CAPITOLUL IV

SISTEME MULTIAGENT ŞI SOCIETĂŢI DE AGENŢI

4.1 Inteligenţa Artificială Distribuită (Distributed Artificial Intelligence)

Inteligenţa Artificială Distribuită **(IAD)** se ocupă cu studiul, dezvoltarea şi aplicarea sistemelor multiagent. Inteligenţa Artificială Distribuită este un nou model de dezvoltare şi cercetare care îmbină mai multe domenii precum: inteligenţa artificială tradiţională, informatica, sociologia, psihologia. **Sistemele cu inteligenţă artificială distribuită** (pe care le vom numi în continuare **sisteme cu IAD**) pot fi definite ca şi sisteme cooperative, unde un grup de agenţi lucrează împreună pentru pentru a rezolva o problemă dată.

Se pune întrebarea de ce este mai bine să împărţim rezolvarea unei probleme pe mai multe sisteme de calcul, din moment ce ea poate fi rezolvată, de cele mai multe ori la fel de eficient sau mai eficient şi pe un singur calculator? Răspunsul la această întrebare este unul foarte natural:

- în primul rând agenţii nu sunt creaţi pentru a acţiona izolat, ci de cele mai multe ori aceştia vor interacţiona în cel mai fericit dintre cazuri doar cu alţi agenţi, dacă nu chiar şi cu oameni;
- un alt motiv este faptul că există probleme de aşa natură încât doar o abordare distribuită poate duce la o rezolvare eficientă a acestora; acestea sunt probleme distribuite. Topologia sistemelor în care apar astfel de probleme complexe este una extrem de dinamică, de aceea este greu ca un singur agent să îndeplinească toate cerinţele utilizând date corecte şi în acelaşi timp consistenţa datelor furnizate de acesta să nu aibă de suferit;
- un ultim motiv ar fi acela că la rezolvarea unei proleme, în societatea umană, de obicei, se adună mai mulţi specialişti din domeniile ce ţin de problema în discuţie, astfel pentru rezolvarea unei probleme am putea utiliza mai mulţi agenţi specializaţi pe domeniile cerute de aceasta. Am putea desigur construi un agent specialist în mai multe domenii, dar acesta ar putea fi folosit doar în rezolvarea unui număr mic de probleme, astfel încât să fie folosit la capacitate maximă, în restul prolemelor doar o mică parte din cunoştinţele şi abilităţile sale ar fi folosite. Mai putem avea în vedere şi faptul că o variantă distribuită este de cele mai multe ori mai uşor de înţeles şi de implementat, şi nu putem nega faptul că astfel de abordări pot duce la desoperirea unor algoritmi foarte utili, care nu ar fi descoperiţi dacă rezolvarea problemelor, mai ales a celor distribuite, ar avea o abordare centralizată.

Există patru mari tehnici de abordare a problemelor de dimensiuni şi complexitate considerabile: modularizarea, distribuirea, abstractizarea şi inteligenţa, unde inteligenţa se referă la modul în care sunt căutate şi modificate informaţii. Utilizarea unor module distribuite şi inteligente combină toate aceste patru tehnici conturând o abordare de tip

IAD. Conform unei astfel de abordări agenţii trebuie să fie distribuiţi prin tot sistemul, unde aceştia pot acţiona din diferite ipostaze precum: aplicaţii–program inteligente, resurse active de informaţie sau servicii on–line în reţea. Agenţii au cunoştinţe locale despre resurse şi cooperează pentru a asigura acces global la ele şi o bună utilizare a acestora. Din motive practice (sistemele sunt mari şi dinamice), pentru obţinerea soluţiilor globale, agenţii trebuie să acţioneze autonom şi să fie dezvoltaţi independent.

Raţiunea pentru interconectarea agenţilor cu sistemele expert este pregătirea lor pentru a coopera în rezolvarea de probleme, pentru a împărtăşi expertiza (în cazul sistemelor expert, să pună la dispoziţia agenţilor expertiza lor), pentru a lucra în paralel la probleme comune, pentru a reprezenta multiple puncte de vedere şi cunoştinţe ale mai multor experţi şi pentru a fi reutilizaţi în rezolvarea altor probleme. Posibilitatea ca un agent să interacţioneze cu alţi agenţi îl face pe constructorul acestuia să privească proiectarea şi construcţia agentului dintr–un alt punct de vedere. De aceea proiectantul unui agent se gândeşte la ceea ce ştie agentul său şi, de asemenea, la modalităţi prin care alţi agenţi pot accesa cunoştinţele acestui agent. Această perspectivă poate duce la o nouă reprezentare a cunoştinţelor agentului. Acestea vor fi reprezentate declarativ dând altor agenţi posibilitatea să le acceseze, în loc să fie ascunse în codul agentului.

Sistemele multiagent sunt cea mai bună cale de proiectare şi caracterizare a sistemelor de calcul distribuite. Lucru dovedit în momentul în care poblema de rezolvat este distribuiă geografic, de exemplu, sau din motive de confidenţialitate, pe motive concurenţiale, mai multe organizaţii nu vor ca datele să fie centralizate şi când, metoda de rezolvare a problemei folosind un singur agent este, în mod evident, depăşită.

Procesarea informaţiei este omniprezentă. Există sisteme de procesare a datelor în toate aspectele vieţii umane. Se găsesc, spre exemplu, chiar şi în bucătăriile oamenilor, şi nu sunt puţine. De exemplu în cuptorul cu microunde, în prăjitorul de pâine, în filtrul de cafea şi există încă o mulţime de electrocasnice pe care nu le–am menţionat, şi nu am vorbit încă de sistemul electric, care foloseşte probabil o mulţime de astfel de procesoare de informaţie pentru a aduce curentul electric în bucătărie (casă). Numărul mare de procesoare de informaţie şi multitudinea de modalităţi în care acestea interacţionează fac din sistemele computaţionale distribuite paradigma dominantă a zilelor noastre.

În momentul în care electrocasnicele din bucătărie sunt destul de inteligente pentru a fi considerate agenţi inteligenţi, se poate vorbi despre ele în termeni antropomorfici. De exemplu: vom putea spune *„prăjitorul de pâine **ştie** când pâinea este prăjită”*.

Scopul IAD este să dezvolte anumite metode, sau mecanisme, care să le permită agenţilor să interacţioneze cu alţi agenţi, sau cu oameni, în acelaşi mod în care interacţionează omenii între ei. Probleme centrale in IAD sunt următoarele:

- cum să interacţioneze agenţii?
- când să interacţioneze agenţii?
- cu cine să interacţioneze fiecare dintre agenţi?

Aceste probleme apar în IAD şi pentru a găsi soluţii, trebuie să vedem de ce ele nu au fost semnalate şi în inteligenţa artificială tradiţională. Atât inteligenţa artificială tradiţională cât şi cea distribuită studiază aspectele computaţionale ale inteligenţei, dar, din perspective diferite. În inteligenţa artificială tradiţională agenţii sunt priviţi ca „sisteme inteligente individuale”, inteligenţa este privită ca o proprietate a sistemelor izolate, formate dintr–un singur individ, şi procesele cognitive (de cunoaştere) sunt considerate doar în cadrul unui singur individ. Spre deosebire de inteligenţa artificială

tradiţională, în IAD agenţii sunt studiaţi ca făcând parte din „sisteme inteligente interconectate”, inteligenţa este considerată a fi o proprietate a sistemelor în care agenţii interacţionează şi procesele cognitive sunt considerate în cadrul grupurilor de indivizi, (între aceştia).

Sistemele cu IAD sunt bazate pe diferite tehnologii cum ar fi: sistemele expert distribuite, planificarea sistemelor sau sistemele tip *tablă*. Noutatea în comunitatea IAD este necesitatea unei metodologii pentru dezvoltarea şi întreţinerea sistemelor cu IAD. În acest scop se folosesc formalisme abstracte pentru reprezentarea proprietăţilor de bază ale IAD.

Modelarea conceptuală a IAD

Deoarece IAD implică interacţiuni între comunităţile de agenţi, există câteva preocupări principale în modelarea conceptuală a IAD. Vom introduce aceste concepte făcând referire la câteva principii ale sistemelor din IAD:

- Actori multipli.

  În general, sistemele IAD cuprind mai mulţi agenţi care acţionează concurent şi asincron.
- Conflictul între cunoştinţele şi acţiunile pragmatice, fixate şi cele la distanţă.

  Toate acţiunile (şi cunoştinţele) în sistemele cu IAD sunt bine plasate în timp şi spaţiu. Dar trebuie să luăm în considerare şi posibilitatea ca acţiunile şi cunoştinţele din diferite momente sau locuri să fie modificate (în trecut sau viitor).
- Perspective multiple.

  Întrucât toate acţiunile şi cunoştinţele sunt locale, agenţii individuali şi colectivi pot să vadă lumea (mediul) din perspective diferite, şi aceste perspective pot fi sau nu în conflict. De exemplu, agenţi diferiţi pot folosi modele conceptuale diferite, dar totuşi să aibă nevoie să interacţioneze.
- Limitarea resurselor.

  Agenţii în sistemele cu IAD împart resurse limitate; resursele şi alocarea resurselor furnizează un mecansim important de interacţiune între agenţi.

Angajamentul. Modele individualiste şi sociale

Angajamentul este important deoarece reprezintă o metodă prin care agenţii îşi stabilesc continuitatea acţiunilor lor. Numeroşi cercetători studiază angajamentul ca fiind la baza multor concepte de IA şi IAD cum ar fi: scopurile, negocierea, faptele şi cunoştinţele. Evident noţiunile ca *fapt*, *cunoştinţă*, *intenţie* şi *scop* presupun existenţa de-a lungul timpului a unor angajamente pentru a avea continuitate în identitatea şi utilitatea unui fapt, unei stări a cunoştinţelor sau unui agent.

Conceptul de angajament poate fi definit ca şi atribut al unui agent individualist sau ca şi rezultatul interacţiunilor între comportamentul mai multor agenţi diferiţi. Deci angajamentul poate fi înţeles din punct de vedere individual sau din punct de vedere social.

În cadrul modelului individualist, *angajamentul* este un fel de alegere raţională făcută de un actor individual. Angajamentul este strâns legat de *intenţie* şi se bazează pe concepte cum ar fi *opinii* (beliefs) şi *scopuri* (goals).

În cazul modelelor sociale, angajamentul este privit ca organizarea participării unui agent în mai multe cadre simultan.

Organizarea. Două modele conceptuale opuse

Organizarea este un alt aspect fundamental în sistemele cu IAD. Noţiunii de organizare i s-a acordat atenţie în literatura IAD şi au fost propuse un număr mare de modele conceptuale particulare de organizare cum ar fi:

- Structurile ierarhice sau centralizate de agenţi, care interacţionează prin intermediul canalelor de comunicaţie.
- Comunităţile pluraliste de agenţi cu reguli de comportament interpretabile local.

Modele individuale de organizare. Toate modelele conceptuale de organizare au la bază arhitecturile agenţilor individuali şi relaţiile dintre ei şi tratează organizarea ca şi rezultatul alegerilor individuale (de acţiuni).

Arhitecturile agenţilor individuali în acest model conceptual nu pot fi schimbate de agenţi sau de comunitate. Agentul are propriile sale cunoştinţe, opinii şi scopuri şi ia decizii locale. Ca urmare, cunoştinţele, acţiunile şi deciziile organizaţiei pot fi reduse la cunoştinţe, acţiuni şi decizii locale agregate. Cu cât sunt mai bune regulile de decizie sau cunoştinţele agenţilor cu atât este mai mare performanţa organizaţiei.

Modele sociale de organizare. În modelele sociale de IAD, elementele principale sunt interacţiunile între participanţi. Agenţii sunt definiţi prin interacţiunile la care participă, nu invers. Relaţiile dintre părţi sunt angajamente. Angajamentele pot fi materializate ca şi reprezentări.

Un prim pas în realizarea modelelor conceptuale sociale de IAD este dezvoltarea interacţiunilor între agenţi şi tratarea agenţilor ca entităţi distribuite care participă împreună la realizarea unui proces. Un alt pas important ar fi reconceperea arhitecturilor agenţilor individuali în aşa fel încât agenţii să rezulte ca urmare a unor procese continue ce au loc în sistemele cu IAD – acest lucru înseamnă că agenţii vor fi entităţi flexibile implicate în mai multe acţiuni simultan şi posibil cu implementări distribuite.

Cele mai importante probleme de IA, pentru care au fost construite modele de IAD, sunt descompuse recursiv în subprobleme, şi apoi pot fi aplicate operaţii primitive pentru a rezolva subproblemele. Din perspectivă socială, agenţii vor fi compuşi şi descompuşi în funcţie de problemă. În rezolvarea unei probleme complexe, dificultatea constă în construirea unui agent potrivit (prin compunere sau descompunere) şi în găsirea unei definiţii potrivite a problemei prin structurarea mulţimii de interacţiuni şi angajamente.

Tehnici de modelare conceptuală în IAD

Există o varietate de instrumente de modelare conceptuală disponibile în IAD, multe dintre ele fiind destinate problemelor specifice ale sistemelor cu IAD. Acestea includ:

- Limbaje de programare în IAD. Diferite limbaje de programare sunt acum utilizate pentru descrierea la nivel conceptual a sistemelor cu IAD. O trăsătură comună a tuturor acestor limbaje este folosirea agenţilor. Printre limbajele folosite se numără LISP, PROLOG şi limbaje specializate cum ar fi limbaje de programare concurentă bazată pe obiecte (OBCP), şi, în special limbajul OBCP care include reflecţia.
- Instrumente de reprezentare a cunoştinţelor în IAD. Din această categorie fac parte cadrele, obiectele, predicatele logice.
- Arhitecturile IAD. Dintre arhitecturile IAD amintim sistemele tip *tablă* (blackboard), sistemele *tablă* distribuite şi arhitecturile IAD cu structuri de control flexibil.

Ca o concluzie a celor prezentate în acest capitol, cercetătorii IAD ar trebui să se concentreze asupra modelelor conceptuale care sunt descentralizate la toate nivelele şi al căror nivel de analiză este social, nu individual. Acest lucru înseamnă că elementele de bază ale organizării sunt schimbate de la fapte sau afirmaţii individuale la angajamente, interacţiuni şi acţiuni reunite între participanţii la un sistem.

Aceste observaţii au mai multe implicaţii asupra arhitecturilor IAD. În primul rând, ele furnizează baza teoriilor sistemelor cu IAD care sunt distribuite şi ne permit să analizăm şi să extindem aceste sisteme. În prezent nu avem o astfel de teorie socială pentru IAD – există mecanisme şi arhitecturi ad-hoc, sau bazate pe modelele conceptuale individualiste.

4.2 Sisteme multiagent şi societăţi de agenţi

Agenţii acţionează în medii care pot fi fizice sau computaţionale (virtuale), închise sau deschise, în care acţionează doar un singur agent sau mai mulţi.

Deşi există situaţii în care un agent poate opera mai eficient de unul singur, creşterea legăturilor şi a numărului reţelelor de calculatoare face ca asemenea situaţii să fie rare, şi de obicei agenţii să interacţioneze unul cu celălalt. Uneori, numărul agenţilor este prea mare pentru a-i putea trata individual, şi atunci este mult mai convenabil să îi tratăm ca pe un colectiv, o societate de agenţi.

Această secţiune prezintă modul de analiză, descriere şi proiectare a mediilor în care agenţii îşi desfăşoară activitatea eficient şi interacţionează unul cu celălalt.

După cum am definit deja în secţiunea 2.7, un **sistem multiagent** este un sistem format din mai mulţi agenţi, care interacţionează între ei. În cel mai general caz, agenţii vor acţiona în sprijinul utilizatorilor având diferite scopuri şi motivaţii. Pentru a

interacţiona cu succes, agenţii au abilitatea de a *coopera*, a se *coordona* şi a *negocia* între ei, aşa cum şi oamenii o fac.

În sistemele multiagent mediul asigură o infrastuctură computaţională pentru interacţiuni între agenţi prin:

- protocoale pentru ca agenţii să comunice- să schimbe mesaje între ei şi să le înţeleagă;
- protocoale pentru ca agenţii să interacţioneze - prin conversaţii (schimburi structurate de mesaje).

Spre exemplu, un protocol de comunicare poate specifica faptul că următoarele tipuri de mesaje pot fi schimbate între doi agenţi:

- propunerea unui plan de acţiune;
- acceptarea unui plan de acţiune;
- refuzarea unui plan de acţiune;
- retragerea unui plan de acţiune;
- dezacordul cu un plan de acţiune;
- propunerea unui alt plan de acţiune.

Vom în continuare un exemplu concret de ***protocol de comunicare.*** Pe baza mesajelor existente, între doi agenţi *Ag1* şi *Ag2* poate avea loc următoarea ***conversaţie*** – o instanţă a unui ***protocol de interacţiune*** pentru negociere:

- *Ag1* propune o acţiune lui *Ag2*;
- *Ag2* evaluează propunerea şi
  - trimite un mesaj de acceptare *sau*
  - trimite un mesaj ce conţine o propunere a unei alte acţiuni *sau*
  - trimite un mesaj de respingere a acţiuni *sau*
  - trimite un mesaj prin care îşi exprimă dezacordul cu acţiunea propusă.

Dar de ce ar trebui să fim interesaţi de distribuţia sistemelor de agenţi? Într-adevăr, soluţiile centralizate sunt, în general, mult mai eficiente: orice se calculează într-un sistem distribuit poate fi mutat pe un singur calculator şi optimizat pentru a fi cel puţin eficient. Totuşi, calculele distribuite sunt uneori mai uşor de înţeles şi de dezvoltat, în special când problema de rezolvat este ea însăşi distribuită.Există situaţii în care metoda centralizării nu poate fi aplicată, deoarece sistemul şi datele aparţin unei organizaţii independente care doreşte să păstreze confidenţialitatea lor pe motive concurenţiale. Sistemele multiagent sunt cele mai bune mecanisme de caracterizare şi proiectare a sistemelor de calcul distribuite.

Între agenţii dintr–un sistem multiagent protocoale de interacţiune frecvent utilizate sunt protocoalele de coordonare şi cele de cooperare.

Caracteristicile unui sistem multiagent sunt:

- mediile multiagent asigură o infrastructură specificând protocoalele de comunicare şi intercaţiune;
- medile multiagent sunt de obicei deschise şi nu au un proiectant central;
- mediile multiagent conţin agenţi care sunt autonomi şi distribuiţi care pot *necooperativi* (*individualişti* - pot acţiona în propriul lor scop) sau pot fi *cooperativi*. Agenţii dintr-un sistem multiagent comunică de obicei pentru a-şi atinge propriile scopuri sau obiectivul întregului sistem. Comunicarea între agenţi le permite sa-şi coordoneze comportamentul şi activitatea.

Majoritatea sistemelor din lumea reală sunt sisteme distribuite şi de acea sistemele multiagent se potrivesc perfect cu aceste medii distribuite.Câteva exemple de domenii unde sunt folosite cu succes sitemele multiagent: controlul traficului aerian şi naval, sistemele multi-robot.

Sistemele multi-agent sunt:

- **modulare** – agenţii pot fi adaugaţi şi eliminaţi fară mari modificări ale sistemului;
- **adaptive** –agenţii se adapteză la modificările sistemului (schimbarea mediului sau a scopului);
- **concurente** – agenţii acţionează în paralel şi asincron (astfel că timpul necesar atingerii obiectivului scade);
- **dinamice** – agenţii colaborează pentru a forma grupuri dinamice în scopul rezolvării unei probleme specifice.

Sistemele multiagent existente pâna în prezent au un număr limitat de agenţi (câteva sute), dar se prevede ca în viitor sistemele să poată conţine mii sau chair milioane de agenţi. O astfel de schimbare poate duce însă la pierderea preciziei şi la creşterea complexităţii sistemelor.

Sistemele multi-agent au apărut pentru:

- problemele reale complexe existente;
- modelarea sistemelor deschise, distribuite;
- capacitatea limitată a unui singur agent de a rezolva probleme reale;
- reutilizarea şi integrarea in sistemele noi a vechilor pachete software;
- aplicaţii modelate cu ajutorul agenţilor;
- necesitatea utilizării resurselor şi expertizei distribuite în reţea;
- creşterea performanţei sistemelor:
    - viteza de calcul;
    - extensibilitate;
    - fiabilitate;
    - robusteţe;
    - flexibilitate;
    - întreţinere.

4.2.1 Transferul de informaţii între agenţi. Protocoale de comunicare

Într-un sistem multiagent , un agent este privit ca un ***obiect activ*** capabil să:

- perceapă mediul;
- raţioneze;
- acţioneze asupra mediului.

În cazul general, un agent are:

- o *cunoaştere iniţială despre mediu* şi *capacitatea de a face inferenţe* (deduce concluzii) pe baza acestor cunoştinţe;
- *capacitatea de a comunica* (schimba mesaje) care este :
    - o parte **percepţie**(capacitatea de a primi mesaje);
    - o parte **acţiune**(capacitatea de a trimite mesaje).

Capacitatea de comunicare a agenţilor dintr-un sistem multiagent creşte eficienţa sistemului. Să ne imaginăm, spre exemplu, mai mulţi roboţi, cu diferite capacităţi, colaborând pentru a găsi cea mai bună soluţie a unei probleme, fiecare având sarcina lui, şi anume, o subproblemă a problemei de rezolvat. Pentru a colaborara cu succes, roboţii, în primul rând, trebuie să fie capabili să efectueze un schimb de informaţii. Pentru ca acest schimb de informaţii să se efectueze cu succes au fost introduse protocoalele de comunicare.

4.2.2 Coordonare

Agenţii comunică de obicei pentru a-şi atinge propriile scopuri sau obiectivul societăţii/sistemului în care ei există. Comunicarea între agenţi le permite să-şi coordoneze comportamentul şi activitatea

Coordonarea poate fi definită ca fiind proprietatea sistemelor de agenţi care acţionează într-un mediu partajat. Coordonarea este necesară deoarece:

- există dependenţe între acţiunile agenţilor;
- este necesar să se respecte anumite constrîngeri globale;
- este necesară distribuirea competenţei, resurselor şi informaţiei pentru a atinge scopurile sistemului;
- informaţia şi resursele sunt distribuite în tot sistemul.

Cooperarea poate fi definită ca fiind coordonarea între agenţi non-antagonici. În general, pentru a coopera cu succes, fiecare agent trebuie să pastreze un model al celorlalţi agenţi, şi deasemenea să dezvolte un model al interacţiunilor viitoare. Acest aspect presupune abilitate socială.

Negocierea poate fi defnită ca fiind coordonarea între agenţi antagonici (care au scopuri antagonice - agenţii aflaţi în competiţie sau agenţi individualişti, cu scop propriu).

Figura 4.1 prezintăo clasificare a diferitelor modalităţi prin care agenţii îşi pot coordona comportamentul şi activităţile.

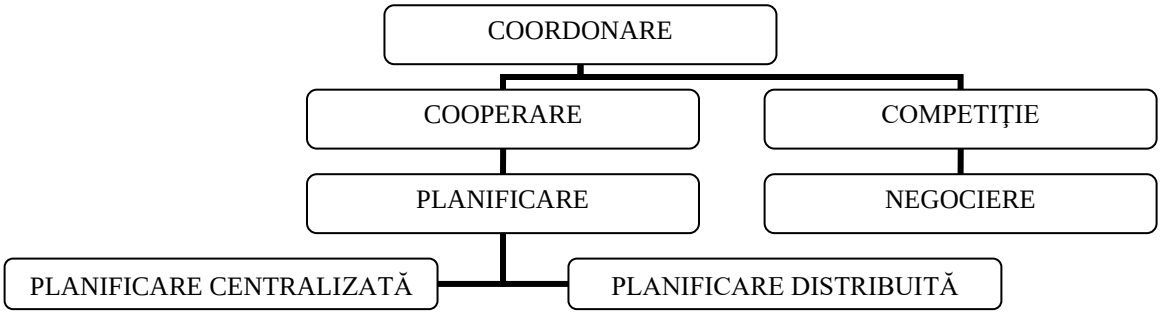

Figura 4.1. O taxonomie a unor modalităţi prin care agenţii îşi pot coordona comportamentul şi activităţile

4.2.3 Tipuri de mesaje

Este foarte important pentru agenţii cu diferite capacităţi să fie capabili să comunice. Comunicarea este definită la diferite nivele, începând cu comunicarea la cel mai jos nivel pentru agenţii mai puţin capabili. Pentru a fi interesaţi unul de celălalt, agenţii trebuie să participe la dialog, rolul lor în dialog putând să fie activ, pasiv sau ambele,

Un agent poate trimite sau primi informaţii (mesaje) prin intermediul reţelei de comunicaţie. Aceste mesaje pot fi de două tipuri: *aserţiuni* (informaţii de la o sursă externă) sau *interogări* (formularea de întrebări şi răspunsuri la întrebări). Fiecare agent trebuie să fie capabil să accepte informaţie, în cea mai simplă formă această informaţie îi este comunicată agentului dintr–o sursă externă sub forma unei afirmaţii (aserţiuni).

Pentru a avea un rol pasiv într-un dialog, agentul trebuie să:

- accepte o întrebare dintr-o sursă externă;
- să trimită răspuns către sursă sub forma unei aserţiuni.

Se observă că există diferenţe între afirmaţiile nesolicitate şi cele făcute pentru a răspunde unei întrebări.

Pentru a-şi asuma un rol activ în dialog, un agent trebuie să pună întrebări şi să facă afirmaţii. Astfel, un agent poate controla un alt agent, constângându-l să răspundă sau să accepte informaţia propusă.

Protocoalele de comunicare sunt specificate, în general, pe mai multe nivele. Nivelul inferior specifică metoda de interconectare, nivelul de mijloc specifică formatul sau sintaxa informaţiei care este transferată, iar la nivelul superior se specifică semnificaţia sau semantica informaţiei. Semantica nu se referă doar la substanţa (conţinutul) mesajului, ci şi la tipul acestuia.

Protocoalele de comunicare pot fi atât binare, cât şi n-are. Un protocol binar implică un singur expeditor şi un singur receptor, în timp ce un protocol n-ar implică un singur expeditor şi mai mulţi receptori (uneori numiţi *broadcast* sau *multicast*).

Un protocol este, în general, specificat de structuri de date cu următoarele câmpuri:

- expeditor;
- receptor(i);
- limbajul folosit în cadrul protocolului;
- funcţii de codificare şi decodificare;
- acţiuni efectuate de către receptor(i).

4.2.4 Limbaje de comunicare între agenţi (Agent Communication Language - ACL)

4.2.4.1 KQML

Un limbaj de comunicare şi coordonare între agenţi este limbajul KQML (Knowledge Query and Manipulation Language).

O decizie fundamentală pentru interacţiunea între agenţi este separarea semanticii unui protocol de comunicare (care trebuie să fie independent de domeniu) de semantica mesajului ataşat (care ar putea să depindă de domeniu). Protocolul de comunicare trebuie

să fie concis, universal partajat de toţi agenţii şi să aibă un număr limitat de fraze de comunicare. KQML este un protocol utilizat în schimbul de informaţii şi cunoştinţe atât între agenţi cât şi între agenţi şi aplicaţii–program. Acest schimb este ilustrat în Figura 4.2.

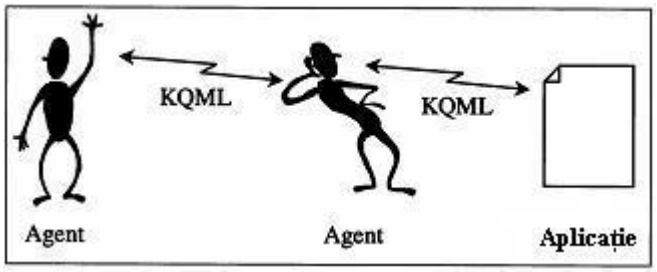

Figura 4.2. Schimbul de informaţii între agenţi şi aplicaţii folosind KQML

Limbajul KQML, dezvoltat iniţial ca o parte a proiectului "DARPA Knowledge Sharing", devine un standard pentru limbajele de comunicare între agenţi.

Eleganţa acestui limbaj constă în faptul că toată informaţia necesară pentru a înţelege conţinutul mesajului este inclusă chiar în comunicare. Protocolul de bază este descris de următoarea structură:

(**frază–KQML**
:sender < *cuvânt* >
:receiver < *cuvânt* >
:language < *cuvânt* >
:ontology < *cuvânt* > // specifică mulţimea cuvintelor folosite în mesaj;
:content <*expresie*> // mesajul propriu-zis;
...
)

Fraza KQML specifică o aserţiune sau o interogare folosită pentru a examina sau modifica o bază de cunoştinţe virtuală a agentului aflat la distanţă. Tabelul 4.3 prezintă o listă a frazelor definite în KQML.

Frază KQML	**Semnificaţia pentru trimiţătorul S (sender) şi receptorul R (receiver) în raport cu baza virtuală de cunoştinţe (Virtual Knowledge Base - VKB)**
achieve	S doreşte ca R să realizeze ceva în mediul său fizic
advertise	S doreşte ca R să ştie că S poate şi va procesa un mesaj ca acela din *:content*
ask-one	S doreşte o instanţiere a lui R referitoare la *:content* care este adevărată pentru R
ask-all	S doreşte toate instanţierile lui R referitoare la *:content* care sunt adevărate pentru R
ask-if	S doreşte să ştie dacă *:content* este în VKB-ul lui R
broadcast	S doreşte ca R să trimită mesaje tuturor agenţilor pe care îi cunoaşte R

broker-all	S doreşte ca R să afle toate răspunsurile la o *frază* (un agent altul decât R va furniza răspnsul)
broker-one	S doreşte ca R să afle un răspuns la o *frază* (un agent altul decât R va furniza răspnsul)
delete-all	S doreşte ca R să-şi şteargă toate potrivirile cu *:content* din VKB
delete-one	S doreşte ca R să-şi şteargă o potrivire cu *:content* din VKB
deny	negaţia propoziţiei este în VKB-ul lui S
discard	S nu mai doreşte răspunsurile lui R la mesaje anterioare
error	S consideră mesajul anterior al lui R ca fiind eronat
eos	marcajul de sfârşit al unui stream ce conţine un răspuns multiplu
forward	S doreşte ca R să forward-eze mesajul agentului specificat în *:to* (R ar putea fi acel agent)
insert	S îi solicită lui R să adauge aserţiunea în VKB-ul său
next	S doreşte următorul răspuns al lui R la un mesaj trimis anterior de S
ready	S este pregătit să răspundă la un mesaj primit anterior de la R
recommend-all	S doreşte să ştie toţi agenţii care pot răspunde la o anumită *frază*
recommend-one	S doreşte să ştie un agent care poate răspunde la o anumită *frază*
recruit-all	S doreşte să ştie toţi agenţii care sunt potriviţi pentru a răspunde la o anumită *frază*
recruit-one	S doreşte să ştie un agent care este potrivit pentru a răspunde la o anumită *frază*
register	S îşi anunţă lui R prezenţa şi numele simbolic
rest	S doreşte ca răspunsurile lui R rămase la un mesaj trimis anterior de S
sorry	S înţelege mesajul lui R dar nu poate furniza un răspuns de informare
standby	S doreşte să-l anunţe pe R că este pregătit să răspundă la mesajul conţinut în *:content*
stream-all	versiunea de răspuns multiplu la o frază *ask-all*
subscribe	S doreşte actualizări ale răspunsului lui R la o *frază*
tell	a aserţiune din VKB-ul lui S
transport-address	S asociază numele său simbolic cu o nouă adresă de transport

Tabelul 4.3. Frazele KQML

Sintaxa KQML seamănă cu cea a limbajului LISP, argumentele identificate de cuvintele cheie precedare de“**:**” pot fi date în orice ordine. KQML este modelat conform conceptelor bazate pe actul vorbirii. Deci, semantica limbajului KQML este independentă de domeniu, în timp ce semantica mesajului este definită de câmpurile: *:content* (mesajul efectiv), *:language* (limbajul în care este scris mesajul) şi *:ontology* (ontologia , vocabularul cuvintelor din mesaj). Putem spune că acest limbaj “încapsulează” un mesaj într–o structură înţeleasă de către toţi agenţii. Pentru a putea înţelege mesajul, receptorul trebuie să înţeleagă limbajul în care acesta este scris şi este necesar ca acesta să aibă

acces la vocabularul (ontologia) acestuia. Termenii *:content, :language* şi *:ontology*, delimitează semantica mesajului. Celelalte argumente, precum *:sender*, *:receiver*, *:reply_with* şi *:in_reply_to*, sunt parametri ce ţin de transmiterea mesajului. KQML suportă comunicări asincrone. Câmpurile *:reply_with*, al unui emiţător, şi *:in_reply_to,* al unui receptor, leagă un mesaj care este trimis cu un răspuns aşteptat.

KQML, ca şi rezultat, face parte dintr–un efort susţinut de dezvoltare a unei metodologii de distribuire a informaţiei între diverse sisteme. O parte a acestui efort o reprezintă definirea KIF (Knowledge Interchange Format). KIF este o sintaxă formală pentru reprezentarea cunoştinţelor. KIF este un limbaj logic particular, care a fost propus ca un standard pentru a descrie informaţii în cadrul sistemelor expert, a bazelor de date, a agenţilor inteligenţi. KIF este o versoune prefixată a logicii de ordinul I, cu extensii pentru gestionarea raţionamentului nemonoton.

Un exemplu în acest sens poate fi dat utilizând ca ontologie ***Lumea blocurilor (cuburilor) (Blocks–World)***. Este vorba de mai multe blocuri (cuburi), dispuse pe o suprafaţă netedă. Pentru descrierea stărilor se folosesc anumite predicate. Noi ne vom referi la două predicate:

- Block(X), a cărui semnificaţie este faptul că X este un bloc (cub);
- ON(X, Y), a cărui semnificaţie este faptul că X se află dispus pe Y.

În acest caz, mesajul: *„blocul A se află pe blocul B”*, poate fi transmis astfel:

(**tell**
:sender *Agent1*
:receiver *Agent2*
:language *KIF*
:ontology *Blocks-World*
:content *(**AND** (**Block** A) (**Block** B) (**ON** A B))*
)

Limbajul unui mesaj KQML nu este restricţionat la KIF, pot fi folosite şi alte limbaje precum LISP, PROLOG, SQL sau orice alt limbaj de comunicare între agenţi.

Vom da în continuare un exemplu de comunicare între doi agenţi. Domeniul în care se face comunicarea este tot **lumea blocurilor**. Presupunem că este vorba despre doi agenţi *Agent1* şi *Agent2* situaţi într-un anumit mediu din lumea blocurilor, fiecare percepând doar o parte a mediului. Ca urmare, este necesar schimbul de mesaje între agenţi.

Spre exemplu, *Agent1* îl poate interoga pe *Agent2* în legătură cu percepţia sa despre mediu, folosind un mesaj KQML al cărui conţinut este descris în Prolog:

(**ask-all**
:sender *Agent1*
:receiver *Agent2*
:language *Prolog*
:ontology *Blocks-World*
:content "*ON(X, Y)*"
)

Agent2 îi va putea răspunde lui *Agent1*, căutând potriviri cu aserţiunile din baza sa de date (să presupunem că *Agent2* percepe din mediu că blocul **A** este situat pe blocul **B** şi blocul **C** este situat pe blocul **D**). Ca urmare, va comunica percepţia sa agentului *Agent1* sub forma:

(**tell**
:sender *Agent2*
:receiver *Agent1*
:language *Prolog*
:ontology *Blocks-World*
:content "*[ON(a, b), ON(c, d)]*"
)

Agenţii care comunică prin KQML dezvoltă relaţii de genul *client–server*. Comunicarea acestora poate fi atât asincronă cât şi sincronizată. Ambele moduri de comunicare sunt reprezentate în Figura 4.4.

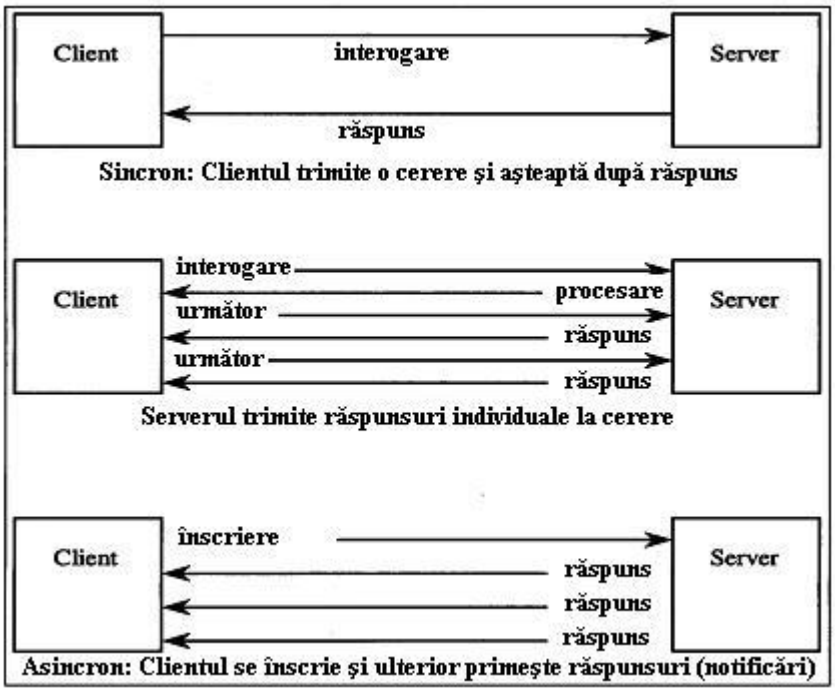

Figura 4.4

Într–o comunicare sincronizată, un agent emiţător aşteaptă răspunsul. Într–o comunicare nesincronizată (asincronă), agentul emiţător îşi continuă activitatea după trimiterea mesajului, activitatea fiindu–i întreruptă de primirea răspunsului.

Un lucru interesant de precizat este faptul că mesajele KQML pot fi intermediate. Dacă un agent nu poate comunica direct cu alt agent, atunci acesta va folosi un intermediar căruia îi va trimite un mesaj, care are ca şi conţinut mesajul pentru destinatarul iniţial, căruia intermediarul îi va transmite mesajul.

În transmiterea de mesaje utilizând KQML, sunt importante de luat în considerare următoarele probleme:

- emiţătorul şi receptorul trebiue să înţeleagă limbajul de comunicare utilizat;
- vocabularul trebuie să fie creat şi trebuie să fie accesibil tuturor agenţilor care comunică;

- KQML operează corect în cadrul unei infrastructuri de comunicare care le permite agenţilor să se localizeze, infrastuctura nefiind parte a specificării KQML;
- KQML este un proiect în dezvoltare, semantica lui nu a fost complet definită şi nu există încă o specificare oficială a acestuia.

4.2.4.2 FIPA ACL

Un alt limbaj de comunicare între agenţi este limbajul de comunicare **FIPA ACL**.

FIPA (The Foundation for Intelligent Physical Agents) este o organizaţie internaţională dedicată promovării industriei agenţilor inteligenţi prin dezvoltarea de specificaţii care suportă interoperabilitate între agenţi şi aplicaţii bazate pe agenţi. Mai multe despre standardul FIPA se pot găsi la adresa **www.fipa.org**.

Sistemele multiagent (MAS) sunt deseori caracterizate ca fiind extensii ale sistemelor orientate obiect (OO). De multe ori, proiectanţii de sisteme multiagent au probleme încercând să surprindă caracteristicile unui sistem MAS folosind mecanisme specific OO. Ca un răspuns la această problemă, FIPA dezvoltă un limbaj de modelare bazat pe agenţi AUML, ca o extensie a limbajului de modelare UML (Unified Modeling Language).

4.2.5 Protocoale de interacţiune între agenţi

Alte probleme care apar în sistemele multiagent, de care însă se ocupă protocoalele de interacţiune, sunt:

- Coordonarea.
- Cooperarea.
- Negocierea.
- Încrederea în afirmaţiile făcute de către agenţii dintr–un sistem multiagent.

4.2.5.1 Protocoale de coordonare

Într–un mediu cu resurse limitate, agenţii trebuie să–şi coordoneze activităţile pentru a satisface atât interesele proprii cât şi pe cele ale grupului. Acţiunile agenţilor trebuie să fie coordonate deoarece există dependenţe între ele, şi nici un agent nu are suficientă competenţă, resurse sau informaţii pentru a atinge scopurile sistemului. Exemple de coordonări includ informarea celorlalţi agenţi, asigurând faptul că acţiunile agenţilor sunt sincronizate şi evitând rezolvarea de probleme redundante.

Pentru a produce sisteme coordonate, majoritatea cercetătorilor în IAD s–au concentrat pe tehnici distribuite, atât a controlului cât şi a datelor. Controlul distribuit se referă la faptul că agenţii au autonomie în generarea de noi acţiuni şi în deciderea scopurilor pe care să le urmeze. Dezavantajul distribuirii controlului şi a datelor este reprezentat de faptul că avem o dispersare a cunoştinţelor în întreg sistemul, astfel fiecare agent posedă perspective parţiale şi imprecise. Există, deci, un grad ridicat de incertitudine despre acţiunea fiecărui agent şi este mult mai dificilă menţinerea unui comportament global coerent.

4.2.5.2 Protocoale de cooperare

Cooperarea este foarte importantă în sistemele multiagent, atât în cazul celor formate din agenţi antagonici, cât şi în cazul celor formate din agenţi nonantagonici. În cazul primelor cooperarea este importantă pentru ca agenţii să rezolve cu succes problema ce reprezintă scopul lor comun. În cel de–al doilea caz cooperarea este rezultatul negocierilor duse de agenţi. Unul dintre agenţii implicaţi în negociere obţine cooperarea celuilalt în atingerea scopului, bineânţeles după încheierea negocierilor şi stabilirea unui compromis.

O strategie de bază folosită de multe protocoale de cooperare este descompunerea şi, apoi, distribuirea sarcinilor. Un astfel de exemplu este metoda *„Împarte şi cucereşte" (Divide et Impera),* care reduce complexitatea unei probleme prin împărţirea în probleme mai mici care pot fi distribuite unor agenţi de complexitate redusă şi cer mai puţine resurse. Descompunerea problemei poate fi făcută de proiectantul sistemului, sau de agenţi. După descompunerea în probleme mai mici, distribuirea sarcinilor se face respectând următoarele criterii:

- se evită supraîncărcarea critică a resurselor;
- se repartizează sarcinile către fiecare agent, potrivit capabilităţilor fiecăruia;
- se desemnează un agent competent să distribuie sarcinile;
- se repartizează responsabilităţile corespunzător fiecărui agent, pentru a obţine coerenţă.

4.2.5.3 Sistemele tip Tablă (Blackboard)

„Imaginaţi–vă un grup de oameni sau agenţi specialişti aşezaţi alături de o tablă mare. Specialiştii cooperează pentru a rezolva o problemă, folosind tabla ca spaţiu de lucru pentru găsirea soluţiei. Rezolvarea începe atunci când problema şi datele iniţiale sunt scrise pe tablă. Specialiştii privesc tabla, căutând o oportunitate de aplicare a expertizei lor pentru soluţia în dezvoltare a problemei. Când un specialist găseşte suficiente informaţii pentru a–şi aduce contribuţia, el le înregistrează pe tablă. Ceilalţi specialişti pot utiliza aceste informaţii adiţionale pentru a–şi aduce contribuţia (expertiza lor). Acest proces de adăugare a contribuţiilor pe tablă continuă până când problema este rezolvată."

Acest enunţ prezintă ***sistemele Blackboard*** de rezolvare a problemelor ***(sistemele tip tablă)***. Enunţul este o metaforă a sistemelor Blackboard, şi subliniază o serie de caracteristici importante ale acestora. Printre aceste caracteristici se numără:

- **Independenţa expertizei**

  Specialiştii (numiţi surse de cunoştinţe sau KSs (Knowledge Sources)) nu sunt antrenaţi pentru a lucra doar cu acest grup de specialişti, fiecare dintre ei este expert în unul sau mai multe aspecte ale problemei şi poate contribui la găsirea soluţiei problemei independent de tipul celorlalţi specialişti din grup.

- **Diversitatea tehnicilor de rezolvare a problemelor**

  În Sistemele Blackboard reprezentarea internă şi inferenţele utilizate de fiecare specialist sunt ascunse şi nu este posibilă o vizualizare directă a lor.

- **Limbajul comun de interacţiune**

  Specialiştii în Sistemele Blackboard trebuie să fie capabili să interpreteze corect informaţiile scrise pe tablă, de către oricare dintre aceştia. În realitate însă, nu există o reprezentare general înţeleasă de toţi specialiştii, ci o expresie a unei reprezentări specializate, reprezentativă doar pentru câţiva dintre specialişti.

- **Reprezentarea flexibilă a informaţiei**

  Modelul tablei nu presupune nici o restricţie legată de informaţiile ce pot fi scrise pe tablă.

- **Activarea bazată pe evenimente**

  Acţiunile specialiştilor sunt generate atât de evenimente legate de tabla pe care se rezolvă problema cât şi de evenimente externe. Evenimentele legate de tablă sunt: adăugarea de noi informaţii, modificarea de informaţii, ştergerea de informaţii. În loc de scanarea tablei, fiecare specialist informează sistemul despre evenimentele care îl interesează, iar în momentul în care un eveniment are loc sistemul informează specialiştii interesaţi.

- **Nevoia de control**

  Există în sistemele Blackboard o componentă de control separată de specialişti. Această componentă poate fi privită ca şi un specialist în probleme de organizare şi de stabilire a direcţiei de rezolvare a problemei. Când specialistul curent selectat pentu a-şi aduce contribuţia termină analiza celor aflate pe tablă, componenta de control alege specialistul cel mai indicat dintre toţi cei care au fost activaţi şi acesta îşi va aduce contribuţia la rezolvarea problemei. În momentul în care un specialist este ales să contribuie la rezolvarea problemei, acesta evaluează posibila sa contribuţie şi informează componenta de control în legătură cu această evaluare, fără a efectua efectiv calculele prin care ar putea să–şi aducă aportul la găsirea soluţiei. Pe baza acestor evaluări, primite de la specialişti, componenta de control va decide care este cea mai bună direcţie de rezolvare a problemei, care apoi va fi utilizată.

- **Obţinerea incrementală a soluţiei**

  Fiecare dintre specialişti contribuie la determinarea soluţiei în modul potrivit, uneori rafinând rezultate existente, alteori contrazicând un rezultat propus sau propunând rezultate.

4.2.5.4 Negocierea

O formă frecventă de interacţiune care apare între agenţi cu scopuri diferite este *negocierea*. Negocierea este un proces prin care o decizie comună este obţinută de doi sau mai mulţi agenţi, fiecare dintre aceştia având un scop (sau obiectiv) individual. În primul rând agenţii îşi comunică părerile (poziţiile cu privire la o afirmaţie: acord,

dezacord indiferenţă, noi propuneri). Aceste păreri pot fi contradictorii. Apoi aceştia încearcă să ajungă la o înţelegere prin concesii sau căutând alternative.

Principalele elemente ce apar în negociere sunt:

- limbajul utilizat de agenţii participanţi;
- protocolul urmat de agenţi pe parcursul negocierii; procesul de decizie utilizat de fiecare agent pentru a stabili care îi sunt poziţiile, concesiile şi criteriile de înţelegere.

Multe grupuri de agenţi au dezvoltat sisteme şi tehnici de negociere. Acestea pot fi centrate pe mediu sau pe agent. Proiectanţii de sisteme centrate pe mediu se concentrează pe rezolvarea următoarei probleme: *„Cum pot fi proiectate regulile mediului astfel încât agenţii care operează în acesta să interacţioneze corect şi productiv, indiferent de originea, capacităţile sau intenţiile acestora?”*. Mecanismul de negociere ideal ar trebui să aibă următoarele atribute:

- **eficienţă**: agenţii nu ar trebui să risipească resurse în procesul de negociere;
- **stabilitate**: nici un agent nu are posibilitatea să devieze de la stategii asupra cărora şi–a dat acordul;
- **simplitate**: un mecanism de negociere trebuie să impună sau să implice un număr mic de calcule sau cerinţe pentru agenţii implicaţi în negociere;
- **distribuţie**: mecanismele de negociere nu trebuie să necesite un arbitru central în luarea deciziilor;
- **simetrie**: mecanismul nu trebuie să fie construit împotriva vreunui agent.

Proiectanţii de sisteme centrate pe agent se concentrează pe rezolvarea următoarei probleme: *„Fiind dat un mediu în care operează agenţii, care este cea mai bună strategie pe care aceştia ar trebui să o urmeze?”*. Cele mai multe strategii de negociere au fost determinate pentru probleme concrete, de aceea nu sunt formulate multe caracteristici generale. Există totuşi două direcţii generale de abordare, bazate pe presupuneri legate de tipurile particulare de agenţi implicaţi. O direcţie presupune capacităţi legate de actul vorbirii şi o anumită semantică a cuvintelor, care ar caracteriza protocolele de negociere şi componentele acestora. Cea de–a doua direcţie este bazată pe presupunerea că agenţii sunt raţionali din punct de vedere economic. Aceştia, în acest caz, sunt în număr mic, au un limbaj şi un mod de abstractizare comun şi trebuie să ajungă la o soluţie comună.

Agenţii care utilizează acest protocol crează un ***pact***, care este reprezentat de un plan comun al agenţilor, în care toate scopurile acestora sunt îndeplinite. Utilitatea acestei înţelegeri, raportată la fiecare agent, este costul pe care acesta este dispus să îl suporte din care scădem costul pactului.

4.2.5.5 Păstrarea credibilităţii agenţilor în sistemelor multiagent

În sistemele multiagent este necesar ca fiecare agent să se poată baza pe faptul că afirmaţiile celorlalţi agenţi din sistem au un fundament corect şi urmăresc scopul cunoscut al acestora. Pentru a asigura acestă cerinţă, este nevoie într–un sistem multiagent de un sistem de păstrare a credibilităţii (**truth–maintenance system - TMS**).

Un sistem TMS în cadrul sistemelor multiagent poate servi drept exemplu de interacţiune la nivel înalt între agenţi. Un astfel de sistem este proiectat pentru a asigura

integritatea cunoştinţelor agenţilor, care ar trebui să fie stabile, să aibă o bază corectă şi să fie consistente din punct de vedere logic.

Având în vedere modul în care credinţele, justificările şi datele sunt reprezentate, o stare stabilă a unei baze a cunoştinţelor este una în care:

- fiecare dată care are o justificare validă este considerată corectă;
- fiecare dată care nu are o justificare validă nu este de încredere.

Consistenţa logică este atinsă în momentul în care există stabilitate la momentul determinării consistenţei şi nu există nici o contradicţie. Datele care stau la baza unor cunoştinţe sunt considerate consistente, dacă nici una dintre aceste nu este în acelaşi timp şi corectă şi incorectă sau nu este nici corectă nici incorectă. Alte caracteristici dorite pentru o bază a cunoştinţelor sunt: completitudinea, acurateţea şi eficienţa.

Un TMS considerat pentru un singur agent încearcă să menţină baza cunoştinţelor sale într–o stare stabilă, bine fundamentată, prin stabilirea datelor corecte şi a celor incorecte. Pentru agenţii dintr–un grup de agenţi este important să poată fi siguri de integritatea informaţiei comunicate, precum şi a cunoştinţelor proprii şi să poată menţine această integritate. Un TMS considerat pentru mai mulţi agenţi poate asigura aceste cerinţe.

Să considerăm în ceea ce urmează, un TMS modificat, bazat pe justificări. În acesta, fiecare dată are un set de justificări şi se află în una din următoarele trei stări: *internă* (credibilă, pe baza unei justificări locale valide), *externă* (credibilă, pentru că un alt agent a afirmat–o) sau *eliminată* (nu este credibilă). Într-o reţea formată din mai mulţi agenţi cu opinii (credinţe) imparţiale şi independente, pentru a asigura o comunicare bine fundamentată, o dată comunicată trebuie să fie *internă* pentru cel puţin unul din agenţii care o consideră adevărată şi *internă* sau *externă* pentru ceilalţi. De aceea, un agent nu poate schimba starea unei date din proprie iniţiativă, el trebuie să aibă acordul celorlalţi şi să se coordoneze cu aceştia pentru menţinerea consistenţei, astfel data va avea starea corectă stabilită pentru toţi şi de toţi agenţii.

Un TMS considerat pentru mai mulţi agenţi acţionează în momentul în care o justificare este adăugată su ştearsă, respectând următoarele principii:

- schimbarea credinţelor trebuie rezolvată implicând cât mai puţini agenţi;
- schimbarea credinţelor trebuie rezolvată modificând cât mai puţine credinţe.

Un TMS acţionează astfel:

- scoate o dată din starea în care este, chiar date justificate recent şi presupuse a fi consecinţe ale acesteia;

  Această dată a cărei stare e acum incertă poate afecta unul sau mai mulţi agenţi. Dacă o dată este incertă pentru un agent atunci acea dată devine incertă pentru toţi agenţii care o partejează.
- alege stări potrivite pentru datele incerte;
- fiecare agent trece datele incerte în stările corespunzătoare alese, respectând cerinţele impuse de data partajată.

  Dacă vreunul dintre agenţi nu poate trece o dată în starea aleasă atunci se revine la pasul anterior. Se alege o nouă stare pentru această dată sau se trec alte date în stare incertă.