform o de failure segons hagi anat bé o s'hagi produit un error.

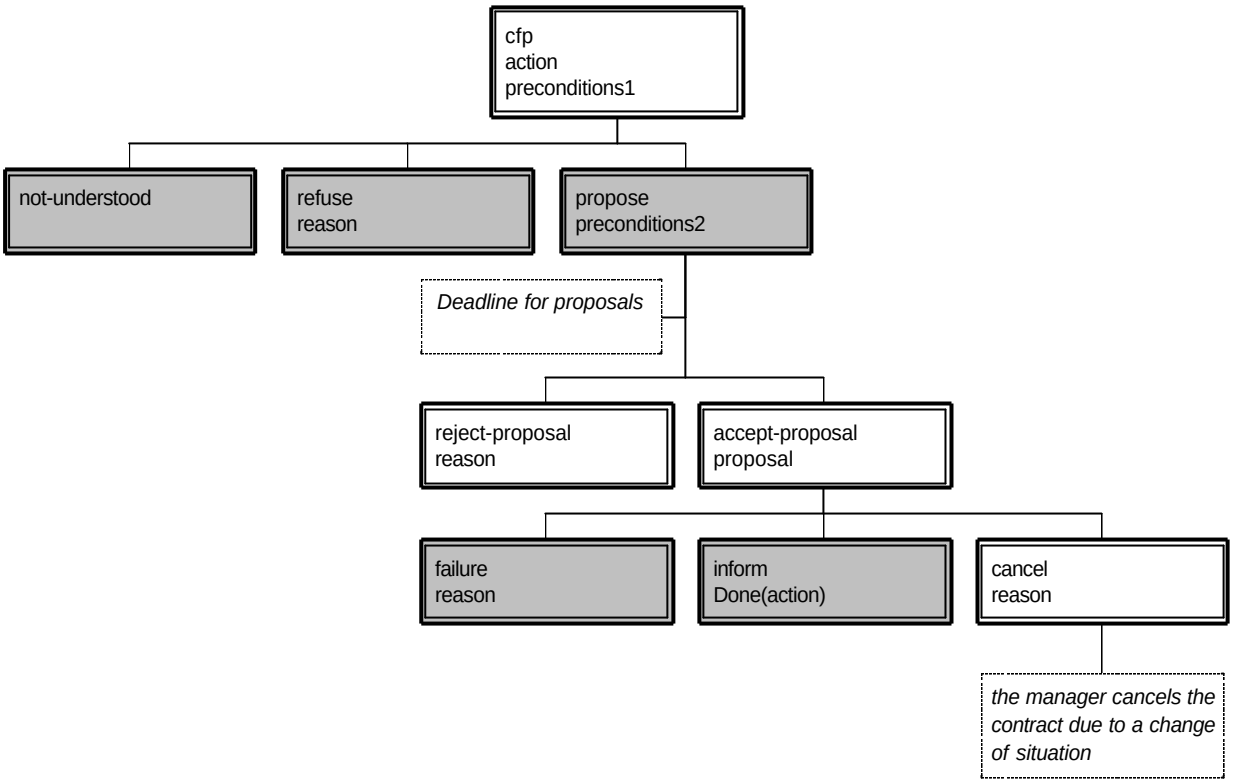

Figura 3.3. Estructura del FIPA Contract-net Protocol

3.6 Els paquets de JADE

Com hem esmentat abans, JADE està composat per una sèrie de paquets escrits en JAVA. Els més importants són els que llistem a continuació[16]:

- *jade.core:* És el nucli del sistema. Proporciona la classe jade.core.Agent que és imprescindible. A més, també conté el paquet jade.core.behaviour dins del qual s'hi troben les classes de comportaments suportats.

- *jade.lang*: Conté les classes específiques per a suportar un llenguatge de comunicació entre agents (ACL). Dintre d'ell hi ha els paquets jade.lang.acl i jade.lang.sl.

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

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

- *jade.tools*: Tobem algunes eines que faciliten el desenvolupament d'aplicacions i l'administració de la plataforma:

    1. Remote Management: Agent (RMA): Aquest actua com una consola gràfica per a l'administració i control del sistema.
    2. Dummy Agent: és una eina de monitorització i depuració, composta per una interfície gràfica i un agent JADE. Amb l'ajuda de la interfície ens és possible crear missatges ACL i enviar-los a altres agents, tot això podent visualitzar-los.
    3. Sniffer: agent que intercepta missatges ACL i els visualitza gràficament.
    4. SocketProxyAgent: Es tracta d'un simple agent que actua com un gateway bidireccional entre la plataforma JADE i una connexió TCP/IP.

3.7 Entorn gràfic del JADE

A més de tot el que hem explicat, el JADE també ens proporciona un entorn gràfic on podem manipular el nostre SMA fàcilment. Treient o posant agents, creant-ne de nous, etc.

Aquest entorn s'engega activant una opció a la línia de comandes. Així, si posem la opció -gui s'ens engegarà l'entorn gràfic del JADE on podrem veure el funcionament del nostre sistema. Els missatges que s'envien entre els agents, el que reben, etc. Tot això gràcies a que el sistema posseeix un Sniffer capaç de fer-ho.

La imatge de l'entorn Jade seria el representat en la figura 3.4.

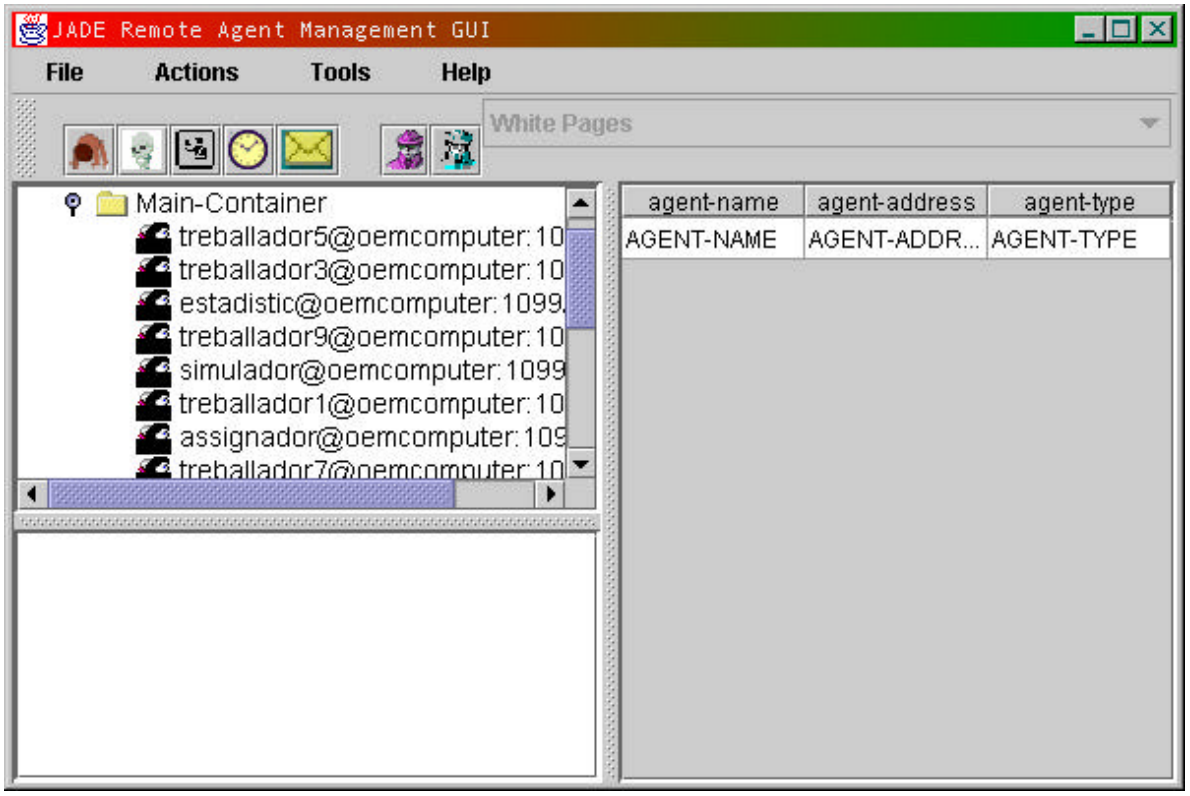

Figura 3.4. Pantalla principal

4. DESCRIPCIÓ DEL PFC

Aquest projecte final de carrera és un Simulador de grups de treball. La seva funció és la de realitzar diverses simulacions de cada una de les diverses assignacions que es puguin fer. Al final tindrem els resultats dels grups que han treballat d'una forma més òptima, els que han tingut menys errades, etc.

A continuació explicarem quins han estat els objectius a l'hora de crear el projecte, justificarem el perquè de l'elecció d'un SMA, les decisions de disseny que hem hagut de pendre tals com per quins paràmetres havia de tenir el sistema o quins camps haviem de tenir en compte en cada treballador. Després explicarem la arquitectura del sistema. A continuació descriurem el funcionamet del sistema. Per a finalitzar, inclourem un joc de proves, unes conclusions i els possibles futurs treballs que es poguessin realitzar amb la idea en la que està basada aquest sistema.

4.1 Objectius del Projecte

El problema que ens va dur a realitzar aquest treball va ser la dificultat que es plantejava a l'hora de seleccionar el grup de treball més idoni per a una feina. En un principi no sembla potser una feina excessivament dura, s'agafen les persones que tenen el nivell indicat per a realitzar una tasca i el problema ja hauria d'estar solucionat. Però no és el cas, es poden plantejar problemes derivats de que no tothom actúa de la mateixa manera i tampoc poseeix un mateix grau d'experiència, fins i tot que no s'entenguin entre ells. Aquest projecte el que vol és servir de guia per a casos com aquests. Les seves pretencions són les de donar una guia més o menys fiable de com aniran les coses. Intentant trobar quins serien els casos per els que es tardaria menys o més temps, els més costosos, els que podrien tenir un major nombre de fallides (Una fallida es pot produir per dues causes, o bé perquè un treballador no acaba, o bé perquè es sobrepassa el deadline imposat). El que ens interessa veure és el resultat final de tot l'equip.

El projecte tampoc no vol ser una solució definitiva. Tots sabem que el que succeirà no es pot determinar a priori, però si que es pot intentar preveure amb més o menys encert. En certa manera, la feina d'aquest projecte és semblant a la que realitzen els homes del temps. Basant-se en unes dades reals que són considerades fiables, extreuen unes conclusions del que ells consideren que pot succeir.

Així doncs els objectius d'aquest projecte són els d'intentar ajudar a un Cap de projecte a seleccionar entre els seus treballadors quins són els més idonis per a realitzar una tasca, i fer-ho utilitzant tècniques d'Intel·ligència Artificial. En el nostre cas, un Sistema Multi-Agent.

4.2 Justificació de l'elecció d'un SMA

El fet d'haver escollit un SMA per a implementar aquest simulador es basa en tres motius bàsics que amb un altre tipus d'implementació no hauriem tingut o com a molt algun, i no amb el mateix grau en que es troba en el cas d'un SMA. Aquets són els següents:

1. Modularitat: Ens permet realitzar la feina de forma distribuida, és a dir, en el nostre cas, aquesta ens permet reduir la complexitat que podria arribar a tenir el nostre Sistema Simulador. Aquest, sinó, s'hauria pogut convertit en un programa abarrotat i difícil d'entendre. El més costós, sobretot, hauria estat controlar tots els passos de la simulació.

2. Fiabilitat: Al tenir el sistema composat per un alt nombre d'elements podem suposar que el fet que un element del sistema deixi de funcionar no impedeix que la resta continuin fent-ho, a no ser que aquest sigui escencial. En el nostre cas ens podria passar, per exemple, amb l'assignador. Però si ens passes amb un d'Obrer no tindriem pas cap problema. També podem aconseguir augmentar la seguretat duplicant serveis escencials.

3. Flexibilitat: Aquesta ens permet que poguem realitzar simulacions amb diversos nombres d'agents, podent decidir en qualsevol moment quants n'hi vols posar, sabent que funcionarà igualment per vuit que per quinze.

A banda dels avantatges que acabem de descriure, un dels motius pels que també ens interessava la utilització d'un Sistema Multi-Agent és per l'autonomia presentada pels diferents agents. Aquesta els permet tenir un coneixement propi on podrem guardar les característiques de cada treballador.

4.3 Requeriments

Al decidir com volia que fos el nostre Simulador vam haver de definir certs paràmetres, camps a tenir en compte pels treballadors, etc.

1) Paràmetres del sistema:

Una de les primeres coses a tenir en compte són els paràmetres del sistma tals com el nombre d'assignacions a realitzar o el número de vegades a realitzar cada assignació. Tots aquests paràmetres són entrats per l'usuari:

- El número d'assignacions a realitzar es referix al número de diferents grups amb que realitzarem les corresponents simulacions.
- Número d'agents que es canviaran a cada assignació
- Número de simulacions per assignació. Es refereix a la quantitat de vegades que l'agent simulador haurà de simular una assignació.

2) Graf de tasques:

Per tal de poder repartir la feina entre els treballadors es va decidir dividir-la en una sèrie de tasques que s'han de realitzar en un cert ordre. Conseqüentment, un punt important a tractar és com ha de ser el graf de tasques que reflexi aquest comportament, quines limitacions ha de tenir i com el guardarem. Per a fer-ho vam decidir que aquest s'entrés a través d'un fitxer, tasques.txt. Els paràmetres del graf de tasques són:

- Deadline: Aquest especifica la durada màxima que podrà tenir un projecte. Sobrepassar aquest límit en una simulació provoca una fallada.
- A través del fitxer de tasques serà on l'usuari especificarà el nombre d'assignacions a realitzar, el número d'agents a canviar per simulació i el número de simulacions per cada assignació.
- El nombre de tasques.
- Les tasques i els seus paràmetres: la duració aproximada de cada una, el seu grau de dificultat i les tasques successores, és a dir, les que toca realizar un cop hagi acabat. L'identificador d'una tasca vindrà donat per la posició d'aquesta en el fitxer.

3) Treballadors:

Serà als qui se'ls hi destinaran directament les tasques. De tipus de treballadors en tindrem tres depenent del seu nivell professional:

- ADP (Agent Director de Projecte). És el tipus de treballador amb un nivell professional més elevat.
- AEQ (Agent Enginyer Químic). El seu nivell professional és intermig.
- AT (Agent Tècnic). És el tipus de treballador que presenta un nivell professional més baix. Després d'ell només hi haurien els agents.

Cada un d'ells, a banda del seu nivell professional, tindrà una sèrie d'atributs que serviran per a modelar-lo en funció del seu homònim real. Aquests atributs són els següents:

- Nom: Li servirà d'identificador en el sistema..
- Grau d'experiència: Aquest camp serà el que usarem per a calcular la possibilitat de una fallida durant el procés de simulació.
- Sociabilitat: L'utilitat d'aquest camp és la de decidir si requerirà l'ajuda d'un Agent obrer o preferirà realitzar la tasca sol. Quan es decideix emprar l'ajuda d'un obrer, el temps que es tarda en la realització de la tasca sol ser menor.

4) Obrers:

Els agents obrers, en canvi, no es distingiran en tipus i no se'ls hi podrà assignar una tasca directament, hauran d'esperar a que algun agent treballador requereixi els seus serveis. A més, a part del nom, només tindran una característica personal que serà el grau d'Experiència. En aquest cas no és necessari el grau de Sociabilitat ja que no es necessita decidir si volen col·laborar o no.

4.4 Arquitectura

Podriem fer una divisió, amb caràcter general, dels agents que hi ha dins del nostre SMA. Els dos grans grups que podem considerar són, per una banda, els agents que emulen el comportament d'una persona física i per l'altra els agents dedicats a fer funcionar tot el procés de la simulació.

Dels primers ja hem descrit breument les característiques que poseeixen en l'apartat anterior. Com ja hem esmentat, aquests es divideixen entre Agents Treballadors i Agents Obrers. Els treballadors hauran d'interactuar directament amb l'agent Simulador, un dels membres del segon grup.

El grup dels agents dedicats a fer funcionar tot el procés de la simulació poseeix tres membres: l'assignador, el simulador i l'estadístic:

- **Agent Assignador.** És l'encarregat de crear les diverses assignacions que es simularan, i d'anar-les enviant a l'agent Simulador.

- **Agent Simulador.** La feina d'aquest, que resulta ser la més important, serà la de rebre les assignacions produides per l'agent assignador i realitzar la seva execució, o sigui, anar informant als treballadors de les tasques que aquests han de realitzar i emmagatzemant aquests els resultats que generin aquestes assignacions.

- **Agent Estadísitc** Serà qui es dedicarà a analitzar la informació produida per les diverses simulacions. Per exemple, mirarà quina ha estat el grup que ha tardat menys temps, o ha tingut un cost elevat, etc.

A l'hora d'entrar les dades i visualitzar-les tenim dos casos difernciats. El primer és un fitxer (tasques.txt) que, tal i com hem dit a l'apartat anterior, usarem per a entrar el graf de tasques. Aquest fitxer serà llegit per l'agent Assignador.

L'altre opció que hem utilitzat és la de crear una interfície gràfica per a cada Agent, exceptuant el simulador. Aquesta la utilitzarem per a entrar les dades personals als treballadors i obrers, indicar-li a l'assignador que ja pot començar. També l'usarem per a visualitzar els resultats analitzats per l'agent Estadístic.

Per a aquest SMA s'ha utilitzat JADE. És per això que, a banda dels agents que hem esmentat anteriorment, també s'hi trobaran els agents bàsics que inclou aquest sistema com poden ser el DF i l'AMS. Conseqüentment, l'estructura final, quedaria com a la Figura 4.1

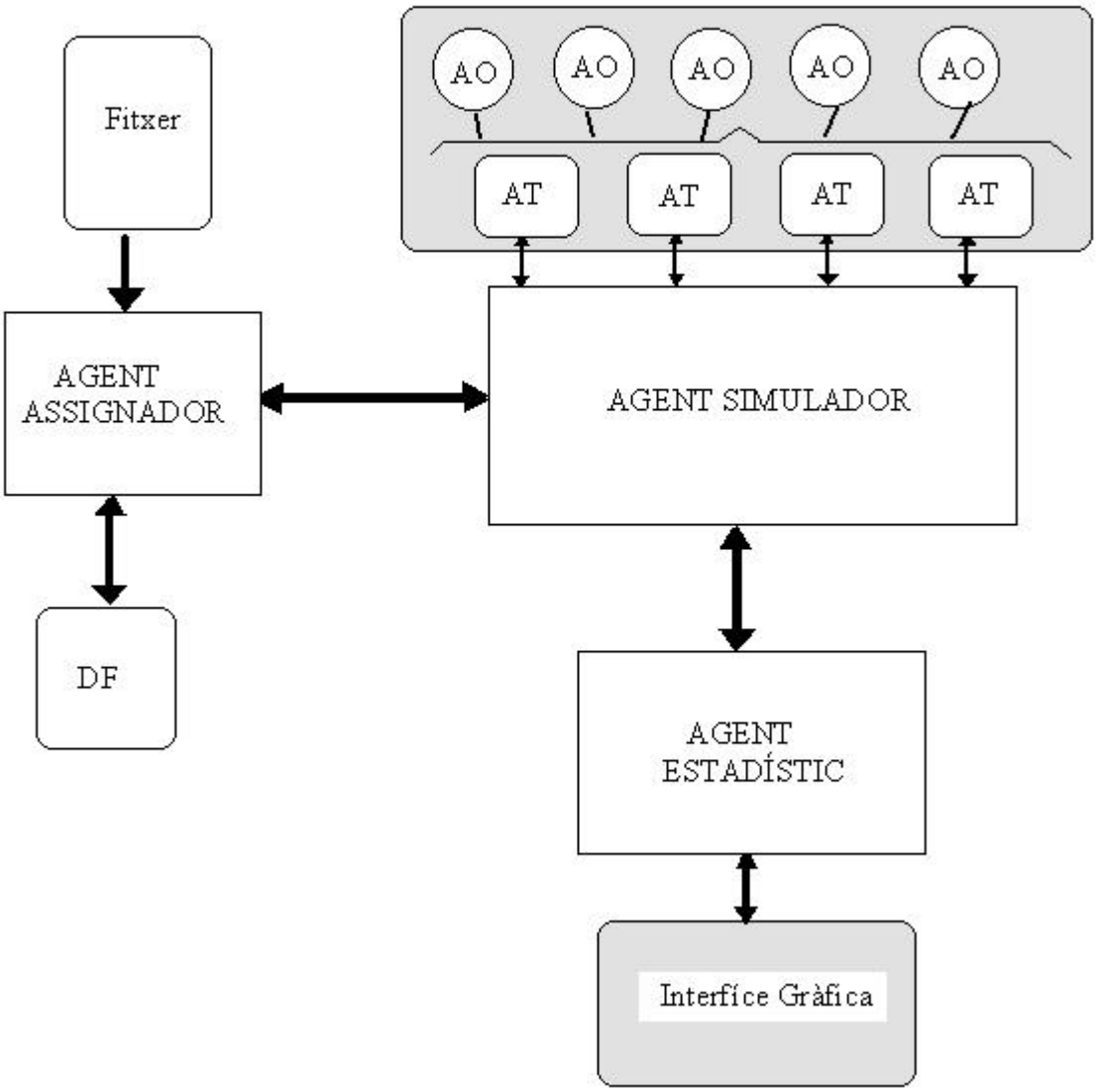

Figura 4.1 Arquitectura del SMA Simulador de Grups de Treball

En la Figura 4.1 podem apreciar la situació de les interfícies gràfiques. Una es troba situada amb contacte amb l'agent estadístic. Les altres són les corresponents als agents treballadors i als obrers. Aquest últims, tant els obrers com els treballadors, ni que en l'esquema no es pugui apreciar, disposen cadascú d'una interfície gràfica propia. La interfície apareix representada en gris.

A tot el SMA, degut a que hem utilitzat el JADE, hem utilitzat les especificacions dictades per la FIPA.

4.5 Funcionamet del procés de simulació

Ara ja hem vis com està estructurat el nostre sistema. Un Assignador que llegeix d'un fitxer les tasques, crea les assignacios i les envia al Simulador per a que les executi. Aquest, va informant als corresponents treballadors del que ténen que fer i va esperant els resultats. Aquests, de mentres, si volen, poden demanar requerir l'ajuda d'un Agent Obrer. Tant els treballadors com els Obrers recolleixen la informació sobre el seu comportament i grau d'especialització quan entren al sistema. Cada cop que s'acaba una simulació l'agent Simulador envia cap a l'Estadístic la informació que ha recol.lectat per a que aquest l'analitizi i pugui ser posteriorment visualitzada per l'usuari.

Per tal de poder seguir fàcilment aquest procés, mentre anem explicant-lo, mostrarem com es faria amb un exemple.

Abans de fer res, haurem d'omplir el fitxer tasques.txt amb el contingut adient. Aquest s'explicarà amb detall en la següent secció.
Quan engeguem el sistema ens apareixeran un seguit de finestres que nosaltres haurem d'anar omplint amb les dades corresponents. Ens trobarem amb tres tipus de finestres, les dels agents treballadors, les dels agents obrers i la de l'agent assignador. La finestra d'un Obrer (Figura 4.2), presenta quatre camps. En el primer hi haurem de ficar el nom del treballador (Recordem que aquest ha de ser únic en el sistema ja que s'utilitza com a identificador). També ens trobem un camp de Tipus ; és en aquest punt en el que hem d'especificar de quin tipus d'agent volem que es tracti, d'un ADP, un AEQ o un AT. A continuació ens trobem amb el camp en el que s'indicarà el grau d'Experiència d'aquest treballador (grauE), aquest pot ser o bé Alt, Mitja o Baix. Per acabar sols ens queda omplir el camp de sociabilitat. En aquest hem de posar Obert si volem que sigui sociable; si desitgem el contrari haurem de posar que es tracta d'un Agent Tancat.

Figura 4.2 Interfície Agent Treballador

En la finestra d'un obrer (Figura 4.3) solament hi trobarem dos camps a omplir. El nom, que realitza la mateixa funció que en el cas anterior; i el grau d'Experiència.

Figura 4.3 Interfície Agent Obrer

Finalment, i quan ja hagem omplert tots els camps de tots els agents i haguem confirmat la informació pitjant el botó d'Ok, serà quan ens tocarà pitjar el botó que apareix en la finestra de l'Assignador (Figura 4.4).

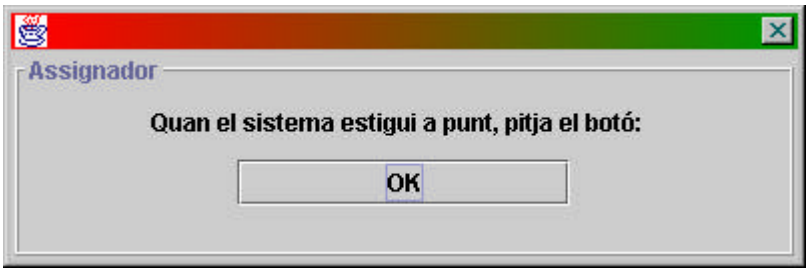

Figura 4.4 Agent Assignador

En el nostre cas hem entrat set Agents treballadors i dos Obrers.

4.5.1 Assignació de treballadors a les tasques

La feina de l'assignació de tasques als agents li correspon a l'agent Assignador. Aquest agent, a l'iniciar-se, realitza una lectura del fitxer tasques.txt d'on extreu la informació del deadline, el número d'assignacions a fer, el número de simulacions a fer, el número de tasques i les tasques en sí, cada tasca conté el seu tipus (aquest indicarà quin tipus d'Agent requereix : 1) ADP (Agent Director de Projecte); 2) AEQ(Agent EnginyerQuímic); 3)AT (Agent Tècnic)), la duració aproximada i els seus successors. L'identificador de la tasca vindrà donat per la seva situació en el fitxer. El format del fitxer és el següent:

```
deadline
numero de vegades a simular cada assignació
numero de vegades a assignar
```

```
numero d'agents a canviar per assignacio
numero de tasques
tipus_tasca   duracio_aprox successor1 successor2
tipus_tasca   duracio_aprox successor1 successor2
... (Així tantes tasques com hi hagin)
```

En el nostre cas, podem suposar que el graf de tasques utilitzat és el de la figura 1.1 (Primer capítol). Suposem, llavors, que volem realitzar 10 assignacions diferents i que per cada assignació volem canviar dos treballadors i volem realitzar quatre simulacions. El deadline volem que sigui de 48. Llavors, el fitxer ens quedaria de la següent manera:
.

48
4
10
2
7
1 10 2 3 ← En aquest punt comencen les tasques
2 6 4 5
2 9 6
2 5 7
2 8 6
1 7 7
3 12

Un cop llegit el fitxer realitzarà els càlculs necessàris fins a obtenir els antecessors de les tasques (informació necessària per al simulador) i les tasques paral·leles. Això últim es necessita per tal de que noed destini el mateix treballador a dues tasques que es podrien realitzar en el mateix instant de temps.

A l'hora de les assignacions, degut a que no seria escalable generar totes les assignacions possibles, es realitza de la següent manera. Primer es fa una assignació amb els agents de més grau, sempre i que això sigui possible pels paral·lelismes. Hem de tenir en compte que una tasca destinada a un AT pot ser realitzada per un AEQ o un ADP (Amb l'increment del cost corresponent). Una tasca destinada a un AEQ pot ser realitzada per un ADP, però no per un AT. Després de realitzar la primera assignació n'anirà creant d'altres. Per a fer-ho, per cada assignació cambiarà un nombre determinat d'agents (indicat per l'usuari en el fitxer de tasques). Però aquest cop els canviarà per agents del tipus que correspongui a aquesta tasca.

4.5.2 Realització d'una Simulació

Un cop s'ha realitzat una assignació, el resultat d'aquesta s'enviarà cap a l'agent Simulador. La funció d'aquest agent, com ja hem esmentat anteriorment serà la d'anar informant als agents corresponents la tasca que els hi toca realitzar. Guardar els resultats i enviar-los a l'agent Estadístic.

Quan rep una assignació per part del Assignador, el primer que es fa és guardar les dades corresponents i iniciar tot seguit el procés de simulació. Recordem que per cada assignació haurem de realitzar diverses simulacions. Per cada simulació el procés a realitzar serà el següent:

- En primer lloc es passarà a comprovar quines són les tasques que no ténen antecessores que necessitn que es realizin abans que elles. Normalment, en els grafs, hi soldrà haver una tasca inicial. Al llarg del procés de simulació s'anira comprovant constantment una taula on hi emmagatzemarem les tasques que han acabat i el moment en el que ho han fet. En quan es detecti que totes les antecessores d'una tasca han acabat, aquesta s'envia a executar. Per exemple, en el nostre cas, per a poder executar la tasca 2 necessitarem que primer acabi la tasca 1.

- Es produiran simulacions d'una mateixa assiganció el número de cops que s'hagi demanat

- Al llarg d'una Assignació, l'agent Simulador anirà informat als treballadors del que han de fer i esperarà les seves respostes. En aquest procés és quan el treballador realitza els calculs de si acabarà o no la tasca que se li ha encomanat. En el cas que no la pogués finalitzar ens trobariem amb una fallada (o fallida). De fallades n'hi poden haver de dos tipus, la que acabem d'esmentar i la produida per que s'ha excedit el deadline. Si l'agent veu que ho pot fer, llavors mira si vol requerir els serveis d'un Agent Obrer per a que l'ajudi a realitzar la tasca. En cas que no en vulgui cap farà el càlcul de temps del que trigarà i li passarà la informació al simulador. En el cas que volgués tenir un ajudant haurà d'escollir-ne un d'entre tots els que hi han al sistema. Per a la negociació d'aquesta col·laboració s'empra el Protocol descrit per la FIPA Contract Net.

- Cada cop que es finalitza una simulació, els resultats daquesta s'envien a l'Agent Estadísitc.

4.5.3 Estadístiques

L'agent Estadístic és l'encarregat d'analitzar totes les dades que es generen durant el procés de Simulació i mostrar-les a l'usuari através de la interfície gràfica que poseeix.

En primer lloc podriem diferenciar la informació que obtenim de l'assignador en **gràfiques** i **consultes**. En l'apartat de les gràfiques podrem veure quina ha estat la evolució dels costos i del temps de duració al llarg de totes les assignacions. També les podrem comparar mitjançant una gràfica en la que hi surten les dues components. Una visió de com és el simulador és la de la figura 4.5.

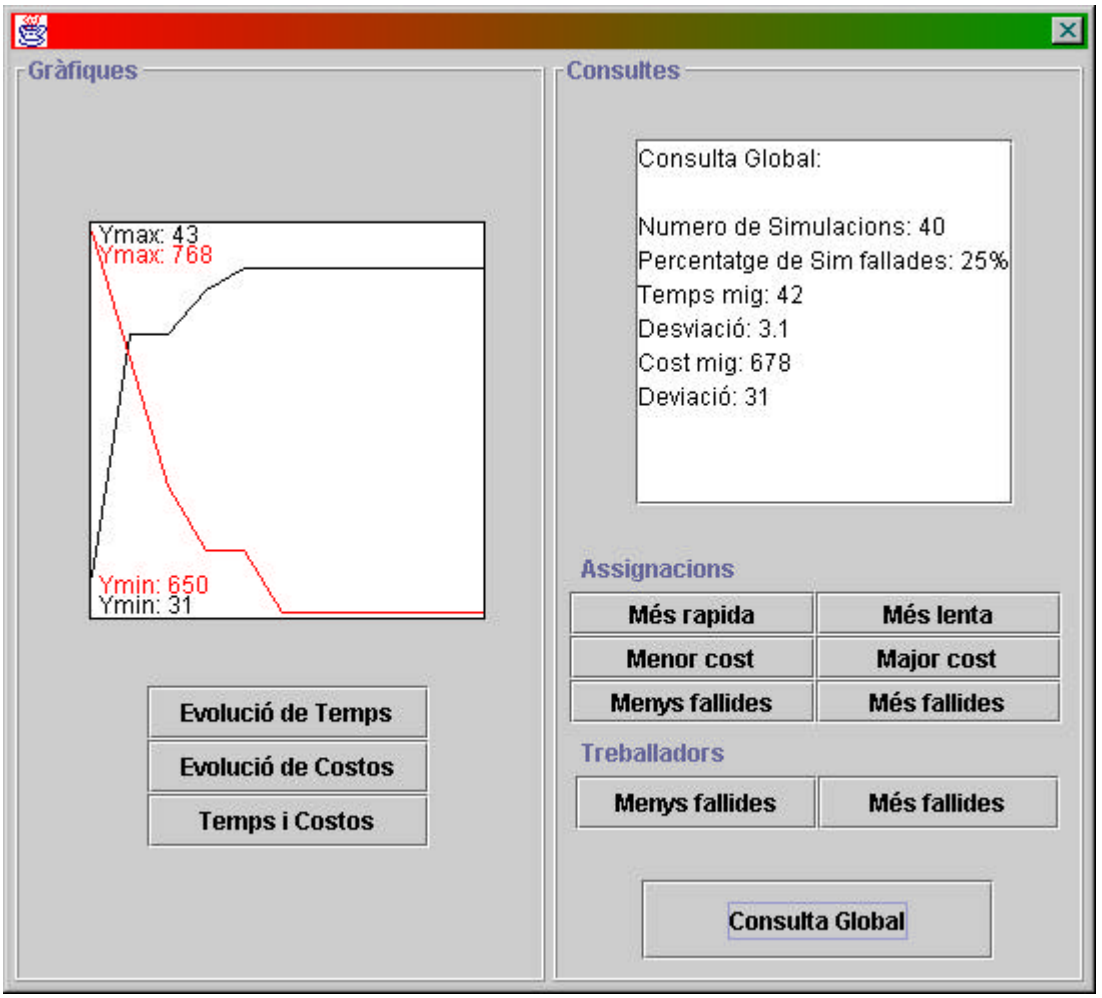

Figura 4.5 Visió general de l'agent Estadísitc

En l'apartat de consultes, tenim nou possibles consultes, una de Global, sis que fan referència a les assignacions i un parell dedicades als treballadors.

Consulta Global: En aquesta hi podem veure el nombre de simulacions realitzat, el porcentatge de simulacions fallides, el temps mig , el cost mig i les dues desviacions d'aquests. En el nostre cas, el resultat que hem obtingut ha estat el següent:

Consulta Global:

Numero de Simulacions: 40
Percentatge de Sim fallades: 25%
Temps mig: 42
Desviació: 3.1
Cost mig: 678
Deviació: 31

Podem apreciar que en aquest cas concret el nombre de fallides que hem tiungut és molt elevat, d'un quinze per cent. En canvi, el temps mig és bastant baix (Hem de tenir en compte que en aquest cas, si no es reduïa cap valor, la finalització d'aquest cas tardava 45 unitats de temps).

- Consultes de treballadors: Solament en disposem de dues, el treballador/s amb més fallides i els treballadors amb menys fallides. En el nostre cas particular, els treballadors amb menys fallades han estat:

  Numero de fallades: 0
  Nom: Josep
  Nom: Enric
  Nom: Tomas

Podem comprovar que el nombre de fallades que han tingut és el més baix que haurien pogut tenir.

- Les consultes sobre les assignacions en són sis. La més lenta, la més ràpida, la més cara, la més barata, la que ha tingut més fallides i la que ha tingut menys fallides. La estructura que mostren totes elles és la mateixa. Per a compendre-ho millor anem a veure quina ha estat la simulació més ràpida en el nostre cas:

  Generals:
  Deadline:60

  Assignació:

  Cost: 770
  Temps mig: 34
  Numero Fallades: 1 de 4
  Tasca nº: 1
  Agent: Clara ADP
  Ajudants: 0 de 4

  Tasca nº: 2
  Agent: Clara ADP
  Ajudants: 2 de 4

  Tasca nº: 3
  Agent: Josep ADP
  Ajudants: 0 de 4

  Tasca nº: 4
  Agent: Enric ADP
  Ajudants: 0 de 4

Tasca nº: 5
Agent: Clara ADP
Ajudants: 0 de 4

Tasca nº: 6
Agent: Enric ADP
Ajudants: 0 de 4

Tasca nº: 7
Agent: Enric ADP
Ajudants: 1 de 4

El primer camp que se'ns exposa és el del deadline, seguit per el cost particular d'aquesta assiganció, el temps mig d'aquesta i el nombre de fallides total (sumem totes les simulacions amb fallides). A continuació ja vénene els agents amb les Tasques que ténen assignades. Per cada agent ens surt el nivell professional al que pertanyen i en quantes simulacions han tingut ajudants. Per exemple, l'Enric ha tingut un ajudant en una de les quatre simulacions per a la tasca 7.

En el referent al tema gràfic. Com ja hem esmentat abans, podem representar tres gràfiques, dues de simples i una de composta. La composta (temps i cost) seria la que es pot veure en la figura 4.5. En ella tenim la component del cost de color vermell i el temps de color negre. Es pot apreciar la relació que existeix entre el cost i l'eficiència, a més cost, més eficiència, a menys cost, menys eficiència.

4.5.4 Protocols

Ara explicarem quins protocols s'han usat per a dur a terme la comunicació entre els agents. En primer lloc, un dels que queda perfectament definit és el que existeix entre els treballadors i els obrers. Aquest haurà de ser un FIPA ContractNet en el que l'iniciador serà el treballador i els receivers els agents obrers. En els altres casos hem implementat el FIPA Request. L'assignador actuarà com un initiator del FIPA Request, en aquest cas el receiver serà l'agent Simulador.

L'agent Simulador, a banda de ser el receiver del Assignador, també serà l'initiator d'altres enllaços de comunicació. Implementarà FIPA Request per a comunicar-se amb els treballadors. També implementarà un FIPA Request per a comunicar-se amb l'Estadístic.

4.5.5 Ontologia

Un cop decidits els protocols a utilitzar en el nostre sistema, ens toc decidir el llenguatge que utilitzaran els agents per a comunicar-s entre ells. Hem definit varis frames, sis objectes i quatre predicats.

En la comunicació entre l'agent Assignador i l'estadístic usarem el predicat/funció simular_assig (Simular Assignació). Aquest contindrà un llistat de totes les tasques a realitzar i un altre llistat per als agents assignats. També contindrà el camps deadline, número de cops a assignar i número de cops a simular l'assignació. El camp de les tasques, el nombre d'assignacions i el nombre de simulacions només estaran inclosos en el primer missatge donat a que seran els mateixos durant tota la simulació.

En la comunicació entre l'Assignador i l'Estadísitc, els missatges enviats per el primer contindran un incloure_a_est (Incloure a Estadística). Aquests estaran composats per una assignació i els resultats de la simulació.

En el cas de l'Assignador als Treballadors, el primer enviarà un missatge contenent el predicat tasca_enviar. Dintre hi haurà la informació referent a la tasca que ha de realitzar així com l'instant de temps en que es troba a l'iniciar la tasca (També hi haurà l'assignació en la que ens trobem i el número de simulació de l'assignador, aquests camps, però, només seran útils per a l'obrer). Un cop l'hagi feta, en el missatge d'inform el treballador inclourà un predicat (temps) en el que s'ind icarà el temps que ha trigat a realitzar la tasca i un altre slot en el que informarà de si ha tingut ajudant o no.

Al comunicar-se amb els obrers, els missatges que s'enviaran seran els mateixos, serà en aquest cas en el que el número d'assignació i el número de simulació de l'assignació adquidiran sentit.

4. 6 Conclusions i Futurs treballs.

Hem pogut apreciar que la implementació d'aquest Sistema de Simulació no ha estat pas senzilla, per relativament poques dades hem hagut de realitzar molta feina. Tot i això, s'ha pogut comprovar la seva eficiència a l'hora de simular.

Amb les dades que manegem en aquest cas tampoc es poden fer meravelles, però ja hem pogut observar-hi certes pautes de comportament bastant lògiques.

Amb una ampliació de les característiques psicològiques i d'habilitat més dels agents es podrien aconseguir resultats veritablement sorprenents. I en certa manera aquest és le propòsit d'un futur projecta que aiat podria veure la llum.

Aquest Sistema que hem implementat és la base d'un futur projecte molt més ambiciós en el que no solament s'utilitzaran comportaments molt més complexos, sinó que s'intentarà ampliar les possibilitats dels sistema per a que realitzi d'altres tasques igualment interessants tal i com poden ser la organització i creació del graf de tasques.

Esperem que aquest projecte serveixi per a ajudar una mica més a la incorporació de la utilitat de la IA en el món quotidià.