Hiperspațiu AOM

Cristina Feier, Dorian Gorgan

Facultatea de Automatică si Calculatoare, Universitatea Tehnică din Cluj-Napoca
E-mail: cristina@cs-gw.utcluj.ro, Dorian.Gorgan@cs.utcluj.ro

Rezumat. În această lucrare se analizează posibilitățile de extindere ale modelului AOM, în sensul definirii conceptului de hiperspațiu AOM. Noul concept este definit prin analogie cu conceptele omonime folosite în alte domenii. Două sensuri ale acestui termen sunt larg folosite în literatura de specialitate, unul referindu-se la spații n-dimensionale, iar celălalt la WWW. Definirea noului concept prin analogie cu WWW se dovedeşte mai fructuoasă, *hiperspațiul* AOM văzut ca si colecție de *hipermodele* AOM interconectate, oferind o mult mai mare expresivitate pentru dezvoltarea aplicațiilor.

Cuvinte cheie: AOM, hiperspațiu, hipermodel**.**

1. INTRODUCERE

Modelul de Obiecte Active (AOM)[1,3,4,5] este un sistem care oferă suport pentru modelarea structurilor bazate pe entități active, dezvoltarea și investigarea tehnicilor de programare vizuale, navigarea în spațiul virtual, utilizatorul având posibilitatea de control atât asupra interfeței cât şi asupra entităților aplicație. Pornind de la termenul de hiperspațiu, în această lucrare investigăm modul în care conceptele denotate de acest termen pot fi definite în contextul AOM.

Sensurile cele mai întâlnite ale termenului sunt cel de spațiu n-dimensional, respectiv de rețea de noduri hipertext [6,7,8,10]. Dacă în ceea ce priveşte primul sens, acesta poate fi aplicat direct în contextul AOM, spațiul caracteristicilor unui model fiind un spațiu n-dimensional, rețeaua bazată pe hipertext va fi folosită ca o metaforă pentru definirea hiperspațiului AOM, unde termenul de metaforă este folosit în sensul definit de Lakoff [9] şi anume "un mod de gândire care poate fi înțeles ca o mapare de la un domeniu sursă la un domeniu țintă; domeniul sursă este o schemă conceptuală pe care am construit-o bazat pe experiențele noastre într-un mediu particular." De fapt maparea se va realiza la nivelul nodurilor, de la hipertext la hipermodel (secțiunea 3). Trei moduri diferite pentru definirea noțiunii de legătură sunt prezentate în secțiunile 3.1, 3.2, 3.3. De asemenea se introduce conceptul de model AOM parametrizat, a cărui execuție depinde de modul în care sunt instanțiați parametrii săi de intrare. În secțiunea 3.7 se prezintă un exemplu care foloseşte construcțiile introduse anterior, evidențiind utilitatea acestora.

Acest mod de a introduce noi noțiuni sau a explica noțiuni existente prin analogie cu alte noțiuni este un procedeu larg folosit in ştiință, modul în care este conceptualizat WWW-ul în prezent fiind bazat pe metafora spațială [2,8].

2. MODELUL DE OBIECTE ACTIVE

Acest model a fost dezvoltat la Universitatea Tehnică din Cluj-Napoca ca un cadru pentru dezvoltarea de aplicaţii multimedia complexe, medii de programare vizuale, interfeţe grafice inteligente, modelarea realităţii virtuale şi crearea de aplicaţii didactice. Două tipuri de abstracţii sunt folosite pentru construirea unui model: *entităţi active* şi *entităţi pasive*. *Obiectele active* fac parte din prima categorie şi reprezintă principala abstracţie a modelului deoarece influenţează direct evoluţia dinamică a acestuia. Din cea de-a doua categorie fac parte: *comportări*, *traiectorii*, *poziţii explicite de traiectorie (ETP)*, *reguli*, *acţiuni*, *prezentări*, *interactori*. Acestea joacă rolul unor resurse globale ce sunt partajate între entităţile active.

Starea unui obiect activ este dată de numele obiectului, poziţia în spaţiul abstract, un indicator de stare (una din stările play, stopped, waiting, paused), un indicator de prezentare (obiectul este vizibil/invizibil) şi o mulţime de atribute.

Obiectele active au o evoluţie descrisă de un comportament bazat pe reguli. Un obiect evoluează de-a lungul unei traiectorii într-un spaţiu, executând o secvenţă definită de reguli în anumite puncte de pe traiectorie (ETP). O regulă constă dintr-o condiţie şi o mulţime de acţiuni. O mulţime de parametrii caracterizează comportamentul unui obiect pe lângă traiectorie, poziţie şi nume.

Aspectul grafic al unui obiect este dat de prezentarea asociată acestuia. Există trei tipuri de prezentări: audio, video şi grafice.

O entitate membru a unui obiect activ care descrie interacţiunea utilizatorului cu obiectul activ şi metodele de acces la sistemul de fişiere este interactorul. Fiecare interactor are definită o listă de evenimente mouse, tastatură şi/sau ceas. Un set de reguli descriu evoluţia obiectului în cazul apariţiei evenimentului. Zonele în care aceste evenimente au semnificaţie pentru obiect se numesc zone active.

Valorile diferitelor variabile dintr-un model pot fi legate folosind mecanismele `bound_from` şi `bound_to`. Aceste mecanisme permit propagarea rapidă a schimbărilor şi sunt folosite în conjuncţie cu atributele obiectelor active, parametrii comportărilor şi poziţiile entităţilor. Expresia `bound_to` specifică elementele din model care trebuie modificate când se modifică elementul curent. În cazul expresiei `bound_from`, de câte ori este cerută valoarea elementului curent, aceasta este înlocuită de valoarea obţinută prin evaluarea argumentului expresiei.

Fiecare obiect al unui model cât şi modelul însuşi are ataşate patru procesoare ce asigură execuţia acţiunilor, execuţia comportării obiectelor, prezentarea acestora şi tratarea evenimentelor : procesoarele server, comportament, prezentare, interactor.

3. HIPERSPAŢIU AOM

Pornind de la metafora hiperspaţiului, ca şi o reţea de entităţi hipertext, se analizează modul în care se poate defini hiperspaţiul AOM, ca si reţea de

hipermodele AOM. Principala abstracție a hipertextului, noțiunea de legătură, va fi definită în cadrul modelelor AOM. Un model AOM ce implementează această abstracție se va numi *hipermodel*. Există mai multe moduri în care funcționalitatea modelelor AOM poate fi extinsă în acest sens (secțiunile 3.1, 3.2, 3.3). În secțiunea 3.5 se analizează modul de explorare al unei legături. Un câştig în expresivitatea modelului este adus de introducerea modelelor AOM parametrizate (secțiunea 3.6). Un exemplu este prezentat în final care îmbină toate aceste concepte.

3.1 Acțiuni pentru explorare legături

O modalitate de referire a altor modele este existența unor acțiuni a căror execuție să consiste în explorarea unei legături. Reguli ataşate ETP specifică condițiile în care astfel de acțiuni pot fi executate. O astfel de acțiune va avea tipul generic *link:*

```
<action> ::= <link_act> | <create_act> | ...

<link_act> ::= link <link_name>|link <link_name>
{<link_block>}

<link_block> ::= <position>; <link_address>;
```

Implementare:

Acțiunile de acest tip vor fi executate de către procesorul server al modelului, care este responsabil de localizarea şi lansarea în execuție a noului model, precum şi de controlul execuției propriului model după explorarea legăturii (v. secțiunea 3.4).

Comentariu asupra acestei soluții:

În acest caz utilizatorul nu are control direct asupra legăturilor ce vor fi explorate, această informație fiind captată de model. Totuşi acesta poate influența alegerea unui legături prin modificarea parametrilor care sunt evaluați în precondițiile regulei. Aceste legături pot fi folosite pentru obținerea de informații mai detaliate despre un sistem, permițând explorarea altor modele ce reprezinta evoluția sistemului dintr-o altă perspectivă în situații de criză (situații ce pot fi codificate în precondițiile unei reguli de tipul descris în această secțiune) (v. secțiunea 3.7).

3.2 Legături ataşate pozițiilor explicite de traiectorie

O altă posibilitate este *ataşarea de legături spre alte modele pozițiilor explicite de pe traiectoria unui obiect* AOM. Când un obiect atinge o astfel de poziție ce are ataşată o legătură, în funcție de valoarea unui *fanion* setat de utilizator, se va explora sau nu legătura respectivă. Fanionul poate fi setat în orice moment al execuției. Acesta poate fi definit la *nivel de model*, la *nivel de obiect* sau la *nivel de ETP*. Flexibilitatea cea mai mare se obține prin ataşarea unui fanion fiecărei ETP. Inițial toate fanioanele au valoarea `false`.

În cazul în care vrea să marcheze o legătură ca pasibilă de explorare, utilizatorul face clic pe un simbol grafic ataşat acesteia. Repetarea acţiunii duce la dezactivarea legăturii. Starea legăturii poate fi dedusă din modul de afişare al simbolului grafic: dacă este inactivă, imaginea afişată va fi mai ştearsă. Simbolul grafic trebuie să conţină informaţie semantică despre modelul referit. Prin poziţionarea cursorului mouse-ului deasupra imaginii, utilizatorul poate vedea adresa modelului referit.

Legăturile sunt vizibile şi pot fi activate doar dacă (atâta timp cât) sunt plasate pe traiectoria unor obiecte vizibile. Presupunând că utilizatorul a activat o legătură, după care obiectul corespunzător a devenit invizibil şi a ajuns la poziţia de pe traiectorie căreia îi corespunde legătura activă, aceasta este ignorată. Astfel, o condiţie necesară şi suficientă pentru explorarea unei legături este ca atât fanionul de vizibilitate al obiectului cât şi fanionul ataşat legăturii să fie setate pe `true`.

Pentru introducerea noului tip de legătură, un nou element `etp_link` este definit ca membru al unei ETP, ce are la rândul său următoarele subcomponente: adresa modelului referit, simbolul grafic ataşat legăturii, reprezentat aici printr-o zonă activă, şi un fanion ce indică dacă legătura este sau nu activă:

```
<etp_link> ::= etp_link <link_name>|etp_link
<link_name> {<etp_link_block>}

<etp_link_block> ::=
<position><link_address><link_symbol><link_flag>
```

Implementare

Acest tip de legătură poate fi modelat folosind construcţiile existente în limbajul AOML, inclusiv cel de acţiune de tipul `link` definit în secţiunea anterioară. Totuşi, efortul de editare al unei astfel de legături folosind construcţiile existente ar fi destul de mare. De aceea, la editare, pentru a include o legătură, se defineşte doar structura `etp_link`. Există o fază intermediară între faza de editare şi faza de execuţie a unui model, fază în care definiţiile `etp_link` din cadrul unui model sunt expandate în structuri cu funcţionalitate echivalentă descrise în AOML extins.

Pentru fiecare legătură de acest tip, este definit un interactor ce are ca zonă activă simbolul grafic asociat acesteia, interactor care tratează evenimentele de apăsare a butonului stânga al mouse-ului, respectiv de mişcare a mouse-ului. Apăsarea butonului stânga al mouse-ului când acesta se află poziţionat în interiorul zonei active duce la setarea sau resetarea fanionului de activare.

Pentru afişarea adresei legăturii un nou obiect activ `o1_aux` este introdus, a cărui traiectorie are o singură `ETP`, iniţial invizibil şi a cărui prezentare este mesajul ce conţine adresa legăturii. Controlul afişării acestei informaţii se face în secvenţa de tratare a evenimentului de mişcare mouse. Atâta timp cât cursorul mouse-ului se afla deasupra zonei active obiectul auxiliar este vizibil, în momentul

în care mouse-ul se deplasează în afara acestei regiuni, obiectul devenind invizibil. Poziția acestui obiect şi a componentelor sale este fixă, în sensul că toate obiectele auxiliare definite pentru diferite legături vor avea aceeasi poziție, informația despre adresa legăturii fiind afişată într-o zonă predefinită.

Zona activă asociată legăturii are aceeaşi stare de vizibilitate ca şi obiectul activ, o legătură `bound_to` fiind specificată de la fanionul de vizibilitate al obiectului la fanioanele de vizibilitate ale eventualelor zone active asociate legăturilor.

Două reguli sunt incluse în cadrul unei ETP pentru fiecare legătură de tipul `etp_link` asociate acesteia: ele indică că atunci când fanionul de vizibilitate al zonei active şi fanionul asociat legăturii au valoarea `true` se va executa o acțiune de tipul `link`, după care se va reseta fanionul de activare al legăturii. A doua regulă este necesară pentru a nu se explora de mai multe ori acelaşi model.

Comentariu asupra acestei soluții:

Prin ataşarea unui fanion fiecărei ETP a fiecărui obiect ce are asociată o legătură, utilizatorul poate restrânge setul de legături ce pot fi explorate de model, dar nu are control total asupra ordinii de explorare a legăturilor sau a legăturii care va fi explorată. De asemenea, este posibil ca utilizatorul să fi marcat una sau mai multe legături ca active, dar nici una din ETP-urile asociate să nu fie atinsă de către un obiect. Totuşi, în cazul în care modelul conține obiecte cu comportament predictibil, această metodă dă rezultate deterministe.

Deşi oferă o flexibilitate mai mare, ataşarea unui fanion fiecărei ETP are şi dezavantaje: pentru obiecte invizibile, utilizatorul nu va putea niciodată să schimbe valoarea eventualelor fanioane ataşate ETP-urilor. Ataşarea unui fanion unic pe model ar înlătura această problemă, dar aceasta ar însemna mărirea incertitudinii asupra legăturii ce va fi explorată prima.

3.3 Hipermodel = model + 'hipertext'

În ambele abordări descrise anterior, utilizatorul nu poate controla (exact, in prima; deloc, în a doua) momentul în care se va explora o legătură. Aceasta deoarece noțiunea de legătură e asociată unei locații de pe traiectorie care trebuie atinsă. Legăturile ce pot se permită utilizatorului explorarea instantanee a unui nou model nu trebuie să fie dependente de atingerea unei anumite locații. Ca atare ele pot să fie plasate oriunde în model, singura cerință fiind ca apariția lor grafică (imagine, text evidențiat) să dea un indiciu despre modelul referit. Astfel de legături sunt similare cu legăturile HTML. Un nou element `<html_link>` este definit ca şi entitate a unui model.

```
<entity_decl> ::= <agent_decl> | <html_link> |...
<html_link> ::= html_link <link_name> | html_link
<link_name> {<html_link_block>}
```

```
<html_link_block> ::= <position>; <link_address>;
<link_symbol>;

<link_symbol> ::= <zona_activa>
```

Definiţia este similară cu cea pentru `etp_link`, exceptând fanionul care aici nu există. La nivelul funcţionalităţii, deosebirea este foarte mare între cele două tipuri de legături, în acest caz utilizatorul controlând atât locaţia cât şi momentul când doreşte ca o anumită legătură să fie explorată. Prin executarea unui clic de mouse asupra simbolului grafic asociat unui `html_link` se trece automat la explorarea legaturii.

Implementare:

Ca şi în cazul anterior, aceste legături pot fi implementate prin descrierea lor prin structuri cu funcţionalitate echivalentă, a căror implementare a fost deja definită. La editare se specifică doar structura `html_link`. Sistemul va genera automat şi include în model un obiect activ ce conţine un interactor, ce tratează evenimentele mouse de apăsare a butonului stânga şi mişcare cursor pentru fiecare zonă activă definită în cadrul unei legături `html_link`. Câte un obiect adiţional ce are ca şi prezentare textul ce conţine adresa legăturii va fi generat pentru fiecare legătură, la fel ca în secţiunea anterioară. Atunci când se apasă butonul stânga se execută o acţiune de tipul `link` descrisă anterior, iar atunci când cursorul este poziţionat deasupra unei zone active obiectul auxiliar devine vizibil.

Comentariu asupra acestei soluţii:

Această abordare poate fi privită ca fiind complementară cu cea menţionată în [1], unde se sugerează construirea de *plug-in*-uri AOM pentru navigatoarele de reţea. Prin metoda descrisă se poate considera că se construieşte un *plug-in* HTML (un subset de fapt - legături) pentru *view*-erul AOM.

3.4 Discuţie asupra modalităţilor de referenţiere

Din punct de vedere formal cele trei modalităţi de referenţiere discutate în secţiunile precedente pot să coexiste într-un model AOM. Trebuie analizat şi în ce măsură sunt utile toate cele trei abordări.

D.p.d.v. al uşurinţei de utilizare, ultimul tip, cel similar cu legăturile HTML, este cel mai intuitiv. Acest tip de legătură nu trebuie să lipsească din AOM. Prin folosirea lui se pot face referinţe la modele apropiate ca tematică cu modelul curent. Al doilea tip necesită resurse cognitive mult mai mari din partea utilizatorilor, aceştia trebuind să anticipeze dacă, în eventualitatea că o poziţie de traiectorie este atinsă, ar fi util să se urmeze legătura specificată în poziţia respectivă. Totuşi, aceste tipuri pot fi văzute ca o relaxare a `link_action`-urilor, cele din urmă executându-se indiferent de dorinţa utilizatorilor. Cum regulile de

acest ultim tip sunt utile (monitorizare, etc.) considerăm că şi tipul discutat anterior trebuie menținut, cele două tipuri putând fi folosite alternativ.

3.5 Modul de explorare a legaturilor

O dată ce s-a ales o legătură ce va fi explorată se pune problema *cum* se va explora legătura respectivă. În cazul hipertextului, se permite specificarea modului de vizualizare a unei noi pagini (într-o fereastră nouă, în aceeaşi fereastră, într-un anumit cadru, etc.), atât la momentul editării (HTML), cât şi la momentul navigării (navigator). Aceeaşi politică este utilă şi în cadrul hipermodelelor AOM.

Pentru legăturile similare cu legăturile HTML, atât AOML cât şi navigatorul AOM vor oferi posibilitatea specificării modului de explorare al legăturii. Dacă în cazul legăturilor `link_action` nu se pune problema specificării de către utilizator a acestui gen de informații, în ceea ce priveşte legăturile `etp_link` se pune problema dacă se lasă şi această sarcină utilizatorului. O nouă variabilă binară ar trebui introdusă pentru fiecare legătură ce ar trebui setată pentru a indica comportamentul dorit. Aceasta ar fi prea împovărător pentru utilizator, de aceea această informație este codificată în model. Un nou fanion `new_window` este introdus in definiția celor trei tipuri de legături.

Ce se întâmplă cu vechiul model atunci când este explorată o legătură definită în acesta? Continuă să se execute sau execuția lui se suspendă? Dacă noul model se execută într-o nouă fereastră, modelul inițial îşi continuă execuția. În caz contrar, dacă execuția sa se suprapune peste modelul inițial, execuția celui din urmă se suspendă. Analog, în momentul revenirii la un model anterior, execuția modelului curent se suspendă. O istorie a modelelor recent vizualizate este păstrată pentru a se putea reveni la modelele anterioare.

3.6 Modele AOM parametrizate

Limbajul HTML permite, prin intermediul formelor, transmiterea de informații de către un document HTML către un script; o noua pagină HTML poate fi generată dinamic pe baza informațiilor procesate de scriptul respectiv. Ar fi util să existe posibilitatea transmiterii de informație între modele AOM interconectate, execuția modelului referit să depindă de starea execuției modelului inițial. Se introduce noțiunea de model AOM parametrizat. Un astfel de model este caracterizat, pe lângă entitățile active şi pasive din componența sa, de o mulțime de parametrii de intrare ce trebuie legați la lansarea în execuție a modelului. Valori ale diferitelor variabile din model pot fi legate de aceşti parametrii de intrare.

Folosirea mecanismelor de legare `bound_to` şi `bound_from` pentru stabilirea unor relații între parametrii de intrare ai modelului referit şi atributele modelului inițial poate conduce la următoarele cazuri de inconsistență:

- dacă execuţia modelului referit se suprapune peste execuţia modelului iniţial şi există parametrii de intrare legaţi prin `bound_from` de variabile din modelul iniţial;
- dacă se revine la un model anterior şi există legături `bound_to` de la variabile din modelul respectiv la parametrii de intrare din modelul current;
- în cazul unui lanţ de modele interconectate ce se execută simultan în ferestre diferite, pot apărea inconsistenţe la sistarea execuţiei unui model;

Soluţia adoptată este aceea ca în cazul în care valoarea unei variabile dintr-un model cu execuţia suspendată sau terminată este cerută prin `bound_from` sau se încearcă prin `bound_to` actualizarea valorii unei variabile în aceeaşi stare, variabila care este legată prin `bound_from` să rămână la valoarea curentă, iar inexistenţa variabilei ce se vrea a fi actualizată să fie ignorată.

Fluxul de date între modele interconectate este acelaşi cu direcţia legăturii: o variabilă din modelul iniţial nu va fi legată cu `bound_from` de un parametru al modelului referit, la fel cum un parametru din modelul referit nu va fi legat prin `bound_to` de variabile din domeniul iniţial.

Pentru a putea specifica fluxul de date între două modele, un nou tip de argument este introdus pentru `link_act, etp_link,` şi `html_link`, şi anume o triplă ce conţine o variabilă din modelul curent, un parametru de intrare din modelul referit şi tipul legăturii dintre acestea. Tipul legăturii poate fi `bound_from`, `bound_to` sau `equal`. Implicit `bound_from` este de la parametrul de intrare la variabilă, iar `bound_to` de la o variabilă la un parametru de intrare. Cel de-al treilea tip, `equal`, presupune setarea parametrului de intrare la valoarea variabilei:

```
<link_param>::=<l_p_name>|<l_p_name>(<l_p_block>);
<l_p_block>::={<variable_name>;<input_name>;<bind_
type>;}
```

Pentru ca modelele parametrizate să poată fi executate şi atunci când nu sunt referite de alte modele se pot specifica valori default pentru parametrii de intrare. Mecanismul de transmitere a parametrilor de-a lungul legăturii, precum şi legăturile posibile între parametrii unui model şi variabilele din modelul respectiv sunt evidenţiate în Fig.1. Modelul 1 conţine o legătură spre modelul 2. Legătura specifică valorile parametrilor de intrare ale modelului 2 $i_1, i_2, ..., i_n$ în funcţie de variabilele modelului 1 $a_1, a_2, ..., a_m$. Variabilele modelului 2, $b_1, b_2, .., b_p$ pot fi definite ca funcţii de parametrii de intrare $i_1, i_2, ..., i_n$ prin conexiuni `bound_from` sau `bound_to` sau prin intermediul actiunilor `set`. Sensul săgeţilor ce evidenţiază relaţii(`b_to, b_from, equal, set`) este de la primul parametru al relaţiei la cel de-al doilea.

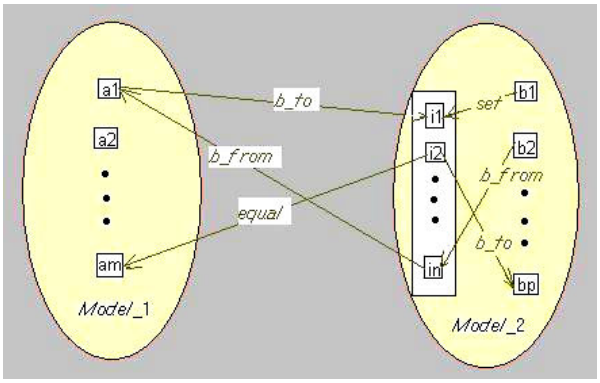

Fig1. Legătura spre un model parametrizat

3.7 Exemplu

Fie un model AOM a cărui execuție constă în reprezentarea evoluției unei pacient. Pentru aceasta, folosind un interactor ce tratează evenimentul de ceas, se monitorizează câțiva parametrii importanți pentru starea pacientului ca ritmul cardiac, tensiunea, glicemia, etc. Pe baza parametrilor monitorizați, se calculează un parametru abstract ce indică "stabilitatea" pacientului.

Modelul reprezintă evoluția pacientului prin prisma parametrului abstract, şi anume ca şi grafic al acestuia în funcție de timp: la fiecare moment de timp se crează un obiect activ cu o singură ETP ce are ca şi coordonate timpul, respectiv valoarea parametrului abstract la momentul respectiv; prezentarea obiectului este un punct de pe grafic, a cărui culoare indică dacă parametrul abstract se încadrează în limitele normale sau nu.

În cadrul procedurii de tratare a evenimentului de ceas există o regulă care specifică ca atunci când s-a depăşit valoarea normală a parametrului abstract să se exploreze un nou model, într-o nouă fereastră, model ce descrie evoluția parametrului concret care a contribuit cel mai mult la această depăşire. Modelul referit este un model parametrizat, având un parametru de intrare ce este legat de parametrul concret care a deviat cel mai mult de la limitele normale printr-o legătură `bound_to`. Există trei modele diferite, câte unul pentru fiecare parametru, ce pot fi referite de modelul inițial. Un fanion este setat la explorarea unui model pentru a evita reexplorarea acestuia.

Noul model conține un obiect activ vizibil cu trei ETP-uri. Fiecare ETP corespunde unei stări posibile a valorii parametrului: low, normal, high. Obiectul va evolua pe traiectorie în funcție de intervalul în care se încadrează valoarea unicului parametru de intrare. Câte un `etp_link` şi câte o zonă activă se defineşte pentru fiecare poziție de pe traiectorie. Legăturile respective trimit spre modele care dau o descriere mai detaliată a stării respective, de exemplu un model

care prezintă informații despre ce înseamnă glicemie mare. Utilizatorul poate activa una sau mai multe din legăturile respective, iar când obiectul va ajunge în una din ETP-urile asociate, legătura va fi explorată şi automat dezactivată.

În acest exemplu s-au folosit primele două din cele trei tipuri de legături introduse în aceasta lucrare. De asemenea s-a utilizat şi conceptul de model parametrizat, cele trei modele referite de modelul inițial având fiecare câte un parametru de intrare.

4. CONCLUZII

În această lucrare pornind de la metafora hiperspațiului, ca şi o rețea de entități hipertext, s-a definit hiperspațiul AOM, ca şi rețea de *hipermodele* AOM. Trei tipuri de legături între modele au fost definite, în funcție de rolul pe care îl are utilizatorul în explorarea acestor legături. Pentru a oferi posibilitatea transmiterii de informație între modele de-a lungul legăturii s-a introdus noțiunea de model AOM parametrizat. Aşa cum reflectă şi exemplul prezentat aceste construcții sunt utile, oferind posibilitatea reprezentării unor modele cu comportamente complexe.

Noile concepte introduse pot sta la baza creării unor biblioteci de modele AOM. Un exemplu ar fi modele ce permit reprezentarea grafică a unei caracteristici în funcție de alta, sau a unei caracteristici în funcție de timp. Legături spre diferite instanțe ale acestora pot să fie plasate într-un model pentru a reflecta diferite aspecte ale hiperspațiului caracteristicilor modelului. Pentru a permite refolosirea, modelele trebuie să fie parametrizate.

Referințe

1. *AOM- Web,* http://users.utcluj.ro/~gorgan/res/aom/aom.html
2. Boechler, P. M. *How Spatial is HyperSpace? Interacting with Hypertext Documents: Cognitive Processes and Concepts*, CyberPsychology&Behavior, 1(4), p. 23-46, 2001.
3. Gorgan, D. and Duce, D. A. *Fuzzy Learning in Multi-Agents Based Interactive Systems*, Rutherford Appleton Laboratory, Research Report, January, 1997, pp.1-43.
4. Gorgan, D. and Duce, D. A. *Multimedia SynchronizationThrough Interactive Active Objects*, Proceedings of the EUROGRAPHICS'97 UK Conference, pp. 131-155, Norwich, East Anglia University, UK. March 1997.
5. Gorgan, D. and Duce, D. A. *The Notion of Trajectory in Graphical User Interfaces*, Research Report, Rutherford Appleton Laboratory, January 1997
6. HTML 4.0 Specification, http://www.w3.org/TR/1998/REC-html40-19980424/
7. *Hyper Hypertext,* http://www.labyrinth.net.au/~saul/essays/04hyper.html
8. *Hypertext Paradigm,* http://www.faced.ufba.br/~edc708/biblioteca/interadividade/web paradigma
9. Lakoff, G: *What is metaphor?* In: Barnden, J.A, Holyoak, K.J., (eds.) Advances in connectionist and neural computation theory: Analogy, metaphor and reminding. New Jersey: Ablex, 1994
10. Pellegrini, U. *Hypertext: The New Paradigm For Electronic Writing and Reading,* http/users.unimi.it/metis/METIS-3MKB/articoli/hypertext.htm