Revista Română de Interacțiune Om-Calculator 1 (2008), 163-186

Controlul accesului în aplicații de comerț electronic utilizând maşini de stare

Mihaela Ordean[1], Dorian Gorgan[2]

[1]iQuest Technologies SRL Cluj-Napoca
Calea Motilor nr 6-8, 400001, Cluj-Napoca
E-mail: mihaela.ordean@iquestint.com

[2]Universitatea Tehnica Cluj-Napoca
Gh. Baritiu nr. 26-28, Cluj-Napoca
E-mail: dorian.gorgan@cs.utcluj.ro

Rezumat. Lucrarea abordează un domeniu particular al autorizării şi propune modelul SCAR-ACE pentru controlul accesului bazat pe roluri în aplicații de comerț electronic având la bază maşini de stare. Aplicațiile de comerț electronic devin din ce în ce mai complexe, impunând accesul la resurse eterogene a utilizatorilor cu roluri diferite. Controlul accesului în aplicațiile de comerț electronic este un subiect important al cercetării ştiințifice actuale. Lucrarea de față îşi propune realizarea unui model sigur de control al accesului bazat pe roluri care să abordeze dintr-o perspectivă diferită controlul accesului fără a utiliza cookies. Modelul propus permite doar utilizatorilor autorizați să acceseze resursele sistem, printr-un control bazat pe ierarhii de roluri şi printr-o gestionare a rolurilor mutual exclusive. Pentru a controla într-o aplicație distribuită fluența şi accesul la resurse se introduce noțiunea de rol ca un element intermediar între utilizator şi setul de permisiuni ale acestuia. Fiecărui rol i se ataşează un set de permisiuni (regăsite în unele accepțiuni sub denumirea de privilegii), de acces la operații şi resurse. Modelul este validat prin realizarea de teste şi obținerea de rezultate experimentale.

Cuvinte cheie: controlul accesului, autorizare, maşini de stare.

1. Introducere

Controlul accesului are drept obiectiv protejarea resurselor față de accesul neautorizat şi se realizează prin acordarea de permisiuni. Un sistem de control al accesului implică o politică prin care se specifică cine poate accesa o anumită resursă şi în ce manieră o poate face. Politicile sunt în general exprimate prin starea curentă a sistemului şi stările care pot rezulta

din aceasta.

În Internetul de astăzi există un număr din ce in ce mai mare de aplicații care necesită decizii de autorizare. Aceste aplicații includ (dar nu sunt limitate la) comerțul electronic, gestionarea și partajarea resurselor distribuite, execuția și descărcarea de cod de la distanță. Autorizarea acestor aplicații este semnificativ diferită de cea a sistemelor centralizate și chiar de cea a sistemelor distribuite relativ mici. În sistemele de autorizare pe Internet există mai multe entități, multe dintre acestea necunoscându-se una pe cealaltă și, de multe ori, neexistând o autoritate centrală de încredere pentru toți actorii. Regulile conform cărora se realizează deciziile de acordare a controlului pot necesita modificări ale acestor aplicații, deciziile pentru permiterea accesului la o resursă specifică depinzând de tipul aplicației.

Un sistem de control al accesului este totodată și o instanță a unei scheme de control al accesului și specifică tipurile de reguli relativ la tranzițiile între stări (Tripunitara și Li, 2004). Un exemplu de model pentru controlul accesului este modelul bazat pe matrici de acces (Graham și P.J. Denning, 1975). Matricea de acces este un model conceptual care specifică drepturile pe care le are fiecare utilizator asupra fiecărui obiect. Acest model se utilizează în special in sistemele de operare (Tolone et all, 2005).

2. Starea actuală a cercetării în controlul accesului

Încă din 1960 controlul accesului a fost un subiect de interes în special în managementul bazelor de date și în sistemele de operare. Obiectivul acestuia era, pe de o parte, cel de a proteja resursele sistemului față de accesul neautorizat si, pe de altă parte, de a permite accesul autorizat.

Primele modele în domeniul controlului accesului au fost: Mandatory Access Control (MAC) și Discretionary Access Control (DAC).

MAC a fost introdus in domeniile militar și guvernamental, realizând controlul accesului prin introducerea de etichete de securitate. Acest model, formalizat inițial de Bell și LaPadula (1975), atașează etichete sau clasificări de securitate fiecărui obiect. Deoarece diferitele forme ale modelului Bell-LaPadula, determinate de multe variațiuni, au dus la confuzii, Sandhu introduce un model minimal (1993) numit BLP care încapsulează părțile esențiale ale modelului Bell-LaPadula.

Modelele MAC sunt destul de rigide. Acestea suportă un set fix de clase de securitate şi permit doar administratorilor de sistem să modifice politica de securitate.

DAC îşi are originile în aria academică (B. Lampson, 1971). El permite sau restricționează accesul la un obiect pe baza identității utilizatorului care îl accesează. Utilizatorilor le este permis accesul prin permisiuni asupra obiectelor proprii. Totodată utilizatorilor le este permis să delege drepturile proprii altor utilizatori. Un exemplu clasic de DAC este Access Control List (ACL) în care obiectele sunt asociate la o listă de utilizatori (care formeaza grupuri) cărora le este permis accesul.

Aceste abordări au avut atât avantaje cât şi dezavantaje. Problemele apar de obicei la inserarea de utilizatori, la ştergerea anumitor utilizatori existenți sau la crearea şi ştergerea obiectelor.

Una dintre primele implementări în domeniul autorizării în sisteme distribuite este Grapevine (Birrell et all, 1982) in care serverele determină dacă un client este membru al unui grup autorizat. O abordare similară se regăseşte în Sun Yellow Pages unde fişierele gestionate centralizat (precum *etc/group*) sunt consultate prin autorizarea utilizatorilor. În ambele cazuri, cu toate că deciziile de autorizare se realizează local, clientul trebuie să obțină datele de autorizare de la un server aflat la distanță.

Un alt model de autorizare pentru sistemele distribuite este Kerberos (Miller şi Neuman, 1988). Acest model se bazează pe principiul că fiecare utilizator este autorizat pentru un numar de servicii şi fiecare serviciu îşi gestionează utilizatorii propri.

Proiectul SESAME (McMahon, 1995), (Parker şi Pinkas PP95) extinde sistemul de autorizare Kerberos şi defineşte o schemă pentru propagarea sigură a privilegiilor, inclusiv roluri şi grupuri, de la clienți la serverele de aplicație.

Alte sisteme propuse oferă soluții de autorizare off-line, precum delegarea certificatelor în Arhitectura de Securitate a Sistemelor Digitale Distribuite (Gasser et all., 1989), (Gasser şi McDermott, 1990), mecanismul de autentificare în cascadă a lui Sollins (1988), autorizarea proxy-based a lui Neuman (1993) şi autorizarea bazată pe certificate a lui Woo and Lam (1998).

Controlul accesului bazat pe roluri se regaseste in literatura de specialitate sub denumirea de RBAC (Role-Based Access Control).

In (Barkley, 1995) se descrie o implementare RBAC într-un limbaj de programare obiectual. Fiecare rol este reprezentat printr-o clasă diferită. La rulare, obiectele rol servesc ca şi proxy-uri pentru aplicațiile care cer accesul la anumite permisiuni. Abordarea din (Barkley, 1995) nu ia în considerare ierarhiile de roluri şi constrângerile pe care modelul SCAR-ACE le include şi le implementează.

Sistemul hyperDRIVE (Bartz, 1997) utilizează standardul LDAP pentru a implementa servicii de autentificare şi control al accesului. Sistemul face uz de tehnologia Java Applet. Abordarea SCAR-ACE diferă în concept şi implementează rolurile şi privilegiile asociate acestora prin introducerea maşinilor de stare.

Sistemul I-RBAC (Tari şi Chan, 1997] implementeaza controlul accesului în Intranet. O caracteristică a I-RBAC este utilizarea paralelă a ierarhiilor de roluri locale şi globale. I-RBAC nu permite definirea rolurilor mutual exclusive. Spre deosebire de această implementare, SCAR-ACE este un model pentru aplicații pe Internet care acceptă roluri globale şi permite existența rolurilor mutual exclusive.

In (Giuri, 1999) se descrie un mecanism RBAC bazat pe servleți Java. Se propune o arhitectură denumită JRBAC-WEB care se bazează pe execuția cererilor HTTP utilizându-se un servlet Java de securitate. Implementarea suportă definirea ierarhiilor de roluri şi constrângerilor de separare a îndatoririlor. JRBAC-WEB foloseşte autentificarea HTTP. In mod opțional abordarea permite păstrarea sesiunilor de lucru prin utilizarea cookies sau prin tehnici de rescriere a URL-ului pentru servleți Java. SCAR-ACE nu utilizează servleți Java (ci credențiale) şi nici cookies.

RBAC/Web (Ferraiolo et all., 1998) este o implementare a unui serviciu de control al accesului bazat pe roluri pentru servere web. RBAC/Web nu conține un mecanism specific de autentificare şi suportă ierarhii de roluri, constrângeri de cardinalitate şi separarea dinamică şi statică a îndatoririlor. Implementarea modelului SCAR-ACE nu este un simplu serviciu ci o aplicație care oferă o suită de servicii şi include autentificarea utilizatorilor pe bază de nume utilizator şi parola.

3. Modelul SCAR-ACE pentru controlul accesului bazat pe roluri

Modelul SCAR-ACE se referă la controlul accesului în sisteme distribuite, şi este exemplificat pentru o aplicație de comerț electronic. Conceptul de bază introdus pentru descrierea şi implementarea modelului constă în maşini de stare, abordare care este nouă față de modelele existente care utilizează diferite alte metodologii (vezi şi secțiunea 2).

În continuare se prezintă componentele modelului raportate la RBAC + modelul standard de control al accesului şi reliefând similitudinile şi diferențele față de acesta.

3.1. Componentele modelului

Componentele de bază ale modelului standard RBAC de control al accesului bazat pe roluri sunt (Ferraiolo et all., 2001): Nucleul RBAC, Ierarhiile RBAC, Relații statice, Relații dinamice şi Extensii.

Modelul SCAR-ACE realizează abordarea componentelor RBAC într-o manieră originală astfel încât să fie soluționate aspectele legate de politica de securitate prin utilizarea unor maşini de stare (vezi figura 1). Necesitatea introducerii maşinilor de stare s-a impus in contextul aplicării controlului accesului în sisteme distribuite şi deschise (precum aplicațiile de comerț electronic). Modelul standard RBAC se referă la conceptele fundamentale ale unui sistem de control al accesului, fiecare dintre modelele existente pe piață aducând propriile formalisme în funcție de domeniul de aplicare.

Nucleul modelului SCAR-ACE este similar nucleului RBAC. Nucleul RBAC include un set de şase elemente de bază şi anume: utilizatori, roluri, sesiuni, obiecte, operații şi permisiuni.

Nucleul modelului SCAR-ACE conține urmatoarele elemente: utilizatori, roluri, sesiuni, permisiuni, operații şi obiecte, maşina de stare şi relațiile dintre aceste elemente.

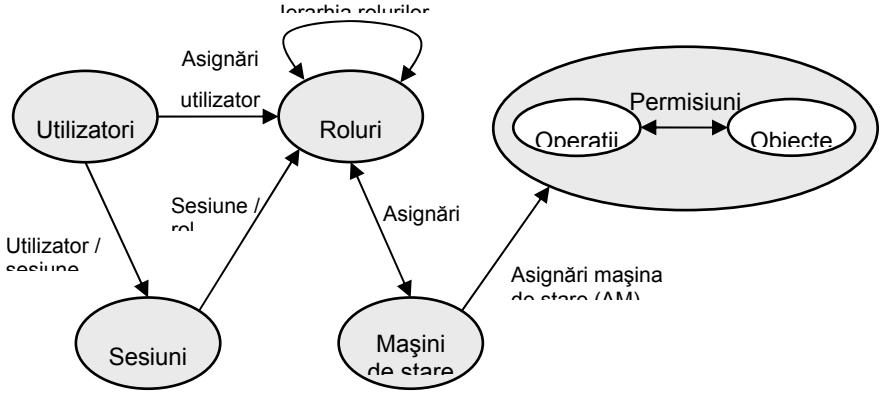

Figura 1. Componentele modelului SCAR-ACE

Fiecare rol are ataşată o *maşină de stare* specifică doar modelului SCAR-ACE. Accesul unui rol la un set de permisiuni nu poate avea loc decât prin intermediul maşinii de stare, aceasta fiind o diferentă majoră față de modelele existente.

O ierarhie a rolurilor este din punct de vedere matematic o ordine parțială sau totală care defineşte relații între roluri, unde rolurile senior dobândesc permisiunile celor junior (Ferraiolo et all., 2001). Ierarhiile sunt structuri de roluri care reflectă nivelul autorității şi al responsabilității.

Există mai multe tipuri de ierarhii de roluri (Sandhu et all., 2000):

ierarhii generale. În acest caz există o ordine parțială între roluri şi include moştenirea multiplă a permisiunilor;

ierarhii limitate. In acest caz se introduc restricții în cadrul ierarhiei rolurilor.

Ierarhia SCAR-ACE este o ierarhie limitată de tip arbore. Rolurile în ierarhie sunt restricționate la un singur descendent imediat şi la un singur ascendent imediat.

Relațiile RBAC sunt relații de separare a îndatoririlor şi impun constrângeri asupra modelului. Relațiile statice (SSD) sunt relații de separare a îndatoririlor şi sunt utilizate pentru a evita conflictul de interese în cadrul unei organizații (pentru a preveni un utilizator de a exceda nivelul de autorizare corespunzător poziției sale).

Modelul SCAR-ACE conține relații statice şi implementarea acestora are loc prin utilizarea unei maşini de stare pentru fiecare rol.

Relațiile dinamice (DSD), similar celor statice, limitează permisiunile utilizatorilor. Relațiile DSD diferă de cele SSD prin contextul în care aceste limitări sunt impuse. Cerințele DSD limitează drepturile prin plasarea constrângerilor asupra rolurilor care pot fi activate în timpul unei sesiuni utilizator (cu alte cuvinte, in mod dinamic, la rulare).

Relațiile DSD în modelul SCAR-ACE sunt realizate prin restricția impusă la asignarea rolurilor. Astfel nu pot fi activate simultan de diverşi utilizatori anumite roluri precum cel de administrator, recepționer al unei cereri sau distribuitor al unui produs. Pentru a realiza relațiile DSD, modelul SCAR-ACE utilizează tot maşini de stare.

In afara componentelor amintite mai sus, există şi alte caracteristici specifice anumitor modele, caracteristici denumite extensii ale modelului RBAC.

Delegarea permite unui utilizator să împuternicească alți utilizatori cu o parte a drepturilor sale.

Noțiunea de delegare este strâns interconectată cu cea de revocare, care permite utilizatorilor delegați să revoce un anumit rol, sau permite rolurilor să fie revocate utilizând constrângeri de timp. Arhitecturile care suportă delegarea şi revocarea diferă prin modul în care acestea au fost implementate.

Scalabilitatea este un factor important în cadrul sistemelor de control al accesului. Un sistem RBAC centralizat s-ar putea să nu respecte cerințele unei organizații mari dispersate geografic; astfel este esențială considerarea sistemelor distribuite şi a scalabilității acestora.

3.2. Maşina de stare

Lucrarea de față propune un model pentru controlul accesului în aplicațiile de comerț electronic pe baza uneia sau a mai multor maşini de stare. În mod extins orice aplicație are o secvență de operații (procese) şi impunerea unei ordini asupra acestora determină posibitatea tratării prin maşini de stare.

Se notează cu:

U, R şi *A* setul de utilizatori, setul de roluri şi respectiv setul de acţiuni în cadrul unui sistem distribuit.

M, S şi *T* pentru maşina de stare, stări şi respectiv tranziţii între stări.

Conform (E. Bertino et all, 1999) o *maşină de stare* este o listă de stări, tranziţii şi operaţii specifice unui rol. Tranziţiile le notez cu [T_1, T_2, …, T_n] unde fiecare T este o 3-tuplă:

$T = <A_i, (RS_i, >_i), act_i>$

unde:

$A_i \in A$,

$RS_i \subseteq R$ este un set de roluri autorizate să execute A_i,

$>_i$ este o ordine locală a rolurilor,

act_i reprezintă numărul posibil de activări ale acţiunii A_i.

Definirea noţiunii de *maşină de stare* într-un sistem distribuit conform (D. Georgakopoulos, M. Hornik, A. Sheth, 1995) determină ca în condiţiile în care suprapunem peste un model distribuit o ordine (parţială sau totală) acesta poate fi tratat prin intermediul maşinilor de stare care modelează şi controlează execuţia proceselor într-o aplicaţie distribuită. Fiecare tranziţie este definită ca şi un set de operaţii coordonate care se execută în vederea obţinerii anumitor rezultate. O acţiune poate fi alcătuită din mai multe procese şi realizează deservirea unei cereri prin accesul la o resursă sau un serviciu. Acţiunile într-un sistem distribuit peste care este suprapusă o ordine sunt interdependente reflectând o activitate coordonată.

3.3. Modele conceptuale

În această secţiune sunt analizate modelele conceptuale RBAC şi, prin paralelism, modelele conceptuale SCAR-ACE.

Pentru a analiza diferite versiuni RBAC s-a definit o familie de patru modele conceptuale (Sandhu et all., 1996). $RBAC_0$ este modelul de bază şi

este cerința minimă pentru un sistem de control al accesului bazat pe roluri. Modelele avansate $RBAC_1$ şi $RBAC_2$ includ modelul $RBAC_0$. $RBAC_1$ introduce ierarhiile de roluri în timp ce $RBAC_2$ adaugă constrângeri. Modelul consolidat $RBAC_3$ include modelele $RBAC_1$ şi $RBAC_2$ şi, prin tranzitivitate, $RBAC_0$.

Modelul SCAR-ACE se referă la un $RBAC_3$ şi, pentru realizarea SSD (Static Separation of Duties) şi DSD (Dynamic Separation of Duties), impune constrângeri. Prin paralelism cu modelul RBAC, modelul SCAR-ACE este compus din SCAR-ACE_0, SCAR-ACE_1, SCAR-ACE_2 şi SCAR-ACE_3. In continuare se va face referință la toate nivelele conceptuale ale modelului SCAR-ACE şi la specificul acestora (Ordean şi Gorgan, 2007).

Modelul de bază SCAR-ACE_0.

Modelul de bază SCAR-ACE_0 constă din următoarele entități: utilizatori (U), roluri (R), maşini de stare (M), permisiuni (P) şi sesiuni (S). Diferența majoră față de componentele modelului RBAC constă în introducerea maşinilor de stare şi, drept consecință a tuturor relațiilor pe care acestea le impun. Necesitatea acestei abordări se datorează domeniului particular de aplicare şi anume cel al sistemelor distribuite pentru care logica de acordare a drepturilor este diferită de cea din cadrul altor sisteme. Alegerea introducerii maşinilor de stare se datorează cerințelor propuse şi anume implementarea controlului accesului în sisteme distribuite.

In relație cu $RBAC_0$ am introdus definirea formală a modelului SCAR-ACE_0 care are următoarele componente:

U, R, P, S şi M (mulțimile de utilizatori, roluri, permisiuni, sesiuni şi respectiv maşini de stare);

Relația utilizator – roluri: $R_{UR} \subseteq U \times R$, o relație de asignare n:m pentru utilizatori-roluri

$R_{UR} = \{ur \mid ur = (u, r), (\forall u \in U, \exists r \in R) \wedge (\forall r \in R, \exists u \in U)\}$;

Relația sesiune - utilizator: $R_{SU} \subseteq S \rightarrow U$, o relație 1:1 care mapează fiecare sesiune unui singur utilizator (constant pentru timpul de viață al sesiunii)

$R_{SU} = \{su \mid su = (s, u), (\forall s \in S, \exists ! u \in U)$;

Relația rol - maşina de stare: $R_{RM} \subseteq R \rightarrow M$ o relație 1:1 care mapează fiecare rol la o singură maşină de stare

$R_{RM} = \{rm \mid rm = (r, m), (\forall r \in R, \exists ! m \in M)$;

Relația maşina de stare - permisiuni: $R_{MP} \subseteq M \rightarrow P$, o relație 1:n care mapează fiecare maşină de stare la un set de permisiuni

$R_{MP} = \{mp \mid mp = (m, p), (\forall m \in M, \exists p \in P)$.

Ierarhiile de roluri şi abordarea acestora în SCAR-ACE$_1$

Modelul SCAR-ACE1 introduce ierarhiile de roluri. Ierarhiile de roluri sunt discutate în literatură în mod invariabil împreună cu rolurile (Ferraiolo şi Kuhn 1992), (Hu et all., 1995), (Nyanchama şi Osborn, 1994), (H. von Solms şi I. van der Merwe, 2000). Modelul SCAR-ACE$_1$ este definit pentru ierarhia de tip arbore.

În continuare este prezentată definiția formală a modelului SCAR-ACE$_1$:

U, R, P, S, M aceleaşi cu cele din SCAR-ACE$_0$;

$IR \subseteq R \times R$, reprezintă o ordonare parțială pe *R* denumită ierarhie de roluri sau relație de dominanță a rolului, notată $\geq$;

Relațiile sunt aceleaşi cu cele din SCAR-ACE$_0$.

Modelul constrângerilor în SCAR-ACE$_2$

SCAR-ACE$_2$ este nemodificat față de SCAR-ACE$_0$ cu excepția faptului că sunt necesare constrângeri pentru a determina nivelul de acceptare al diferitelor componente SCAR-ACE$_0$. Modelul SCAR-ACE$_2$ realizează două seturi de constrângeri şi anume constrângeri statice şi constrângeri dinamice.

Componentele modelului formal SCAR-ACE$_2$ sunt:

U, R, P, M şi *S* (mulțimile de utilizatori, roluri, permisiuni, maşini de stare şi respectiv sesiuni);

$P \in SP$, permisiunile *P* sunt parte a unui set *SP* (se impun constrângeri asupra permisiunilor şi se specifică permisiunile acceptate şi nu cele care sunt respinse);

$Ri \neq Rj$, $\forall Ri, Rj \in R$, definesc rolurile mutual exclusive;

maşina de stare: $M \leftrightarrow R$, o funcție biunivocă care mapează fiecare maşină de stare unui rol şi fiecare rol unei maşini de stare. Prin intermediul acesteia se realizează constrângerile relativ la roluri.

Modelul SCAR-ACE$_3$

SCAR-ACE$_3$ este caracterizat de ierarhii şi constrângeri şi combină modelele SCAR-ACE$_1$ şi SCAR-ACE$_2$. Sunt definite constrângeri asupra ierarhiilor de roluri, interacțiuni şi constrângeri de cardinalitate.

Modelul SCAR-ACE$_3$ acceptă atât o ierarhie limitată de roluri, cât şi constrângeri asupra rolurilor, permisiunilor (prin maşina de stare) şi sesiunilor (constrângeri de timp).

4. Definirea formală a modelului

Pentru implementarea modelului SCAR-ACE (descris anterior în secțiunea 3) se impune definirea unui formalism care să asigure realizarea într-o manieră consistentă a componentelor, a relațiilor dintre acestea precum şi a constrângerilor intervenite.

Pentru definirea formală a modelului SCAR-ACE am utilizat un limbaj de exprimare a constrângerilor de autorizare (Ordean şi Gorgan, 2007). Acest limbaj de specificare a constrângerilor asupra controlului accesului defineşte un set de simboluri constante, variabile şi predicate.

Definiție: O *constantă* poate fi orice membru al uneia dintre următoarele mulțimi: *U* (setul de utilizatori), *R* (setul de roluri), *A* (setul de acțiuni), *S* (setul de stări), *E* (setul de evenimente), *T* (setul de tranziții) şi *P* (setul de parametri).

Astfel sistemul definit lucrează cu constante din toate aceste tipuri (de exemplu rolurile constante R_VIZITATOR, R_CUMPARATOR, R_RECEPTIONER$\in R$, stările constante ST_HOME, ST_VIEW$\in S$ şi evenimentele constante EV_HOME, EV_VIEW, EV_SELECT etc).

Definiție: Pentru fiecare set de constante se definește tipul *variabil.* Astfel, există șapte seturi de variabile notate cu V_U, V_R,, V_A, V_S, V_E, V_T și V_P.

Variabilele pot fi cel mai ușor exemplificate pentru cele de tip rol și anume un utilizator intră în cadrul sistemului în mod obligatoriu pe un rol de securitate redusă (exemplu R_VIZITATOR) și, la login, el comută pe un rol cu securitate mai mare (exemplu R_CUMPARATOR) modificându-și astfel starea și deci și rolul.

În continuare se va nota cu UT mulținea termenilor utilizator care definesc atât constantele cât și variabilele utilizator UT = U $\cup V_U$. Similar RT, AT, ST, ET, TT, PT denotă seturile $V_R \cup R$, $V_A \cup A$, $V_S \cup S$, $V_E \cup E$, $V_T \cup T$ și respectiv $V_P \cup P$.

Un alt set de elemente utilizate în cadrul definirii formale a modelului SCAR-ACE sunt predicatele. Astfel, există mai multe seturi de predicate:

- un set de *predicate de specificație* care definesc modelul formal al aplicației;
- un set de *predicate de planificare* care realizează restricțiile;
- un set de *predicate de execuție* pentru execuția aplicației.

Tabelul 1. Predicate de specificație

Predicat	**Definire**
Rol	dacă `rol`(r_i) este adevărat atunci $r_i \in$ RT
Utilizator	dacă `utilizator` (u_i) este adevărat atunci $u_i \in$ UT
Stare	dacă `stare` (s_i) este adevărat atunci și $\in$ ST
Eveniment	dacă `eveniment` (e_i) este adevărat atunci $e_i \in$ ET
Tranziție	dacă tranziție (s_i, s_j, t_k) este adevărat atunci $\exists$ stările s_i, $s_j \in$ ST pentru care este definită tranziția $t_k \in$ TT
Parametru	dacă `parametru` (t_i, p_j) este adevărat atunci tranziția $t_i \in$

	TT necesită parametrul $p_j \in$ PT
Aparține	dacă `aparține` (u_i, r_j) este adevărat atunci utilizatorul $u_i \in$ UT are rolul $r_j \in$ RT
>	dacă $r_i > r_j$ atunci rolul r_i moşteneşte toate funcționalitățile lui r_j sau, cu alte cuvinte, r_i domină pe r_j în ordinea rolurilor

Tabelul 2. Predicate de planificare

Predicat	**Definire**
$cannot\_do_r$	dacă `cannot_do`$_r$ (r_i, a_j) este adevărat, atunci acțiunea a_j nu poate fi asignată rolului r_i
$cannot\_do_t$	dacă `cannot_do`$_t$ (t_m, s_i, s_j) este adevărat, atunci tranziția t_m din starea şi în starea s_j nu poate fi realizată
$must\_execute_r$	dacă `must_execute`$_r$ (r_i, a_j) este adevărat, atunci acțiunea a_j trebuie să fie executată de către un utilizator aparținând rolului r_i
$must\_execute_t$	dacă `must_execute`$_t$ (t_m, s_i, s_j) este adevărat, atunci tranziția t_m trebuie să fie executată din starea şi în starea s_j
check	dacă `check` (t_i, s_j, s_k) este adevărată atunci constrângerile legate de tranziția t_i din starea s_j în starea s_k pot fi verificate în afara executiei aplicației
panic	dacă `panic` este adevărată atunci există cel puțin o constrângere care nu poate fi satisfăcută

Tabelul 3. Predicate de execuție

Predicat	Definire
$exec_u$	dacă $exec_u(u_i, a_j, s_k, e_l)$ este adevărat, atunci acțiunea a_j este executată de utilizatorul u_i prin comutarea de la starea s_k la apariția evenimentului e_l
$exec_r$	dacă $exec_r(r_i, a_j, s_k, e_l)$ este adevărat, atunci acțiunea a_j este executată de utilizatorul aparținând rolului r_i prin comutarea de la starea s_k la apariția evenimentului e_l
$exec_t$	dacă $exec_t(r_i, t_j, s_k, e_l, p_n)$ este adevărat, atunci tranziția t_j este executată de utilizatorul aparținând rolului r_i prin comutarea de la starea s_k la apariția evenimentului e_l şi cu parametrii p_n.
abort	dacă abort (a_j, s_k, e_l, p_n) este adevărat atunci acțiunea a_i nu a putut fi executată cu succes la comutarea din starea s_k, la apariția evenimentului e_l. Aceasta se poate întâmpla fie dacă evenimentul e_l nu este definit pentru starea s_k, fie dacă parametrii p_n nu sunt corespunzători.
success	dacă success (a_j, s_k, e_l, p_n) este adevărat atunci acțiunea a_i a putut fi executată cu succes la trecerea din starea s_k la apariția evenimentului e_l cu parametrii p_n.

Definiție: Modelul sistemului SCAR-ACE notat M_M este alcătuit din setul maşinilor de stare şi constrângerile corespunzătoare.

5. Rezultate experimentale

5.1. Aplicația de comerț electronic care implementează modelul SCAR-ACE

Pentru a realiza un proces de achiziție sistemul realizează următorii paşi:

1. Clientul foloseşte un navigator Web şi accesează sistemul. Nivelul de autorizare inițial este minim şi anume rolul său este cel de *vizitator*;
2. Serverul cu catalogul organizației furnizoare autentifică clientul şi îi permite acestuia să navigheze şi să selecteze articole. După autentificare clientul trece în rolul de *cumpărător;*
3. *Cumpărătorul* mapează comanda într-o cerere, o încapsulează într-un obiect şi transmite cererea de comandă organizației furnizoare utilizând Internetul;
4. Clientul specifică orice adnotări necesare comenzii şi furnizorul realizează aprobarea internă;
5. Organizația furnizoare începe distribuirea comenzii.

5.2. Funcționalități implementate

Vizitatorul este utilizatorul cu nivelul de securitate cel mai scăzut şi orice încercare de acces a unui utilizator îl direcționează pe acesta spre pagina de "Home", ca fiind prima funcționalitate implementată. Din această pagină utilizatorul poate vizualiza catalogul de produse, poate intra pe pagina de comparare a produselor sau se poate autentifica în sistem prin login sau înregistrare.

În cadrul catalogului de produse se pot vizualiza produsele în funcție de anumite criterii: produse de acelaşi tip, produse ale unui furnizor sau cele care au o anumită caracteristică selectată.

Aceste funcționalități sunt moştenite de rolurile de cumpărător şi de vânzător.

Un utilizator are rolul de cumpărător după înregistrare şi îşi poate modifica informațiile de identificare de pe pagina cu profilul său. Această pagină conține date de identificare gen nume şi adresă precum şi informații pentru care ar dori notificări.

Totodată cumpărătorul poate selecta produse în coş şi poate emite cereri de livrare. Chiar dacă au fost selectate unul sau mai multe produse pentru achiziționare, acestea pot rămâne în coş fără a se materializa într-o cerere fermă. Informațiile precum coşul de cumpărături şi cererile ferme sunt păstrate în baza de date şi pot fi accesate în orice sesiune după autentificare.

Cumpărătorul moşteneşte de asemenea, toate funcționalitățile oferite vizitatorului.

5.3. Teste de performanță

Testele funcționale determină pe de o parte nivelul de performanță al aplicației distribuite prin evaluarea timpilor de execuție şi, pe de altă parte, modificarea acestor timpi la scalarea aplicației (teste de scalabilitate). Ceea ce s-a urmărit au fost aspectele legate de performanța unei aplicații de comerț electronic care implementează modelul SCAR-ACE deoarece, prin introducerea unei maşini de stare, pot apărea eventuale întârzieri. Testele au demonstrat timpi de răspuns mici la o încărcare simultană a aplicației cu 300 de utilizatori şi au validat modelul SCAR-ACE din punct de vedere funcțional.

Ipoteza de verificat: timpul de răspuns la introducerea maşinilor de stare creşte la încărcarea / descărcarea unei maşini de stare în / din memorie.

Aplicația de comerț electronic a fost testată pentru două tipuri de achiziții: pentru maşini şi pentru calculatoare. Funcționalitățile dezvoltate au fost cele pentru rolurile de vizitator, cumpărător, vânzător şi administrator şi, au fost implementate acțiunile necesare realizării unei comenzi.

Testele efectuate au fost cele de verificare a timpului de răspuns în condițiile în care au fost lansate 300 de operații în paralel. Sistemul a fost testat în următoarea configurație: serverul - o maşină HP cu procesor Pentium(R) 4 CPU 2.53GHz,

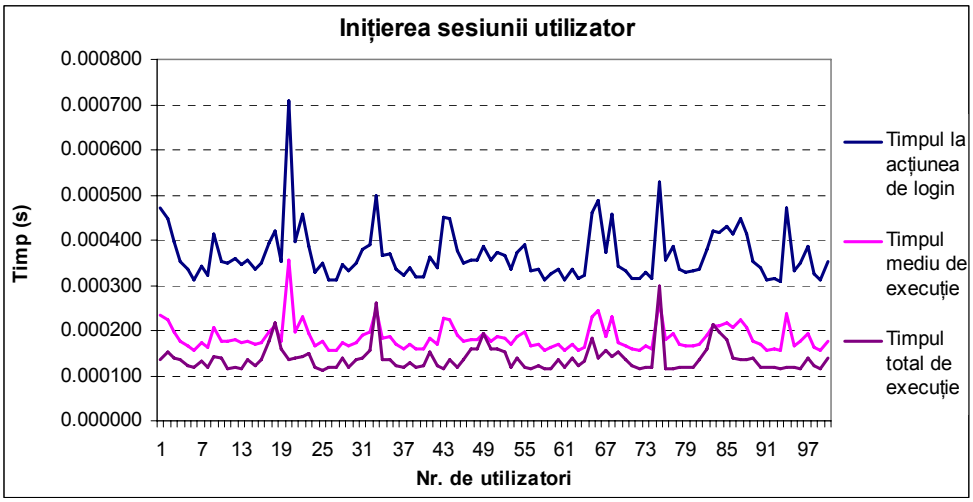

Figura 2. Timpii la inițierea sesiunii utilizator

1GB RAM şi 2 clienti cu aceeaşi configurație. Pentru scalabilitate s-au utilizat două servere şi doi clienți, toate cu configurația de procesor Pentium(R) 4 CPU 2.53GHz şi 1GB RAM.

Testele au constat din lansarea în execuție a 300 de operații care să ruleze în paralel. Astfel:

- primele 100 de operații realizează inițierea de sesiuni de către utilizatori (în rolul de vizitator). Pentru fiecare dintre aceste operații serverul a executat următoarele acțiuni:
  - crearea unui identificator utilizator şi păstrarea acestuia în memorie;
  - încărcarea maşinii de stare vizitator în memorie din baza de date;
  - memorarea stării curente (ST_HOME) în memorie ca şi stare actuală a utilizatorului nou creat;
  - trimiterea paginii "Home" pe maşina client împreună cu datele de identificare a utilizatorului şi a stării curente.

- următoarele 100 de operații sunt pentru trecerea utilizatorilor din rolul de vizitator în rolul de cumpărător prin autentificare. Pentru fiecare task serverul execută următoarele acțiuni:
    - verificarea datelor de autentificare;
    - descarcărea maşinii de stare vizitator din memorie şi încărcarea maşinii de stare cumpărător în cazul în care datele de autentificare au fost corecte (actiunea efectiva de login);
    - emiterea unui mesaj de eroare dacă datele de autentificare au fost greşite;
    - trimiterea unei pagini de răspuns pe maşina client (fie pagina cu informații personale a cumpărătorului fie o pagină de eroare) împreună cu datele de identificare a utilizatorului şi a stării curente.
- ultimele 100 de operații reprezintă încheierea sesiunilor de lucru. Serverul execută următoarele acțiuni în acest caz:
    - descărcarea maşinii de stare curente din memorie (acțiunea de logoff);
    - descărcarea datelor de context din memorie pentru fiecare utilizator (starea, informații de acces);
    - scrierea datelor ce necesită a fi păstrate persistent în baza de date (dacă a modificat datele de identificare sau non-identificative, coşul de cumpărături sau cererile ferme.

Testele care s-au făcut au urmărit următorii trei tipuri de informație temporală: timpul de execuție; timpul mediu al execuției; timpul total al execuției.

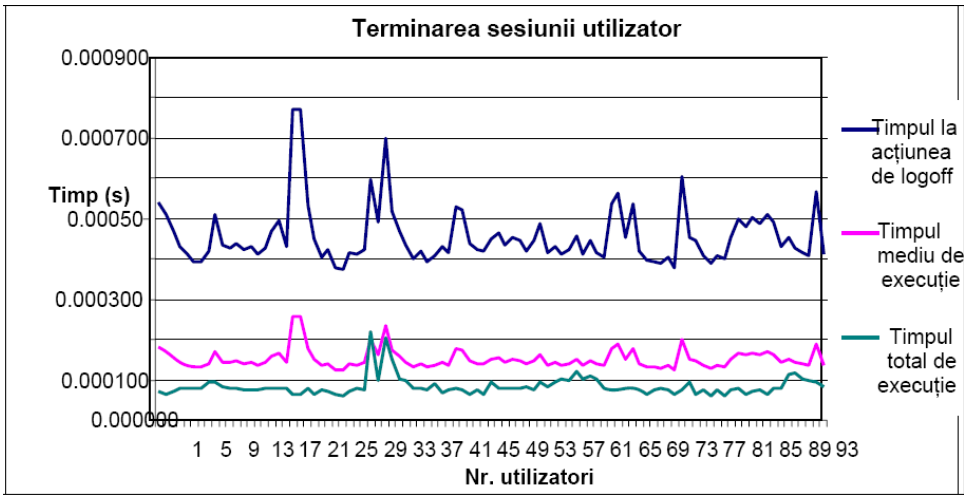

Figura 3. Timpii la terminarea sesiunilor utilizator

Rezultatele obținute sunt cele din figurile de mai jos (figura 2 şi figura 3).

Concluzie: introducerea maşinilor de stare implică creşterea timpului de execuție la încărcarea/descărcarea acestora în/din memorie. Toți timpii de execuție subsecvenți nu cresc. Timpul total de execuție nu este puternic influențat de lucrul cu maşini de stare.

6. Concluzii şi dezvoltări ulterioare

Prezenta analiză poate fi considerată un pas inițial în cercetarea politicilor controlului accesului bazat pe roluri în sistemele deschise şi distribuite. Există mai multe posibile dezvoltări ulterioare precum vom ilustra mai jos.

Politica abordată extinde modelul de bază RBAC astfel încât sunt suportate componentele standard cărora li se adaugă componente specifice necesare gestionării accesului cu ajutorul maşinilor de stare. Pe viitor se poate studia dacă aceste componente pot fi utilizate în tranzitia de la un sistem RBAC existent la altul. Modelul SCAR-ACE poate deveni destul de complex dacă se iau în considerare ierarhii diferite de roluri, modelul actual

fiind specific ierarhiilor de roluri de tip arbore. O extensie ar putea lua in considerare şi alte tipuri de ierarhii.

Abordarea curentă organizează şi structurează politicile de control ale accesului în aplicațiile de comerț electronic dar acest model poate fi investigat şi pentru alte tipuri de aplicații. Maşinile de stare care constituie extensiile, sunt în general nişte grafuri, iar implementarea acestora necesită cunoştințe avansate. O direcție nouă ar putea implementa o interfață grafică pentru proiectarea unei aplicații bazate pe modelul SCAR-ACE care să aibă drept intrare stările, evenimentele şi tranzițiile dintre stări şi să genereze la ieşire maşinile de stare corespunzătoare.

Aplicația de comerț electronic este un exemplu de implementare a modelului SCAR-ACE pentru controlul accesului bazat pe roluri. Această aplicație include atât componente de business cât şi componente de administrare, componente care fie formează grupuri fie sunt predefinite în sistem. Unele componente cer îndeplinirea prealabilă a unor condiții. O direcție de cercetare constă în extinderea părții de business astfel încât să poată fi gestionate diverse facilități dezvoltate pe verticală (precum achiziții de cărți cu includerea profilului cumpărătorilor pentru notificare şi includerea contextului în care cumpărarea are loc).

Pe de altă parte politica de componente poate fi marcată cu elemente de context, permițând astfel administratorului să grupeze componentele în funcție de anumite condiții. Aceasta ar permite posibilități de exploatare a politicilor de securitate prin intermediul unor unelte de vizualizare. În funcție de punctul de vedere considerat s-ar putea determina de exemplu, cumpărători care satisfac anumite criterii fie din punct de vedere al securitatii fie din punct de vedere al bunurilor achiziționate.

O altă extensie posibilă a modelului implică asocierea de roluri la procesele de business. În acest caz s-ar permite utilizarea modelului pentru un workflow în care maşina de stare urmăreşte fluența părții de business a unei aplicații. Chiar în contextul actual modelul urmăreşte fluența controlului şi, în acest sens, modificările ar fi minore şi ar consta în adăugarea de etichete de proces.

SCAR-ACE ia în considerare restricții la nivel înalt considerând primitivele modelului ca fiind rolurile, drepturile şi maşina de stare. Pentru a se diminua diferențele dintre diferite modele RBAC ar fi de interes o cercetare în care primitivele ar fi modelate sub forma unor triplete (obiect, acces, acțiune) şi ar fi localizate politicile echivalente, corespunzătoare

acestei modelări. Această cercetare ar consta, în special, în determinarea constrângerilor.

O posibilitate de extindere ar fi prin încapsularea componentelor modelului SCAR-ACE într-o interfață între diferite modele RBAC existente. Fiecare model ar fi o stare în cadrul maşinii de stare şi s-ar putea realiza cooperarea între diferite modele RBAC. În astfel de medii, politicile interfeței se pot utiliza în medierea diferențelor dintre diferitele domenii de control ale accesului. Primul scop al politicilor de interfață ar fi cel de a permite accesul între politicile de domenii. În cazul în care este posibilă o cooperare, aceasta ar putea depinde şi de politicile de administrare. În anumite cazuri tranzițiile între modele s-ar putea realiza pe bază de încredere adică fără necesitatea autentificării. Astfel apare posibilitatea extinderii pentru politici de conformitate pentru deciziile colaborării inter-domenii, decizii bazate pe încredere, după care s-ar putea utiliza politicile de interfață pentru a realiza în mod automat colaborarea între modele.

Politica de gestionare a drepturilor furnizează un mijloc de control al accesului la o granularitate foarte fină. Aceste drepturi sunt obtinute în SCAR-ACE prin intermediul maşinii de stare. Aceste drepturi pot fi însă ordonate într-o ierarhie care să permită o politică mai concisă de acordare a acestora. O extensie posibilă a drepturilor ar fi asocierea acestora cu un factor de risc. Astfel, acțiunile permise nu sunt egale din punct de vedere al răului potențial pe care îl pot aduce în sistem la o folosire defectuoasă. Prin asocierea acestora cu un factor de risc se pot emite anumite politici care să specifice supervizări suplimentare în acordarea accesului pentru drepturile cu factor ridicat de risc. Astfel, un risc ar fi o valoare care, în funcție de unul sau mai multe praguri, ar avea sau nu nevoie de supervizări suplimentare.

Printre motivațiile importante ale cercetării prezente este dezvoltarea unei politici de acces în sistemele distribuite şi deschise care să controleze accesul în funcție de rolul utilizatorului. O extensie ar fi adoptarea acestei politici în cadrul diverselor aplicații din domeniile: sănătate, telecomunicații, gestionarea resurselor umane şi materiale, şi alte tipuri de aplicații în care este necesară definirea de roluri multiple.

Referințe

J. Barkley, *Implementing role based access control using object technology,* Proceedings of the ACM Workshop of Role Based Access Control, November 1995, pp. 20-es., ISBN:

0-89791-759-6

L. Bartz, *hyperDRIVE: Leveraging LDAP to implement RBAC on the web,* Proceedings of the ACM Workshop of Role Based Access Control, 1997, pp.69-74, ISBN: 0-89791-985-8

D. E. Bell, L. J. LaPadula, *Secure computer systems: Unified exposition and Multics interpretation*, Technical Report MTR-2997, The MITRE Corporation, July 1975, www.csrc.nist.gov/publications/history/bell76.pdf

A.D. Birrell, R. Levin, R. M. Needham, M.D. Schroeder, *Grapevine: An exercise în distributed computing,* Communication of the ACM, pp. 260-274, April 1982, ISSN: 0001-0782

D. Ferraiolo, J. Barkley, D. Kuhn, *A role-based access control model and reference implementation within a corporate intranet*, ACM Transaction on Information and System security, 2, February 1998, pp.34-64, ISSN:1094-9224

D. Ferraiolo, R. Kuhn, *Role-based Access Control,* Proceedings of 15th NIST-NCSC National Computer Security Conference, Gaithersburg, 1992, pp.554-563, csrc.nist.gov/rbac/

E. Bertino, E. Ferrari, V. Atluri*, The specification and enforcement of authorisation constraints in workflow management systems*, ACM Transactions on Information Systems Security, November 1999, pp. 65-104, ISBN: 1094-9224

David F. Ferraiolo, Ravi Sandhu , Serban Gavrila , D. Richard Kuhn , Ramaswamy Chandramouli, *Proposed NIST standard for role-based access control*, ACM Transactions on Information and System Security (TISSEC), v.4 n.3, pp.224-274, August 2001 (1), ISSN:1094-9224

M. Gasser, A. Goldstein, C. Kaufman, B. Lampson, *The Digital distributed system security architecture*, Proceedings of the 1989 National Computer Security Conference, pp. 305-319, October 1989

M. Gasser, E. McDermott, *An architecture for practical delegation in a distributed system*, Proceedings of the 1990 IEEE Symposium on Security and Privacy, pp. 20-30, May 1990, ISBN: 63835-X

D. Georgakopoulos, M. Hornik, A. Sheth, *An overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure,* Distributed and Parallel Databases, pp 119-153, 1995, ISSN:0926-8782

L. Giuri, *Role-base access control on the web using java*, Proceedings of the ACM Workshop on Role-Based Access Control, 1999, pp. 11-18, ISBN:1-58113-180-1

G. S. Graham and P.J. Denning, *Protection – principles and practice*, Procedeengs of the fifth ACM symposium on Operating systems principles, May 1975, pp. 14-24, ISSN:0163-5980

M.Y.Hu, S.A. Demurjian, T.C. Ting, *User-Role based Security în the ADAM Object-Oriented Design and Analyses Environment*, Database Security VIII: Status and Prospects, 1995, pp. 333-348, ISBN:0-89791-808-8

S.H. von Solms, I. van der Merwe, The Mangement of Computer Security Profile, 2000

B. Lampson, Protection, Proceedings of the Fifth Annual Princeton Conference on Information Sciences and Systems, pp 437–443, Princeton University, 1971

P.V. McMahon, *SESAME V2 Public Key and Authorization Extensions to Kerberos,* Proceedings of the 1995 Symposium on Network and Distributed System Security*,* pp 114-131, February 1995

S.P. Miller, B.C. Neuman, J. I. Schiller, J.H. Saltzer, *Section E.2.1: Kerberos Authentication and Authorization System*, Project Athena Technical Plan, MIT Project Athena, Cambridge, Massachusetts, October 1988, http://web.mit.edu/Kerberos/papers.html

B. C. Neuman, *Proxy-based authorization and accounting for distributed systems, P*roceedings of the 13th International Conference în Distributed Computing Systems, pp 283-291, May 1993, ISBN: 0-8186-3770-6

M. Nyanchama, S. Osborn, *Access Rights Administration în Role-Based Security Systems,* Database Security VIII: Status and Prospects, 1994, pp. 37-56

M. Ordean, D. Gorgan, *"FORMAL DEFINITION OF SCAR-ACE ROLE-BASED ACCESS CONTROL MODEL"*, Proceedings CSCS-16, 16th International Conference on Control System and Computer Science, 22-25 May 2007, Bucharest, Vol. 2, pp 155-159, ISBN:978-973-718-743-7

M. Ordean, D. Gorgan, *RBAC and SCAR-ACE Role Based Access Models*, VIPSI 2007 Venice, International Conferences on Advances in the Internet, Processing, Systems, and Interdisciplinary Research, March 19-22, 2007, Italy, Book of Abstracts, ISBN 86-7466-117-3, pp. 12

T. Parker, D. Pinkas, *"SESAME V4 – Overview*", December 1995, http://www.esat.kuleuven.ac.be/cosic/sesame/doc-txt/overview.txt

R. Sandhu, *Lattice-based access control models*, IEEE Computer, 26(11), pp. 9–19, 1993, ISSN: 0018-9162

R. Sandhu, E.Coyne, H. Feinstein, C. Youman, *Role-based access control models,* IEEE Computer, pp 38-47, 1996, csrc.nist.gov/rbac/sandhu96.pdf

R. Sandhu, D. Ferraiaolo, R. Kuhn, *The NIST model for role-based access control: Towards a unified standard*, Proceedings of the Fifth ACM Workshop on Role-Based Access Control*,* July 2000, pp. 47-63, ISSN:1094-9224

K. R. Sollins, *Cascaded Authentication*, Proceedings of the 1988 IEEE Symposium on Security and Privacy, IEEE Computer Society, 1988, pp. 156, ISBN: 0-8186-0850-1

Z. Tari, S. Chan, *A role based access control for intranet security,* IEEE Internet Computing, September/October 1997, pp. 24-34, Digital Object Identifier 10.1109/4236.623965

W. Tolone et all, *Access control in collaborative systems*, ACM Computing Surveys (CSUR), March 2005, pp. 91-100, ISSN:0360-0300

M. V. Tripunitara, N. Li, *Comparing the expressive power of access control models,*

Proceedings of the 11th ACM conference on Computer and communications security, October 2004, pp. 62-71, ISBN:1-58113-961-6

T.Y.C. Woo, S.S. Lam, *Designing a Distributed Authorization Service*, Proceedings of IEEE INFOCOM'98, March 1998, ISBN: 0-7803-4383-2 Hall, 2004.

Card, S. Moran, T. and Newell, A. *The Psychology of Human-Computer Interaction*. Cariere. Lawrence Erlbaum Associates, 1983.

Chin, J. P., Diehl, V. A., and Norman, K. L., *Development of an instrument measuring user satisfaction of the human-computer interface*. In CHI '88: Proceedings of the SIGCHI conference on Human factors in computing systems, pages 213–218, New York, NY, USA, 1988. ACM Press

Ivory, M. Y. and Hearst, M. A., *The state of the art in automating usability evaluation of user interfaces*, ACM Computing Surveys, vol. 33, no. 4, pp. 470–516, 2001.

Kiczales, G., Lamping, J., Menhdhekar, A., Maeda, C., Lopes, C. Loingtier, and Irwin, J.-M., *Aspect-Oriented Programming, in Proceedings European Conference on Object-Oriented Programming*. Springer-Verlag, 1997, vol. 1241, pp. 220–242.

Maes, P., *Social Interface Agents: Acquiring Competence by Learning from Users and Other Agents*. In O. Etzioni, editor, Softwareware Agents — Papers from the 1994 Spring Symposium (Technical Report SS-94-03, pages 71–78. AAAI Press, 1994.

Moldovan, G. S. and Tarța. A. M., *Developing an Usability Evaluation Module Using AOP*. In International Conference on Computers, Communications & Control, ICCCC 2006, pages 320–325, Felix Spa, România, 2006 (a).

Moldovan, G. S., and Tarța, A. M., *A Comparison of Using AOP and Java Accessibility for Usability Evaluation*. In CD Proceedings of 4th European Conference on Intelligent Systems and Technologies, ECIT 2006, Iaşi, România, 2006 (b).

Nielsen, J., *Usability Engineering*. Academic Press, 1993.