Comunicarea în modelul obiectelor active

Raluca Vartic
Universitatea Tehnică Cluj-Napoca
Catedra de Calculatoare
Bariţiu 28
Raluca.Vartic@cs.utcluj.ro

Dorian Gorgan
Universitatea Tehnică Cluj-Napoca
Catedra de Calculatoare
Bariţiu 28
Dorian.Gorgan@cs.utcluj.ro

REZUMAT

Modelul obiectelor active permite tratarea independenta a elementelor grafice ale unei aplicaţii, folosind programarea orientată pe obiecte. Autonomia acestor elemente încurajează tratarea lor ca agenţi într-un sistem multi-agent. Pentru o colaborare şi sincronizare adecvată, trebuie abordată problema modalităţii în care agenţii comunică. Soluţia propusă ţine cont de tendinţele în definirea de limbaje de comunicare între agenţi, dar şi de caracteristicile particulare ale modelului obiectelor active.

Categorii şi descriptori ai subiectelor

D.1.7 [**Programming Techniques**]: Visual Programming

D.1.3 [**Programming Techniques**]: Object Oriented Programming

D.2.12 [**Software Engineering**]: Interoperability – *interface definition languages.*

Termeni generali

Languages, Design, Human Factors, Performance.

Cuvinte-cheie

Agenţi, comunicare, limbaj, grafică.

1. INTRODUCERE

Termenul de agent a devenit o metaforă des utilizată pentru desemnarea aplicaţiilor soft autonome, proactive şi reactive. Dacă aplicaţiile au fost scrise pentru a coopera, apare necesitatea abilitaţilor sociale. Comunicarea este mecanismul prin care este facută posibilă coordonarea şi colaborarea în sistemele multi-agent.

Odată cu evoluţia în spaţiul Active Objects Model (AOM) apare şi nevoia schimbului de informaţii între obiecte. Astfel, modelul obiectelor active poate fi văzut ca un sistem multi-agent, în care obiectele active complexe sunt cele care joacă rolul agenţilor.

Secţiunea 2 este o prezentare generală a modelului AOM, iar secţiunea 3 descrie propunerile cele mai importante pentru a rezolva problema comunicării în sistemele multi-agent. Pentru a stabili un model de comunicare în AOM, mai întâi se va încerca identificarea situaţiilor în care se va apela la schimbul de mesaje între obiecte, în secţiunea 4, iar secţiunea 5 oferă o soluţie. Concluziile sunt expuse în secţiunea 6.

2. MODELUL OBIECTELOR ACTIVE

Modelul Obiectelor Active (Active Objects Model – AOM) este o abordare orientată pe obiecte a elementelor grafice ale unei interfeţe sau aplicaţii, menită să uşureze controlul acestora prin modelarea unor structuri bazate pe entităţi. Această abordare permite analiza şi dezvoltarea tehnicilor de programare vizuală, funcţionalitate distribuită şi procesare paralelă.

Modelul este alcătuit din entităţi, care pot fi active sau pasive. Entităţile sunt *obiecte*, instanţe ale unor *clase*, iar obiectele comunică prin *mesaje*. Fiecare obiect este caracterizat de o identitate, o stare şi un comportament [5].

Entităţile pasive sunt resurse folosite de entităţile active, şi sunt împărţite în următoarele tipuri: comportamente, traiectorii, pozitii explicite ale traiectoriei, reguli, expresii, acţiuni (*create, delete, instantiate, append, get, set, assign, call, jmp*), şi prezentări. Entităţile active pot fi obiecte active (caracterizate de stare, părinte, componente, comportament, interactor, prezentator şi multime de procesare) sau variabile (entităţi globale folosite pentru a reprezenta stări în evoluţia modelului) [1].

Evoluţia modelului constă în evoluţia concurentă şi paralelă a obiectelor active componente. Evoluţia unui obiect activ este definită de o traiectorie, compusă din poziţii explicite (ETP), care au asociate reguli privind execuţia obiectului când ajunge în respectiva poziţie [6].

Tema acestei lucrari este tratarea aspectelor legate de comunicare în AOM.

3. COMUNICAREA ÎN SISTEME MULTI-AGENT

Pentru a depăşi problemele legate de comunicarea între doi sau mai mulţi agenţi, aceştia trebuie să aibă un limbaj comun. În prezent în AOM nu se utilizează un limbaj standard. În continuare sunt prezentate limbajele FIPA-ACL şi KQML.

3.1 FIPA-ACL

FIPA este organizaţia pentru agenţi fizici inteligenţi, care încearcă dezvoltarea de specificaţii pentru tehnologii legate

de agenţi, în vederea furnizării unui nivel înalt de interoperabilitate între aplicaţii. În ceea ce priveşte comunicarea inter-agenţi, FIPA a propus un limbaj: FIPA-ACL, care se ocupă cu mesaje ACL, protocoale de interacţiune prin schimb de mesaje, acte de comunicare bazate pe teoria "speech act" – a actelor discursive, şi reprezentarea limbajelor conţinute.

FIPA-ACL propune următoarele acte de comunicare: *Accept Proposal, Agree, Cancel, Call for Proposal, Confirm, Disconfirm, Failure, Inform, Inform If, Inform Ref, Not Understood, Propagate, Propose, Proxy, Query If, Query Ref, Refuse, Reject Proposal, Request, Request When, Request Whenever, Subscribe* [4].

Această mulţime de acte de comunicare furnizează, în viziunea FIPA, setul de bază al primitivelor de comunicare pentru orice sistem multi-agent.

Un agent îşi planifică îndeplinirea obiectivelor prin comunicarea cu alţi agenţi (trimiteri de mesaje şi primiri de răspunsuri). Actele de comunicare vor fi selectate de agent pe baza relevanţei rezultatului, sau a efectului raţional scontat.

3.2 KQML

Interacţiunile existente în mediile computaţionale actuale au pus bazele consorţiului KSE (Knowledge Sharing Effort), preocupat de problemele de partajare a cunoştinţelor, adică înţelegerea mutuală a acestora şi comunicarea lor. KSE propune limbajul şi protocolul de comunicare KQML (Knowledge Query and Manipulation Language).

Un limbaj comun pentru reprezentarea conţinutului bazelor de date folosit de KQML este KIF (Knowledge Interexchange Formalism), limbaj bazat pe logica de ordinul întâi, care include atât specificarea sintaxei, cât şi specificarea semanticii limbajului. KIF este folosit pentru translatarea conţinutului bazelor de date ce folosesc reprezentări diferite. În afară de limbajul comun, agenţii trebuie de asemenea să partajeze un framework comun al cunoştinţelor, având ca scop posibilitatea interpretării mesajelor interschimbate, pentru care se folosesc ontologii partajate. Pe lângă sintaxă şi semantică, KQML oferă suport pentru pragmatică, ceea ce pentru agenţi înseamnă să cunoască cu cine vor să comunice şi cum să găsească acest partener de comunicare, şi să ştie să iniţieze şi să păstreze un schimb de mesaje [2].

4. SCENARII DE COMUNICARE

În continuare vom încerca identificarea principalelor cazuri în care ar fi utilă comunicarea între agenţi în AOM.

Un scenariu simplu ar fi unul în care comunicarea are loc pentru:

- interogarea stării interne a altui obiect
- modificarea stării interne a altui obiect

Singurele primitive de comunicare sunt *set(Agent, Attribute, Value)* şi *get(Agent1, Agent2, Attribute)*. De exemplu, dacă avem 2 obiecte: un obiect grafic, şi un numărător care ţine evidenţa click-urilor date de utilizator pe obiectul grafic, la un nou eveniment mouseClick, obiectul grafic ar interoga numărătorul pentru a-i afla valoarea curentă, şi i-ar seta noua valoare, obţinută prin incrementarea valorii vechi.

O primă limitare a acestei abordări este lipsa explicită a delegării de sarcini către alte obiecte. Dacă ar exista o primitivă pentru a cere unui alt obiect executarea unei acţiuni în limbajul înţeles de agenţi (*do(Agent, Action)*), codul pentru obiectele implicate s-ar simplifica substanţial. Pentru exemplul prezentat anterior, numărătorul ar avea o metodă de incrementare a valorii reţinute, iar obiectul grafic ar trebui doar să apeleze această metodă în cazul unui eveniment mouseClick. Adăugarea acestei primitive la protocolul cunoscut de obiecte oferă avantaje nete: comunicaţie redusă, cod mai citibil.

Un exemplu mai elocvent pentru a ilustra avantajele delegării de acţiuni ar fi o aplicaţie grafică pentru simularea jocului de biliard. Tacul şi mingile sunt obiecte active. De asemenea, mai există obiecte reprezentate prin linii care arată traiectoria bilelor în funcţie de poziţia curentă a tacului. Dacă nu ar exista un act de comunicare pentru determinarea executării de acţiuni, obiectul "tac" ar trebui să calculeze traiectoriile tuturor bilelor, pentru a seta orientarea liniilor ajutătoare. De asemenea, la executarea loviturii, obiectul "tac" este cel care calculează poziţiile bilelor. Cu noua abordare, calculele se pot delega obiectelor implicate, care le pot efectua în paralel, şi viteza de calcul va fi îmbunătăţită.

În cazul în care un obiect îşi schimbă starea (poziţia, culoarea), iar starea lui poate fi folosită de alţi agenţi, însă obiectul în discuţie nu are nevoie să ştie în ce mod această stare este folosită, ar fi utilă o acţiune de informare a agenţilor interesaţi de schimbarea produsă. Pentru generalizare, presupunem că agenţii ar putea sesiza şi proprietăţi ale mediului, care nu ţin neaparat de agenţi, dar care ar putea fi informaţii utile pentru comunitate. Limbajul va fi extins astfel cu primitiva de informare *inform(Fact)*. Informaţia va fi transmisă întregii comunităţi. Agentul transmiţător furnizează doar o informaţie, felul în care aceasta este folosită intră în atribuţiile celorlalţi agenţi. Dacă în schimb ştim că doar unii agenţi sunt interesaţi de o anumită informaţie, nu are rost să încărcăm canalele de comunicare prin informarea tuturor obiectelor. Ar fi suficientă o primitivă *tell(Agent, Fact)*.

5. SOLUTIA PROPUSĂ

Pentru comunicarea între agenţi s-au dezvoltat limbaje speciale, care pun la dispoziţie acte de comunicare pe care aceştia trebuie să le folosească pentru a comunica între ei sau cu utilizatori umani. Un scop principal al acestor

limbaje este furnizarea unei mulțimi standard de acte de comunicare (CA). Alegerea acestei mulțimi se face pe baza unui compromis între generalitate (pentru reutilizare) şi expresivitate. O bibliotecă de acte de comunicare este FIPA ACL.

Catalogul FIPA include 22 de acte de comunicare. Un agent *i* are atitudini (convingeri B_i, incertitudini U_i, opțiuni C_i). Se presupune că formulele din mulțimea următoare se exclud reciproc: (B_i $\neg\phi$, U_i $\neg\phi$, U_i ϕ, B_i ϕ). Între aceste atitudini există relații. Înțelesul acțiunilor se modelează prin efectul rațional şi precondițiile acestora.

Se identifică cinci tipuri majore de interacțiuni sociale [3]:

1. Schimb de mesaje, cu următoarele CA: inform, confirm, disconfirm, failure.
2. Executare de acțiune 1 (Solicitant – Solicitat), CA: request, request when, request whenever, cancel.
3. Schimb de informații (între: cel care încearcă să găsească o informație şi furnizorul informației), CA: query-if, query-ref, inform-if, inform-ref.
4. Executare de actiune II (solicitantul nu e sigur de conditiile în care solicitatul va executa actiunea), CA: cfp, propose.
5. Mediere, CA: propagate, proxy.

Aceste interacțiuni se pot clasifica în acțiuni sociale volitive şi informaționale.

Analizând scenariile din secțiunea anterioară, observăm că setul de acte de comunicare propus de FIPA acoperă scenariile propuse conform tabelului Tabel 1.

Tabel 1 Echivalenta FIPA – acte de comunicare în AOM

AC în AOM	**AC în FIPA**
set	request, request when, request whenever, propose, cfp
get	query ref, inform ref
do	request, request when, request whenever, propose, cfp
inform, tell	inform, confirm, disconfirm, failure

Se observă de asemenea că fiecărei acțiuni cerere îi corespunde o acțiune răspuns, fiind astfel stabilit un protocol simplu de comunicare.

Revenind la modelarea jocului de biliard prezentată în secțiunea anterioară, se va ilustra modul în care se poate aplica noua metodă propusă pentru comunicarea inter-agenti. Pentru simplitate, să presupunem că singurele obiecte prezente în scenă sunt: masa de biliard, tacul, bila obiect şi o bilă colorată. Traiectoria tacului şi viteza fiind stabilite de către utilizator, în punctul final al acestei traiectorii se trimite un mesaj bilei obiect, prin care aceasta este informată de acțiunea executată, şi trimiterea unui alt mesaj prin care se cere începerea executiei agentului *BilaObiect*. Trimiterea mesajului se poate realiza prin mecanismul rule-action deja existent în AOM, cu condiția să se introducă un nou tip de acțiune: *send agent(<numeAgent>),<mesaj>*, obținând următorul cod pentru agentul *Tac*:

```
agent Tac{
...
behavior Hit{
...
 trajectory T1{...
  trjposition ETP1{...}, ...
  ETPn{
   rule R1{
    action A1{
     send agent(BilaObiect), inform(hit, traseu, viteză);
    },A2{
     send agent(BilaObiect), request(set, state, PLAY);
 ...
};
```

La rândul ei, bila obiect va cere să fie informată de pozițiile celorlalte bile de pe masa, în cazul propus una singură (BilaColorată1), trimițând câte un mesaj fiecăreia, după care işi va calcula traiectoria în funcție de datele primite. În punctele de pe traiectorie în care bila obiect întâlneşte alte obiecte mobile, le va trimite un mesaj prin care sunt informate de viteza şi directia loviturii, şi le va cere să-şi înceapă execuțiile.

Codul pentru agentul *BilaObiect* va arăta astfel:

```
agent BilaObiect{
...
behavior move{
...
 trajectory T1{...
  trjposition ETP1{...
   rule R1{
    action A1{
     send agent(BilaColorata1), request(get position);
     };
     <calcul traiectorie>;
   };
  }, ...
  ETPi{...
   rule R1{
    action A1{
     send agent(BilaColorată1), inform(hit, traseu, viteză);
    },A2{
     send agent(BilaColorată1), request(set, state, PLAY);
 ...
};
```

În cazul în care bila ajunge într-unul din buzunare, agentul BilaObiect îşi va trimite un mesaj în care îşi va opri execuţia.

Comportamentul agentului *BilaColorată1* va fi asemănător cu cel al agentului *BilaObiect*. Pentru a păstra un nivel scăzut de complexitate, în acest exemplu ilustrativ nu am inclus unele aspecte legate de regulamentul acestui joc.

Dacă până acum era nevoie de multe tipuri de actiuni pentru interacţiunea agenţilor, acum singura actiune necesară este cea de *send*, restul informatiei fiind inclusă în conţinutul mesajului.

Modelul Obiectelor Active nu a ajuns încă la maturitate, şi probabil pe măsură ce se vor dezvolta noi aplicaţii folosind acest model, vor apărea şi scenarii noi de comunicare, pe care nu le putem intui încă. Se poate chiar ca noile cazuri să nu fie acoperite nici chiar de librăria FIPA ACL. În acest caz avem trei variante:

1. definirea unei noi librării, care să includă actele de comunicare relevante domeniului
2. rămânerea la FIPA CAL, şi codarea de informaţie adiţională în conţinutul mesajului
3. folosirea FIPA CAL şi a unor subcataloage noi de CA

Ultima alternativă este cea mai avantajoasă, furnizând atât acte de comunicare generale standard cât şi acte de comunicare specifice, care asigură expresivitatea.

În proiectarea unor extensii specifice contextului s-au propus în literatura urmatorii 3 paşi [3]:

1. identificarea tipurilor de interacţiuni sociale care pot fi suportate direct de FIPA ACL, nesuportate dar reutilizabile în alte aplicaţii, şi nesuportate, specifice sistemului.
2. identificarea actelor de comunicare
3. formalizarea actelor de comunicare

Se observă că soluţia propusă poate fi astfel uşor extinsă pentru aplicaţii care necesita mai multa expresivitate în ceea ce priveste comunicarea între agenţi.

6. CONCLUZII

O idee îndrăzneaţă în acest moment este că agenţii ar putea înţelege limbajul natural şi l-ar putea folosi în comunicare. Pentru abordarea limbajelor naturale, teoria limbajelor formale şi gramaticile de structură a frazei (în particular gramatici lipsite de context) sunt unelte utile. Propoziţiile sunt parsate, iar interpretarea semantică se realizează de o gramatică augmentată. Câteva probleme care apar în cazul limbajelor naturale sunt ambiguitatea şi contextul.

Deocamdată însă, agentii interacţionează prin schimb de semnale, emise pentru a realiza anumite scopuri. Abordarea propusă în lucrarea de faţă a vizat modalităţile de comunicare specifice Modelului Obiectelor Active, şi aduce îmbunătăţiri substanţiale în ceea ce priveşte autonomia agenţilor (singurele interacţiuni permise sunt schimburile de mesaje, care se conformează protocolului stabilit). De asemenea, metoda este scalabilă, modificările putând fi realizate uşor, specific aplicaţiilor care le necesită.

7. REFERINŢE

[1] Cornea, V.S. *Construirea tehnicilor de interacţiune în spaţiul virtual (AGML – Agent Modelling Language)*, UTCN, 2003.

[2] Finin, T., Fritzson, R., McKay, D., McEntire, R. KQML as an Agent Communication Language, *Proceedings of the 3rd International Conference on Information and Knowledge Management* (CIKM'94) 1997.

[3] Serrano, J.M., Ossowski, S. The design of communicative act libraries: a linguistic perspective. *Applied Artificial Intelligence*, Vol. 16, No. 9/10, 2002.

[4] * * *, *FIPA Communicative Act Library Specification*: http://www.fipa.org/specs/fipa00037/SC00037J.html#_Toc26729690

[5] * * *, *AOM*: http://bavaria.utcluj.ro/~gorgan/res/aom/aom.html

[6] * * *, *Visual Programming* http://bavaria.utcluj.ro/~gorgan/res/vp/vp.html