

# Tipuri de agenți software

## Interfețe utilizator inteligente și adaptabile, sisteme multi-agent

– Sabin-Corneliu Buraga

În acest articol vom prezenta mai întâi o serie de aspecte ale *agenților de interfață*, un domeniu foarte atractiv pentru specialiștii în interacțiunea dintre om și calculator. Factorii care disting agenții de interfață de alte abordări sunt atributele de autonomie și de flexibilitate.

În ultima perioadă s-a pus tot mai mult în evidență nevoia de interactivitate sporită a aplicațiilor. Caracteristicile principale ale unei interfețe-utilizator (funcționalitate, ușurință în utilizare, ergonomie, „intelligență”, simplitate etc.) sunt cu atât mai dezirabile cu cât calculatorul devine bun al tuturor, cerințele pentru o interfață inteligentă fiind următoarele: interfață intuitivă, ușor de manipulat, fără instruirea în prealabil a utilizatorului, având posibilități de comunicare și de înțelegere a cerințelor utilizatorului, flexibilă (suport pentru dispozitive multiple de intrare/ieșire, comunicare în diverse moduri: limbaj natural, semne, percepție globală).

### Prezentare generală

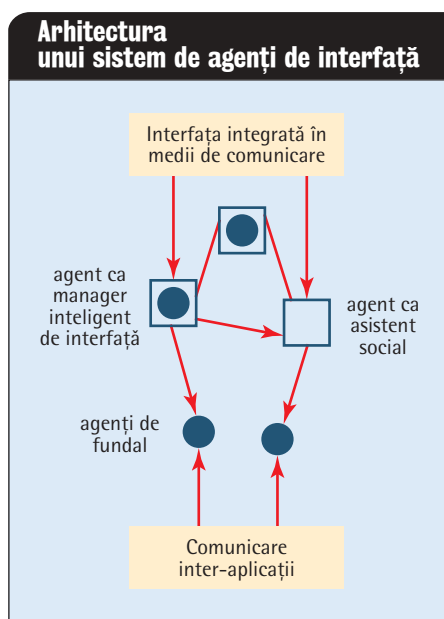
De câțiva ani buni (începând cu Xerox Star, apoi cu Apple Macintosh, pe urmă cu Microsoft Windows), interfețele *manipulate direct* de către utilizator au devenit standard de facto, utilizatorii fiind informați constant (prin intermediul meniurilor și butoanelor) ce anume pot face sau li se permite să facă. Problemele care pot apare în dialogul cu sisteme distribuite, de cele mai multe ori la nivel planetar, așa cum se întâmplă cu Web-ul, sunt: imposibilitatea prin tehnici tradiționale de indexare de a accesa informațiile pe care le dorește utilizatorul, planificarea activităților într-un mediu imprecizabil, rigiditatea în operare, lipsa de flexibilitate a interfețelor actuale etc.

Aici ar putea interveni agenții de interfață. Rolurile unui agent de interfață ar putea fi următoarele:

- asistarea utilizatorului în comunicarea cu sistemul;
- învățarea așa-numitului *profil* al utilizatorului (preferințe, opțiuni mai des folosite, operații specifice, pretenții etc.);
- selectarea funcționalității dorite de utilizator.

Aspectul cel mai interesant al unui agent de interfață este cel al *personificării*, agentul putând simula comportamentul unui personaj (avatar), ducând la o apropiere mai bună a utilizatorului uman de mașină. Un exemplu de astfel de agent antropomorf este Maes.

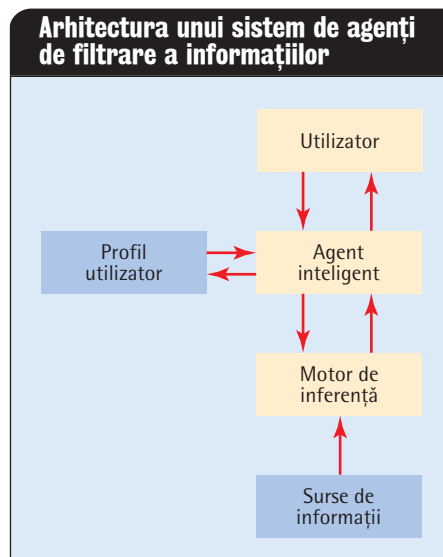
Arhitectura generală a unui sistem de interfață prin intermediul agenților este cea din figura „Arhitectura unui sistem de agenți de interfață”.



### Proiectarea unui agent de interfață

Una dintre principalele probleme în proiectarea unui agent de interfață este colectarea de informații referitoare la interesele, scopurile și preferințele generale ale utilizatorului, în vederea alcătuirii unui așa-numit *model utilizator*. În cadrul modelului utilizator se realizează „reprezentări ale unor caracteristici și atitudini ale operatorilor umani, utile pentru adoptarea unei interacțiuni individualizate și adecvate stabilită între mediile computaționale și oameni” (Paiva, Self și Hartley). Un model utilizator va reține scopul sau activitatea utilizatorului, interesele, planurile și intențiile pe termen scurt, cunoștințele an-

terioare în domeniu, educația, punctul de vedere al subiectului, experiența și abilitățile sale, diferitele preferințe.



Pot fi abordate aici tehnici tradiționale ca învățarea automată (folosirea clasificării simbolice, rețelele neuronale sau a altor metode). O altă variantă este învățarea direct din mediul de operare. Se au în vedere stereotipurile, analiza acțiunilor utilizatorului, modul de interacțiune cu sistemul (folosindu-se inferența sau recunoașterea automată).

Un agent de interfață poate fi de mai multe tipuri:

**Agent de filtrare a informației.** Componenta agent a unui astfel de sistem se bazează pe metafora asistentului inteligent, în care agentul este separat de interfață și joacă rolul unui mijlocitor între calculator și utilizator, asistându-l pe acesta din urmă în dialogul cu sistemul. Ca exemple de astfel de abordări pot fi date *HOMR*, un sistem dezvoltat la MIT, funcționând pe *paradigma filtrării colaborative automate* (*ACF - Automated Collaborative Filtering*) sau *SIFT*, implementat la Universitatea Stanford.

**Agent de achiziție de informații.** Deosebirea de tipul considerat mai înainte constă din fap-

tul că agentul de interfață are rol în modificarea stării sistemului. Fiecare sursă separată de informație va putea fi modelată de alți agenți și agentul de interfață va interacționa cu aceștia pentru schimbul de date. Agentul de interfață, primind fragmente de informație din mai multe surse va fi capabil să le organizeze în mod inteligent și să le prezinte utilizatorului într-o formă unitară. Acest aspect este deosebit de interesant într-un mediu distribuit pe arie largă precum Web-ul. O astfel de abordare este proiectul *KIMSAC* constând dintr-un număr de chioșcuri multimedia exploatate de agenți de interfață.

Posibile utilizări ar fi cele din domeniul educației asistate de calculator, învățământ virtual, diverse tipuri de simulări etc.

**Asistenți digitali personali.** Această clasă de agenți (*PDA - Personal Digital Assistant*) probabil este cea mai reprezentativă clasă de agenți de interfață (amintim doar „vrăjitorii” din suitele de birou integrate de tip *Office*).

Ideal, un asemenea agent posedă cunoștințe despre capabilitățile sistemului și despre abilitățile și preferințele utilizatorilor. Agentul va oferi ajutor și/sau sfaturi pe baza acestor cunoștințe în funcție de activitățile întreprinse de utilizator. Modelul formal al acestor sisteme se bazează pe rețelele Bayesian exploatate de grupul de cercetare de la Microsoft și de Adaptive System Group. Acțiunile utilizatorului pot fi modelate prin intermediul unui graf aciclic (rețeaua Bayesian) care va reprezenta o serie de distribuții probabilistice ale comenzilor asupra interfeței. Folosind algoritmi specifici de învățare, putem antrena rețeaua să realizeze predicții ale răspunsurilor utilizatorului în cazul unei anumite probleme.

Un proiect referitor la proiectarea asistenților digitali personali este *Lumiere*, inițiat încă din anul 1993 de către Microsoft pentru creșterea potențialului interfeței cu utilizatorul. Partile vizibile ale acestui proiect sunt *Office Assistant* și *Microsoft Agent*.

Un exemplu de sistem interactiv multimedia bazat pe animație și voce, folosind arhitectura deschisă a agenților de interfață este *InfoWiz*, compus din diverse module extensibile precum: animatorul interfeței, sistemul de recunoaștere a vocii, controlorul Web, managerul de dialog în limbaj artificial, gestionarul de cunoștințe.

## Sisteme multi-agent

Cercetările asupra sistemelor compuse din mai mulți agenți au fost întreprinse din perspectiva inteligenței artificiale distribuite și au fost divizate în două domenii: *rezolvarea problemelor în context distribuit* și *sistemele multi-agent*.

Prima direcție ia în considerație rezolvarea unei probleme particulare de un număr de module (noduri) care cooperează cu scopul de a găsi soluția sau soluțiile acelei probleme.

Toate strategiile de interacțiune sunt încorporate ca parte integrantă a sistemului.

Prin contrast, cercetările în domeniul sistemelor multi-agent se concentrează asupra comportamentului unei colecții de agenți autonomi care încearcă să rezolve problema. Un sistem multi-agent poate fi definit ca o rețea slab conectată de entități care lucrează împreună la rezolvarea unei probleme care nu poate fi soluționată în mod individual. Asemenea entități - agenții - sunt autonome și pot fi eterogene.

Caracteristicile sistemelor multi-agent sunt următoarele:

- fiecare agent posedă informații sau capabilități incomplete;
- nu există un control global al sistemului;
- datele sunt descentralizate;
- calculul se desfășoară în manieră asincronă.

O parte dintre problemele cheie ale proiectării și implementării unor astfel de sisteme pot fi enumerate mai jos:

- Cum trebuie formulate, descrise, descompuse și alocate problemele care se doresc a fi rezolvate și cum se sintetizează rezultatele oferite de grupul de agenți inteligenți?
- Cum și când vor comunica și vor interacționa agenții?
- Cum se va asigura faptul ca agenții să fie coerenti în luarea deciziilor și execuția acțiunilor?
- Cum vor reprezenta și vor manipula agenții acțiunile, planurile și cunoștințele?
- Cum se vor detecta și reconcilia intențiile conflictuale și punctele de vedere diferite asupra problemei de rezolvat între agenți?
- Cum se va organiza alocarea resurselor limitate?
- Cum se vor proiecta și construi sisteme multi-agent reale? Care vor fi metodologiile și tehnologiile care vor fi folosite?

**Actori.** Unul dintre primele modele al rezolvării problemelor cu ajutorul multi-agenților este cel bazat pe *actori*. Actorii pot fi priviți ca primitive universale pentru efectuarea calculului concurrent, componente autonome interacționând prin intermediul schimbului asincron de mesaje.

Primitivele de bază sunt următoarele:

- create va crea un actor pe baza unei descrieri de comportament și a unui set de parametri (incluzând eventual și alți actori deja existenți);
- send va trimite un mesaj unui actor;
- become va schimba starea locală a unui actor.

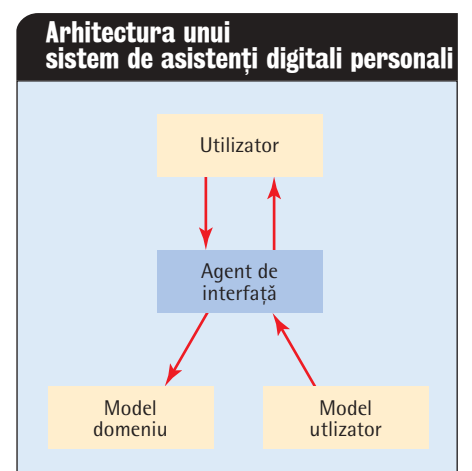
Modelul actor poate fi extins și la domeniul sistemelor deschise.

**Alocarea activităților.** Un alt aspect legat de rezolvarea problemelor în context distribuit este cel al *protocolului de contract în rețea*, în acest protocol agenții putând să joace, în mod dinamic, două roluri: de *manageri* și de *contractori*.

Data fiind o activitate de rezolvat, un agent denumit contractor mai întâi va deter-

mina dacă poate fi divizată în sub-activități (conform metodei *divide et impera*) care vor fi soluționate de alte noduri (gazde) de rețea. Un nod care recepționează un anunț de rezolvare a unei activități va replica cu o descriere a cât de bine poate îndeplini acea activitate. Contractorul va colecta toate datele primite și va decide care nod va executa activitatea cel mai bine.

Protocolul asigură alocarea dinamică a sarcinilor, permițând agenților să-și ofere serviciile, oferind un balans natural al încărcării sistemului (agenții ocupați nu vor putea să propună contractorului să le furnizeze spre executare alte activități).



## Interacțiuni între multi-agenți cooperativi.

În prezent, interesul pentru aplicații utilizând agenți cooperând la realizarea unui scop comun a crescut foarte mult. Se pot da ca exemple de astfel de aplicații integrarea informațiilor pe Internet, sistemele de divertisment interactiv sau antrenarea virtuală într-un anumit domeniu.

Pentru rezolvarea problemelor de interdependență între agenți care cooperează s-au propus diverse modele dintre care poate fi menționat mai ales *modelul FA/C (Functionally Accurate Model)*. Agenții nu au nevoie de toate informațiile la nivel local pentru a-și rezolva activitățile, ci pot interacționa prin intermediul co-rutinelor asincrone în vederea schimbului de rezultate parțiale. Pe baza acestui model, au fost dezvoltate o serie de scheme de control pentru coordonarea agenților, ca utilizarea unui meta-nivel static de specificare a organizării structurale a agenților sau a unui meta-nivel dinamic de informații. Interacțiunile dintre agenți pot lua forma unor planuri și scopuri de comunicare la un anumit nivel de abstractizare. Agenții au, de asemenea, diverse cunoștințe comune referitoare la cum și când se vor utiliza aceste informații pe care le dețin. Un cadrul de dezvoltare în timp-real modelat pe aceasta abordare este *Taems*.

Desigur, există și alte soluții precum negocierea datelor inspirată din societatea umană (sistemul *Persuader*) sau *modelul Bazaar* care include capacități de învățare a multi-agenților în locul interacțiunilor.

### Dezvoltare și implementare

Designerilor de aplicații multi-agent li se pun la dispoziție diverse shell-uri și medii de dezvoltare precum Agent Builder (Java), DECAF (Java), AgenTalk (Lisp), KnowBot (Python), UMPR (C++) sau Multi-Agent System Tools (C++).

Ca aplicații importante se pot aminti cele *industriale* (controlul producerii produselor, controlul proceselor, telecomunicațiile, controlul traficului aerian, sistemele de transport), cele *comerciale* (managementul informațiilor, comerțul electronic, managementul proceselor bancare), cele de *divertisment* (jocuri, cinematograful și teatru interactiv), sau cele *medicale* (monitorizarea pacienților, supravegherea sănătății).

*Sabin-Corneliu Buraga este doctorand în Computer Science la Universitatea „A.I. Cuza” din Iași, putând fi contactat prin e-mail la adresa busaco@infoiasi.ro și pe Web la <http://www.infoiasi.ro/~busaco>. ■ 98*

### Referințe bibliografice

- B.Alwyn - „An Overview of an Agent-Oriented Language for Concurrent and Distributed Programming”, ROSYCS'96 Proceedings, Iasi, 1996
- M.Barbuceanu, M.Fox - „Integrating Communicative Action, Conversations and Decision Theory to Coordinate Agents”, Agents'97 Proceedings, ACM, 1997
- M.Barbuceanu, R.Teigen, M.Fox - „COOL - un limbaj de coordonare a agenților”, PC-Report, vol7/1 (64), ian.1998
- K.Bharat, L.Cardelli - „Visual Obliq - Distributed Applications In A Hypermedia Setting”, CHI'95 Video Proceedings, 1995
- J.Bradshaw - „Software Agents”, AAAI Press, 1997
- A.Cheyer, L.Julia - „InfoWiz: An Animated Voice Interactive Information System”, SRI International Artificial Intelligence Center, Menlo Park, 1999
- A.Florea - „Sisteme multi-agent inteligente”, PC-Report, vol7/1 (64), ian.1998
- S.Green, F.Somers - „Software Agents: A Review”:  
[http://www.cs.tcd.ie/research\\_groups/aig/iag/iag.html](http://www.cs.tcd.ie/research_groups/aig/iag/iag.html)
- K.Höök et al. - „Edited Adaptive Hypermedia: Combining Human and Machine Intelligence to Achieve Filtered Information”, Flexible Hypertext Workshop, Hypertext'97 Proceedings, ACM, 1997
- E.Horvitz - „Lumiere Project: Bayesian Reasoning for Automated Assistance”, Decision Theory & Adaptive Systems Group, Microsoft Research, Microsoft Corp. Redmond, WA: 1998:  
<http://research.microsoft.com/research/dtg/horvitz/1um.htm>
- N.Jennings et al. - „A Roadmap of Agent Research and Development”, Autonomous Agents and Multi-Agent Systems, no.1, Kluwer Academic, Boston, 1998
- P.Maes - „Agents that Reduce Work and Information Overload”, Communications of the ACM 37(3), 1994
- D.Martin et al. - „The Open Agent Architecture”, Applied Artificial Intelligence Journal, vol.13, ian.-mart.1999
- C.Petrie - „Agent-Based Engineering, the Web and Intelligence”, IEEE Expert, dec.1996
- D.Weerasooriya et al. - „Design of a Concurrent Agent-Oriented Language”, in „Intelligent Agents: Theories, Architectures and Languages”, LNAI890, Springer Verlag, Amsterdam, 1995
- M.Wooldridge, N.Jennings - „Intelligent Agents: Theory and Practice”, Knowledge Engineering Review Journal, 10(2), 1995
- \*\*\* - „AgentWeb”: <http://www.cs.umbc.edu/agents>
- \*\*\* - „Microsoft Research”: <http://research.microsoft.com>
- \*\*\* - „Bibliography of Learning in Multi-Agent Systems”:  
<http://www7.informatik.tu-muenchen.de/~weissg/bib-mal.html>