UNIVERZITET U NOVOM SADU
FAKULTET TEHNIČKIH NAUKA
U NOVOM SADU

mr Goran Sladić

Model kontekstno zavisne kontrole pristupa u poslovnim sistemima

— doktorska disertacija —

Novi Sad 2011.

Sadržaj

Predgovor

Kontrola pristupa (*access control*) odnosno autorizacija, u širem smislu, razmatra na koji način korisnici mogu pristupiti resursima računarskog sistema i na koji način ih koristiti. Kontrola pristupa predstavlja samo jedan aspekt složene i obimne oblasti bezbednosti računarskih sistema, ali je i jedna od njenih najvažnijih delova. Ona učestvuje u obezbeđivanju poverljivosti i integriteta informacija. Poverljivost informacija podrazumeva da samo autorizovani korisnici mogu da pročitaju informacije, dok integritet informacija podrazumeva da samo autorizovani korisnici mogu da menjaju podatke u skladu sa sigurnosnom politikom posmatranog sistema.

Sistemi za upravljanje poslovnim tokovima (*Workflow Management Systems*, WFMS) se danas u velikoj meri koriste u softverskom inženjerstvu za realizaciju softverskih sistema za podršku poslovnim procesima u različitim oblastima. Sa sve većom primenom ovih sistema u različitim oblastima problem efikasnog sprovođenja kontrole pristupa u takvim sistemima dolazi do izražaja. U literaturi su identifikovana tri osnovna zahteva koje kontrola pristupa u sistemima za upravljanje poslovnim tokovima treba da zadovolji:

- *Striktne najmanje privilegije.* Princip najmanjih privilegija podrazumeva da se korisniku dodele samo one privilegije koje su neophodne za izvršenje određenog zadatka. Ovaj princip sprečava mogućnost da korisnik izvršava nepotrebne, potencijalno opasne operacije kao prateći efekat dozvole za izvršenje određene operacije. Striktno poštovanje principa najmanjih privilegija zahteva da korisnik ima različite privilegije u zavisnosti od funkcije koju trenutno obavlja.

- *Redosled događaja* zahteva da se određene privilegije mogu dodeliti tek nakon što su neke druge već bile dodeljene, tj. privilegije da se izvrši neki zadatak se dodeljuju tek pošto je prethodno izvršen neki drugi zadatak.
- *Razdvajanje obaveza.* Iako postoji više varijacija razdvajanja obaveza (*Separation of Duties, SoD*), SoD je u suštini zahtev da se izvršenje kritičnih operacija raspodeli izmedu dve ili više osoba, tako da pojedinačno nijedna osoba ne može ugroziti bezbednost. Primarni cilj razdvajanja obaveza je da se očuva semantički integritet poslovnih procesa kao i podataka koje ti procesi obrađuju.

Jedan od najčešce korišćenih model za kontrolu pristupa u savremenim poslovnim sistema je model kontrole pristupa zasnovane na korisničkim ulogama (*Role Based Access Control*, RBAC), gde je pristup objektima sistema baziran na ulozi korisnika u sistemu. Osnovni RBAC model čine sledeći entiteti: *korisnici*, *korisnička sesija*, *korisničke uloge* i *privilegije*, pri čemu se privilegije sastoje od *operacija* koje se izvršavaju nad *objektima*. Centralni deo RBAC-a predstavlja koncept uloge oko koje su formulisana prava pristupa. Osnovna relacija modela je povezivanje uloga i privilegija. U RBAC-u, ulogama se dodeljuju privilegije, a korisnicima se dodeljuju određene uloge. Prednost RBAC modela u odnosu na druge modele kontrole pristupa je u efikasnijem definisanju prava pristupa i administraciji sistema. RBAC entiteti obično se definišu na istom nivou apstrakcije kao i poslovni procesi organizacije za koju se RBAC model definiše. Osim osnovnog RBAC modela u literaturi, a i u praksi se susreću različite modifikacije i proširenja ovog modela. RBAC standardom pored osnovnog modela definisana su i njegova tri proširenja: hijerarhija uloga, statičko (*Static Separation of Duties*) i dinamičko (*Dynamic Separation of Duties*) razdvajanje obaveza, i kombinacija hijerarhije uloga sa razdvajanjem obaveza.

Kontekstno zavisna bezbednost eksplicitno razmatra kontekst (stanje okruženja) u specifikaciji i implementaciji sigurnosnih mehanizama. Pojavom kontekstno zavisne bezbednosti pojavili su se i novi problemi sa stanovišta bezbednosti. Kako je kontekst uglavnom dinamičan, tj. često se menja, potrebno je obezbediti da se bezbednosni mehanizmi adaptiraju tim promenama, tj. da budu u stanju da efikasno reaguju na promene konteksta.

U današnje vreme kroz sisteme za upravljanje poslovnim tokovima se realizuju složeni dinamički poslovni procesi. U praksi realizacija kontrole pristupa

za takve sisteme često je zavisna i od stanja okruženja, npr. tekućeg vremena, lokacije, istorije događaja, itd. Prirodan način rešavanja kontrole pristupa u ovakvim uslovima je uspostavljanje kontekstno zavisne kontrole pristupa koja omogućuje da stanje okruženja utiče na mehanizam sprovođenja kontrole pristupa.

Tema doktorske disertacije pripada oblasti kontrole pristupa u poslovnim sistemima. Kontroli pristupa u poslovnim sistemima posvećena je relativno velika pažnja u dosadašnjim istraživanjima ako se kao kriterijum usvoji broj objavljenih rezultata koji se odnose na kontrolu pristupa u širem smislu. Sa druge strane postojeći rezultati u ovoj oblasti problem kontekstno zavisne kontrole pristupa razmatraju samo delimično, tj. uticaj konteksta u predstavljenim modelima kontrole pristupa za poslovne sisteme je ograničen.

Cilj disertacije je razvoj modela kontekstno zavisne kontrole pristupa u poslovnim sistemima (*COntext-sensetive Business processes Access Control model*, COBAC) zasnovanog na standardnom RBAC modelu kontrole pristupa koji je proširen entitetima *poslovnog procesa*, *aktivnostima*, *kontekstom* i *kategorijama resursa*. Uvođenjem entiteta *poslovnog procesa* i *aktivnosti* omogućeno je efikasnije definisanje i sprovođenje kontrole pristupa za poslovne procese. S obzirom da je moguće da na kontrolu pristupa utiču i faktori iz okruženja samog sistema pa i faktori koji čine sam sistem ali ne i eksplicitno model kontrole pristupa, predloženi model proširen je i *kontekstom*. *Kategorizacija* resursa omogućuje definisanje prava pristupa za čitavu kategoriju (grupu) resursa i time potencijalno smanjuje broj prava pristupa koje je potrebo definisati.

Prikazani model prevazilazi sledeće probleme postojećih modela:

- Baziran je na RBAC modelu koji je proširen konceptom poslovnog procesa i aktivnosti. Korišćenjem ova dva koncepta moguće je direktno povezati određene segmente kontrole pristupa sa aktivnostima umesto da se povezuju sa tekućom sesijom korisnika i na ovaj način obezbediti efikasnije definisanje prava pristupa i ograničenja za različite poslovne sisteme. Pored toga, korišćenje poslovnog procesa i aktivnosti obezbeđuje nezavisnost odgovarajućih segmenata kontrole pristupa od broja sesija u kojima se aktivnosti izvršavaju. Kako se u COBAC model poslovni proces posmatra kao sekvenca aktivnosti koje se izvršavaju, ulogama su dodeljene aktivnosti koje one imaju pravo da izvrše, dok su privilegije

(za izvršenje operacija nad resursima) potrebne za izvršenje aktivnosti dodeljene aktivnostima. Na ovaj način obezbeđena je preciznija kontrola najmanjih privilegija.

- Prava pristupa mogu biti definisana na nivou određene instance procesa (instance aktivnosti) i na taj način zadovoljiti različite sigurnosne zahteve koje različite instance procesa mogu da imaju. Sa druge strane, zahtevanje da se za svaku instancu procesa eksplicitno definišu prava pristupa značajno komplikuje proces definisanja autorizacija, stoga COBAC model podržava definisanje prava pristupa i na nivou definicije procesa (definicije aktivnosti). Prava pristupa definisana na nivou definicije procesa primenjuju se na sve instance tog procesa.
- Modelom je definisano više statičkih i dinamičkih ograničenja potrebnih za efikasno sprovođenje kontrole pristupa u sistemima za upravljanje poslovnim tokovima.
- Kako realni poslovni procesi mogu da imaju sofisticirane autorizacione zahteve koji mogu da zavise od više različitih faktora iz okruženja, relacije u COBAC modelu proširene su kontetkstno zavisnim uslovima. Na ovaj način obezbeđena je podrška za kontekstno zavisnu kontrolu pristupa. Pored ovoga, proširena je definicija uloge sa kontekstnim informacijama.
- Predloženi model konteksta razvijen je koristeći ontologije kako bi se na efikasan način mogle opisati različite kontekstne informacije, obezbediti semantička interoperabilnosti između različitih kontekstno zavisnih sistema i relativno jednostavno proširiti kontekst za konkretni sulučaj korišćenja.
- Resursi u COBAC modelu mogu biti hijerarhijski organizovani, što može da utiče na smanjenje broja privilegija koje je potrebno definisati.

Tekst disertacije sastoji se iz šest poglavlja.

Prvo poglavlje, koje sadrži uvodna razmatranja, započinje prikazom dosadašnjih istraživanja i rezultata koji se odnose na oblast kontrole pristupa zasnovane na korisničkim ulogama (RBAC). Analizirana je primena RBAC modela u različitim sistemima, a takođe su predstavljeni i relevantni postojeći modeli za kontrolu pristupa zasnovani na RBAC modelu. Pregled istraživanja u oblasti kontekstno zavisne kontrle pristupa dat je na kraju poglavlja.

Centralni deo disertacije dat je u drugom poglavlju. Ono sadrži formalnu specifikaciju COBAC modela za kontrolu pristupa u poslovnim sistemima. Formalnom specifikacijom obuhvaćeni su:

- model konteksta koji se koristi u ovom modelu,
- entiteti COBAC modela i njihove međusobne relacije,
- modeli različitih ograničenja koji su definisani COBAC modelom, i
- COBAC model sprovođenja kontrole pristupa.

U trećem poglavlju opisana je softverska arhitektura sistema. Opisana je arhitektura četiri osnovna podsistema prezentovanog modela:

- podsistema za administraciju,
- podsistema za proveru identiteta,
- podsistema za autorizaciju, i
- kontekstnog podsistema.

Na kraju poglavlja prikazano je okruženje implementacije prototipa kojim je realizovan COBAC model.

U četvrtom poglavlju disertacije izvršena je verifikacija prikazanog modela na konkretnom realnom sistemu — Postupak izbora u nastavna zvanja na Fakultetu tehničkih nauka u Novom Sadu. Verifikacija je sprovedena pomoću razvijenog prototipa COBAC-a, a obuhvata sledeće aktivnosti:

- analiza procesa,
- specifikacija resursa i operacija nad resursima,
- specifikacija korisničkih uloga,
- specifikacija modela konteksta,
- specifikacija prava pristupa i ograničenja, i
- specifikacija kontekstnih modula.

U petom poglavlju prikazana je verifikacija COBAC modela na realnom poslovnom procesu u bibliotekama – Obrada bibliografskih zapisa. Za razliku od procesa predstavljenog u četvrtom poglavlju gde je težište kontrole pristupa bilo na nivou procesa, poslovni proces u ovom poglavlju karakteriše težište kontrole pristupa na nivou resursa, tj. bibliografskih zapisa, pošto je potrebno sprovoditi kontrolu pristupa nad delovima bibliografskih zapisa. Cilj ove studije slučaja je da pokaže da se COBAC protip može iskoristiti u ovakvim i sličnim slučajevima, gde se COBAC koristi za kontrolu pristupa na

nivou procesa, a neki drugi namenski sistem se koristi za kontrolu pristupa na nivou delova resursa.

Šesto poglavlje sadrži zaključna razmatranja, analizu rezultata i doprinosa disertacije, kao i analizu pravaca daljih istraživanja.

Bibliografija sadrži bibliografske jedinice koje su direktno ili indirektno pomenute u tekstu disertacije.

Zahvaljujem svim članovima Komisije koji su svojim korisnim sugestijama doprineli da disertacija bude jasnija i preglednija. Posebnu zahvalnost dugujem mentoru prof. dr Branku Milosavljeviću i prof. dr Zori Konjović za nesebičnu podršku u toku izrade disertacije.

Novi Sad, februar 2011.

Goran Sladić

Poglavlje 1

Uvodna razmatranja

1.1 Kontrola pristupa

Kontrola pristupa ili autorizacija, u širem smislu, kao koncept postoji otkad čovek ima potrebu da nešto zaštiti. Straže, kapije i brave su korišćene još od najranijeg doba da bi se sprečio pristup dragocenim resursima. U današnjem dobu informacionih tehnologija autorizacija, obično, razmatra na koji način korisnici mogu pristupiti resursima računarskog sistema ili "*ko šta može da radi*". Autorizacija predstavlja, neosporivo, fundamentalni sigurnosni mehanizam koji je danas u upotrebi.

Kontrola pristupa je samo jedan aspekt složene i opširne bezbednosti računarskog sistema, ali je i jedan od njegovih najvažnijih delova. Ugrožavanje resursa, sa sigurnosnog aspekta, može se podeliti u sledeća četiri tipa [105]:

- poverljivost (*confidentiality*) se odnosi na potrebu čuvanja zaštićenih i privatnih informacija. Ova kategorija može da sadrži sve, od npr. državne tajne do lozinke.
- integritet (*integrity*) se odnosi na koncept zaštite informacija od nedozvoljenih izmena od strane neautorizovanih korisnika.
- dostupnost (*availability*) odnosi se na dostupnost informacija kada je to potrebno. Napadi koji pokušavaju opteretiti web servere su uglavnom napadi na dostupnost.
- neporecivost (*nonrepudation*) - niti pošiljalac niti prilamac ne može poreći transmisiju poruke.

Kontrola pristupa bi trebalo da obezbedi poverljivost i integritet informacija. Obezbeđivanje poverljivosti zahteva da samo autorizovani korisnici mogu da pročitaju informacije, dok obezbeđivanje integriteta zahteva da samo autorizovani korisnici mogu da menjaju informacije u skladu sa sigurnosnom politikom [105].

1.1.1 Provera identiteta i autorizacija

Provera identiteta (*authentication*) i autorizacija su fundamentalni entiteti kontrole pristupa. Iako predstavljaju sasvim različite koncepte, često dolazi do mešanja pojmova. Jedan od razloga konfuzije je bliska povezanost ova dva pojma.

Provera identiteta je proces utvrđivanja da je korisnik zaista onaj za koga se predstavlja. Bazirana je na nekim od sledećih faktora [105]:

- "*nešto što znaš*", poput lozinke, ličnog identifikacionog broja (PIN) i sl.
- "*nešto što imaš*", poput smart kartice, ATM (*Automatic Teller Machine*) kartice i sl.
- "*nešto što jesi*", ili fizičke karakteristike, poput otiska prsta, oblika lica i sl.

Očigledno, provera identiteta je pouzdanija ako se koriste dva ili više faktora. Lozinka može biti otkrivena (pogođena), smart kartica može biti izgubljena, a sistem za prepoznavanje oblika lica radi sa određenim stepenom grešaka u prepoznavanju, tako da korišćenje samo jednog sistema za proveru identiteta potencijalno ne ostvaruje željeni nivo sigurnosti [105]. Ovo je razlog zašto ATM terminali od korisnika zahtevaju i karticu i PIN da bi mu pristup bio omogućen.

Autorizacija je proces odlučivanja šta je korisniku dozvoljeno da radi. Autorizacija se obično odnosi na dozvolu ili zabranu pristupa nekom resursu. Neophodno je da postoji veza između korisnika sistema i resursa u sistemu da bi se znalo koji korisnici imaju pravo pristupa kojim resursima. Zbog ovog, jasno je da se autorizacija oslanja na proveru identiteta korisnika [105].

1.1.2 Korisnici, subjekti, objekti, operacije i privilegije

Gotovo svi sistemi za kontrolu pristupa koriste pojmove: *korisnik, subjekat, objekat, operacija, privilegija* i relacije između ovih entiteta [105].

Pojam *korisnik* odnosi se na osobu koja je u interakciji sa računarskim sistemom. U mnogim implementacijama moguće je da jedan korisnik ima više identifikatora preko kojih se prijavljuje na sistem. Mehanizam provere identiteta omogućuje mapiranje više identifikatora za prijavljivanja na jednog korisnika [105].

Instanca korisnikove interakcije sa sistemom naziva se *sesija* [105].

Proces unutar računarskog sistema koji, u ime korisnika, pristupa resursima predstavlja *subjekat*. U realnosti, sve korisnikove interakcije sa računarskim sistemom odvijaju se kroz određene aplikacije koje se izvršavaju u tom računarskom sistemu. Korisnik može da istovremeno ima više subjekata (pokrenuto je više aplikacija) iako je prijavljen sa samo jednim identifikatorom [105]. U različitoj literaturi se pod subjektom često podrazumeva i sam korisnik.

Objekat je bilo koji resurs računarskog sistema, uključujući fajl sistem, hardverske uređaje, baze podataka, itd. [105].

Operacija je aktivni proces kreiran od strane korisnika, aktivnost koju subjekat u ime korisnika izvršava nad nekim objektom [105].

Privilegije (dozvole) su autorizacije za izvođenje odgovarajuće akcije u sistemu [105].

1.1.3 Najmanje privilegije

Princip najmanjih privilegija predstavlja dodeljivanje minimalnog skupa neophodnih privilegija korisniku potrebnih za izvršenje određenog zadatka. Ovaj princip sprečava korisnika da izvršava nepotrebne, potencijalno opasne, operacije kao usputni efekat dozvole za izvršenje određene operacije. Problem dodeljivanja minimalnog skupa neophodnih privilegija korisniku je pretežno administrativni problem koji zahteva da se za svaku funkciju, koju obavljaju različiti korisnici, izdvoji samo neophodan skup privilegija potrebnih za izvršenje te funkcije i ništa više [105].

Striktno pridržavanje principa najmanjih privilegija zahteva da korisnik (subjekat) ima drugačije privilegije zavisno od fukcije koju **trenutno** obavlja. Ovo znatno usložnjava sistem koji obezbeđuje kontrolu pristupa, ali i administraciju kontrole pristupa jer je potrebna dodela prava pristupa na nivou granularnosti nižem od zadataka koje korisnik obavlja. Isto tako bitno je da su prava pristupa dodeljena samo u vremenskom intervalu u kome se izvršava određeni zadatak (podzadatak), ali ne i van njega [105].

1.2 Kontrola pristupa zasnovana na korisničkim ulogama - RBAC

Tokom 1992 godine NIST (*National Institute of Standards and Technology*) pokrenuo je studiju [101] u vezi bezbednosti informacionih sistema kako u privatnom tako i u državnom sektoru. Kao rezultat ove studije uočeno je da mnogobrojni zahtevi kontrole pristupa, u oba sektora, ne mogu biti zadovoljeni tadašnjim mehanizmima za kontrolu pristupa koji su prvenstveno bili zasnovani na DAC (*Discretionary Access Control*) [187] i MAC (*Mandatory Access Control*) [187] kontroli pristupa. Ferraiolo i Kuhn su u [102] predložili model kontrole pristupa zasnovan na korisničkim ulogama (*Role Based Access Control*, RBAC) kao soluciju koja može da reši probleme kontrole pristupa uočene u NIST-ovoj studiji. U ovom modelu pristup objektima sistema je baziran na ulozi korisnika u sistemu.

Danas, RBAC model predstavlja jedan od najzastupljenijih modela za kontrolu pristupa primenjen u širokom spektru komercijalnih aplikacija. Osnovni razlozi za ovako široku primenu je što RBAC model omogućuje smanjenje kompleksnosti i troškova administracije bezbednosti u aplikacijama. Osim u komercijalnoj primeni RBAC je u značajnoj meri zastupljen i u naučnoj literaturi. Veliki broj radova iz oblasti bezbednosti bavi se RBAC modelom, odnosno modelima kontrole pristupa koji u osnovi predstavljaju proširenje RBAC-a.

RBAC model predložen u [102] evoluirao je RBAC model koji je modularno organizovan. Pored osnovnog modula RBAC-a, koji definiše osnovne funkcionalnosti, postoje još tri ključna modula: modul koji obezbeđuje hijerarhiju uloga i moduli za statičku i dinamičku separaciju obaveza [103, 105, 106].

1.2.1 Osnovni RBAC model

Osnovni RBAC model čine sledeći entiteti: *korisnici*, *korisničke uloge*, *sesije* i *privilegije*, gde se privilegije sastoje od *operacija* koje se izvršavaju nad *objektima* [103, 105, 106].

Prednost RBAC modela u odnosu na druge modele kontrole pristupa je u efikasnijem definisanju prava pristupa [103]. Korisnicima se dodeljuju određene uloge na osnovu njihovih kompentencija i obaveza. Na jednostavan način moguće je ukloniti određene uloge korisniku kao i dodeliti mu nove uloge. U RBAC modelu privilegije se nikad direktno ne dodeljuju korisnicima, već su

im privilegije pridružene posredno preko uloga. Korišćenje uloga kao posrednika u dodeli privilegija korisnicima predstavlja ključ u efikasnoj administraciji sistema [103]. Moguća je promena privlegija pridruženih određenoj ulozi bez da se privilegije menjaju za svakog korisnika ponaosob. Promena privilegija za svakog korisnika može biti dugotrajan posao, sa većom verovatnoćom podložan greškama. Koncept *sesije* predstavlja mapiranje između korisnika i aktiviranog podskupa uloga koje su dodeljene korisniku [106].

Alternativa dodeljivanja privilegija korisnicima posredstvom uloga je postupak baziran na kloniranju privilegija pridruženih nekom korisniku. Kloniranje je postupak dodeljivanja privilegija nekom korisniku kopiranjem privilegija nekog drugog korisnika koji obavlja sličnu funkciju u sistemu. Kloniranje se obično obavlja bez obraćanja detaljne pažnje na pojedinačne privilegije pridružene korisniku. Ovo je razlog zašto se kloniranje smatra lošom praksom u administraciji sistema za kontrolu pristupa, pa se zato i ne preporučuje za korišćenje [105].

Druga bitna prednost RBAC modela je da se RBAC entiteti definišu na istom nivou apstrakcije kao i poslovni procesi organizacije za koju se RBAC model definiše. Uloge u RBAC modelu, u određenom smislu, odgovaraju radnim mestima (pozicijama poslova) u organizaciji. Privilegije pridružene ulozi ograničavaju članove te uloge da izvršavaju samo određeni skup aktivnosti predviđen za odgovarajuće radno mesto na koje je korisnik raspoređen u organizaciji [105].

Osnovni RBAC model formalno je definisan na sledeći način [106]:

Definicija 1.1 *Neka U, $ROLES$, OPS i OBS predstavljaju skupove korisnika, uloga, operacija i objekata, tim redosledom. Tada važi:*

- $UA \subseteq USERS \times ROLES$ *je mapiranje više-na-više korisnik-uloga, tj. relacija dodele uloga korisnicima.*
- $assigned\_users(r : ROLES) \rightarrow 2^{USERS}$ *je mapiranje uloge r na skup korisnika, tj.* $assigned\_users(r) = \{u \in USERS \mid (u, r) \in UA\}$.
- $PRMS = 2^{OPS \times OBS}$ *je skup privilegija.*
- $PA \subseteq PRMS \times ROLES$ *je mapiranje više-na-više privilegija-uloga, tj. relacija dodele privilegija ulogama.*
- $assigned\_permissions(r : ROLES) \rightarrow 2^{PRMS}$ *je mapiranje uloge r na skup privilegija, tj.* $assigned\_permissions(r) = \{p \in PRMS \mid (p, r) \in PA\}$.

- $SESSIONS$ *predstavljaju skup sesija.*
- $user\_sessions(u : USER) \rightarrow 2^{SESSIONS}$ *je mapiranje korisnika na skup njegovih sesija.*
- $session\_roles(s : SESSIONS) \rightarrow 2^{ROLES}$ *je mapiranje sesije na skup uloga aktivnih u toj sesiji.*
- $avail\_sessions\_perms(s : SESSIONS) \rightarrow 2^{PRMS}$ *je mapiranje sesije na skup privilegija dostupnih u toj sesiji.*

1.2.2 Hijerarhija uloga

Potreba za uvođenjem hijerarhije uloga javila se uočavanjem da se funkcije pojedinih uloga preklapaju, tj. korisnici koji su članovi različitih uloga mogu da imaju zajedničke privilegije. Na primer, postoje funkcije koje obavljaju svi zaposleni u organizaciji (npr. podnošenje zahteva za odsustvo). U slučaju nepostojanja hijerarhije uloga potrebno je da se za svaku ulogu dodele privilegije potrebne za obavljanje opšte (zajedničke) funkcije u organizaciji ili da postoji uloga kojoj bi bile dodeljene privilegije da obavlja sve opšte funkcije, a svim korisnicima bi bila pridružena ova uloga. Oba pristupa znatno usložnjavaju i otežavaju administraciju sistema. U prisustvu hijerarhije uloga, kolekcija privilegija potrebnih za obavljanje neke funkcije u organizaciji može biti raspoređena na više korisničkih uloga, a svaka od ovih uloga može biti korišćena u deljenju opštih privilegija i formulisanju novih uloga [105].

Ogranizovanje uloga u hijerarhiju obezbeđuje efiksniju administracija sistema za kontrolu pristupa. Međutim administracija sistema se može još više pospešiti uvođenjem veznih uloga. Vezne uloge postoje u hijerarhiji uloga zbog pogodnosti definisanja kolekcije semantički grupisanih privilegija koje će biti nasleđivane od nekih uloga na višim hijerarhijskim nivoima. Vezne uloge se nikad direktno ne dodeljuju korisnicima tako da one ne bi ni postojale da ne postoji hijerarhijsko organizovanje uloga [105].

U praksi postoje dva tipa hijerarhije korisničkih uloga [106]:

- *Uopštena hijerarhija uloga* omogućuje višestruko nasleđivanje. Bilo koja uloga može može da ima više neposrednih nadređenih uloga i isto tako svaka uloga može da ima više neposrednih podređenih uloga.
- *Ograničena hijerarhija uloga* uvodi određena ograničenja na hijerarhiju uloga u cilju pojednostavljenja administracije uloga. Tako, npr. može se uvesti ograničenje da uloga može da ima jednu ili više neposredno

nadređenih uloga, ali uloga može da ima samo jednu neposredno podređenu ulogu.

Uopštena hijerarhija uloga formalno je definisana na sledeći način [106]:

Definicija 1.2 *$RH \subseteq ROLES \times ROLES$ je parcijalno uređenje nad $ROLES$, tj. relacija nasleđivanja, u oznaci $\succeq$, gde važi $r_1 \succeq r_2$ samo ako su sve privilegije r_2 ujedno i privilegije r_1, i svi korisnici r_1 su i korisnici r_2. Formalno: $r_1 \succeq r_2 \Rightarrow authorized\_perms(r_2) \subseteq authorized\_perms(r_1) \wedge authorized\_users(r_1) \subseteq authorized\_users(r_2)$ gde je:*

- *$authorized\_users(r : ROLES) \rightarrow 2^{USERS}$ je mapiranje uloge r na skup korisnika u prisustvu hijerarhije uloga. Formalno: $authorized\_users(r) = \{u \in USERS \mid r' \succeq r, (u, r') \in UA\}$*
- *$authorized\_perms(r : ROLES) \rightarrow 2^{PRMS}$ je mapiranje uloge r na skup privilegija u prisustvu hijerarhije uloga, tj. $assigned\_perms(r) = \{p \in PRMS \mid r' \succeq r, (p, r') \in PA\}$.*

Iako uopštena hijerarhija uloga nudi veću fleksibilnost u definisanju kompleksnih struktura uloga, u većini RBAC implementacija zastupljen je neki vid ograničene hijerarhije uloga. Razlog za ovo je daleko jednostavnija implementacija hijerarhije uloga sa izvesnim ograničenjima nego li implementacija uopštene hijerarhije uloga kao i znatno efikasnija administracija ovakvog sistema [105].

Jedna od bitnih aktivnosti prilikom uvođenja RBAC baziranog sistema za kontrolu pristupa u nekoj organizaciji je postupak definisanja uloga kao i formiranja njihove hijerarhije. Ovaj proces, poznat kao *role engineering* [73], idenitifikovan je kao jedna od najskupljih aktivnosti prilikom realizacije RBAC-a. U literaturi se sreće značajan broj radova na ovu temu. Rad [256] se bavi formiranjem uloga u *workflow* okruženjima. Formiranje uloga u velikim složenim organizacijama opisano je u [213]. *Role engineering* proces zasnovan na modelu scenarija (opis korišćenja sistema u formi sekvence akcija i događaja [73]) opisan je u [180]. Colantonio i dr. predložili su metriku za ocenu kvaliteta potencijalnih uloga i na osnovu toga formiraju uloge u sistemu [65]. Rad [226] bavi se modelovanjem informacija sistema (*system-centric*) u cilju olakšavanja *role engineering* procesa. Probabilistički metod za *role engineering* opisan je u radu [110], takođe, data je i mera za ocenu kvaliteta postignutog rezultata.

Formiranje hijerarhije uloga zasnovano na optimizaciji grafova opisano je u [272].

1.2.3 Separacija obaveza

Iako postoji više varijacija pojma razdvajanja obaveza (*Separation of Duties*, SoD), SoD je u suštini zahtev da se izvršenje kritičnih operacija raspodeli između dve ili više osoba, tako da pojedinačno nijedna osoba ne može ugroziti bezbednost sistema [48, 148, 227]. Npr. banke zahtevaju da dvoje zaposlenih moraju biti prisutni prilikom postavljanja/preuzimanja novca u ATM automate. Cilj je da se spreči da čitav tok visokorizičnih operacija ne kontroliše jedna osoba već više njih.

U literaturi je predloženo više tipova SoD ograničenja, ali se samo neki od njih i primenjuju u postojećim RBAC implementacijama. Standardni RBAC model definiše dva tipa SoD ograničenja: *statička SoD* i *dinamička SoD* [106]. Podela je izvršena na osnovu vremena kada se SoD ograničenja primenjuju na uloge. Kod statičkih SoD, ograničenja na uloge primenjuju se u vreme administracije uloga, odnosno u trenutku kada je korisnik autorizovan za određenu ulogu (npr. ako je korisniku dodeljena uloga R_1 ne može mu biti dodeljena i uloga R_2). Dinamička SoD ograničenja primenjuju se kada korisnik aktivno koristi sistem, tj. kada postoji korisnička sesija (npr. korisniku su dodeljene uloge R_1 i R_2, ali u tekućoj sesiji korisnik može da ima samo jednu od ove dve uloge) [227].

SoD relacije se često koriste da bi se onemogućio "sukob interesa". Sukob interesa može nastati kao posledica dodele privilegija korisniku, pri čemu su privilegije pridružene *konfliktnim ulogama* (korisniku su dodeljene konfliktne uloge). Jedan od načina za sprečavanje sukoba interesa je posredstvom statičkog SoD, tj. postaviti ograničenja na dodelu uloga korisnicima [48].

Statički SoD se mora posebno razmatrati u slučajevima nepostojanja i postojanja hijerarhije uloga. U slučaju nepostojanja hijerarhije uloga statički SoD se svodi na onemogućavanje korisnika da bude član jedne ili više uloga. U drugom slučaju mora se voditi računa ne samo o ulogama koje se direktno pridružuju korisniku već i o njihovim nadređenim ulogama u hijerarhiji [105].

Formalna definicija statičkih SoD ograničenja bez hijerarhije uloga i u prisustvu hijerarhije uloga data u [106] je sledeća:

Definicija 1.3 *Statička SoD ograničenja,* $SSD \subseteq (2^{ROLES} \times N)$*, predstavljaju kolekciju parova* (rs, n)*, gde je* rs *skup uloga, a* n *prirodan broj* ≥ 2*, sa osobinom da ni jednom korisniku ne može biti dodeljeno* n *ili više uloga iz skupa* rs *u svakom* $(rs, n) \in SSD$ *paru. Formalno:* $\forall (rs, n) \in SSD, \forall t \subseteq rs : |t| \geq n \Rightarrow \bigcap_{r \in t} assigned\_users(r) = \emptyset$*.*

U prisustvu hijerarhije uloga uvodi se sledeća izmena: $\forall (rs, n) \in SSD, \forall t \subseteq rs : |t| \geq n \Rightarrow \bigcap_{r \in t} authorized\_users(r) = \emptyset$*.*

Kod dinamičkih SoD ograničenja korisnik može biti član međusobno konfliktnih korisničkih uloga, ali ograničenja postoje prilikom sesije korisnika. Dinamičke SoD relacije, poput statičkih SoD ograničavaju dostupnost privilegija korisniku, međutim dinamička SoD se razlikuju od statičkih SoD po kontekstu u kome su ova ograničenja primenjena. Dinamička SoD ograničavaju dostupnost privilegija korisniku ograničavanjem uloga koje mogu biti aktivne tokom korisničke sesije [227].

Formalna definicija dinamičkih SoD ograničenja bez hijerarhije uloga i u prisustvu hijerarhije uloga, data u [106], je sledeća:

Definicija 1.4 *Dinamička SoD ograničenja,* $DSD \subseteq (2^{ROLES} \times N)$*, predstavljaju kolekciju parova* (rs, n)*, gde je* rs *skup uloga, a* n *prirodan broj* ≥ 2*, sa osobinom da ni jedan korisnik ne može aktivirati* n *ili više uloga iz skupa* rs *u svakom* $(rs, n) \in DSD$ *paru. Formalno:* $\forall (rs, n) \in DSD \Rightarrow n \geq 2 \wedge |rs| \geq n$ *i* $\forall s \in SESSIONS, \forall rs \in 2^{ROLES}, \forall role\_subset \in 2^{ROLES}, \forall n \in N, (rs, n) \in DSD, role\_subset \subseteq rs, role\_subset \subseteq session\_roles(s) \Rightarrow |role\_subset| < n$*.*

Dinamička SoD ograničenja obezbeđuju proširenu podršku za princip najmanjih privilegija. Svaki korisnik, zavisno od zadatka kojeg trenutno izvršava ima različite privilegije. Ovim je obezbeđeno da su korisniku pridružene privilegije samo u vremenskom intervalu potrebnom da se određeni zadatak izvrši [105, 227].

Većina postojećih RBAC sistema, u nekoj formi, podržava hijerarhiju uloga, a hijerarhija uloga ima značajan uticaj na SoD ograničenja i obrnuto, SoD ograničenja utiču na hijerarhiju uloga. Kada neka uloga nasleđuje jednu ili više drugih uloga sistem mora obezbediti da struktura nasleđivanja ne narušava SoD ograničenja [105]. Uočeno je da su tri ograničenja (karakteristike) bitna za razumevanje interakcije između SoD ograničenja i hijerarhije uloga [105]:

- dve uloge R_1 i R_2 mogu biti međusobno isključive (konfliktne) samo ako R_1 nije nadređena (direktno ili indirektno) uloga ulozi R_2 i obrnuto, R_2 nije nadređena (direktno ili indirektno) ulozi R_1,
- ako su dve uloge međusobno isključive (konfliktne) ne može postojati neka treća uloga kojoj će obe ove uloge biti nadređene, i
- ako su bilo koje dve uloge međusobno isključive tada ne može da postoji "*root*" ili "*superuser*" uloga tj. uloga koja poseduje sve privilegije u sistemu odnosno uloga koja nasleđuje sve ostale uloge.

1.2.4 Nedostaci RBAC modela

Neke kritike na RBAC standard [16] usvojen od strane ANSI (*American National Standards Institute*) navedene su u [153]. U ovom radu uočeno je nekoliko ograničenja predloženog standarda kao i pojedinih elemenata koji standardom nisu potpuno jasno razjašnjeni što bi po autorima moglo da izazove probleme u različitim RBAC implementacijama, tj. semantika pojedinih elemenata RBAC standarda bi mogla biti različita u različitim rešenjima. Autori predlažu da sesija ne bude definisana u osnovnom RBAC-u, već kao poseban modul kao i da standard eksplicitno definiše mogućnost aktivacije samo jedne uloge u sesiji jer u pojedinim apliakcijama nije neophodno postojanje sesije odnosno dozvoljena je aktivacija samo jedne uloge. Takođe, navodi se potreba za jasnim razdvajanjem osnovnih i izvedenih relacija definisanih u modelu. Za hijererhiju uloga autori smatraju da je potrebno jasno definisati semantiku nasleđivanja uloga, jer su moguće različite interpretacije nasleđivanja.

Sa primenom RBAC-a u različitim okruženjima uočeno je da na zahteve kontrole pristupa često, osim RBAC entiteta, utiču i drugi faktori. U tu svrhu predložena su različita proširenja RBAC modela kako bi se zadovoljili odgovarajući zahtevi kontrole pristupa. Detaljniji opisi različitih proširenja RBAC modela i razlozi za njihovo uvođenje dati su u odeljcima 1.4 i 1.7.3.

1.2.5 XACML RBAC profil

XACML (*Extensible Access Control Markup Language*) [185] specifikacija definiše jezik za specifikaciju prava pristupa i mehanizam za sprovođenje kontrole pristupa. Ova specifikacija se koristi za opis opštih zahteva kontrole pristupa u informacionom sistemu [233]. Za predstavljanje RBAC baziranog mo-

dela XACML definiše *XACML RBAC profil* [183] kojim je moguće predstaviti osnovni RBAC, hijerarhiju uloga kao i SoD ograničenja.

Neki nedostaci XACML RBAC profila kao i predlog proširenja XACML RBAC profila u cilju otklanjanja tih nedostataka opisani su u [1]. Autori smatraju da predloženi profil ne može efikasno da izrazi različita ograničenja koja se sreću u RBAC modelima. Takođe, predloženi način organizovanja hijerarhije uloga može da stvara komplikacije ako se podržava delegacija uloga (jedan korisnik delegira neku svoju ulogu drugom korisniku). Osim ovoga, navodi se potreba za postojanjem negativnih prava pristupa, uvođenja konteksta kao i da se umesto uloga kao subjekat prava pristupa može navesti i korisnik. Iako ovi elementi nisu definisani standardnim RBAC modelom u praksi se susreće veliki broj različitih RBAC baziranih modela koji definišu i ove elemente, pa stoga autori smatraju da bi bilo dobro da postoje i u XACML RBAC profilu jer su već definisani XACML-om.

1.3 Primene RBAC-a

U ovom odeljku prikazane su primene RBAC pristupa u različitim okruženjima (sistemima). Prikazane su primene u *workflow* sistemima, web okruženjima, operativnim sistemima, sistemima za upravljanje bazama podataka i u programskim jezicima.

1.3.1 RBAC u workflow sistemima

Prema [257] *workflow* predstavlja računarski podržanu automatizaciju poslovnih procesa, u celini ili delimično, tokom koje se podaci prosleđuju od jednog do drugog učesnika u cilju izvršavanja aktivnosti (zadatka, *task*-a) poslovnog procesa u skladu sa skupom predefinisanih proceduralnih pravila, dok je sistem za upravljanje poslovnim tokovima (*Workflow Management System*, WfMS) sistem koji kompletno definiše, upravlja i izvršava *workflow* uz podršku softvera čiji je redosled izvršavanja determinisan računarskom reprezentacijom *workflow* logike.

Mehanizam kontrole pristupa u *workflow* sistemima ima za cilj da implementira sigurnosne polise organizacije za potrebe kontrole pristupa *workflow* sistema, tj. potrebno je da zadovolji sigurnosne zahteve i ciljeve organizacije.

Kako u današnje vreme organizacija postaje sve dinamičnija i poslovni procesi se često razvijaju i menjaju tokom vremena potrebno je, zbog veće efikasnosti, razdovojiti model kontrole pristupa od samog *workflow* modela [267].

Poslovna pravila koja su korišćena u fazi dizajna *workflow* sistema (za opis procesa i zadataka *workflow*-a) su jedan od važnijih faktora za definisanje zahteva kontrole pristupa *workflow* sistema u fazi njegovog dizajna [105]. Osnovni zahtevi, identifikovani u različitoj literaturi, koje je potrebno definisati su [18, 33, 46, 160]:

- *Striktne najmanje privilegije* - Koncept najmanjih privilegija zahteva da korisnik ima samo one privilegije koje su mu potrebne da bi izvšio svoje obaveze. Striktne najmanje privilegije pojačavaju koncept najmanjih privilegija u smislu da su neke od privilegija u određenim situacijama neprikladne (nepoželjne), tj. nisu potrebne korisniku za izvršenje njegovih obaveza. Šta više, posedovanje tih privilegija može da naruši sigurnosnu politiku organizacije.
- *Redosled događaja* zahteva da se određene privilegije mogu dodeliti, tek nakon što su neke druge bile dodeljene, tj. privilegije da se izvrši neki zadatak se dodeljuju tek pošto je prethodno izvršen neki drugi zadatak (za čije izvršenje su takođe bile dodeljene privilegije).
- *SoD* - Primarni cilj razdvajanja obaveza (SoD) u *workflow* sistemima je da se očuva semantički integritet poslovnih procesa kao i podataka koje ti procesi obrađuju.

U slučaju RBAC modela za kontrolu pristupa definisani zahtevi kontrole pristupa u *workflow* sistemima utiču na sledeće elemente RBAC modela [105, 140]:

- *dodeljivanje uloga korisnicima* - definisani statički SoD zahtevi (ograničenja) moraju se primeniti prilikom dizajna sistema, odnosno prilikom administracije sistema (definisanja uloga, odnosno dodeljivanje uloga korisnicima mora biti u skladu sa statičkim SoD ograničenjima).
- *definisanje hijerarhije uloga* kao i samih uloga bazirano je i na obavezama korisnika u izvršenju različitih procesa, a ne samo na strukturi organizacije.
- *dodeljivanje privilegija ulogama* - privilegije direktno dodeljene korisničkim ulogama treba da budu apstraktne i povezane sa semantikom *work-*

flow procesa (zadaci procesa su semantički entiteti koji predstavljaju korisničke operacije). Pored dodele privilegija korisničkim ulogama, i zadacima je potrebno dodeliti privilegije da bi zadaci mogli uspešno pristupati određenim resursima (objektima) u sistemu u toku svog izvršavanja.

Kontrola pristupa u međuorganizacionim *workflow*-ovima razmatrana je u radu [140]. Kao dva dodatna zahteva sa stanovišta kontrole pristupa za međuorganizacione *workflow* sisteme autori navode sledeće:

- *Separacija organizacione od međuorganizacione bezbednosne infrastrukture* - Potreba za ovim javlja se jer više različitih organizacija učestvuje u međuorganizacionom *workflow*-u, takođe jedna organizacija može da učestvuje u različitim međuorganizacionim *workflow*-ovima. Organizacije mogu da zamenjuju jedna drugu u *workflow*-u, može doći do spajanja više organizacija i slično. Sve ove situacije nameću potrebu za razdvajanjem mehanizama bezbednosti međuorganizacionih *workflow*-ova od bezbednosti definisane za pojedinačne organizacije.
- *Različiti nivoi granularnosti* ne samo za međuorganizacione *workflow*-ove već i za klasične potrebno je obezbediti različite stepene granularnosti sa stanovišta kontrole pristupa. Potrebno je obezbediti ne samo kontrolu na nivou čitavog procesa, već i kontrolu na nivou *task*-ova kao i na nivou resursa kojima se pristupa u okviru tih *task*-ova.

Prototipska implementacija modula za kontrolu pristupa koja podržava navedene zahteve realizovana je u okviru SALSA [139] sistema za upravljanje međuorganizacionim *workflow*-ovima.

1.3.2 RBAC u web okruženjima

Web bazirane aplikacije, prvenstveno zbog heteregenosti okruženja i distribuiranosti, zahtevaju fleksibilan model za kontrolu pristupa koji će omogućiti efikasnu kontrolu pistupa i efikasnu administraciju kontrole pristupa. Zbog ovih zahteva RBAC model kontrole pristupa nameće se kao jedno moguće rešenje [133]. Većina novijih implementacija RBAC modela kontrole pristupa u web okruženjma bazirana je na infrastrukturi javnih ključeva [133] (*Public Key Infrastructure*, PKI) [266].

Implementacije RBAC modela za web okruženja imaju dva pristupa [198]:

- U jednom pristupu RBAC se implementira na nivou web aplikacije. Sama implementacija je, u velikoj meri, nezavisna od web servera na kome će se aplikacija izvršavati. Loša strana ovog pristupa je što svaka aplikacija ima svoju, nezavisnu implementaciju kontrole pristupa što otežava uspostavljanje autentifikacije bazirane na *Single Sign On* (SSO) [121] principu.
- U drugom pristupu, implementacija RBAC modela vrši se na nivou web servera, a aplikacije se po potrebi oslanjaju na ovu implementaciju radi uspostavljanja svoje kontrole pristupa. Ovaj pristup omogućuje primenu jedne RBAC implementacije za različite web aplikacije, a samim time je olakšano uspostavljanje autentifikacije bazirane na *Single Sign On* [121] principu. Loša osobina ovog pristupa je što je moguće da aplikacije u određenoj meri budu zavisne od konkretne RBAC implementacije (od RBAC API-a), pa time i od web servera na kome se izvršavaju.

Park i dr. identifikovali su dva načina preuzimanja korisničkih atributa u web okruženju. Pri tome atributi, između ostalog, obuhvataju uloge i grupe. [198]:

- U *user-pull* arhitekturi, korisnik preuzima svoje uloge sa nekog servera uloga. Kada korisnik pristupa željenom web serveru, odnosno web aplikaciji, pored autentifikacionih podataka treba da prosledi i dobijene uloge. Web server, odnosno web aplikacija mora imati mogućnost da proveri da li su dobijene uloge zaista pridružene korisniku. Preuzimanje uloga ne mora da se vrši prilikom svakog pristupa korisnika web serveru, već je moguće skladištenje uloga na klijentskom računaru određeni vremenski period.
- U *server-pull* arhitekturi web server, odnosno web aplikacija preuzima uloge korisnika sa servera na kome su smeštene uloge. U ovoj arhitekturi korisnik ne mora da prosleđuje svoje uloge, već je samo potrebno da web serveru prosledi identifikacione podatke.

Za obe arhitekture postoje dva moguća pristupa (režima rada) [198]:

- *Računarski baziran pristup*, za proveru identiteta, prosleđuje autentifikacione podatke korisnikovog računara (npr. IP adresa ili simbolička adresa). U ovom pristupu uloge su nezavisne od korisnika, vezane su za računare (računarske mreže).

- *Korisnički baziran pristup*, za proveru identiteta, prosleđuje autentifikacione podatke samog korisnika. Ovaj pristup podržava veću portabilnost korisnika u odnosu na prethodni.

Izbor arhitekture i/ili režima rada zavisi od web aplikcije, tj. od njenih karakterisitka. Ako postoji veliki broj korisnika, a njihove uloge se ne menjaju tako često verovatno je *user-pull* režim bolji da bi se poboljšale performanse web servera, jer u ovom slučaju on ne mora preuzimati uloge [198]. Sa druge strane, ako su uloge korisnika dinamične, odnosno ako se zahtevaju striktno trenutne uloge korisnika (*user-pull* arhitektura vrši keširanje uloga na klijentskom računaru) tada je *server-pull* arhitektura bolje rešenje [198]. *Računarski baziran* režim rada primenjuje se kada identifikacija konkretnog korisnika nije bitna [198]. Npr. univerziteti obično imaju pravo pristupa digitalnim bibliotekama koje nije ograničeno na konkretne studente i profesore, već svi koji pristupaju sa univerzitetske računarske mreže imaju ista prava pristupa. *Korsinički baziran* režim rada se koristi kada je neophodno da se prava pristupa vežu za samog korisnika [198]. Npr. u poslovnim informacionim sistemima za prava pristupa bitan je korisnik, a ne računar sa koga se pristupa sistemu.

Referentna implementacija RBAC modela za web servere (RBAC/Web) opisana je u članku [104]. U ovom modelu privilegije predstavljaju HTTP metode koje krajnji korisnici mogu da izvrše nad kontrolisanim URL-ovima. RBAC/Web realizovan je tako da može da se koristi u simbiozi sa postojećim servisima web servera za proveru identiteta (*username/password*) i poverljivost (SSL). Na web serveru je da izvrši proveru identiteta korisnika i da ona bude poverljiva, a potom da identifikacione podatke prosledi RBAC/Web-u kako bi on vršio kontrolu pristupa. RBAC/Web ne zahteva nikakve izmene na klijentskoj strani, tj. nije potreban nikakav namenski web brauzer.

U implementaciji za UNIX/Linux platformu pored osnovnog RBAC-a podržana je hijerarhija uloga kao i statička i dinamička SoD ograničenja. Za ovu platformu moguća su dva različita načina korišćenja, tj. realizovane su dve implementacije. Jednostavnija implementacija realizovana je posredstvom CGI skriptova i ne zahteva izmene u izvornom kodu servera. Dobra strana ove implementacije je što je relativno jednostavna za instalaciju i korišćenje, međutim postoje situacije kada nije dovoljno efikasna. Druga implementacija realizovana je kroz modifikaciju izvornog koda web servera, tako što je u kod web

servera ubačen poziv RBAC/Web API-a. Pojedini web serveri, poput Apache-a, raspodelili su svoje operacije u više koraka i dozvoljavaju da se svaki od ovih koraka naknadno unapredi ili u potpunosti promeni. Ovo pruža mogućnost da se RBAC/Web ugradi u te web servere na mnogo efikasniji način, izmenom sekvence poziva, tj. dodavanjem novog koraka u proces izvršavanja [104].

Referentna imlementacija za Windows platforme ne podržava dinamička SoD ograničenja jer bi to zahtevalo izmenu mehanizma sesije koji postoji na Windows platformama. Kako je mehanizam bezbednosti za Windows platforme uniforman između samog Windows OS-a i mnogih web servera koji se izvršavaju na toj platformi, RBAC/Web za Windows takođe može da se iskoristi i za kontrolu pristupa fajl sistemu [104].

Implementacija RBAC modela sa hijerarhijom uloga, posredstvom *sigurnih cookie-a*, u web okruženju opisana je u radu [211]. Pod sigurnim *cookie*-jem podrazumeva se *cookie* koji obezbeđuje autentifikaciju, integritet i poverljivost.

Jedan mogući scenario primene RBAC koncepta za kontrolu pristupa na više različitih web servera, odnosno aplikacija koje se izvršavaju na tim serverima, u okviru istog domena opisan je u [225].

Ahn i dr. u [9] opisali su mogući način uspostavljanja RBAC-a u postojećim web baziranim *workflow* apliakcijama. Predložena implementacija bazira se na na *user-pull* arhitekturi i X509v3 sertifikatima [125] koji sadrže podatke o ulogama korisnika. Osnovni cilj ove implementacije je da se realizuje kontrola pristupa sa što manjim izmenama u web baziranim *workflow* aplikacijama.

1.3.3 RBAC u operativnim sistemima

Osnovna motivacija za implementaciju RBAC modela kontrole pristupa za Solaris operativni sistem bila je da se tradicionalni UNIX *superuser* ili *root* zameni korisničkim ulogama. Za mnoge administrativne zadatke kod UNIX baziranih sistema potrebne su privilegije *root* korisnika. Ovakav nedostatak granularnosti u administraciji sistema otežao je administraciju, tj. otežana je raspodela administracije. RBAC model implementiran je tako da ne zahteva izmene postojećih aplikacija da bi se one oslonile na njega [97].

U ovoj ([97]) implementaciji RBAC-a korisnici i uloge predstavljeni su kao tipovi UNIX naloga (*UNIX accounts*). Međutim, uloga se ne može koristiti za primarno prijavljivanje na sistem, niti može biti korišćena od strane korisnika (uključujući i *root* korisnika), a da mu prethodno nije bila pridružena. Uloga

mora biti formalno pridružena korisniku da bi se mogle koristiti njoj pridružene privilegije. Npr. ako je kreirana "root" uloga kojoj su pridružena sva prava *superuser* korisnika, samo autorizovani korisnik koji je član "root" uloge i kome je ta uloga pridružena nakon prijavljivanja na sistem ima prava *superuser* korisnika. Ako korisnik pokuša da se prijavi na sistem pod "root" korisničkim imenom neće imati nikakve privilegije [97].

Implementacija uloga preko korisničkih naloga izvršena je koristeći postojeće mehanizme UNIX-a, bez izmene kernela operativnog sistema. Modul za autentifikaciju (*Pluggable Authentication Module*, PAM) je modifikovan (proširen) da bi mogao da radi sa korisničkim ulogama. Privilegije za nekoliko operacija i dalje su morale biti direktno pridruženo korisnicima, a ne ulogama kao posrednicima. Ove privilegije se prvenstveno odnose na startovanje operativnog sistema uključujući i proces zadužen za prijavljivanje korisnika [97].

Uloge korišćene u ovoj implementaciji nisu hijerarhijski organizovane. Za svaku ulogu moguće je definisati kardinalitet, tj. koliko korisnika može biti član te uloge. Takođe, za uloge se mogu definisati i statička SoD ograničenja, tj. mogu se definisati SoD relacije između uloga [97].

Osim u UNIX/LINUX baziranim operativnim sistemima, RBAC model kontrole pristupa sreće se i u Windows familiji operativnih sistema. U Windows Server 2003 operativnom sistemu ugrađena je podrška za RBAC model. Međutim, RBAC nije u potpunosti zamenio DAC model, već ova dva modela paralelno koegzistiraju. Jedna od najznačajnijih karakteritika ove implementacije je da podržava vrlo fleksibilnu kontrolu pristupa koja se vrlo lako može primeniti i na različite aplikacije, a ne samo za operativni sistem i njegove resurse [62].

1.3.4 RBAC u sistemima za upravljanje bazom podataka

Sistemi za upravljanje bazom podataka (SUBP) omogućuju kontrolu pristupa na različitim nivoima granularnosti, pa čak i kontrolu pristupa zavisnu od sadržaja. Obično se od SUBP-ova zahteva da obezbede kontrolu pristupa velikoj količini podataka od strane različitih korisnika, koji pripadaju različitim ulogama. Primena RBAC pristupa prikazana je kroz karakterističan primer Oracle SUBP.

Oracle SUBP poseduje određen broj predefinisanih uloga koje predstavljaju *sistemske uloge*. Uloge kreirane od strane administratora baze podataka

(*Database Administrator*, DBA) ili drugih korisnika za potrebe kontrole pristupa su *korisnički definisane uloge* [194]. Oracle poseduje tri osnovna tipa sistemskih uloga [194]:

- *CONNECT* uloga omogućuje prijavljivanje na bazu podataka i kreiranje sesije,
- *RESOURCE* uloga poseduje privilegije za kreiranje tabela i procedura, i
- *DBA* uloga ima privilegije da izvršava bilo koje operacije na Oracle serveru uključujući i mogućnost kreiranja uloga, dodeljivanja uloga korisnicima kao i dodeljivanje privilegija ulogama.

Prilikom kreiranja uloga može se navesti lozinka potrebna za aktiviranje uloge. Oracle dozvoljava da se ulozi pridruži više korisnika i da se korisniku dodeli više uloga. Ako je uloga pridružena korisniku sa administratorskom opcijom tada taj korisnik ima pravo adminstriranja te uloge, može da je menja, briše, dodeljuje drugim korisnicima, itd. Podržano je kreiranje hijerarhije uloga, ali nije moguće specificirati minimalan ili maksimalan broj korisnika koji mogu biti pridruženi ulozi kao ni uspostavljanje SoD relacija između uloga [194].

Korisnik, kome je dodeljena jedna ili više uloga, može uloge koje su mu pridružene da aktivira ili deaktivira u tekućoj korisničkoj sesiji. Ako je ulozi pridružena lozinka, korisnik mora da navede lozinku da bi tu ulogu aktivirao. Takođe, korisniku je dozvoljeno da, iz skupa uloga koje su mu pridružene, navede uloge koje će se automatski aktivirati prilikom njegovog prijavljivanja [194].

Privilegije se u Oracle-u mogu podeliti u dve osnovne kategorije: *sistemske privilegije* i *privilegije nad objektima* [194]:

- *Sistemske privilegije* su prava da se izvršavaju različite sistemske operacije (npr. komanda za kreiranje tabele) pri čemu nije specificiran objekat nad kojim se privilegija definiše, već se privilegija definiše za sve objekte odgovarajućeg tipa.
- S druge strane, *privilegije nad objektima* dozvoljavaju korisnicima (ulogama) da izvrše određenu operaciju nad konkretnim objektom (tabelom, pogledom, uskladištenom procedurom, itd).

Grup radova [36, 248, 249, 250] bavi se problemima bezbednosti i načinima njihovog rešavanja (korišćenjem RBAC modela) u *data warehouse* sistemima.

Autori u [250] kao najznačajnije probleme, sa stanovišta kontrole pristupa za *data warehouse* sisteme, identifikovali su slučajeve kada je različite podatke na različiti način potrebno prikazati korisnicima (podaci su multidimenzionalni), kao i potrebu za sprovođenjem kontrole pristupa na različitim nivoima granularnosti. Osim ova dva slučaja kao bitan problem navodi se i način definisanja prava pristupa u slučaju kada podaci u *data warehouse* dolaze iz različitih baza podataka, tj. na koji način je moguće preuzeti prava pristupa iz ovih sistema.

1.3.5 RBAC u programskim jezicima

Savremene tehnologije za razvoj višeslojnih aplikacija poput *Java Enterprise Eddition* (JEE) i *Microsoft .NET* imaju odgovarajuću podršku za RBAC baziranu kontrolu pristupa [111, 206]. U slučaju JEE tehnologije RBAC je implementiran kroz JAAS (*Java Authentication and Autorization Service*) [166]. JAAS omogućuje kontrolu pristupa EJB komponentama, odnosno kontrolu pristupa metodama EJB komponenti, ali je njime moguće realizovati i kontrolu pristupa web aplikacijama, tj. kontrolu pristupa web stranicama u celini ali i delovima stranica ukoliko je to potrebno. Kontrola pristupa dobrim delom sprovodi se na deklarativnom nivou ali ju je, za neke specifične slučajeve, moguće sprovoditi i na programskom nivou.

Trenutno, postupak definisanja prava pristupa svodi se na definisanje uloga u sistemu i identifikovanje koje od metoda svaka od definisanih uloga može da pozove. Iz ovoga proističe da je kontrola pristupa bazirana na operacijama nad komponenatama. Međutim, autori u [178] su uočili da je često namera prava pristupa da štite podatke (RBAC baziran na podacima), a ne metode objekata. Iz ovog proističe nekoliko faktora koji mogu komplikovati RBAC administraciju u ovakvim okruženjima [53]:

- U sistemima u kojima može da se definiše na oba nivoa, na nivou metoda koje se pozivaju i na nivou podataka kojima se pristupa, nekonzistentnost sa stanovišta bezbednosti može lakše da se pojavi. Npr. ulozi može biti zabranjen pristup nekom objektu, ali joj može biti dozvoljeno da pozove neku metodu koja pristupa tom objektu.
- Osoba koja razvija aplikaciju, često i nije osoba koja će izvršiti njeno postavljanje i podešavanje kod klijenta. Šta više, JEE specifikacija [165] eksplicitno razdvaja ove dve funkcije. Stoga administrator koji vrši kon-

figurisanje i postavljanje aplikacije verovatno u potpunosti ni ne zna koja metoda vrši modifikovanje kojih podataka.

U praksi, Java EE i .NET trenutno podržavaju samo RBAC baziran na operacijama. U radu [53] su predložene metode i predstavljen je alat koji pomaže administratorima da rasuđuju (analiziraju) o RBAC-u baziranom na operacijama posredstvom ekvivalentnog RBAC-a baziranog na podacima. Posredstvom predloženih metoda, odnosno razvijenog alata moguće je uočiti ekvivalentnosti između prava pristupa u RBAC-u baziranom na operacijama i prava pristupa u RBAC-u baziranom na operacijama.

1.4 Modeli kontrole pristupa bazirani na RBAC-u

RBAC model danas ima primenu u širokoj sferi različitih sistema i sigurno je jedan od najzastupljeniji modela kontrole pristupa. Sa porastom njegove primene pojavljuju se i različiti RBAC bazirani modeli, kako u naučnoj literaturi tako i u praktičnim rešenjima, koji pored osnovne RBAC funkcionalnosti dodaju i neke nove funkcije u cilju poboljšanja kontrole pristupa za pojedine oblasti. U ovom odeljku predstavljeni su RBAC bazirani modeli kontrole pristupa koji predstavljaju proširenje RBAC modela za potrebe efikasnije kontrole pristupa u određenim oblastima. Predstavljeni su temporalni i prostorni RBAC modeli i modeli bazirani na obligacijama i pravilima. U odeljku su takođe predstavljeni RBAC modeli za kontrolu privatnosti, *workflow* sisteme, XML dokumente, multimedijalne sisteme, i administraciju RBAC modela.

1.4.1 Temporalni RBAC modeli

Bertino i dr. u člancima [26, 27] identifikovali su potrebu da je u određenim situacijama ulogama potrebno dodeliti vremensku dimenziju. Uloge koje korisnik u okviru tekuće sesije može da aktivira nazvane su *omogućene* (*enabled*) uloge, a uloge koje ne može *onemogućene* (*disabled*) uloge. U sistemima gde pojedine uloge nisu uvek omogućene važno je uvesti postupak preko koga će se specificirati proces omogućavanja odnosno onemogućavanja uloga.

Kao rešenje ovih zahteva autori su predložili *Temporal RBAC* (TRBAC) model kontrole pristupa. TRBAC omogućuje periodično omogućavanje i onemogućavanje uloga kao i uspostavljanje zavisnosti između ovakvih akcija. Ova-

kve zavisnosti izražene su preko tzv. *trigera uloga* (*role triggers*). Aktiviranje ovih trigera može da omogući/onemogući ulogu momentalno ili nakon isteka specificiranog vremena. Akcijama omogućavanja/onemogućavanja može biti dat i prioritet što pruža zgodan način za rešavanje konflikta u slučaju da se istovremeno aktivira i omogućavanje i onemogućavanje uloge. Da bi administrator sistema zadužen za bezbednosti mogao da reaguje na vanredne situacije dozvoljeno mu je da dinamički promeni status odgovarajuće uloge posredstvom *run-time* zahteva.

U daljem istraživanju TRBAC modela uočeno je da on ne podržava nekoliko značajnih temporalnih ograničenja [151]. Model ne podržava temporalna ograničenja za relacije dodele uloga korisnicima i dodele privilegija ulogama. Za drugi nedostatak TRBAC modela autori u [151] navode da model ne podržava jasno definisano razlikovanje između omogućavanja uloge i aktivacije uloge. Kao posledica ovih ograničenja, TRBAC model nije u stanju da obradi nekoliko ograničenja koja se odnose na aktivaciju uloga, poput ograničenja na maksimalno vreme aktivacije uloge koje je dozvoljeno korisnicima kao i maksimalan broj aktivacija uloge od strane jednog korisnika u određenom vremenskom intervalu. Kako TRBAC ne razmatra trajanje ograničenja i ograničenja na aktivacije uloga on ne podržava ideju omogućavanja/onemogućavanja ograničenja. Aktivaciona ograničenja bi se trebala jasno definisati u odnosu na vreme omogućavanja uloge. Na kraju, TRBAC ne podržava vremenski zavisnu sematniku hijerarhije uloga i SoD ograničenja. U članku [151] autori su predložili *Generalised TRBAC* (GTRBAC) model koji preuzima osnovne osobine iz TRBAC modela i nadograđuje ih funkcionalnostima potrebnim za rešavanja prethodno navedenih nedostataka TRBAC-a.

U radu [169] uočena su neka ograničenja standardnog RBAC modela sa stanovišta objektno bazirane dinamičke separacije obaveza, tj. da u praksi postoje slučajevi dinamičke separacije obaveza koje nije moguće modelovati standardnim RBAC entitetima. Da bi se prevazišli ovi nedostaci autori predlažu proširenje RBAC modela sa dva nova entiteta (funkcionalnosti). Prvo predloženo proširenje je da se u RBAC model uključi predikat izvršenja (*execution predicate*) definisan kao $exec(s, op, obj)$. Ovim predikatom praktično je opisano da se u okviru sesije s izvršava (ili je izvršena ili će biti izvršena) operacija op nad objektom obj. Međutim, po autorima, za opis nekih realnih situacija ni predikat izvršenja nije dovoljan već je potrebno uvesti i redosled

izvršenja, tj. potrebna je istorija događaja, odnosno vremenska nota u RBAC modelu. Predložena je temporalna logika [163, 164] za predstavljanje vremenski zavisnih entiteta.

1.4.2 Prostorni RBAC modeli

Bertino i dr. u [35] analiziraju potrebene funkcionalnosti kontrole pristupa u GIS sistemima. Dat je obiman pregled zahteva koje kontrola pristupa u ovim sistemima treba da zadovolji. Kao značajan zahtev navodi se i potreba postojanja odgovarajućih sigurnosnih standarda koji će biti povezani sa postojećim geoprostornim standardima kako bi se obezbedila efikasna praktična realizacija mehanizmama za kontrolu pristupa.

U radovima [30, 83] opisan je *GEO-RBAC* model, ekstenzija RBAC modela koja omogućuje rad sa prostornim i lokacijski baziranim informacijama. Ovaj model se oslanja na OGC (Open GeoSpatial Consortium) model [66, 67, 68] za predstavljanje prostornih objekata, pozicija korisnika i geografski ograničenih uloga. Druga bitna krakteristika ovog modela je da podržava realne (fizičke) pozicije kao i logičke pozicije. Osnovni koncept GEO-RBAC sistema čini *prostorna uloga* definisana kao par (r, e), gde je r naziv uloge, a e *prostorno rastojanje (spatial extent)* uloge. Prostorno rastojanje definiše prostorne granice u kojima korisnik može da poseduje tu ulogu.

Skup svih geometrija sadržanih u referentnom prostoru (poligonu) označen je sa GEO, a sa referentni prostor sa MBB. Standardne (neprostorne) uloge se posmatraju kao podskup prostornih uloga koje kao prostorno rastojanje imaju čitav referentni prostor.

Za potrebe definisanja geometrijskog modela u GEO-RBAC-u uvedeni su sledeći formalizmi:

Sa FT_s i FT_{ns} označeni su skupovi prostornih tipova objekata (*spatial feature types*) i neprostornih tipova objekata (*nonspatial feature types*). Skup svih tipova objekata (*feature types*) definisan je sa: $FT = FT_s \cup FT_{ns}$. Slično tome sa F je označen skup svi objekata (*features*), F_s je skup svih prostornih *feature*-a, a F_{ns} je skup neprostornih feature-a u MBB.

Definicija 1.5 *Lokacija feature-a je fukcija: LocObj* : $F \rightarrow GEO \cup \{\perp\}$. *Za dati feature* $f \in F$ *lokacija je ili geometrija u GEO ako je* $f \in F_s$ *ili nedefinisano* ($\perp$) *ako je* $f \in F_{ns}$.

Definicija 1.6 *Neka je R skup svih naziva uloga i neka je $REXT\_FT \subseteq FT$ skup feature tipova prostornih rastojanja uloge. Sa $Ext : FT \rightarrow 2^F$ označeno je mapiranje tipova feature-a na same feature. Skup prostornih rastojanja uloge, $REXT$, je definisan kao $REXT = \bigcup_{ft \in REXT\_FT} Ext(ft)$.*

Definicija 1.7 *Parcijalno uređenje nad skupom FT, $\subseteq_{ft}$, definisano je kao $ft_i \subseteq_{ft} ft_j$ ako važi: $\forall f_i \in Ext(ft_i), \exists f_j \in Ext(ft_i)$ i $LocObj(f_i) \subseteq LocObj(f_j)$.*

Definicija 1.8 *Skup $RPOS \subseteq GEO$ predstavlja skup realnih pozicija u referentnom prostoru. Skup logičkih pozicija $LPOS$ sastoji se od feature-a čiji su tipovi u $LPOS\_FT \subseteq FT$, tj. $LPOS = \bigcup_{ft \in LPOS\_FT} Ext(ft)$.*

Definicija 1.9 *Funkcija za mapiranje lokacije, m_{ft} definisana je kao preslikavanje $m_{ft} : RPOS \rightarrow LPOS$. Funkcija m_{ft} za datu realnu poziciju, rp, vraća logičku poziciju koja odgovara instanci ft koja ima rp kao realnu poziciju.*

Centralna ideja GEO-RBAC modela je razlika između koncepta šeme uloge i instance uloge. Šema uloge definiše neka zajednička obeležja skupa prostorno zavisnih organizacionih funkcija sa sličnim značenjem. Šema uloge ne samo da definiše zajedničko ime za skup prostornih uloga već i ograničava prostor u kome je te uloge moguće aktivirati od strane korisnika. Štaviše, ona definiše i tip logičke pozicije. Instanca uloge je uloga koja zadovoljava ograničenja definisana na nivou šeme. Prostorna uloga ima naziv koji je isti kao i naziv šeme uloge pri čemu su prostorne granice uloge prostorni feature sa preciznom semantikom.

Definicija 1.10 *Šema uloge definisana je kao uređena torka (r, ext, loc, m_{loc}) gde je:*

- *$r \in R$, R - skup imena uloga,*
- *$ext \in REXT\_FT$,*
- *$loc \in LPOS\_FT$,*
- *$loc \subseteq_{ft} ext$,*
- *$m_{loc} \in M$ je funkcija za mapiranje lokacije za tip loc.*

Definicija 1.11 *Za datu šemu uloge r_s, instanca r_i je par (r, e), gde je $r = r_s.r$ i $e \in F$ tako da je tip feature-a od e je jednak $r_s.ext$*

U GEO-RBAC modelu privilegije se mogu dodeliti ili šemi uloga ili direktno instanci uloge. Ukoliko su privilegije dodeljene šemi uloge njih nasleđuju sve instance te šeme uloge. Kada se korisnik prijavi na sistem kreira se korisnička sesija i određen broj uloga je aktiviran, tj. uključen je u skup aktivnih uloga te sesije. Međutim, za razliku od standardnog RBAC modela, uloge je moguće dodeliti korisnicima samo kada je korisnik logički pozicioniran unutar prostornog rastojanja uloge.

Proširenje GEO-RBAC modela predstavlja *GEO-HRBAC* model [83] gde je GEO-RBAC model proširen hijerarhijom uloga. Definisane su dve hijerarhije uloga: *hijerarhija šema uloga* i *hijerarhija instanci uloga*. Na nivou šeme uloga hijerarhija dozvoljava da se definiše parcijalno uređenje između šema uloga. Identično, na nivou instanci, hijerarhija dozvoljava da se definiše parcijalno uređenje između instanci uloga.

U radovima [84, 85] razmatra se na koji način je moguće ostvariti kontinualnu kontrolu pristupa dok se korisnik kreće u nekom okruženju, odnosno kako uspostaviti kontrolu pristupa u mobilnim okruženjima. Da bi se rešio ovaj problem autori predlažu da prostorno zavisni modeli kontrole pristupa podrže ne samo zavisnost od lokacije već i kretanje korisnika u prostoru za vreme dok ona/on pristupa nekom resursu. Predloženo rešenje u [84] realizovano je kroz proširenje GEO-RBAC modela, tzv. *kontinualni GEO-RBAC (GEO–RBAC$_C$)*. Ključni koncept na kome je zasnovano proširenje u modelu *GEO–RBAC$_C$* je pojam kontinualne privilegije, koja omogućuje da se kontrola pristupa ne sprovede samo prilikom pristupa resursu, već da se kontinualno kontroliše. Kontinualna kontrola pristupa je jedna od najznačajniji osobina u *modelima kontrole korišćenja* (*Usage Control*, UCON) [197, 276].

U radu [78] RBAC model proširen je podrškom za prostorno zavisnu kontrolu pristupa i kontrolom pristupa zavisnom od atributa resursa i korisnika. Opisani model zasnovan je na GEO-RBAC modelu [30, 83], Ex-RBAC [79] i modelu zavisnom od atributa (*Role and Attribute Based Access Control model*) [59]. Praktično, privilegije su dodeljene ulogama zavisno od vrednosti specificiranih atributa resursa i korisnika. Takođe, obezbeđeno je maprianje između fizičkih i logičkih lokacija uz oslonac na Google Maps API [120]. U implementaciji modela autori su se oslonili na tehnologije semantičkog weba.

1.4.3 Prostorno-temporalni RBAC modeli

Grupa autora u radu [215] razmatra na koji način je potrebno definisati i sprovesti kontrolu pristupa digitalnih resursa unutar neke organizacije kada ona funkcioniše u normalnim i kriznim uslovima. Autori navode potrebu postojanja adaptivnog sistema kontrole pristupa koji treba da obezbedi različita prava pristupa kao i različiti način sprovođenja kontrole pristupa u normalnim i vanrednim situacijama. Osnovni zahtevi koje jedan takav sistem treba da zadovolji uključuju:

- promena vremenskih ograničenja, poput vremenskog perioda u okviru koga korisnik može izvršiti odgovarajuću aktivnost,
- promena prostornih ograničenja koja korisniku odobravaju/zabranjuju pristup sa određenih lokacija u kriznim uslovima, i
- primena poboljšanog (ojačanog) ili oslabljenog mehanizma za proveru identiteta korisnika.

Na osnovu ovih zahteva predložen je uopšteni, prostorno i vremenski zavisan RBAC (*Generalised Spatio-Temporal RBAC*, GST-RBAC) [17] kao mogući model za sprovođenje kontrole pristupa koji može da zadovolji prethodno navedene zahteve. Ovaj model predstavlja sintezu GTRBAC i GEO-RBAC modela. Kao jedan od najvećih izazova u specifikaciji i razvoju jednog takvog sistema navodi se problem definisanje prava pristupa za različite krizne situacije, tj. problem analiziranja različitih kriznih situacija i definisanja odgovarajućih prava pristupa.

Prostorno-temporalni model zasnovan na RBAC modelu, a prvenstveno namenjen za kontrolu pristupa u *pervasive* računarskim okruženjima opisan je u radovima [209, 210, 251]. U modelu su uvedena dva tipa lokacija: *fizička* i *logička* koje su definisane na sledeći način:

Definicija 1.12 *Fizička lokacija $PLoc_i$ je neprazni skup tačaka $\{p_1, p_2, p_3, \cdots, p_n\}$, gde je svaka tačka p_k predstavljena sa tri koordinate.*

Definicija 1.13 *Logička lokacija predstavlja apstrakciju fizičke lokacije.*

Funkcija m za maprianje fizičke lokacije na logičku i funkcija m' za obrnuto mapiranje definisane su na sledeći način:

Definicija 1.14 *Neka je P skup svih mogućih fizičkih lokacija, a L skup svih logičkih lokacija. Tada važi:*

- $m : P \rightarrow L$
- $m' : L \rightarrow P$
- *za svaku logičku lokaciju Loc_i važi: $m(m'(Loc_i)) = Loc_i$*
- *za svaku fizičku lokaciju $PLoc_i$ važi: $m'(m(PLoc_i)) = PLoc_i$*

Relacija sadržavanja jedne lokacije u drugoj formalno je predstavljena sledećom definicijom:

Definicija 1.15 *Fizička lokacija $ploc_j$ je sadržana u nekoj drugoj fizičkoj lokaciji $ploc_k$, u oznaci $ploc_j \subseteq ploc_k$, ako je zadovoljen uslov $\forall p_i \in ploc_j \Rightarrow p_i \in ploc_k$. Za dve logičke lokacije $lloc_j$ i $lloc_k$ važi relacija $lloc_j \subseteq lloc_k$ samo ako važi $m'(lloc_j) \subseteq m'(lloc_k)$.*

Vreme u ovom modelu može biti predstavljeno kao skup diskretnih tačaka (*vremenskih instanci*) ili kao *vremenski interval*.

Kako se lokacije korisnika i objekta mogu menjati tokom vremena uvedene su relacije $UserLocation(u, t)$ i $ObjLocation(o, t)$ koje za datog korisnika u, odnosno za dati objekat o u nekom vremenskom trenutku ili intervalu t određuju njegovu lokaciju.

Da bi se korisniku mogle dodeliti uloge on prvo mora da zadovolji prostorna i vremenska ograničenja potrebna za dodelu te uloge. Za ulogu se kaže da je *alocirana* kada su zadovoljena prostorna i vremenska ograničenja potrebna za njenu dodelu korisnicima. Funkcija $RoleAllocLoc(r)$ vraća skup lokacija u kojima uloga r može da se alocira, a funkcija $RoleAllocDur(r)$ vraća vremenske intervale u kojima uloga r može da se alocira. Za ulogu se kaže da je *omogućena* kada su zadovoljena prostorna i vremenska ograničenja potrebna za njenu aktivaciju. Funkcija $RoleEnableLoc(r)$ vraća lokacije u kojima uloga r može da se aktivira, a funkcija $RoleEnableDur(r)$ vraća vremenske intervale kada uloga može da se aktivira.

Predikat $UserRoleAssign(u, r, t, l)$ definiše da je korisniku u dodeljena uloga r za vreme intervala t na lokaciji l. Ovaj predikat je zadovoljen ako važi:

$$UserRoleAssign(u, r, t, l) \Rightarrow (UserLocation(u, t) = l) \wedge (l \subseteq RoleAllocLoc(r)) \wedge (t \subseteq RoleAllocDur(r))$$

Predikat $UserRoleActivate(u, r, t, l)$ je zadovoljen ako je korisnik u aktivirao ulogu r u intervalu t na lokaciji l, odnosno:

$$\begin{aligned} UserRoleActivate(u, r, t, l) \Rightarrow & (l \subseteq RoleEnabledLoc(r)) \wedge \\ & (t \subseteq RoleEnabledDur(r)) \wedge \\ & UserRoleAssign(u, r, t, l) \end{aligned}$$

Da bi i privilegija podržala prostorna i vremenska ograničenja dodeljena su joj tri dodatna entiteta: *lokacija korisnika*, *loakcija objekta* i *vreme*. Definisane su i tri nove funkcije koje određuju vrednosti ovih entiteta. Funkcija $PermRoleLoc(p, r)$ definiše dozvoljene lokacije na kojima korisnik kome je dodeljena uloga r mora da se nalaz da bi posedovao privilegiju p. Sa funkcijom $PermObjLoc(p, o)$ definisane su dozvoljene lokacije na kojima objekat o mora da se nalazi da bi korisnik mogao da poseduje privilegiju p koja mu omogućuje pristup tom objektu. $PermDur(p)$ definiše vreme u kome je dozvoljeno posedovanje privilegije.

Ovim modelom definisan je i predikat $PermRoleAcquire(p, r, t, l)$ koji je zadovoljen ako uloga r poseduje privilegiju p tokom vremena t na lokaciji l, tj:

$$\begin{aligned} PermRoleAcquire(p, r, t, l) \Rightarrow & (l \subseteq (PermRoleLoc(r) \cup RoleEnabledLoc(r))) \\ & \wedge (t \subseteq PermDur(r)) \wedge RoleEnabledDur(t))) \end{aligned}$$

Predikat $PermUserAcquire(u, o, p, l, t)$ definiše da korisnik u može da poseduje privilegiju p nad objektom o na lokaciji l za vreme t tj:

$$\begin{aligned} & PermRoleAcquire(p, r, t, l) \wedge UserRoleActivate(u, r, t, l) \wedge \\ & (ObjLocation(o, t) \subseteq PermObjLoc(p, o)) \Rightarrow PermUserAcquire(u, o, p, l, t) \end{aligned}$$

1.4.4 Obligacioni RBAC modeli

U različitoj literaturi [37, 38, 132, 168, 278] uočena je potreba proširenja modela kontrole pristupa sa *obligacijama*. Pod obligacijama se podrazumevaju zahtevi i zadaci koji treba da se izvrše prilikom odobrenja, i eventualno zabrane, izvršenja neke operacije nad resursom.

Minsky i Lockman u [168] prezentovali su motivaciju dodeljivanja obligacija privilegijama. Njihova osnovna ideja bila je da se narušavanje integriteta može tolerisati ako narušavanje može da se otkloni posredstvom obligacija u nekoj

razumnoj budućnosti. Autori su takođe predložili model u kome se obligacije dodeljuju privilegijama i kada je neka privilegija primenjena onda su i aktivirane njene obligacije.

U radu [132] predloženo je inkorporiranje obaveza u RBAC model, gde se pod obavezama podrazumevaju zadaci koje korisnik treba da izvrši. Ovo je u suštini veoma slično principu obligacija.

Bettini i dr. su u [37, 38] formalizovali prava pristupa sa obligacijama i pravilima primene, dozvoljavajući na taj način da se definišu akcije i uslovi koji treba da se izvrše (zadovolje) pre ili posle nego što korisnik iskoristi dodeljene mu privilegije. Takođe predložen je i model upravljanja obligacijama.

U radu [278] predložen je RBAC model proširen sa obligacijama, tako da se svakoj privilegiji koja je dodeljena ulozi može dodeliti skup obligacija. Na ovaj način ista privilegija dodeljena različitim ulogama može da poseduje različite obligacije za svaku od uloga, odnosno da ima iste obligacije za sve uloge. U opisanom modelu obligacije se sastoje od dva parametra: *akcije koja predstavlja obligaciju* i *vremenskog parametra*. Vremenski parametar definiše kada se obligacija može izvršiti. Na osnovu ovog parametra obligacije je moguće izvršiti pre izvršenja zahteva korisnika, po izvršenju zahteva korisnika (bez obzira da li je zahtev odobren ili odbijen) i zajedno sa zahtevom korisnika, tj. zahtev i obligacije se izvršavaju kao atomska celina. Kada je definisano više obligacija potrebno je primeniti algoritam za njihovo kombinovanje. Predloženi model definiše tri načina kombinovanja obligacija: *unija*, *bilo koja od obligacija* i *prva koja je primenjiva*.

XACML (*Extensible Access Control Markup language*) [185] jezik takođe omogućuje definisanje obligacija za prava pristupa. U XACML-u pod obilgacijama se podrazumevaju akcije koje moraju da se izvrše od strane PEP (*Policy Enforcement Point*) XACML modula u isto vreme kada se sprovodi odluka o dozvoli pristupa.

1.4.5 RBAC modeli bazirani na pravilima

U slučajevima kada sistem ima veliki broj korisnika često je nepraktično ili čak nemoguće administratoru da dodeli svakom od tih korisnika odgovarajuće uloge i da ih održava. U radu [10] opisan je model (*Rule Based RBAC*, RB-RBAC) koji omogućuje automatsku dodelu uloga korisnicima na osnovu definisanih pravila za dodelu. Ova pravila razmataju atribute (obeležja) ko-

risnika i definisana ograničenja na korišćenje uloga. Dat je način poređenja pravila i time je moguće uspostaviti relaciju poređenja između bilo koja dva pravila. Osim ove automatske (implicitne) dodele uloga i dalje je moguća eksplicitna dodela od strane administratora kao i kombinovana varijanta.

U radu [11] uočeno je da je moguće da se na osnovu relacija između pravila uspostavi *indukovana hijerarhija uloga* koja postoji paralelno sa postojećom.

Kern i Walhorn su identifikovali tri potencijalna nedostatka RB-RBAC modela [143]:

- za očekivati je da velike organizacije imaju značajan broj pravila, što značajno otežava pregled uloga (privilegija) dodeljenih korisnicima u bilo kom trenutku,
- posledica ovog otežanog pregleda naročito komplikuje vođenje evidencije šta je ko uradio u sistemu, što je jako bitno za određene sisteme, i
- često je komplikovano predvideti uticaj novog pravila ili izmenu postojećeg.

Za moguće rešenje ovog problema autori su predložili razdvajanje administracije pravila od administracije ostatka RBAC-a. Ova administracija pravila podrazumeva dve stvari [143]:

- Unos i brisanje korisnika kao i izmenu atributa postojećih korisnika u RBAC sistemu.
- Na osnovu atributa korisnika i definisanih pravila, odgovarajući proces, određuje koje se uloge trebaju dodeliti, odnosno ukloniti korisnicima što se evidentira u podsistemu za administraciju ostatka RBAC-a, tj. postoji eksplicitna dodela uloga korisnicima. Ovo predstavlja statičku varijantu dodele uloga korisnicima u odnosu na način definisan u modelu RB-RBAC.

Kao osnove prednosti ovog pristupa se navode [143]:

- RBAC sistem sadrži eksplicitnu dodelu uloga korisnicima, što značajno olakšava pregled privilegija koje korisnici poseduju,
- iz istog razloga evidencija šta je ko uradio je znatno pojednostavljena, i
- korišćenje pravila znatno pojednostavljuje administraciju čitavog sistema automatizacijom dodele uloga korisnicima i time smanjuje potencijalne greške u sistemu.

Nedostatak ovog modela u odnosu na RB-RBAC je to što se ovde pravila koriste mnogo statičnije, tj. izmena pravila ili atributa korisnika neće biti primenjena odmah već kada se startuje proces koji određuje dodelu uloga na osnovu atributa i pravla [143].

U radu [5] pokazano je kako se aktivne autorizacije, realizovane posredstvom unapređenih ECA (*Event-Condition-Action*) pravila, tzv. OWTE (*On-When-Then-Else*) pravila, mogu primeniti u različitim RBAC modelima kako u cilju efikasnije realizacije RBAC modela tako i za sprovođenje kontekstno zavisne kontrole pristupa.

Muhammad i dr. u radu [13] analiziraju primenu RBAC modela u SOA (*Service Oriented Architecture*) sistemima. Kao jedan od nedostataka RBAC–a u SOA sistemima autori su identifikovali nemogućnost dinamičke dodele privilegija ulogama, tj. da dodela privilegija ulogama ne bude statička već da zavisi od određenih ograničenja (*Permission Assignment Constraints*, PAC). Ukoliko su ova ograničenja zadovoljena određena privilegija će biti dodeljena ulozi, dok u protivnom neće. Za drugi bitan nedostatak navodi se nemogućnost postojanja parcijalnog nasleđivanja u hijerarhiji uloga, tj. da nadređena uloga nasledi samo neke od privilegija podređene uloge. Kao rešenje ovih nedostataka predložen je *Constraint Based RBAC* (CRBAC) model za kontrolu pristupa u SOA sistemima. Takođe, predstavljen je i jezik visokog nivoa za specifikaciju PAC ograničenja.

1.4.6 Distribuirani RBAC modeli

Wonohoesodo i Tari u radu [265] predlažu RBAC baziran model kontrole pristupa za web servis bazirana okruženja. Za razliku od tradicionalnih RBAC modela, gde su objekti pasivni entiteti, objekti u predloženom modelu su i aktivni i pasivni entiteti. Aktivni entiteti su servisi, a pasivni entiteti su parametri i povratne vrednosti servisa (atributi servisa). Model SWS-RBAC (*Single Web Services RBAC*) sprovodi kontrolu pristupa na dva nivoa: nivou servisa i nivou atributa. Kontrola pristupa na nivou servisa obezbeđuje da samo autorizovani korisnici mogu koristiti taj servis. Kontrola pristupa na nivou atributa primenjuje se tek kada je kontrola pristupa na nivou servisa dozvolila pristup servisu. Za svaki servis potrebno je definisati minimum privilegija za sve njegove atribute. Ovaj minimum privilegija se koristi za sprovođenje kontrole pristupa na nivou atributa. Da bi kontrola pristupa na nivou atributa dozvo-

lila pristup (izvršavanje servisa) privilegije koje su definisane za te atribute i dodeljene servisu moraju da budu podskup privilegija koje su dodeljene korisnikovim ulogama, tj. korisnik mora da sadrži sve privilegije za atribute koje su dodeljene servisu kome on pristupa. Osim SWS-RBAC modela autori su predložili i CWS-RBAC (*Composite Web Services RBAC*) model namenjen za kontrolu pristupa kompozitnih web servisa. Pod kompozitnim (globalnim) web serivsima podrazumevaju se web serivisi koji u cilju svog izvršenja pristupaju i drugim web servisima koji mogu da se nalaze i kod drugih provajdera, dok se pod lokalnim web servisim podrazumevaju servisi koji za svoje izvršenje ne koriste neke druge web servise. Sam CWS-RBAC model u osnovi zasnovan je na prethodno opisanom SWS-RBAC modelu koji je proširen sa podrškom za više servis provajdera kao i sa postojanjem globalnih i lokalnih servisa i uloga. Za pristup globalnim servisima koriste se globalne uloge, a lokalnim, lokalne, pri čemu postoji mapiranje globalnih uloga na lokalne. U cilju postizanja autonomije uloga između provajdera autori predlažu da provajder, čiji servisi se koriste u okviru globalnih servisa drugih provajdera, alociraju različiti skup uloga za različite provajdere. Ovi skupovi uloga treba da se sastoje od minimuma uloga potrebnih za poziv web servisa.

U današnje vreme web aplikacije i web servisi se često koriste za razmenu informacija između više organizacija, tj. posredstvom njih korisnici pristupaju resursima kako u svojoj tako i u organizacijama sa kojima njihova matična organizacija sarađuje. Bammigatti i Rao u radu [21] opisuju *GenericWA-RBAC* model namenjen za kontrolu pristupa u jednom takvom sistemu. Predloženi model uključuje pet osnovnih komponenti: *korisnike, uloge, operacije, objekte* i *organizacije* kao i međusobne veze između ovih komponenti. Uloge i objekti (privilegije) definišu se za svaku organizaciju. Ukoliko korisnik iz neke organizacije pristupa nekom objektu koji takođe pripada toj organizaciji pristup će mu biti odobren shodno ulogama koje su mu dodeljene i privilegijama koje te uloge poseduju. Međutim, ako korisnik pristupa nekom objektu koji ne pripada njegovoj matičnoj organizaciji tada se prvo vrši mapiranje uloga korisnika koje su mu dodeljene u njegovoj matičnoj organizaciji na uloge organizacije kojoj pripada objekat, a potom se proverava da li je tim ulogama dozvoljen pristup zahtevanom objektu.

Lamb i dr. su u [150] predložili model kontrole pristupa za složene sisteme koji obezbeđuju i integrišu podatke sa različitih izvora (*data service integra-*

tion). Konkretno, razvijen je sistem za kontrolu pristupa u okviru Health Data Integration (HDI) [4, 23] projekta koji obezbeđuje povezivanje, integraciju i analizu zdravstvenih (medicinskih) podataka za potrebe različitih istraživanja. HDI projekat realizovan je kroz skup web servisa koji obezbeđuje navedene funkcionalnosti nad podacima koji su pretežno smešteni u različite relacione baze podataka. Predloženi model kontrole pristupa baziran je na RBAC modelu koji je u osnovi proširen sa *projektom*. Za HDI sistem pretpostavljeno je da postoji više različitih projekata koji koriste podatke iz HDI sistema. Kada se korisnik autentifikuje, on takođe treba da specificira i za koji projekat se autentifikuje. Korisnički podaci, projekat za koji se autentifikovao kao i uloge koje poseduje su smešteni u SAML token [184] za potrebe korisničke sesije. Dati model omogućuje korisniku da poseduje različite uloge u različitim projektima, uloge sa istim imenom u različitim projektima mogu da imaju različite privilegije, tj. projekti vrše particionisanje uloga u različite prostore imena (*namespace*-ove).

Autori su predložili mapiranje datog modela na XACML [185] prava pristupa, odnosno XACML RBAC [183] profil. U tu svrhu dat je predlog kako bi se obezbedila podrška i za entitet *projekat* u XACML modelu. Radi pojednostavljenja kreranja XACML prava pristupa definisan je namenski jezik (XML baziran) za direktno predstavljanje prava pristupa definisana u skladu sa predloženim modelom. Međutim ovako kreirana prava pristupa se posredstvom XSLT [61] transformacija transformišu u XACML.

Primena GTRBAC modela (v. odeljak 1.4.1) za kontrolu pristupa u multidomenskim okruženjima gde je bitan uticaj vremenskog faktora opisan je u [205]. Prezentovani algoritam transformiše lokalna prava pristupa na takav način da ona još uvek održavaju originalne autorizacije za lokalne korisinke, pri čemu se odvajaju uloge koje se sada mogu koristiti od strane spoljašnjih entiteta koji pristupaju tom resursu.

Problem kontrole pristupa u distribuiranim sistemima koji funkcionišu na principu pretplate (jedna organizacija obezbeđuje resurse, a druge se pretplaćuju na te resurse) razmatran je u okviru [162]. Za kontrolu pristupa u ovakvim sistemima dat je predlog je distribuirane verzije RBAC modela – *DRBAC*. Predloženi koncept uvodi pojam distribuirane uloge. Organizacija koja obezbeđuje resurs će kreirati i eksportovati distribuirane uloge organizacijama koje su pretplaćene na resurse. Sa obe strane će distribuirane uloge biti

mapirane na lokalne uloge, ali će lokalnim ulogama na strani organizacije koja obezbeđuje resurs biti dodeljene odgovarajuće privilegije, a na strani organizacije koja se pretplatila na resurs lokalne uloge će biti dodeljene korisnicima. Na ovaj način administracija RBAC-a je distribuirana između organizacija. Autori su još definisali temporalna ograničenja koja obezbeđuju vremensku restrikciju na pristup resursu kao i ograničenja kardinaliteta koja ograničavaju maksimalan broj korisnika koji će pristupati resursima.

Coalition Based Access Control (CBAC) model za kontrolu pristupa u koalicionim okruženjima predstavljen je u [64]. Pod koalicijom se podrazumeva asocijacija partnerskih organizacija oformljena radi postizanja odgovarajućeg cilja/ciljeva. Cilj ovog modela je da obezbedi model kontrole pristupa koji će podržati semantiku između koalicionih veza i specifičnih karakteristika kontrole pristupa koje proističu iz jednog takvog okruženja. Autori su definisali četiri CBAC modela. $CBAC_{basic}$ predstavlja osnovni model koji defini osnovne entitete i njihove relacije, a baziran je na RBAC modelu. $CBAC_{team}$ dodaje na osnovni model podršku za timove, a baziran je na TMAC [246] modelu. $CBAC_{task}$ je nadogradnja $CBAC_{basic}$ za kontrolom pristupa zavisnom od zadatka (*task*-ova) koji se obavljaju. $CBAC_{team+task}$ predstavlja kombinaciju prethodna dva modela, tj. njihovu sintezu.

Kompozicija multidomenskih prava pristupa u jednom složenom heterogenom kolaborativnom sistemu može da bude veoma složen proces koji može da izazove postojanje različitih tipova konflikta, kao posledica različitih modela, šema, formata podataka i menahizama sprovođenja kontrole pristupa u svakom od pojedinačnih domena [31, 44, 87, 119, 208]. U članku [221] razmatrani su konflikti koji se javljaju prilikom integracije RBAC baziranih modela za kontrolu pristupa u multidomenskom okruženju. Takođe, predloženo je i okruženje za kompoziciju RBAC prava pristupa iz različitih domena.

1.4.7 RBAC modeli za kontrolu privatnosti

Danas je privatnost postala jedan od značajnih faktora u oblasti bezbednosti informacionih sistema. Privatnost informacija je i zakonski regulisana u većini razvijenih zemalja, npr. u SAD-u pravna akta poput *Health Insurance Portability and Accountability Act* (HIPAA) [188] za zdravstvo i *Gramm Leach Bliley Act* (GLB) [193] za finansijske institucije zahtevaju da institucija obezbedi privatnost njihovih korisnika.

Konvencionalni modeli kontrole pristupa, uključujući i RBAC nisu dizajnirani da sprovode kontrolu privatnosti informacija i da zadovolje odgovarajuće zahteve po pitanju privatnosti [108]. Organizacija za ekonomsku saradnju i razvoj (*Organisation for Economic Co-operation and Development*, OECD) u [109] dala je preporuke za obezbeđenje i sprovođenje privatnosti informacija. Kao najznačajniji faktori ističu se: *namena korišćenja* tj. da podaci prikupljeni za jednu namenu ne smeju da se koriste za drugu bez saglasnosti korisnika, *uslovi* koji moraju da se zadovolje pre izvršenja zahtevane operacije nad nekim resursom i *obaveze (obligacije)* koje predstavljaju akcije koje se moraju izvršiti po izvršenju zahtevane operacije.

Ni i dr. u radu [182] predstavili su *Privacy-aware RBAC* (P-RBAC) model za kontrolu privatnosti informacija. Sa ciljem nasleđivanja klasičnog RBAC model definisana je familija P-RBAC koja s sastoji od *osnovnog P-RBAC modela*, *hijearhijskog P-RBAC modela*, *uslovima proširenog P-RBAC modela* i *univerzalnog P-RBAC modela*.

Osnovni P-RBAC model (Core P-RBAC) definiše osnovne entitete i njhove međusobne relacije. U ovom modelu privilegija je proširena *namenama (purposes)*, *uslovima (conditions)* i *obavezama (obligations)* u skladu sa OECD principima. Stoga osim podatka i akcije koja se izvršava nad njim, privilegije eksplicitno navode namenu zajedno sa uslovima pod kojim privilegija može biti dodeljena i obavezama koje treba da na kraju da se ispune (izvrše).

Hijerarhijski P-RBAC (Hierarchical P-RBAC) uvodi hijerarhiju u okviru uloga, podataka i namene. Pod *hijerarhijom uloga*, kao i u standardnom RBAC modelu, podrazumeva se parcijalno uređenje između uloga. *Hijerarhija podataka* predstavlja stablo koje je parcijalno uređeno tako da svaki objekat podataka ima najviše jednog neposrednog nadređenog. Na identičan način, kao i hijerarhija podataka, uređena je i *hijerarhija namena korišćenja*.

Osnovni P-RBAC model omogućuje kreiranje samo jednostavnih uslova, dok uslovima proširen P-RBAC (Conditional P-RBAC) omogućuje definisanje znatno složenijih uslova.

Univerzalni P-RBAC (Universal P-RBAC) kombinuje funkcionalnosti prethodna dva, osnovnog i uslovima proširenog P-RBAC modela.

Peng i dr. u radu [201] predložili su model kontrole pristupa za zaštitu privatnosti baziran na RBAC modelu. Po autorima, trenutno u modelima za zaštitu privatnosti bitni faktori su *svrha* (*purpose*) zbog koje se pristupa

resursu, *uslovi* (*condition*) koji treba da se zadovolje pre nego što se odgovarajuća akcija izvrši nad resursom i *obazvze* (*obligations*) koje predstavljaju akcije koje je u nekim slučajevima neophodno izvršiti nakon izvršenja same operacije. Osnovne karakteristike predloženog, *Dynamic Purpose-based Access Control* (DPBAC) su:

- zahtev je predstavljen kao trojka $(objekat, akcija, svrha\ pristupa)$,
- uvedene su *uslovne uloge* čija aktivacija je dinamički implementirana zavisno od atributa subjekta i stanja sistema,
- uslovnim ulogama su dodeljene privilegije koje su predstavljene kao trojka $(objekat, akcija, svrha)$,
- svakom objektu su dinamički dodeljene moguće namene (svrhe) njegovog korišćenja u skladu sa tekućim podacima koji se odnose na privatnost, i
- korisniku će biti dozvoljen pristup ako poseduje odgovarajuću privilegiju (uloge koje su mu dinamički dodeljene poseduju odgovarajuću privilegiju) i ako je *svrha pristupa* navedena u zahtevu u skladu sa nekom od svrha korišćenja objekta.

1.4.8 Workflow RBAC modeli

U radu [274] je opisan *Task Role Based Access Control* (TRBAC) model zasnovan na RBAC modelu [105, 106] i na TBAC modelu [247]. TRBAC je konstruisan proširenjem RBAC-a TBAC-om. *Workflow* je definisan kao skup aktivnosti (*task*-ova) koji su međusobno povezani u cilju postizanja određenog cilja. *Task*-ovi mogu biti podeljeni u dve klase na osnovu toga da li oni pripadaju nekom *workflow*-u ili ne. Centralna ideja ovog modela je da je korisnik u relaciji sa privilegijama posredstvom uloge i *task*-a. TRBAC model podržava klasifikaciju *task*-ova na osnovu organizacione strukture i karakteristika kontrole pristupa u poslovnom sistemu. U *workflow*-u privilegije se mogu podeliti u dve grupe, privilegije koje se odnose na *workflow task* definiciju i privilegije koje se odnose na *workflow task* instancu. U skladu sa tim, postoje dve vrste dodele privilegija koje se dodeljuju definicijama *task*-ova odnosno instancama *task*-ova. Razlika između njih je u tome što dodela privilegija za *workflow task* definicije omogućuje definisanje samo statičkih ograničenja, dok dodela privilegija za instance omogućuje definisanje dinamičkih ograničenja.

Sličan model za kontrolu pristupa u *workflow* sistemima dali su Oh i Park u [189, 190, 191]. Njihov model baziran je na klasifikaciji funkcija koje se oba-

vljaju u organizaciji. Identifikovana su tri tipa *task*-ova: *workflow task*-ovi koji formiraju *workflow* proces, *supervizorski task*-ovi namenjeni za supervizorske funkcije u organizaciji i *privatni task*-ovi namenjeni za privatne funkcije koje nisu *workflow*-orijentisane. Identično kao i u prethodnom modelu, ulogama su privilegije dodeljeno posredstvom *task*-ova.

Lin i dr. su, na sličan način kao Oh i Park u [189, 191], predstavili model baziran na RBAC-u i TBAC-u koji definiše dve vrste *task*-ova: *task*-ovi koji su deo *workflow*-a i oni koji nisu [269]. Takođe, autori su u predloženom modelu identifikovali i različita ograničenja sa stanovišta bezbednosti.

Važnost vremenskog faktora u modelima kontole pristupa baziranih na *task*-ovima za *workflow* sisteme uočena je u radovima [115, 129, 279].

Predlog arhitekture za kontrolu pristupa u *realtime workflow* baziranim sistemima (sistemi u kojima je vremenski faktor značajan za njihovo pravilno izvršavanje) data je u radu [223]. Osnovni zahtev jednog ovakvog *workflow* sistema je da obezbedi dostupnost odgovarajućih podataka odgovarajućim osobama u odgovarajuće vreme. Osim ovoga bitno je da postoji mogućnost rekonfigurisanja *workflow* sistema ako odgovarajući zadatak ne može da se izvrši do kraja zahtevanog perioda. Kao model za kontrolu pristupa autori su iskoristili GTRBAC model (v. odeljak 1.4.1). Korišćenje GRBAC modela omogućuje da na proces sprovođenja kontrole pristupa utiče, između ostalog, i vremenska komponenta. Pored toga, podrška trigerima u GTRBAC-u omogućuje dinamičku adaptaciju *workflow*-a zavisno od odgovarajućih događaja koji su se desili u sistemu. U radu [222] opisan je mehanizam adaptacije *workflow* procesa, a da se pri tome zadovolje sva sigurnosna ograničenja koja su definisana.

Wei i dr. u radu [267] predložili su servisno-orijentisan model za kontrolu pristupa u *workflow* sistemima (*Service-Oriented Workflow Access Control*, SOWAC). Osnovna ideja predloženog modela je da se u što većoj meri obezbedi razdvajanje modela kontrole pristupa od *workflow* modela. Da bi to postigli autori su uveli entitet *servisa* kao apstrakciju privilegija i *workflow task*-ova. Za razliku od standardnog RBAC modela, u SOWAC modelu ulogama su dodeljeni servisi posredstvom kojih imaju odgovarajuće privilegije i prava da izvrše odgovarajuće *task*-ove.

Model kontrole pristupa za *workflow* sisteme zasnovan na RBAC modelu proširenim veremenskom komponentom i sigurnosnim nivoom (*security level*) opisan je u [275]. U prezentovanom modelu korisniku i privilegiji je dodeljen

odgovarajući sigurnosni nivo, a korisnik će moći da izvrši neki *task* ako mu je dodeljena odgovarajuća privilegija, posredstvom uloga, ako njegov sigurnosni nivo nije manji od sigurnosnog nivoa privilegije i ako je vreme izvršavanja odgovarajućeg *task*-a u okviru vremenskog perioda definisanog u toj prvilegiji. Predloženi model namenjen je za primenu u oblastima gde se zahteva veći stepen bezbednosti, npr. u vojne svrhe.

U članku [258] Wainer i dr. dali su model kontrole pristupa proširen sa *organizacionim jedinicama* i *slučajevima*. Razlog za uvođenjem organizacionih jedinica autori navode potrebu da je u realnim slučajevima potrebno znati kojoj organizacionoj jedinici korisnik pripada kao i ko mu je nadređeni, što po autorima nije uvek najzgodnije opisivati hijearhijom uloga. Slučaj praktično predstavlja instancu procesa, tj. mogućnost da se odgovarajući bezbednosni zahtevi referenciraju na instancu procesa. Kao potreba za uvođenjem slučaja navodi se problem da koncept sesije u RBAC-u nije baš najjasniji, što može značajno da utiče na interpretaciju dinamičkih ograničenja.

Wang i Zhang u radu [259] takođe ističu značaj postojanja organizacionih jedinica u modelu kontrole pristupa za *workflow* sisteme. Autori su predložili *Organization and Task Based Access Control Model* (OTBAC) model kontrole pristupa u kome su ulogama i organizacionim jedinicama dodeljeni *task*-ovi, a *task*-ovima odgovarajuće privilegije. Model definiše hijerarhiju organizacionih jedinica, a u okviru organizacione jedinice postoji hijerarhija uloga. Identifikovana su dva tipa *task*-ova: *zajednički* i *profesionalni*. Zajednički se dodeljuju odgovarajućoj organizacionoj jedinici (kojoj pripadaju korisnici koji imaju pravo da izvrše taj *task*), a moguće je nasleđivanje ovih *task*-ova kroz hijerarhiju organizacionih jedinica. Profesionalni se dodeljuju određenoj ulozi u određenoj organizacionoj jedinici, a njihovo nasleđivanje nije dozvoljeno.

RBAC model za kontrolu pristupa u *workflow* sistemima naveden u [242] uvodi koncept *aktivnosti* predstavljene kao torka ($p\_set, r\_set, ant\_set, cur\_sta$, a_type) gde je p_set skup privilegija dodeljenih aktivnosti, r_set je skup uloga kojima je dozvoljeno da izvrše aktivnost, ant_set je skup neposrednih podaktivnosti, cur_sta je tekuće stanje aktivnosti i a_type je tip aktivnosti.

Model kontrole pristupa za *workflow* sisteme opisan u [18, 19] zasnovan je na *autorizacionim šablonima (Authorization Template, AT)* preko kojih se definišu statički parametri autorizacija (prava pristupa) tokom faze dizajna

workflow-a. Svaki autorizacioni šablon je dodeljen nekom *task*-u, a prilikom izvršenja tog *task*-a dodeljeni šabloni se koriste za određivanje autorizacija.

Definicija 1.16 *Autorizacija je definisana kao četvorka* $A = (s, o, pr, [t_b, t_e])$ *gde je* s *subjekat kome je dozvoljen pristup objektu* o *sa privilegijama* pr *u periodu od trenutka* t_b *do trenutka* t_e.

Definicija 1.17 *Za dati task* tw_i autorizacioni šablon $AT(tw_i)$ *definisan je kao četvorka* $AT(tw_i) = ((r_i, -), (ot_i, -), pr_i, [t_b, t_e])$ *gde je:*

- $(r_i, -)$ - šablon subjekta *koji može da se zameni sa bilo kojim subjektom kome je dodeljena uloga* r_i.
- $(ot_i, -)$ - šablon objekta*, koji može da se zameni sa bilo kojim objektom tipa* ot_i.
- pr_i - privilegija *koja se dodeljuje subjektu* s_i *nad objektom* o_i.
- $[t_b, t_e]$ - vremenski interval *u okviru koga task treba da se izvrši.*

Primena ovog modela u web baziranom sistemu za kontrolu pristupa *workflow* baziranih aplikacija opisana je u [126].

Liu u Chen opisali su u [159] prošireni RBAC model kontrole pristupa za web servise u poslovnim procesima (*WS-RBAC4BP*) u kojima učestvuje više organizacija. U poslovnom procesu realizovanom pomoću web servisa različite organizacije obavljaju različite uloge i izvršavaju web servise dodeljene ulogama da bi se postigli određeni poslovni ciljevi. Ovde uloga predstavlja dinamički koncept - ona predstavlja dvosmernu poslovnu vezu između organizacija, odnosno uloga predstavlja odgovarajuću poslovnu funkciju u okviru konteksta poslovnog procesa. Iz prespektive organizacije, resursi kojima se pristupa u poslovnim procesima su web servisi. Na nivou poslovnog procesa dovoljno je da se poznaje od koje organizacije je potekao zahtev za izvršenje web servisa, pre nego koji individualni korisnik je uputio taj zahtev. Stoga se u ovom modelu organizacije, umesto korisnika, koriste subjekti. Kako postoji samo jedna operacija (poziv web servisa) element kojim su modelovane operacije u ovom modelu ne postoji.

Model kontrole pristupa predstavljen u [261] obezbeđuje kontrolu pristupa u SOA okruženjima za automatizovane aktivnosti koje izvršava web servis bez učešća korisnika, ali i za aktivnosti gde je potrebno učešće korisnika tj. obezbeđuje kontrolu pristupa za procese opisane WS-BPEL [186] i BPEL4People

[128] specifikacijama. U opisanom modelu uveden je entitet *servisa* (koji je ekvivalntan entitetu korisnika), a njime su predstavljeni web servisi u sistemu. Model definiše dve vrste uloga: *humane uloge* koje se dodeljuju korisnicima i *računarske uloge* koje se dodeljuju servisima. I jednim i drugim ulogama moguće je dodeliti odgovarajuće privilegije.

1.4.9 RBAC modeli za XML dokumente

Sve veća primena XML-a kao formata za modeliranje i razmenu podataka dovodi do izražaja problem kontrole pristupa XML dokumentima. Zbog hijerarhijske strukture XML dokumenata i mogućnosti da XML dokumenti mogu da sadrže podatke različitog stepena dostupnosti javila se potreba za modelima kontrole pristupa specifičnim za XML dokumente. U praksi postoje tri bitna zahteva koje kontrola pristupa XML dokumentima treba da zadovolji [146, 230]:

- Prvi zahtev je da se podrži kontrola pristupa na različitim nivoima granularnosti. U nekim situacijama isto pravo pristupa treba da važi za više dokumenata. U drugim slučajevima različita prava pristupa treba da se odnose na isti dokument, ali da eventualno budu primenjena na različite delove dokumenta. Mehanizam kontrole pristupa mora biti dovoljno fleksibilan da podrži različite nivoe granularnosti na koje se pravo pristupa odnosi.
- Drugi bitan zahtev za kontrolu pristupa XML dokumentima je da se obezbedi kontrola pristupa zavisna od sadržaja dokumenta. Ovo je jako bitan zahtev jer vrlo često dokumenti iste strukture imaju sadržaj različitog stepena sigurnosti. Da bi se ovo obezbedilo potrebno je omogućiti definisanje prava pristupa zavisnih od sadržaja dokumenta.
- Treći zahtev je da se obezbede prava pristupa različitog nivoa prioriteta. U slučaju da se više prava pristupa odnosi na isti dokument ili deo dokumenta primenjuje se pravo pristupa sa najvišim stepenom prioriteta. Ako postoji više prava pristupa jednakog prioriteta nad istim delom dokumenta primenjuje se predefinisano pravilo za razrešenje konflikta (npr. dozvola ima prednost, zabrana ima prednost).

U nastavku ovog odeljka dat je pregled reprezentativnih rešenja za kontrolu pristupa XML dokumentima.

Proširivi sistem za kontrolu pristupa XML dokumentima zasnovanu na korisničkim ulogama (*eXtensible XML Role-Based Access Control Framework*, XXACF) [228, 229, 231, 232] omogućava definisanje prava pristupa i sprovođenje kontrole pristupa na nivou uloga korisnika. Sistem obezbeđuje definisanje prava pristupa različitog stepena prioriteta na različitim nivoima granularnosti. Definisanje različitih nivoa prioriteta i stepena granularnosti pruža mogućnost za efikasno definisanje prava pristupa. Podržano je sprovođenje kontrole pristupa za različite operacije nad dokumentima kao i mogućnost različitih načina sprovođenja kontrole pristupa za istu operaciju. Naročiti značaj XXACF sistema je što analizira i slučajeve čitanja XML dokumenata u prisustvu XML šeme. XXACF pruža mogućnost za reprezentaciju prava pristupa u različitim formatima, odnosno njihovog skladištenja u različitim repozitorijumima podataka. Proširivost i modularnost XXACF sistema omogućuje proširenje osnovnog modela za kontrolu pristupa shodno specifičnim potrebama, kao i dodavanje novih modula u cilju sprovođenja konteksno zavisne kontrole pristupa. Konteksno zavisna kontrola pristupa pruža mogućnost za realizaciju kontrole pristupa koja je zavisna od okruženja u kome se XXACF sistem nalazi. Ovim je omogućena primena XXACF-a u razlicitim okruženjima. U radu [234] analizirana je primena ovog sistema u okviru bibliotečkog informacionog sistema BISIS [127], gde je XXACF iskorišćen za za kontrolu pristupa bibliografskim zapisima predstavljenim u XML formatu.

Kudo i Hada u radu [147] predstavili su RBAC baziran model kontrole pristupa XML dokumentima proširen sa dopunskim (*provisional*) modelom, gde se osnovnim operacijama pridružuju dopunske operacije. Ove operacije omogućuju da se pored izvršenja ili neizvršenja osnovne operacije, zavisno od prava pristupa, izvrše i neke dopunske (dodatne) operacije (npr. digitalno potpisivanje, logovanje, itd.). Predloženi model podržava sprovođenje kontrole pristupa prilikom čitanja i ažuriranja dokumenata.

U radu [49] predstavljen je model kontrole pristupa XML dokumentima u workflow okruženju. Predloženi model definiše dve vrste prava pristupa: prava pristupa koja dozvoljavaju pristup i prava pristupa koja ga zabranjuju. Podržane su operacije čitanja i ažuriranja XML dokumenta. Prava pristupa mogu se definisati na nivou dokumenta, ali i na nivou delova dokumenta.

Sigurnosni model za *native* XML baze podataka koje podržavaju XUpdate jezik predstavljen je u [114]. Model zahteva definisanje privilegija za izvršenje svake XUpdate operacije.

Fundulaki and Maneth predložili su u radu [113] jezik za specifikaciju prava pristupa nad XML dokumentima koji se takođe oslanja na XUpdate specifikaciju.

X-RBAC [41, 43] je sistem za kontrolu pristupa XML dokumentima (čitanje i ažuriranje) namenjen za primenu web servis okruženjima. Kontrola pristupa, pored RBAC entiteta, može da zavisi i od konteksta korisničke sesije kao i od sadržaja dokumenta. Prava pristupa mogu da se definišu na konceptualnom nivou, nivou šeme dokumenta, instnace dokumenta i nivou elementa dokumenta. Pored ovoga, omogućena je propagacija prava pristupa niz hijerarhiju dokumenta. Modelom nije podržano definisanje prava pristupa koja zabranjuju pristup.

XML ACP (Access Control Processor) [81, 82] predstavlja sistem za kontrolu pristupa XML dokumentima u web baziranim okruženjima. Sistem funkcioniše kao plug-in postojećim web serverima. Kontrola pristpa je moguća na nekoliko nivoa granularnosti: nivou DTD-a, nivou instance dokumenta i nivou odgovarajućeg elementa/atributa. Uvedena je hijerarhija prava pristupa od osam nivoa. Definisana prava pristupa mogu da propagiraju niz hijerarhiju dokumenta. U ovom modelu podržano je definisanje prava pristupa koja dozvoljavaju i zabranjuju pristup resursu.

Author-X sistem [28, 29, 32] obezbeđuje definisanje prava pristupa na različitim nivoima granularnosti koja mogu da dozvoljavaju ili zabranjuju pristup. Pored toga, podržana je kontrolisana propagacija prava pristupa, gde je moguće da prava pristupa definisana za DTD budu propagirana na semantički povezane dokumente ili DTD-ove. Sistem podržava rad u dva režima *pull* i *push*. U pull režimu korisnik eksplicitno zahteva dokument. Po prijemu zahteva Author-X formira dokument koji će sadržati samo delove vidljive za tog korisnika. U push režimu rada sistem periodično šalje dokumente svim korisnicima. Iako se isti dokument šalje svim korisnicima oni i dalje neće moći videti čitav dokument jer je on pre slanja šifrovan različitim ključevima. Svaki korisnik poseduje odgovarajuće ključeve u skladu sa pravima pristupa koja poseduje.

Fokus radova [75, 77, 167] je kako sprovesti kontrolu pristupa XML dokumentu nakon što je on već isporučen korisniku. Osnovna ideja je da se iskoriste kriptografske tehnike kojima će se različiti delovi dokumenta, u skladu sa pravima pristupa, biti šifrovani različitim ključevima.

Qi i dr. su u radu [207] predložili model kontrole pristupa XML dokumentima realizovan kroz funkcije pravila. Osnovna ideja je da se prava pristupa predstave kao skup pravila koja sprovode proveru pristupa dokumentu. Ova pravila predstavljaju fragment programskog koda kojim su enkapsulirana prava pristupa. Kao osnovna prednost ovakvog pristupa navodi se viskoka skalabilnost i performanse.

U radu [212] razmatrana je kontrola pristupa XML dokumentima koja može da zavisi od tekućeg sadržaja dokumenta, ali i od podataka iz istorije dokumenta. U ovom slučaju istorija dokumenta sadrži podatke o operacijama izvršenim nad dokumentom i podatke o delovima dokumenta koji originalno potiču iz nekih drugih dokumenata.

Byun and Park [52] predložili su dvofaznu šemu filtriranja kao mehanizam za sprovođenje kontrole pristupa nad XML dokumetnima. U prvoj fazi odabiraju se samo neophodna prava pristupa na osnovu pristiglog korisničkog upita. U drugoj fazi vrše se modifikacija korisničkog upita tako da on vrati samo rezultate dostupne korisniku.

1.4.10 Multimedijalni RBAC modeli

Kontrola pristupa za multimedijalne aplikacije razmatrana je u radu [196]. Po autorima, postojeći sistemi za kontrolu pristupa u multimedijalnim aplikacijama ne omogućuju kontrolu pristupa na dovoljnom nivou granularnosti, npr. kontrola pristupa na nivou određenih površina u frejmu. Kao drugi značajan nedostatak takvih sistema je to što podržavaju kontrolu pristupa za samo određen tip multimedijalnog sadržaja (npr. samo za video ili samo za sliku). Većina postojećih sistema bairana je na MAC (*Mandatory Access Control*) modelu kontrole pristupa što često nije dovoljno fleksibilno za realne potrebe. Autori su predložili *Criterion Based Role Based Multilayer Access Control* (CB-RBMAC) model za kontrolu pristupa za multimedijalne aplikacije koje koriste MPEG-7 standard. CB-RBMAC model zasnovan je na RBAC modelu koji je proširen *sigurnosnim kriterijumima* i *izrazima sigurnosnih kriterijuma*. Svim korisnicima i operacijama dodeljen je odgovarajući skup sigurnosnih kri-

terijuma, gde se pod sigurnosnim kriterijumom podrazumeva nekakvo obeležje (atribut) relavantno za sprovođenje kontrole pristupa. Objekti u sistemu su po MPEG-7 standardu hijererhijski organizovani, a svakom podobjektu dodeljen je izraz sigurnosnog kriterijuma (izraz sačinjen od sigurnosnih kriterijuma i logičkih operatora). Kada korisnik izvršava određenu operaciju nad nekim objektom vrši se izračunavanje izraza sigurnosnih kriterijuma i to u dva koraka. Prvo se svi sigurnosni kriterijumi u izrazu sigurnosnih kriterijum menjaju sa "tačno" ili "netačno" po sledećem pravilu: Svi sigurnosni kriterijumi u izrazu sigurnosnih kriterijuma koji nisu zajednički za korisnika i operaciju se menjaju sa netačnom logičkom vrednošću, dok se zajednički menjaju sa tačnom logičkom vrednošću. Sledeći korak je da se izračuna vrednost izraza sigurnosnog kriterijuma u skladu sa Bulovom algebrom. Ako je rezultat izračunavanja izraza tačan, znači da su kriterijumi zaštite podobjekta zadovoljeni, tj. da on nije dostupan. U suprotnom podobjekat je dostupan.

Potrebu za sofisticiranim mehanizmom kontrole pristupa u sistemima koji rade sa multimedijalnim sadržajem uočena je i u [57]. U radu je opisano okruženje za kontrolu pristupa slikama i video sadržaju *Secured Multimedia Application by adopting RBAC, XML and ORDBMS* (SMARXO). Opisani sistem omogućuje kontrolu pristupa na različitim nivoima granularnosti i za slike i za video. Model kontrole pristupa realizovan u SMARXO sistemu baziran je na RBAC modelu proširenim: *objektnim ulogama* (omogućuju grupisanje objekata), *temporalnim ulogama* (omogućuju vremenski zavisnu kontrolu pristupa) i *ulogama IP adresa* (omogućuju kontrolu pristupa zavisnu od računara sa koga se pristupa).

Model za kontrolu baza podataka medicinskih slika opisan je u [255]. Predloženi model zasnovan je na RBAC modelu proširenim ograničenjima za dodelu privilegija ulogama. Korišćenjem RBAC-a postignut je fleksibilan model kontrole pristupa, a upotrebom ograničenja kontrola pristupa je zavisna od sadržaja, domena (npr. određenog departmana zdravstevene ustanove), vremena i od relacije korisnik-pacijent (odnosno da samo određeni korisnici mogu pogledati slike pacijenta).

1.4.11 Administrativni RBAC modeli

Administracija RBAC sistema može da bude vrlo složen problem u slučaju kada je RBAC sistem primenjen za kontrolu pristupa u nekom složenom

poslovnom sistemu. Vrlo često administracija takvih sistema treba da bude decentralizovana, odnosno da više administratora vrši administraciju sistema, pri čemu svaki administrator može da vrši administraciju samo određenog podskupa korisnika, uloga i privilegija u sistemu (npr. svaka organizaciona jedinica ima svog administratora). Iz ovog proističe potreba za kontrolom pristupa i samih sistema koji sprovode kontrolu pristupa.

Kao jedan mogući model za kontrolu pristupa predložen je *Administrative Role Based Access Control Model* (ARBAC) [192, 217, 218] model. Ovaj model kontrole pristupa zasnovan je na RBAC modelu, a predstavlja namenski model kontrole pristupa za RBAC bazirane sisteme.

Osim ARBAC modela u literaturi se susreću još neki modeli zasnovani na ARBAC-u [89, 142, 154, 161, 238, 277] koji su namenjeni za administraciju nekih specifičnih modela zasnovanih na RBAC modelu, odnosno nadograđuju pojedine funkcionalnosti ARBAC modela za namenske potrebe.

Primena RBAC modela baziranog na ograničenjima (*Constraint Based RBAC*, CRBAC) [13] (v. odeljak 1.4.5) za administraciju RBAC-a opisana je u radu [12].

1.5 Modeli hijerarhije uloga

Uloge i hijerarhija uloga identifikovani su kao jedan od najkompleksnijih elemenata prilikom definisanja RBAC bazirane kontrole pristupa. Pararelno sa različitim RBAC baziranim modelima kontrole pristupa u literaturi se javljaju i različiti modeli formiranja hijerarhije uloga. U ovom odeljku dat je pregled različitih modela uloga i njihove hijerarhije.

U većini RBAC modela seniorska i njena juniorska uloga su povezane relacijom nasleđivanja koja ima dva semantička dela: *nasleđivanje privilegija* i *aktivaciju uloga* [216]. Semantika nasleđivanja privilegija dozvoljava seniorskoj ulozi da nasledi sve privilegije koje su dodeljene juniorskoj ulozi, dok semantika aktivacije uloga dozvoljava svim korisnicima kojima je dodeljena seniorska uloga da aktiviraju i juniorsku ulogu. Većina RBAC modela koristi kombinovanu semantiku hijerarhije koja dozvoljava i nasleđivanje privilegija i aktivaciju uloga. Shandu je pokazao da, u slučaju kombinovane semantike hijerarhije, određena SoD ograničenja ne mogu biti definisana na hijerarhijski povezanim ulogama [216]. Da bi prevazišao ova ograničenja Shandu je pred-

ložio *ER-RBAC* model koji podržava razlikovanje *hijerarhije korišćenja* (*usage hierarchy*) koja koristi isključivo semantiku nasleđivanja privilegija i *aktivacione hijerarhije* (*activation hierarchy*) koja koristi kombinovanu semantiku hijerarhije [216].

Kasnije, Joshi i dr. su uspostavili jasnu razliku između tri tipa hijerarhije uloga [135]: hijerarhija koja omogućuje samo nasleđivanje privilegija (*I-hijerarhija*), hijerarhija koja omogućuje samo aktivaciju uloga (*A-hijerarhija*) i kombinovana hijerarhija (*IA-hijerarhija*). Takođe, pokazano je da u realnim sistemima često nije najbolje rešenje da se koristi samo jedan tip hijerarhije između uloga, već su često potrebna sva tri tipa, tj. potreba da postoji *hibridna* hijerarhija uloga [135].

Primer hibridne hijerarhije uloga realizovan je u GTRBAC sistemu (v. odeljak 1.4.1) [134, 136, 137, 151]. Za potrebe formalnog opisa hijerarhije uloga u GTRBAC modelu definisani su predikati navedeni u tabeli 1.1.

Predikat	**Značenje**
enabled(r,t)	Dozovljena je dodela uloge r u vremenu t
u_assigned(u,r,t)	Korisniku u je dodeljena uloga r u vremenu t
p_assigned(p,r,t)	Privilegija p je dodeljena ulozi r u vremenu t
can_activate(u,r,t)	Korisnik u može da aktivira ulogu r u vremenu t
can_acquire(u,p,t)	Korisnik u može preuzeti privilegiju p u vremenu t
can_be_acquired(p,r,t)	Privilegija p može biti preuzeta posredstvom uloge r u vremenu t
active(u,r,s,t)	Uloga r je aktivna u sesiji s korisnika u u vremenu t
acquires(u,p,s,t)	Korisnik u u sesiji s poseduje privilegiju p u vremenu t

Tabela 1.1: Predikati za definisanje hijerarhije uloga

Veze između predhodno navedenih predikata definisane su sa sledeće četiri relacije. Takođe, ove relacije preciznije identifikuju semantiku preuzimanja privilegija i aktivaciju uloga [134, 135].

1. $p\_assigned(p, r, t) \Rightarrow can\_be\_acquired(p, r, t)$
2. $u\_assigned(u, r, t) \Rightarrow can\_activate(u, r, t)$
3. $can\_activate(u, r, t) \wedge can\_be\_acquired(p, r, t) \Rightarrow can\_acquire(u, p, t)$
4. $active(u, r, s, t) \wedge can\_be\_acquired(p, r, t) \Rightarrow acquires(u, p, s, t)$

Relacija (1) definiše da ako je privilegija dodeljena ulozi tada se ona može preuzeti preko te uloge. Relacijom (2) opisano je da svim korisnicima kojima je dodeljena neka uloga mogu tu ulogu i da aktiviraju. Sa relacijom (3) definiše se da ako korisnik može da aktivira ulogu tada mu mogu biti dodeljene sve privilegije koje se mogu preuzeti preko te uloge. Na kraju, relacija (4) opisuje da ako postoji korisnička sesija u kojoj je korisnik aktivirao određenu ulogu, tada korisnik poseduje sve privilegije koje se mogu pribaviti posredstvom te uloge.

Na osnovu predikata navedenih u tabeli 1.1 definisani su neograničeni *I*, *A* i *IA* tipovi hijerarhije uloga. Sledeće definicije ne razmatraju vremenski period kada je dozvoljeno **omogućavanje** hijerarhijski povezanih uloga (pretpostavlja se da je u svakom trenutku dovoljeno omogućavanje podređene i nadređene uloge) pa se stoga koristi termin "neograničen" [135, 137].

Definicija 1.18 Neograničena I-hijerarhija. *Neka su x i y uloge tako da važi $x \geq_i y$, tj. između x i y postoji samo relacija privilegijskog nasleđivanja u vremenu t, tada važi:*

$$\forall p, (x \geq_i y) \wedge can\_be\_acquired(p, y, t) \Rightarrow can\_be\_acquired(p, x, t)$$

Definicija 1.19 Neograničena A-hijerarhija. *Neka su x i y uloge tako da važi $x \geq_a y$, tj. između x i y postoji samo relacija aktivacijskog nasleđivanja u vremenu t, tada važi:*

$$\forall u, (x \geq_a y) \wedge can\_activate(u, x, t) \Rightarrow can\_activate(p, y, t)$$

Definicija 1.20 Neograničena IA-hijerarhija. *Neka su x i y uloge tako da važi $x \geq y$, tj. između x i y postoji relacija generalnog nasleđivanja u vremenu t,*

tada važi:

$$\forall p, \forall u, ((x \geq y) \wedge can\_be\_acquired(p, y, t) \Rightarrow can\_be\_acquired(p, x, t)) \\ \wedge ((x \geq y) \wedge can\_activate(u, x, t) \Rightarrow can\_activate(p, y, t))$$

Prva definicija govori da privilegije koje se mogu preuzeti posredstvom uloge x uključuju sve privilegije dodeljene ulozi x (po relaciji (1)) i sve privilegije koje se mogu preuzeti posredstvom uloge y. Drugom definicijom navedeno je da ako korisnik u može da aktivira ulogu x, a x je u *A-relaciji* sa y tada u takođe može da aktivira ulogu y, iako mu ta uloga nije eksplicitno dodeljena. Treća definicija u stvari predstavlja sintezu prethodne dve. Za bilo koju od hijerarhija važi i tranzitivnost.

Ako se razmatra i vreme kada je moguće aktivirati hijerarhijski povezane uloge tada postoji *slabo ograničena* i *čvrsto ograničena* forma hijerarhije. U tabeli 1.2 prikazani su slučajevi nasleđivanja za ove dve forme hijerarhije u disjunktnim vremenskim periodima τ_1 i τ_2, gde uloga r_1 nasleđuje ulogu r_2. U periodu τ_1, r_1 nje moguće aktivirati (*disabled*), a r_2 jeste (*enabled*), dok u periodu τ_2 važi obrnuto.

Period/ Tip hijerarhije	$\tau = \tau_1$ r_1 disabled, r_2 enabled	$\tau = \tau_2$ r_1 enabled, r_2 disabled
I slaba	Bez nasleđivanja u τ	Nasleđivanje privilegija u τ
I čvrsta	Bez nasleđivanja u τ	Bez nasleđivanja u τ
A slaba	Nasleđivanje aktivacije u τ	Bez nasleđivanja u τ
A čvrsta	Bez nasleđivanja u τ	Bez nasleđivanja u τ
IA slaba	Nasleđivanje aktivacije u τ	Nasleđivanje privilegija u τ
IA čvrsta	Bez nasleđivanja u τ	Bez nasleđivanja u τ

Tabela 1.2: Semantika nasleđivanja za različite hijerarhije uloga

Sun i dr. u radu [241] definišu model hijerarhije uloga za fleksibilni RBAC model za *workflow* sisteme opisan u [242] (v. odeljak 1.4.8). Aktivnosti su podeljene u dva tipa: *privatne* i *javne*. Privatne se koriste za opis privatnih funkcija, odnosno funkcija vezanih za odgovarajuće radno mesto i ne mogu se nasleđivati. Javne aktivnosti je moguće nasleđivati u hijerarhiji uloga. Definisana su dva stila nasleđivanja hijerarhije uloga sa stanovišta nasleđivanja aktivnosti: *normalno* i *ograničeno* nasleđivanje. Kod normalnog nasleđivanja

tip aktivnosti ostaje nepromenjen, dok kod ograničenog nasleđivanja tip aktivnosti koji je nasleđen od juniorske uloge postaje privatan da se aktivnost ne bi mogla dalje nasleđivati. Na osnovu ovoga moguće je da neka seniorska uloga parcijalno nasledi aktivnost juniorske uloge.

Botha i Eloff u radu [47] predložili su *tipiziranu* hijerarhiju uloga, zasnovanu na ulogama specificiranih tipova. Definisano je šest tipova uloga:

- *virtuelne uloge* - namenjene su da se uspostavi jedinstvena hijerarhija uloga, zbog efikasnije administracije takve strukture.
- *organizacione uloge* - predstavljaju organizacionu strukutru (obično vertikalnu),
- *poslovne uloge* - opisuju radna mesta korisnika u širem smislu (npr. sekretarica, menadžer, računovođa),
- *pozicione uloge* - odnose se na specifičnu poziciju u organizaciji (npr. finansijski menadžer),
- *privatne uloge* - zabranjuju nasleđivanje njima dodeljenih privilegija kroz hijerarhiju, i
- *task uloge* - predstavljaju uloge koje se odnose na specifične *task*-ove u organizaciji.

1.6 Modeli ograničenja

Pored statičkih i dinamičkih separacija obaveza definisanih standardnim RBAC modelom [103, 105, 106], kako u teoriji tako i u praksi pokazala se potreba i za nekim drugim modelima ograničenja. Ovaj odeljak sadrži opis najčešće korišćenih ograničenja (pored statičkih i dinamičkih SoD ograničenja) u RBAC baziranim modelima.

Gligor i dr. u radu [117] dali su detaljniju podelu SoD ograničenja u odnosu na statičku i dinamičku koja se sreće u većini RBAC literature. Autori su analizirali postojeće podele SoD ograničenja u različitoj literaturi [60, 103, 177, 219, 227] i sistematizovali su sledećih 11 različitih SoD ograničenja.

Statičko SoD (SSoD) ograničenje onemogućuje da isti korisnik bude član neke dve uloge. Za ovakve dve uloge se kaže da su *isključive (konfliktne)*.

Čvršća verzija prethodnog ograničenja definisana je *striktnim statičkim SoD* (SSSoD) ograničenjem koje kaže da a) korisnik ne može biti član dve

isključive uloge i b) da te uloge nisu autorizovane da izvrše bilo koje operacije nad istim objektom.

Jednokoračno striktno statičko SoD (1sSSSoD) ograničenje definiše da a) korisnik ne može biti član dve isključive uloge, b) bilo koja uloga autorizovana je da izvrši najviše jednu operaciju nad svakim od objekata i c) da te uloge nisu autorizovane da izvrše bilo koje operacije nad istim objektom.

Dinamičko SoD (DSoD) ograničenje onemogućuje da istom korisniku budu dodeljene dve isključive uloge u okviru iste sesije, tj. da korisnik ne može da aktivira dve isključive uloge.

Objektno bazirano dinamičko SoD (DSoD) ograničenje omogućuje da isključive uloge mogu imati zajedničke korisnike i da ih ti korisnici mogu aktivirati u isto vreme, ali ni jedan korisnik ne može da izvrši operaciju nad zahtevanim objektom, ako je prethodno već izvršio neku operaciju nad tim objektom.

Sledeća dva ograničenja predstavljaju statičku varijantu prethodnog.

Objektno bazirano statičko SoD (ObjSSoD) ograničenje dozvoljava da korisniku budu dodeljene isključive uloge, ali da te dodeljene uloge nisu autorizovane da izvrše više od jedne operacije nad svakim od objektata.

Objektno bazirano statičko SoD ograničenje po ulozi (RObjSSoD) je zadovoljeno ako nijedna uloga nije autorizovana da izvrši više od jedne operacije nad svakim objektom.

Još jedna fleksibilna alternativa statičkom SoD je operativno SoD ograničenje.

Operativno statičko SoD (OpSSoD) ograničenje dozvoljava da korisniku budu dodeljene isključive uloge, ali da te dodeljene uloge nisu autorizivane da izvrše sve operacije bez obzira na ciljni objekat.

Operativno statičko SoD ograničenje po ulozi (ROpSSoD) je zadovoljeno ako ne postoji uloga koja je autorizovana da izvrši sve operacije u sistemu, bez obzira na ciljni objekat.

Operativno statičko SoD ograničenje ima sledeću dinamičku varijantu:

Operativno dinamičko SoD (OpDSoD) ograničenje omogućuje da isključive uloge mogu imati zajedničke članove, i da ih ti članovi mogu istovremeno aktivirati, sve dok unija svih operacija koje uloge mogu da izvrše ne sadrži sve operacije potrebne da bi se izvršio neki poslovni proces.

Generalizacija operativnih i objektno baziranih dinamičkih SoD ograničenja predstavljena je kroz *Istorijski bazirano dinamičko SoD* (HDSoD) ograni-

čenje koje omogućuje da isti korisnik može da poseduje i aktivira isključive uloge, i da na osnovu toga može da izvrši sve operacije u okviru poslovnog procesa, ali sve operacije ne mogu da se izvrše nad istim objektom.

Simon i Zurko naveli su u [227] da HDSoD ograničenje praktično definiše sekvencu dozvoljenih ili obaveznih koraka i da se svaki korak mora sastojati od akcija *zavisnih* ili *nezavisnih* od redosleda. Akcije zavisne od redosleda, tj. ograničenja koja definišu zavisnost od redosleda, podrazumevaju da se određene akcije moraju izvršiti u određenom redosledu, dok kod akcija nezavisnih od redosleda to nije bitno.

U članku [145] data je kategorizacija SoD ograničenja u 7 kategorija: *statička SoD*, *dinamička SoD*, *objektno bazirana SoD (statička i dinamička)*, *operacijski bazirana SoD (statička i dinamička)* i *istorijski bazirana SoD ograničenja*. Takođe, predstavljene su tranzicije iz stanja jednog ograničenja u stanje drugog ograničenja. Analiziran je uticaj ovih ograničenja na proces delegacije uloga.

SoD ograničenja za GTRBAC (v. odeljak 1.4.1) model kontrole pristupa data su u [138, 151]. Specificirana ograničenja imaju i vremensku komponentu, tj. mogu biti zavisna od vremena. Definisana je *slaba* i *čvrsta* forma vremenski baziranih ograničenja. Slaba forma ograničenja podrazumeva da u okviru specificiranog intervala ne postoji neki trenutak u kome su konfliktne uloge dodeljene istom korisniku. Međutim, slaba forma ograničenja ne zabranjuje konfliktnim ulogmama da budu dodeljene istom korisniku u različitim vremenskim trenucima. Čvrsta forma statičkog SoD ograničenja podrazumeva da ako je u okviru posmatranog intervala u nekom trenutku korisniku dodeljena jedna od konfliktnih uloga, tada ni u jednom drugom trenutku tog perioda korisniku ne može biti dodeljena konfliktna uloga.

Ahn i Sandhu su u članku [8] predstavili *Role-Based Constraints Language 2000* (RCL 2000) jezik za formalno opisivanje ograničenja u RBAC modelima. Opisani su osnovni elementi, sintaksa i formalni osnov jezika. RCL 2000 predstavlja generalizaciju RSL 99 [7] jezika u cilju podrške različitim tipovima ograničenja, a ne samo statičkim i dinamičkim SoD ograničenjima. RCL 2000 je upotrebljen kao jezik za specifikaciju ograničenja u okruženju namenjenom za specifikaciju sigurnosnih modela, specifikaciju prava pristupa kao i validaciju specificiranih modela i prava [6].

Osim opisa RCL 2000 jezika, autori su identifikovali i 6 različitih SoD ograničenja, od kojih su neke autori prvi put eksplicitno definisali u literaturi. Navedena ograničenja su sledeća:

1. Nijednom korisniku ne smeju biti dodeljene dve međusobno konfliktne uloge, tj. konfliktne uloge ne smeju imati zajedničke korisnike.
2. Korisnik može posedovati, posredstvom dodeljenih mu uloga, najviše jednu konfliktnu privilegiju.
3. Prethodno ograničenje dozvoljava da uloga poseduje dve konfliktne privilegije i time ta uloga postaje beskorisna. Stoga je uvedeno ograničenje koje ne dozvoljava da dve konfliktne privilegije budu dodeljene istoj ulozi, ne razmatrajući pri tome korisnici-uloge relaciju.
4. Ako se konfliktnim privilegijama proširi ograničenje 1) dobija se ograničenje koje sprečava da se dve konfliktne privilegije dodele dvema nekonfliktnim ulogama. Neposredna posledica ovoga je da korisnik može imati najviše jednu konfliktnu privilegiju posredstvom uloga koje su mu dodeljene.
5. Ovo ograničenje uvodi pojam konfliktnih korisnika. Ograničenje definiše da dvojici konfliktnih korisnika ne mogu biti dodeljene uloge koje su međusobno konfliktne.
6. Poslednje ograničenje praktično predstavlja sintezu ograničenja 4) i 5), tj. onemogućuje da su konfliktni korisnicima dodeljene uloge iz skupa konfliktnih uloga, pri čemu konfliktne uloge mogu da imaju najviše jednu konfliktnu privilegiju.

Crampton je u radu [74] predložio model za specifikaciju ograničenja u RBAC modelu. Model je baziran na teoriji skupova i ima jednostavniju sintaksu nego li neki prethodno predloženi modeli opisani u [8, 130, 227]. Predloženi model podržava specifikaciju statičkih, dinamičkih i istorijskih SoD ograničenja. Formalno, ograničenja se definišu kao trojka (s, c, x), gde je s oblast (opseg) ograničenja, c je ograničavajući skup, a x je kontekst ograničenja i može biti statički (s), dinamički (d) i istorijski (h). Pokazano je da se ovim formalizmom mogu opisati sva ograničenja navedena u [130], a podeljena su u četiri osnovne grupe: *korisnički bazirana ograničenja*, *ograničenja bazirana na ulogama*, *ograničenja bazirana na privilegijama* i *ograničenja bazirana na objektima*.

U radu [76] dat je model za definisanje različitih ograničenja u *workflow* sistemima, pri čemu je model za definisanje ovih ograničenja nezavisan od samog mehanizma za sprovođenje kontrole pristupa. Predloženi model omogućuje definisanje SoD ograničenja i ograničenja kojima se zahteva da određene zadatke izvrši jedan te isti korisnik. Takođe, ovim modelom moguće je definisati i ograničenja koja se odnose na relativnu seniornost korisnika koji izvršavaju određene zadatke (npr. korisnik koji izvršava neki zadatak mora biti nadređeni, u organizacionoj strukturi firme, korisniku koji je izvršio neki drugi zadatak), kao i kontekstualna, korisnički zavisna ograničenja.

Bertino i dr. u člancima [33, 34] razmatraju SoD ograničenja u RBAC modelima primenjenim u *workflow* sistemima. Autori navode da, do tada, veoma mali broj RBAC modela koji se koristi u *workflow* sistemima (pa i u sistemima za upravljanje bazama podataka) ima podršku za različite tipove ograničenja uključujuči i SoD ograničenja. U ovom članku prezentovan je jezik (baziran na logičkim programima) za definisanje ograničenja za dodelu *task*-ova ulogama i korisnicima. Ovaj jezik, između ostalog, podržava i definisanje separacije obaveza. Takođe, dat je algoritam za proveru konzistentnosti ograničenja, kao i algoritam za planiranje dodele korisnika, odnosno uloga *task*-ovima na način da se ne narušavaju definisana ograničenja.

Liu et. al. u članku [158] razmatraju SoD ograničenja u *workflow* okruženjima, pri čemu se vodi računa o relacijama između *task*-ova kao i o međusobnoj zavisnosti *task*-ova (ako dva ili više *task*-ova pristupaju istim objektima).

Autori u [263] su uočili potrebu da je pored specifikacije ograničenja na nivou jedne instance *workflow* procesa potrebno definisati ograničenja koja se odnose više različitih instanci procesa (uključujući i različite vrste procesa). Takođe, uočena je i potreba da, u određenim situacijama, na ograničenja treba da utiče i istorija događaja. Predložen je i jezik za specifikaciju takvih ograničenja.

U modelu predstavljenom u [258] (v. odeljak 1.4.8) definisana su četiri tipa dinamičkih ograničenja:

- *dinamička SoD* ograničenja,
- *povezivanje obazveza* - predstavlja suprotnost SoD ograničenjima, a zahteva da dve operacije mora da izvrši ista osoba,
- *ograničenja između slučajeva (instanci procesa)* - omogućuju da se ograničenja odnose na različite instance procesa, i

- *recipročna separacija obaveza* - sprečava nedozvoljene koalicije, npr. dozvoljeno je da korisnik A u procesu P1 izvrši zadatak (*task*) T1, a korisnik B izvrši T2, ali ako se desio prethodni slučaj onda nije dozvoljeno da korisnik A u procesu P2 izvrši *task* T2, a korisnik B izvrši T1.

Predstavljeno je i proširenje modela ograničenja sa prioritetima, jer je uočeno da je u određenim situacijama neka ograničenja potrebno ignorisati (zaobići), dok za druga to nije dozvoljeno.

Sistem za administraciju SoD ograničenja (prvenstveno statičkih) opisan je u člancima [202, 203]. Sistem je baziran na tzv. "konfliktnim entitetima" i prvenstveno je namenjen za rad u *workflow* sistemima sa konfliktnim entitetima definisanim u [48], tj: *konfliktne privilegije*, *konfliktnim korisnici*, *konfliktne uloge* i *konfliktni task*-ovi.

Takabi i dr. u [243] dali su model za predstavljanje SoD ograničenja u RBAC-u korišćenjem *fuzzy* relacija. Koncept poverljivosti, koji je po svojoj prirodi *fuzzy*, je iskorišćen za predstavljanje ovog modela. Po autorima, u poređenju sa ne-*fuzzy* metodama, ova metoda je pragmatičnija i konzistentnija sa realnim slučajevima. Autori takođe smatraju da je ekspresivnost ove metode veća u odnosu na ne-*fuzzy* metode. Ograničenja su definsiana u odnosu na pojmove *korisnikove poverljivosti* i *stepena poverljivosti zahtevanog od uloge* opisane u [244] (v. odeljak 1.7.3). Definisana su tri tipa SoD ograničenja koja su predstavljena posredstvom *fuzzy* relacija:

Fuzzy Static Separation of Duty (FSSoD) ograničenje predstavljeno je kao $f_{ssod}(\{p_1, p_2, \cdots, p_n\}, td)$. $FSSoD \subseteq (2^{PRMS} \times TD)$ je kolekcija parova (ps, td) sa osobinom da samo skup korisnika koji zajedno imaju dovoljan stepen poverljivosti mogu da izvrše zadatak koji zahteva sve privilegije u $\{p_1, p_2, \cdots, p_n\}$. $PRMS$ je skup privilegija, a TD je skup stepena poverljivosti (i uloga i korisnika). Kod FSSoD ograničenja, suprotno SSoD ograničenjima, nije potrebno odrediti tačan broj korisnika. Dovoljno je proveriti za određeni skup korisnika da li imaju dovoljan stepen poverljivosti da izvrše određeni zadatak ili ne.

Fuzzy Statically Mutually Exclusive Role (FSMER) ograničenje je izraženo kao $f_{smer}(\{r_1, r_2, \cdots, r_n\}, td)$. $FSMER \subseteq (2^{ROLES} \times TD)$ je kolekcija parova (rs, td) sa osobinom da korisniku mogu biti dodeljene uloge iz $\{r_1, r_2, \cdots, r_n\}$ pri čemu njihov zajednički stepen poverljivoti nije veći od td. $ROLES$ je skup uloga u sistemu.

Fuzzy Dynamically Mutually Exclusive Role (FDMER) ograničenje je izraženo kao $f_{dmer}(\{r_1, r_2, \cdots, r_n\}, td)$. $FDSMER \subseteq (2^{ROLES} \times TD)$ je kolekcija parova (rs, td) sa osobinom da korisnik u sesiji može da aktivira uloge iz $\{r_1, r_2, \cdots, r_n\}$ čiji zajednički stepen poverljivoti nije veći od td.

U člancima [152, 155] autori su napravili jasnu razliku između pojma *separacije obaveza* i *međusobno isključivih uloga*. Po autorima separacija obaveza nije nešto što je vezano za RBAC model kontrole pristupa, već se ona može koristiti i u drugim modelima kontrole pristupa. Statičke međusobno isključive uloge *(Statically Mutually Exclusive Roles, SMER)* predstavlja mehanizam za sprovođenje statičkih SoD principa u RBAC modelu. Slično ovom, za dinamički SoD princip autori su definisali dinamičke međusobno isključive uloge (*Dynamically Mutually Exclusive Roles*, DMER) kao mehanizam za sprovođenje.

Metodologija za specifikaciju, verifikaciju i sprovođenje kontrole pristupa zasnovane na RBAC modelu predstavljena je u [235]. U ovom članku autori su pokazali da se različita autorizaciona ograničenja, kako vremenski nezavisna tako i vremenski zavisna mogu predstaviti preko različitih formalizama. Konkretno, autori su ponudili dva načina za predstavljanje ograničenja: preko linearne temporalne logike prvog reda (*Linear Temporal First-Order Logic*, LTL)[118] i preko UML (*Unified Modeling Language*)[15, 45, 214] i OCL (*Object Constraint Language*)[262] jezika. Takođe, pokazano je da se ovako specificirana ograničenja kasnije mogu i formalno verifikovati u cilju pronalaženja eventualnih konflikta odnosno ograničenja koja nedostaju.

Problem generisanja skupa ograničenja za definisanje statičkih SoD ograničenja razmatran je u [56]. Takođe razmatran je i način generisanja tih ograničenja da ona budu kompatibilna sa postojećom hijerarhijom uloga i da budu minimalna u smislu da ne postoji ni jedan drugi skup ograničenja koji je manje restriktivan, a da zadovoljava statička SoD ograničenja i da ta ograničenja budu kompatibilna sa hijerarhijom uloga.

1.7 Kontekstno zavisna kontrola pristupa

1.7.1 Kontekst i kontekstno zavisno računarstvo

Kada čovek komunicira sa čovekom, oni su u stanju da u komunikaciji implicitno koriste informacije trenutne situacije (*konteksta*) u kome se komu-

nikacija odvija kako bi ta komunikacija imala potpuni smisao. Na žalost, ova mogućnost ne prenosi se direktno na komunikaciju čovek-računar. U tradicionalnim računarskim sistemima korisnici imaju relativno improvizovan mehanizam za prosleđivanje informacija korisniku. Obezbeđujući računaru pristup kontekstu povećava se bogatstvo u načinu komunikacije čoveka i računara i pruža mogućnost da računar obezbeđuje korisnicima mnogo više korisnijih informacija [2].

U cilju efikasnog korišćenja konteksta potrebno je razumeti šta je kontekst, kako se može koristiti i potrebna je odgovarajuća arhitektura sistema. Razumevanje konteksta omogućuje dizajnerima aplikacije da odaberu koji kontekst da koriste u svojim aplikacijama. Razumevanje načina na koji se kontekst može iskoristiti omogućiće pravilan odabir kontekstno zavisnog ponašanja u aplikaciji. Na kraju, arhitektura jednog takvog sistema bi trebalo da obezbedi jednostavan i efikasan način njegove impelmentacije [90].

Kao tri osnovna aspekta konteksta autori u [220] navode: *"gde si" (where are you)*, *"s kim si" (who you are with)* i *"koji resursi su u blizini" (what resources are nearby)*. Pod kontekstno zavisnim računarstvom podrazumeva se sistem sposoban da se adaptira lokaciji korišćenja, korisnicima koji ga koriste ili se nalaze u okruženju sistema, računarima i ostalim uređajima od kojih je sistem sačinjen, kao i da reaguje na promene ovih faktora.

Autori u [2, 90, 91] daju sledeće definicije konteksta i kontekstno zavisnog računarstva:

Definicija 1.21 *Kontekst je bilo koja informacija koja se može koristiti za karakterizovanje situacije (stanja) entiteta. Entitet je osoba, mesto ili objekat koji se smatraju relevantnim za interakciju korisnika i aplikacije, uključujući samog korisnika i aplikaciju.*

Definicija 1.22 *Sistem je kontekstno zavisan ako koristi kontekst da bi obezbedio relevantne informacije i/ili servise korisniku, gde relevantnost zavisi od korisnikovih zadataka.*

Chen i Kutiz u [54] dali su sledeću definiciju konteksta:

Definicija 1.23 *Kontekst je skup stanja okruženja i podešavanja koji ili određuju ponašanje aplikacije ili u okviru kojih se desio odgovarajući događaj interesantan za korisnika.*

Na osnovu ove definicije konteksta data je definicija aktivnog i pasivnog kontekstno zavisnog računarstva:

Definicija 1.24 Aktivna kontekstna zavisnost - *Aplikacija je u stanju da se automatski adaptira detektovanom kontekstu promenom svog ponašanja.*

Definicija 1.25 Pasivna kontekstna zavisnost - *Aplikacija je u stanju da prezentuje novi ili promenjeni kontekst zainteresovanom korisniku ili da perzistira kontekst za kasniji pristup.*

Lokacija, korisnik, aktivnosti i računarski entiteti u [260] se navode kao fundamentalni entiteti koji čine kontekst. Za ove entiteti navodi se da su zajednički za modele konteksta u različitim sistemima.

Tripathi i dr. navode u članku [253] da kontekst može biti baziran na fizičkoj lokaciji korisnika, računarskom okruženju, uređajima koji se koriste prilikom izvršenja zadatka, zahtevima vezanim za različite koordinacije i vremenskim ograničenjima. Autori su klasifikovali kontekst u dve kategorije: *interni* i *eksterni*. Interni kontekst se odnosi na stanje različitih aktivnosti koje se izvršavaju u sistemu. Eksterni kontekst predstavljaju atributi koji se odnose na fizičko okruženje.

U radu [88] kontekst je zasnovan na pet semantičkih dimenzija, tzv. 4WH, definisanih u [3, 254]: *"ko" (who)*, *"gde" (where)*, *"kada" (when)*, *"šta" (what)* i *"kako" (how)*.

U praksi postoji više različitih mehanizama za predstavljanje (modelovanje konteksta). U radovima [176, 199, 239] identifikovano je šest najčešće korišćenih pristupa:

- modelovanje posredstvom parova ključ-vrednost (*key-value*),
- modelovanje XML baziranim jezicima,
- modelovanje grafičkim jezicima (uglavnom UML),
- objektno orijentisano modelovanje,
- modelovanje logičkim jezicima, i
- modelovanje korišćenjem ontologija.

1.7.2 Kontekstno zavisna bezbednost

Kontekstno zavisna bezbednost eksplicitno razmatra kontekst u specifikaciji i implementaciji sigurnosnih mehanizama (kontroli pristupa, određivanju sigurnosnih nivoa, itd.) [50, 170, 171, 172, 173]. Pojavom kontekstno zavisne

bezbednosti pojavili su se i novi problemi sa stanovišta bezbednosti. Kako je kontekst uglavnom dinamičan, tj. često se menja, potrebno je obezbediti da se bezbednosni mehanizmi adaptiraju tim promenama, tj. da budu u stanju da efikasno reaguju na promene konteksta. U [171] data je definicija *sigurnosnog konteksta* kao:

Definicija 1.26 *Sigurnosni kontekst je skup informacija, sakupljenih iz okruženja korisnika i aplikacije, koje su relevantne sa stanovišta bezbednosne arhitekture za korisnika i/ili za aplikaciju.*

1.7.3 RBAC bazirani kontekstno zavisni modeli

U literaturi postoji veliki broj istraživanja na temu kontekstno zavisne kontrole pristupa. Uglavnom, u ovim istraživanjima nastoji se da se prošire već postojeći modeli kontrole pristupa sa kontekstnim informacijama. Većina radova fokusirana je na ekstenziju RBAC modela kontrole pristupa. Pod konteksno zavisnim RBAC modelima mogu se podrazumevati i neki od modela predstavljenih u odeljku 1.4 (npr. temporalni model, prostorni modeli, modeli bazirani na pravilima) s obzirom da koriste "parametre okruženja" prilikom sprovođenja kontrole pristupa. U ovom odeljku dat je pregled kontekstno zavisnih modela kontrole pristupa baziranih na RBAC-u.

U radovima [71, 72, 174] predstavljen je *Generalised Role-Based Access Control* (GRBAC) model kontrole pristupa. Ovaj model predstavlja ekstenziju RBAC model koja omogućuje ne samo subjekat-orijentisan pogled na prava pristupa već i objekat-orijentisan i okruženje-orijentisan pogled, kao i kombinacije ova tri pogleda. GRBAC uklanja limitaciju RBAC modela (samo subjekat-orijentisan pogled) koristeći koncept uloge za organizovanje objekata nad kojim se sprovodi kontrola pristupa kao i za organizovanje okruženja u okviru koga se sprovodi kontrola pristupa. Po ovom modelu postoje tri tipa uloga: *uloge subjekta*, *uloge objekta* i *uloge okruženja*.

Uloge subjekta su analogne tradicionalnim RBAC ulogama. Jedina razlika između GRBAC-ovih uloga subjekta i tradicionalnih RBAC uloga je u načinu na koji se vrši provera prava pristupa. U tradicionalnom RBAC sistemu prava pristupa su u potpunosti bazirana na privilegijama dodeljenim ulogama koje subjekat poseduje. U GRBAC sistemu na pravo pristupa ne utiču samo uloge subjekta već i uloge objekta i uloge okruženja [72].

Uloge objekta omogućuju kreatoru prava pristupa da strukturira prava pristupa po obeležjima resursa sistema. Uloge objekta omogućuju da se po identifikovanju zajedničkih osobina nekih objekata oni grupišu u istu/iste uloge. Po grupisanju objekata moguće je vršiti kontrolu pristupa zavisnu od šeme grupisanja koja je korišćena.

Postoji mnogo realnih slučajeva gde kontrola pristupa ne zavisi samo od osobe koja prstupa nekom resursu već i od okruženja u kome se pristup vrši, tj. od stanja sistema. GRBAC omogućuje opisivanje stanja sistema posredstvom *uloga okruženja*. Uloge okruženja mogu da se baziraju na bilo kom stanju sistema koje može da se precizno evidentira. Aktivacija odnosno deaktivacija odgovarajuće uloge okruženja zavisi od stanja okruženja, tj. od odgovarajućih *uslova okruženja* koji utiču na aktivaciju/deaktivaciju određene uloge okruženja. Za definisanje uloga okruženja, njihove aktivacije, hijerarhije i ograničenja autori su iskoristili PROLOG-baziran jezik.

Da bi subjekat S pristupio objektu O, S mora da poseduje neku ulogu R_S tako da:

- postoji neka uloga objekta R_0 koju poseduje O,
- postoji neka uloga okruženja R_E koja je trenutno aktivna, i
- postoji neka neka privilegija P koja dozvoljava R_S da pristupi objektima u ulozi R_O kada je R_E aktivna.

Grupa autora u radu [123] daje predlog formalnog modela kontekstno zavisne kontrole pristupa. Kontekstno zavisna kontrola pristupa (CSAC) je definisana kao:

Definicija 1.27 $CASC = \{Context,\ Policy,\ Request,\ Algorithm\ Set\}$

Context predstavlja formalnu definiciju konteksta. Pod kontekstom autori podrazumevaju vreme, lokaciju, istoriju događaja, identitet i kalendar. Osim ovih faktora model konteksta treba da obezbedi jednostavan mehanizam za proširenje kako bi omogućio podršku i za neke druge faktore specifične za određene slučajeve.

Definicija 1.28 *Neka je kontekst definisan kao:* $Context = \{Time, Location, History, IdentityLinker, Calendar, Other\}$ *gde je:*

- *Vreme (Time) definisano je kao:* $Time = TimePoint \cup TimeRange \cup LoopTimePoint \cup LoopTimeRange$ *gde je:* $TimePoint$ *vremenski trenutak,* $TimRange$ *vremenski period,* $LoopTimePoint$ *vremenski trenutak koji se periodično ponavlja i* $LoopTimeRange$ *vremenski period koji se periodično ponavlja.*
- *Lokacija (Location) može biti fizička ili logička tj:* $Location = PhysicalLocation \cup LogicalLocation$
- *Istorija (History) predstavlja skup svih događaja koji su se desili u sistemu, tj.* $History = 2^{HistoryEntry}$*. Istorijski događaj (*$HistoryEntry$*) definisan je kao:* $HistoryEntry = Subject \times Operation \times Object \times Time \times Location \times Result$*. Istorijski događaj opisuje koji subjekt (*$Subject$*) je izvršio operaciju (*$Operation$*) nad objektom (*$Object$*) u kom kontekstu (*$Time$ *i* $Location$*) i šta je rezultat (*$Result$*).*
- $IdentityLinker = \{Name, PrincipalIdentity\}$*. Kontekstni podaci mogu mogu da pripadaju nekom korisniku, grupi korisnika, itd.* $IdentityLinker$ *omogućuje izražavanje te pripadnosti.* $Name$ *predstavlja identifikator resursa, a* $PrincipalIdentity$ *identifikator onoga kome resurs pripada.*
- *Kalendar (*$Calendar$*) predstavlja hronološki uređenu istoriju.*
- *Proširivost konteksta realizovana je kroz* $Other$ *element, koji u osnovi predstavlja skup (parametar, vrednost) parova.*

Definicija 1.29 *Pravo pristupa definisano je kao:* $Policy = Sign \times Subject \times Object \times Action \times ContextCondition \times ContextMask$ *gde je:*

- $Sign = \{+, -\}$ *znak prava pristupa, tj. da li pravo pristupa odobrava ili zabranjuje pristup.*
- $Subject$ *- subjekat na koga se pravo pristupa odnosi.*
- $Object$ *- objekat na koga se pravo pristupa odnosi.*
- $Action$ *- akcija na koju se pravo pristupa odnosi.*
- $ContextCondition \subseteq Context$ *određuje uslov primene prava pristupa. Pravo pristupa će biti primenjeno ako kontekst zadovoljava određene uslove.*
- $ContextMask$ *se odnosi na delove konteksta koji su dostupni za navedeno pravo pristupa.* $ContextMask$ *može da ubrza sprovođenje kontrole pristupa jer modul za sprovođenje kontrole pristupa ne mora da proverava svaki deo konteksta navedenog u zahtevu.*

U tradicionalnim sistemima za kontrolu pristupa zahtev mora da specificira ko želi da izvrši koju akciju nad kojim objektom. U kontekstno zavisnim sistemima za kontrolu pristupa pored ovih elemenata zahtev mora da obezbedi i odgovarajuće kontekstne informacije o subjektu, tj. zahtev je formalno definisan kao [123]:

Definicija 1.30 $Request = Subject \times Object \times Action \times SimpleContext$

U prethodnoj definiciji *SimpleContext* je podskup konteksta (*Context*) jer nije moguće u zahtevu obezbediti sve informacije iz konteksta.

Predložena arhitektura za sprovođenje kontrole pristupa data u [123] zasnovana je na tri algoritma, tj:

Definicija 1.31 *Skup algoritama (*Algorithm Set*) definisan je kao:* $Algorithm\ Set = \{Authorization, Revoke, AccessControlEvaluation\}$ *gde je:*

- *Authorization - algoritam za kreiranje novih prava pristupa,*
- *Revoke - algoritam za uklanjanje postojećih prava pristupa, i*
- *AccessControlEvaluation - algoritam za sprovođenje kontrole pristupa, tj. algoritam koji odlučuje da li je subjektu dozvoljeno da izvrši zahtevanu akciju nad zahtevanim objektom pri tekućem stanju konteksta.*

Kumar i dr. predložili su u članku [149] kontekstno zavisan RBAC model u kome su kontekstne informacije uvedene na dva nivoa: na nivou korisnika (sadrži informacije o korisniku relevantne sa aspekta bebednosti) i na nivou objekta (sadrži informacije o objektu relevantne sa aspekta bezbednosti).

Neumann i Strembeck u radovima [181, 240] analiziraju kontekstno zavisno proširenje RBAC-a kroz različita ograničenja definisana RBAC modelom. Autori navode da pored klasifikacije na *statička* (npr. statičko razdvajanje obaveza) i *dinamička* (npr. dinamičko razdvajanje obaveza) ograničenja moguća je i klasifikacija na *endogena* (*interna*) i *egzogena* (*spoljašnja*). Pod endogenim ograničenjima podrazumevaju se ona ograničenja koja se odnose na entitete RBAC modela. Npr. endogeno ograničenje može biti statičko razdvajanje obaveza jer se ono odnosi na uloge koje su jedne od RBAC entiteta. Za razliku od endogenih, egzogena ograničenja odnose se na entitet koji nisu osnovni RBAC entiteti već su oni definisani kao sporadični uslovi izvesnih operacija sistema

za kontrolu pristupa. Primer ovih ograničenja može biti vremensko ograničenje koje ograničava aktivaciju uloge za određeni vremenski period. Osim kategorizacije ograničenja na statička/dinamička i endogena/egzogena autori definišu podelu ograničenja i na *autorizaciona ograničenja* i *ograničenja dodele*. Autorizaciona ograničenja su ograničenja koja unose dodatnu logiku u procesu sprovođenja kontrole pristupa. Tako da, iako korisnik poseduje privilegije za izvršenje određene operacije, izvršenje će se odobriti isključivo ako su i odgovarajuća autorizaciona ograničenja zadovoljena. Npr. ovakva ograničenja mogu se primeniti da bi se implementirala kontrola pristupa zasnovana na istoriji pristupa. Ograničenja dodele su ograničenja kojima se kontroliše dodela privilegija ulogama (npr. maksimalni i minimalni kardinalitet, razdvajanje obaveza). Pojam *kontekstnog ograničenja* autori definišu kao uslove koje određeni atributi konteksta moraju da zadovolje da bi se odobrila zahtevana operacija. U odnosu na prethodno opisane kategorizacije kontekstna ograničenja su definisana kao dinamična egzogena autorizaciona ograničenja.

Kontekstno ograničenje po [240] spada u dinamička egzogena autorizaciona ograničenja, a definisano je preko pojma kontekstnog atributa, kontekstne funkcije i kontekstnog uslova.

Definicija 1.32 Kontekstni atribut *predstavlja odgovarajuće obeležje okruženja čija aktuelna vrednost može dinamički da se menja.*

Definicija 1.33 Kontekstna funkcija *je mehanizam za preuzimanje tekuće vrednosti određenog kontestnog atributa. Kontekstna funkcija može da ima i svoje parametre.*

Definicija 1.34 Kontekstni uslov *je predikat (logička funkcija) koja se sastoji od operatora i dva ili više operanada. Prvi operand je uvek određeni kontekstni atribut, dok drugi operand može biti kontekstni atribut ili konstantna vrednost.*

Definicija 1.35 Kontekstno ograničenje *je klauzula koja sadrži jedan ili više kontekstnih uslova. Ona je zadovoljena akko su svi kontekstni uslovi zadovoljeni.*

Kontekstna ograničenja se koriste za definisanje uslovnih privilegija. U skladu sa prethodno definisanim terminima, *uslovna privilegija* je privilegija

kojoj je pridruženo jedan ili više kontekstnih ograničenja. Predložena kontekstna ograničenja autori su uveli u xoRBAC [179] sistemu.

Bao i dr. predstavili su u [22] *Conditional RBAC* (C-RBAC) model kontrole pristupa. Ovaj model predstavlja proširenje RBAC modela uvođenjem *atributa uloge* i *konteksta* koji je opisan preko *kontekstnih atributa*. Osim ovih etiteta, C-RBAC uvodi i pojam *uslovne uloge* koja je definisana kao par (*uloga*, *uslovni izraz*), gde *uslovni izraz* predstavlja logički izraz koji se sastoji od atributa uloge, kontekstnih atributa i operatora $\{<, \leq, >, \geq, =, \neq, \in \wedge, \vee\}$. U C-RBAC-u pristup resursu je dozvoljen ako je za bilo koju uslovnu ulogu, uloge koja je dodeljena korisniku, dozvoljen pristup resursu i ako je zadovoljen uslov te uslovne uloge.

Predlog UML [15, 45, 214] modela za kontekstno bazirani model za kontrolu pristupa dat je u [100]. Predloženi model identifikuje dva tipa konteksta (logički i fizički), korisnike, resurse, profile korisnika, uređaja kao i servisa (zadataka) koje dati sistem obezbeđuje. Korisnicima su prava pristupa odgovarajućim resursima dodeljena posredstvom konteksta, tj. dozvola pristupa korisnika odgovarajućem resursu zavisi od prava pristupa definisanih za aktivne kontekste.

Sekvencu događaja, *striktne najmanje privilegije* i *separaciju obaveza* autori u radu [58] svrstavaju među najznačajnije zahteve kontrole pristupa u *workflow* sistemima. Da bi se obezbedili navedeni zahtevi predložen je kontekstno zavisan RBAC model za *workflow* okruženja. Pod kontekstnim informacijama u ovom modelu se podrazumevaju definicije i instance procesa i zadataka (*task*-ova) kao i njihove međusobne relacije. Uloga kojoj je dozvoljeno da izvrši odgovarajući *task* identifikovana je relacijom između *task*-ova i uloga. Takođe, između *task*-ova uvedena je relacija kojom su definisani međusobno konfliktni *task*-ovi. Korisnik će posedovati različite privilegije zavisno od tekućeg *task*-a koga izvršava. Ove privilegije ne samo da se razlikuju sa vremenom, već su uvek apsolutni minimum privilegija potrebnih da se izvrši *task*.

Georgiadis et. al. predstavili su u [116] *Context based TMAC* (C-TMAC) model za kontrolu pristupa u kolaborativnim sistemima. Ovaj model predstavlja integraciju TMAC (*Team-based Access Control*) [246] i RBAC koncepata zajedno sa kontekstnim informacijama. Korisniku su pored uloga dodeljeni i timovi, pri čemu tim predstavlja grupu korisnika, sa specifičnim ulogama, formiran u cilju završetka neke aktivnosti u određenom kontekstu. Međutim,

koncept timova se koristi i kao mehanizam povezivanja korisnika i konteksta, ekvivalentno kao što se uloge koriste za povezivanje korisnika i privilegija.

X-GTRBAC okruženje [39, 40, 42] zasnovano je na GTRBAC modelu [151] (v. odeljak 1.4.1), tj. predstavlja implementaciju GTRBAC modela i namenjen je za sprovođenje kontrole pristupa u različitim poslovnim sistemima. Za predstavljanje entiteta definisanih GTRBAC modelom i njihovih međusobnih relacija iskorišćen je XML jezik. Osim za korišćenje u poslovnim sistemima, u radovima [39, 40] pokazana je na koji način bi se X-GTRBAC okruženje moglo primeniti u web servis baziranim sistemima. Za potrebe primene u ovakvim sistemima model X-GTRBAC je proširen da podržava i kontekstno zavisnu kontrolu pristupa zasnovanu na temporalnim kontekstnim uslovima, ali i na kontekstnim uslovima koji nisu temporalni. Formalni model konteksta definisan u X-GTRBAC-u bazira se na GTRBAC modelu, a zasnovan je na sledećim entitetima.

- PN - je skup svih mogućih naziva za kontekstne parametre,
- RT - je skup svih mogućih tipova kontekstnih parametara,
- *kontekstni parametar* definisan je kao struktura podataka p koja ima sledeća obeležja $p \in PN, type \in PT$ i funkciju $getValue()$,
- *skup uloga* - $RR = \{rr_1, rr_2, \cdots, rr_k\}$, gde je $rr_i, i = 1, \cdots, k$ regularna uloga u GTRBAC modelu,
- *skup operacija* - $RO = \{ro_1, ro_2, \cdots, ro_k\}$, gde je $ro_i, i = 1, \cdots, k$ regularna operacija u GTRBAC modelu, i
- *skup servisa* - $SRVS = \{srv_1, srv_2, \cdots, srv_k\}$, gde je $srv_i, i = 1, \cdots, k$ servis, gde se pod pojmom servisa podrazumeva apstrakcija operacija koju obezbeđuje sistem nad nekim svojim resursima. Formalno servis je podskup skupa operacija RO.

Na osnovu ovih entiteta definisani su sledeći pojmovi:

Definicija 1.36 *Kontekstni skup C sastoji se od n kontekstnih parametara $\{p_1, \cdots, p_n\}, n \geq 0$, pri čemu važi $p_i.name \neq p_j.name$ za svaki p_i, p_j gde je $i \neq j, 1 \leq i, j \leq n$, tj. ne mogu da postoje dva parametra sa istim imenom.*

Definicija 1.37 *Zahtev za izvršenje servisa definisan je kao trojka $(role, srv, contextset)$ gde je $role \in RR$, $srv \in SRVS$, a contextset (kontekstni skup) je definisan u skladu sa prethodnom definicijom.*

Na osnovu pristiglog zahteva za izvršenjem servisa sistem određuje primenjiva prava pristupa za traženi servis. Ovaj postupak, pored navedene uloge i servisa zavisi i od ograničenja definisanih za navedenu ulogu kao i od definisanih kontekstualnih podataka. Za svaku ulogu moguće je definisati skup kontekstnih atributa kojima je definisan kontekstno zavisan uslov za tu ulogu. Vrednosti ovih atributa definiše administrator, a te vrednosti se porede sa vrednostima kontekstnih parametara navedenim u zahtevu u cilju određivanja primenljivih prava pristupa. Skup kontekstnih atributa uloge je u stvari podskup kontekstnog skupa C.

Rad [264] razmatra problem kontrole pristupa u web servis baziranim sistemima sa stanovišta kvaliteta (pouzdanosti) identifikacionog mehanizma kao kontekstno zavisnog parametra. Autori su predložili model kontrole pristupa zasnovan na RBAC modelu gde je u hijerarhiju uloga uključen i mehanizam identifikacije kojeg korisnik koristi. U sistemima (aplikacijama) koji poseduju različite mehanizme za proveru identiteta korisnika prava pristupa mogu da zavise od korišćenog mehanizma identifikacije. Npr. ako je korisnik identifikovan preko privatnog ključa i sertifikata može da ima veće privilegije u odnosu na slučaj kada je identifikovan preko korisničkog imena i lozinke. Kako su u klasičnom RBAC modelu različiti nivoi privilegija korisnika agregirani posredstvom uloga, mehanizam identifikacije korisnika osim na njegove privilegije utiče i na uloge i njihovu hijerarhiju. Zavisno od korišćenog mehanizma autentifikacije korisniku će u okviru tekuće sesije biti dodeljen odgovarajući skup uloga.

Kontekstno zavisan, servis-orijentisan RBAC model (*Context-aware Service-oriented RBAC*, CSRBAC) za kontrolu pristupa web servisima opisan je u radu [99]. U predloženom CSRBAC modelu ulogama se dodeljuju privilegije za pristup odgovarajućim servisima, gde se pod servisom podrazumeva apstrakcija operacija koje sistem obezbeđuje nad objektima. Kontekst u CSRBAC modelu predstavlja skup, sa stanovišta bezbednosti relevantnih, informacija u sistemu, npr. vreme, lokacija, prethodna stanja, itd. Odgovarajuća uloga, koja je dodeljena korisniku, će biti aktivna u okviru sesije tog korisnika ako su ispunjeni svi kontekstni uslovi potrebni za aktivaciju te uloge (ako navedeni kontekstni parametri zadovoljavaju zahtevane vrednosti).

Habio i Fan u radu [122] analiziraju problem bezbednosti u web servis baziranim aplikacijama, osvrćući se na problem kontekstno zavisne kontrole

pristupa. Rad prikazuje kontekstno zavisan RBAC model za kontrolu pristupa, namenjen prvenstveno za web servise (*Context-aware Role-based Access Control*, CGRBAC). Opisani model rešava problem kontrole pristupa za procese realizovane posredstvom kompozitnih (globalnih) web servisa. Pod kompozitnim (globalnim) web servisima podrazumevaju se web servisi realizovani od atomskih (lokalnih) web servisa i/ili drugih kompozitnih web servisa. Dok se za atomske web servise prava pristupa mogu specificirati direktno, za kompozitne to nije slučaj. Prava pristupa za kompozitne web servise zavise od njegovih atomskih web servisa kao i od samog kompozitnog web servisa. U datom modelu objekti i operacije su zamenjeni *servisima*. Servis predstavlja skup operacija koje se mogu izvršiti nad određenim objektom, tj. servis predstavlja apstrakciju operacija nad nekim objektom koje dati sistem obezbeđuje. Uloge u modelu su podeljene na globalne i lokalne uloge. Korisniku su dodeljene globalne uloge, a globalnim ulogama se dodeljuju privilegije za izvršenje odgovarajućih globalnih servisa. U slučaju da globalni servis poziva neki lokalni servis, potrebno je izvršiti mapiranje globalnih uloga korisnika na lokalne uloge korisnika pošto su globalne uloge poznate samo provajderu globalnog servisa, ali ne i provajderima lokalnih servisa. Stoga je RBAC model proširen i relacijom koja opisuje mapiranje globalnih uloga nekog provajdera na lokalne uloge drugih provajdera. Aktivacija globalnih uloga za korisnika zavisi od stanja okruženja, tj. od definisanih parametara okruženja, dok aktivacija lokalnih uloga zavisi od toga da li je pozvan lokalni servis za koga su te uloge definisane.

Problem kontrole pristupa u serivs-orijentisanim autonomnim decentralizovanim sistemima razmatran je u radu [271]. U datom radu predložen je model i arhitektura sistema za kontrolu pristupa zavisnog od situacije (*Situation Aware Access Control*, SA-AC). Predloženi SA-AC model baziran je na RBAC modelu koji je proširen sa situacijski zavisnim ograničenjima (SA ograničenja) na dodelu uloga korisnicima i dodelu privilegija ulogama. Formalno, SA ograničenja su modelovana na sledeći način:

- U - je skup svih korisnika.
- R - je skup svih uloga.
- P - je skup svih privilegija.
- $UR \subseteq U \times R$ - je skup svih korisnik-uloga dodela.
- $RP \subseteq R \times P$ - je skup svih loga-privilegija dodela.
- SE - je skup svih izraza kojima se opisuju situacije.

- $SEUR \subseteq 2^{SE} \times UR$ - je skup situacijski zavisnih korisnik-uloga dodela. $seur = (L_{se}, (u, r))$, $seur \in SEUR$) definiše da ako su samo svi situacijski izrazi (situacije) u listi $L_{se} \subseteq SE$ zadovoljeni (tačni) tada je dodela $(u, r) \in UR$ aktivna.
- $SERP \subseteq 2^{SE} \times RP$ - je skup situacijski zavisnih uloga-privilegija dodela. $serp = (L_{se}, (r, p))$, $serp \in SERP$) definiše da ako su samo svi situacijski izrazi (situacije) u listi $L_{se} \subseteq SE$ zadovoljeni (tačni) tada je dodela $(r, p) \in RP$ aktivna.

Kapasails i dr. u radu [141] predstavili su dinamičku, kontekstno zavisnu arhitekturu modula za kontrolu pristupa u web servis baziranim okruženjima. Mehanizam kontrole pristupa baziran je na RBAC modelu u koji su inkorporirane dinamičke kontekstne informacije u formi kontekstnih ograničenja.

Definicija 1.38 *Kontekst je definisan i kategorisan kao:* korisnički kontekst *koji predstavlja status korisnika koji pristupa resursu,* kontekst resursa *koga čini status resursa kome se pristupa,* kontekst okruženja *koji predstavlja status bilo kojih entiteta relevantnih za specifični zahtev i* istorijski kontekst *koji se sastoji od specifičnih, prethodno izvršenih događaja i situacija, koji konstituišu dodatnu dimenziju kontekstnih informacija uključujući istoriju vezanu za korisnike, resurse i okruženje.*

RBAC model je proširen kontekstno zavisnom kontrolom pristupa tako da korisnik koji pripada određenoj ulozi ima određenu privilegiju dodeljenu toj ulozi samo ako i zadovoljava navedeno *kontekstno ograničenje*. Kontekstno ograničenje je definisano kroz pojmove *kontekstnih podataka*, *akvizicije kontekstnih podataka* i *kontekstnog uslova*, pri čemu kontekstni uslov može biti prost i složen. *Kontekstni podatak* predstavlja bilo koji podatak relevantan za entitete kontrole pristupa. *Akvizicija kontekstnih podataka* je u stvari mehanizam pomoću koga su kontekstni podaci sakupljeni. *Prosti kontekstni uslov* je predikat koji proverava validnost uslova i sastoji se od operatora i dva operanda. Prvi operand je uvek kontekstni podatak, dok drugi takođe može biti kontekstni podatak ili predefinisana vrednost. Prosti kontekstni uslov uvek vraća logičku vrednost (tačno ili netačno). *Kompozitni kontekstni uslov* se definiše kao regularni izraz koji se sastoji od logičkih operatora i jednog ili više operanda, pri čemu operandi mogu biti prosti kontekstni uslovi ili drugi složeni kontekstni

uslovi. Kontekstno ograničenje sastoji se od jednog ili više prostih i/ili složenih kontekstnih uslova.

Kontekstno zavisni dinamički RBAC bazirani model (*Context Based Dynamic RBAC*, CDACM) za kontrolu pristupa u web servis okruženjima opisan je u radu [224]. Osnovne karakteristike ovog modela su sledeće [224]:

- mogućnost formiranja hijerarhije privilegija (privilegija sadrži privilegije nižeg nivoa),
- Kontrola pristupa se sprovodi na dva nivoa; nivou servisa i nivou atributa (parametara/povratne vrednost) servisa, i
- ulogama su dodeljeni odgovarajući konteksti, a kontekstima privilegije, tj. dodela privilegija ulogama je posredstvom odgovarajućeg konteksta (kontekstnog ograničenja). Kontekst je definisan kao skup atributa, kontekstni uslov kao logički izraz koji sadrži kontekstni atribut, a kontekstno ograničenja kao skup kontektnih uslova.

U radu [59] predstavljen je kontekstno zavisan model kontrole pristupa dizajniran i realizovan korišćenjem tehnologija semantičkog web-a. Dati sistem omogućuje dinamičku dodelu uloga korisnicima na osnovu korisničkih atributa i atributa okruženja.

Ubiquitous Context-based Security Middleware (UbiCOSM) sistem za sprovođenje kontrole pristupa u heterogenim mobilnim računarskim okruženjima opisan je radovima [69, 70]. Za razliku od klasičnog RBAC pristupa gde uloge predstavljaju posrednika između korisnika i privilegija, u UbiCOSM sistemu indirektna relacija između korisnika i privilegija ostvarena je posredstvom konteksta. Za svaki kontekst definiše se skup privilegija koji mu je pridružen. Kada se subjekat nalazi unutar odgovarajućeg konteksta, za njega se automatski aktivira skup privilegija koje su aktivne za navedeni kontekst. Prilikom promene konteksta privilegije prethodnog konteksta se uklanjaju, dok se privilegije novog konteksta dodeljuju subjektu. Opisani sistem razlikuje dva različita tipa konteksta: *fizički* i *logički*. Fizički kontekst identifikuje fizički prostor, ograničen određenim geografskim koordinatama. U bilo kom vremenskom trenutku korisnik može da pripada (da se nalazi u) samo jednom fizičkom kontekstu. Logički kontekst identifikuje logičko stanje fizičkih konteksta ali i ostalih entiteta koje čine dati sistem (npr. korisnici, servis, itd.) Za razliku od fizičkog konteksta, korisnik u jednom vremenskom trenutku može pripadati većem broju logičkih konteksta. UbiCOSM posmatra identitet korisnika i korisničke uloge

kao specifične tipove logičkog konteksta što omogućuje da se sistem koristi i kao subjekat-baziran sistem za kontrolu pristupa (npr. kao RBAC sistem). Prava pristupa su definisana kao torka (*association_name(context_collection)*, *permissions*). Prvi argument identifikuje kolekciju jednog ili više konteksta (*context_collection*) kojima će biti dodeljene privilegije (*permissions*). Sa *association_name* se definiše na koji način se ostvaruje veza između konteksta i privilegija. Moguće vrednosti za *association_name* su: *simple*, *and*, *or* i *dependence*. Vrednost *simple* omogućuje dodeljivanje privilegija individualnom kontekstu, dok se ostale vrednosti primenjuju na kolekciju od više konteksta. U slučaju *and* asocijacije privilegije su aktivivirane ako su svi navedeni konteksti aktivni u isto vreme, dok će u slučaju *or* asocijacije privilegije biti primenjene ako je bar jedan od navedenih konteksta aktivan. Asocijacija *dependence* omogućuje izražavanje složenijih situacija koje mogu da se pojave. U osnovi ona omogućuje da se nekom mobilnom klijentu u okviru nekog konteksta dozvoli odgovarajuća akcija samo ako je u okviru istog fizičkog konteksta prisutan i neki drugi mobilni klijent. U slučaju ove asocijacije privilegije će biti primenjene kada je logički kontekst za datog mobilnog klijenta aktivan i kada su logički konteksti (koji su navedeni u relaciji) ostalih mobilnih klijenata (u istom fizičkom kontekstu) aktivni.

Liscano i Wang predstavili su u [157] mehanizam nadogradnje dRBAC [112] modela kontrole pristupa da podrži kontekstno zavisnu delegaciju za potrebe distribuiranih heterogenih računarskih sistema. Pod delegacijom se podrazumeva mogućnost da korisnik delegira neke svoje privilegije odnosno uloge nekom drugom korisniku. Kontekstualno proširenje dRBAC modela zasnovano je na modelu ontologija preuzetim iz [55, 90, 156], gde su kontekstualne informacije klasifikovane u *entitete* i *aktivnosti*. Entiteti su osobe, lokacije, vreme i resursi, a aktivnosti su ponašanja entiteta. Da bi se izbegla potreba za korišćenjem zajedniče kontekstualne ontologije između različitih strana, kontekstnualna ograničenja se primenjuju samo na stranu (organizaciju) koja je vlasnik resursa.

U radu [95] predložen je model kontekstno zavisne kontrole pristupa za heterogena mobilna računarska okruženja (*pervasive computing*). Po autorima većina sistema za kontrolu pristupa u ovoj oblasti zasnovana je na RBAC modelu [105, 106] koji je proširen tako što je omogućena kontekstno zavisna dodela privilegija ulogama, međutim dodela uloga korisnicima nije zavisna od kontek-

sta što potencijalno može da izazove otežanu administraciju takvog sistema. Predloženi model zasnovan je na RBAC modelu koji je modifikovan kako bi bio efikasniji za heterogena računarska okruženja. Korisničke uloge su dinamički dodeljene korisnicima na osnovu dugotrajnih kontekstnih informacija, dok je dodela privilegija ulogama zavisna od kratkotrajnih informacija iz okruženja. Pod dugotrajnim kontekstnim informacijama podrazumevaju se informacije koje se ne menjaju unutar nekog vremenskog perioda Δt (npr. vremena trajanja sesije korisnika) dok se pod kratkotrajnim kontekstnim informacijama podrazumevaju informacije koje se unutar tog istog vremenskog perioda Δt mogu promeniti. Na početku sesije korisnicima su dodeljene određene uloge na osnovu definisanih informacija iz okruženja. Da bi se omogućila dinamička autorizacija aktivne privilegije dodeljene ulogama korisnika mogu da se menjaju za svakog korisnika ponaosob promenom okruženja.

Pigeot i dr. uočili su da je kontekstno zavisna kontrola pristupa u *pervasive computing* okruženjima često kompleksna i retko kad korisnički-orijentisana (komplikovana za definisanje). Zbog toga, predložen je gramatika jezika koja omogućuje korisnicima da na relativno jednostavan i efiksan način definišu kontekstno zavisna pravila, na osnovu njihove percepcije konteksta, koja će uticati na kontrolu pristupa [204].

Filho i Martin u radu [107] razmatraju primenu i uticaj *kvaliteta kontektne informacije* (*Quality of Context Information*, QoC) na kontekstno zavisnu kontrolu pristupa u *pervasive* računarskim okruženjima. Pod kvalitetom kontekstne informacije autori podrazumevaju definiciju datu u [51]: "*bilo koja informacija koja opisuje kvalitet informacije koja se koristi kao kontekstna informacija*" (npr. tačnost, preciznost, kompletnost, pouzdanost). Takođe, autori su dali i predlog merenja QoC informacija.

OASIS (*Open Architecture fore Secure Interworking Services*) model [20, 270] dizajniran je da obezbedi kontrolu pristupa servisima u otvorenim distribuiranim sistemima. Uloge u ovom sistemu formirane su na osnovu servisa. Naziv uloga predstavlja odgovarajuću poslovnu funkciju ili zvanje u organizaciji, dok uloga predstavlja naziv uloge pridružen određenom servisu, par $(naziv\ uloge, servis)$, tj. uloge su specifične za odgovarajući servis, a naziv uloge je jedinstven na nivou određenog servisa. Aktivacija uloga kontrolisana je na osnovu odgovarajući pravila definisanih za tu ulogu. Ovim pravilima su definisani uslovi koje korisnik treba da zadovolji kako bi mogao da aktivira

ulogu. Takođe, u pravilima mogu da se navode i parametari okruženja (ograničenja okruženja) čime je praktično obezbeđena kontekstna zavisnost OASIS modela.

Belokosztolszki i dr. razmatraju u radu [25] potencijalne nepoželjne tokove informacija u parametrizovanim (kontekstno zavisnim) RBAC modelima, naročito u onim sistemima koji omogućuju dvosmernu interakciju između internog sistema za kontrolu pristupa i njegovog dinamičkog eksternog okruženja. Dat je i predlog za rešenje ovog problema koji je ilustrovan na OASIS RBAC [20, 270] modelu.

Problem kontekstno zavisne kontrole pristupa u grid aplikacijama razmatran je u radu [273]. Autori opisuju SESAME (*Scalable, Environment Sensitive Access Management Engine*) sistem namenjen za spovođenje kontrole pristupa prvenstveno u grid baziranim sistemima. Sam sistem zasnovan je na *Dynamic Role-based Access Control* (DRBAC) modelu. Predloženi DRBAC model sa stoji se od sledećih entiteta:

- *USERS* - korisnici sistema,
- *ROLES* - uloge sistema,
- *PERMS* - privilegije sistema,
- *ENVS* - skup kontekstnih informacija sistema,
- *SESSIONS* - skup sesija (interakcija između subjekata i objekata) sistema,
- *UA* - mapiranje koje dodeljuje uloge korisnicima, i
- *PA* - mapiranje koje dodeljuje privilegije ulogama.

Ovaj model omogućuje dinamičku dodelu uloga korisnicima, odnosno privilegija ulogama na osnovu kontekstnih informacija. Prikupljanje kontekstnih informacija vrši se posredstvom *kontekstnog agenta*, koji na osnovu prikupljenih informacija vrši dodelu uloga, odnosno privilegija. Dinamička dodela uloga (privilegija) vrši se korišćenjem automata stanja (*automat stanja uloga* i *automat stanja privilegija*) čiji prelaz iz jednog stanja u drugo je definisan odgovarajućim kontekstnim informacijama.

Grupa autora u radu [268] takođe razmatra problem kontekstno zavisne kontrole pristupa u grid apliakcijama. Autori su predložili *RCBAC* model kontrole pristupa koji takođe predstavlja proširenje RBAC modela sa kontekstnim informacijama. Kontekstno proširenje RBAC modela predstavljeno je sledećim entitetima: *kontekstni parametar*, *kontekstni skup*, *kontekstni uslov* i *kontekstno ograničenje*.

Definicija 1.39 Kontekstni parametar (CP) *je predstavljen kao struktura podataka* p *koja ima sledeća obeležja: name* $\in CN$*, type* $\in CT$ *i funkciju* getValue()*. Pri čemu je* CN *skup svih mogućih naziva parametara,* CT *je skup tipova kontekstnih parametara, a funkcija* getValue() *vraća tekuću vrednost parametra* CP*.*

Definicija 1.40 Kontekstni skup (CS) *se sastoji od* n *kontekstnih parametara* $\{CP_1, CP_2, \cdots CP_n\}$*,* $n \geq 0$*,* $\forall(CP_i, CP_j), i \neq j \rightarrow CP_i.name \neq CP_j.name$

Definicija 1.41 Kontekstni uslov (CN) *je definisan kao:* $CN = (CP, OP, VALUE)$*,* $CP \in CS$*,* OP *je logički operator i* $VALUE$ *je specificirana vrednost pri čemu tip od* $VALUE$ *mora da je* $CP.type$

Definicija 1.42 Kontekstno ograničenje (CC) *je definisano kao:* $CC = CL_1 \cup CL_2 \cdots \cup CL_n$*,* $CL_i = CN_1 \cap CN_2 \cdots \cap CN_n$*,* $CN_i \in CN$

Definicija 1.43 Pravo pristupa (AP) *definiše se kao uređena trojka,* $AP = (R, P, C)$*, gde je:* R *uloga iz sistema,* P *je privilegija iz sistema i* C *je kontekstni uslov.*

Pristup nekom resursu je odobren samo ako korisnik pripada ulozi (R) za koju je definisana odgovarajuća privilegija (P) i ako je rezultat izvršenja kontekstnog ograničenja (C) za dato pravo pristupa tačan (*true*).

Trust and Context Based Access Control (TCAC) model namenjen je za kontrolu pristupa u distribuiranim sistemima, a predstavlja proširenje RBAC modela sa pojmovima *poverljivosti* i *konteksta* [98]. Dodela uloga korisnicima u ovom modelu zavisna je od stepena poverljivosti određenog korisnika kao i od kontekstnih informacija. Kada je nivo poverljivosti korisnika veći ili jednak granici poverljivosti definisanoj za određenu ulogu i kada su zadovoljena definisana kontekstna ograničenja korisniku će biti dodeljena uloga. U ovom modelu pod kontekstom se podrazumevaju informacije o korisnicima, vremenu i lokaciji. Formalno kontekst je definisan kao skup kontekstnih parametara $Context = \{p_1, p_2, p_3, \cdots, p_n\}$ gde svaki parametar ima svoje ime i vrednost. Konteksto ograničenje je logički izraz koji se sastoji od kontekstnih parametara, konstanti, logičkih operacija i/ili funkcija.

Predlog kontekstno zavisnog RBAC modela za kontrolu pristupa i zaštitu privatnosti u *pervasive* okruženjima dat je u radu [144]. Pod kontekstom autori

podrazumevaju različite informacije o sistemu u različitim situacijama. Definisana su dva tipa konteksta: *statički* koji se ne menja tako često, a čine ga akreditivi korisnika (atributi korisnika bitni sa stanovišta bezbednosti) i jedinstveni identifikator korisnika i *dinamički* čije su promene češće a sastoji se od podataka o vremenu, okruženju i lokaciji. Kontekstna ograničenja (statička i dinamička) su uvedena na relaciju dodele uloga korisnicima, ali i na relciju dodele privilegija ulogama. Uvođenjem ograničenja na ovu poslednju relaciju omogućeno je da dva korisnika iako imaju iste uloge imaju različite privilegije, što je u nekim situacijama potreban zahtev sa stanovišta bezbednosti.

U radu [244] prezentovan je metod unapređenja RBAC modela korišćenjem teorije *fuzzy* skupova. Definisana su dva parametra koji se odnose na koncept poverenja i poverljivosti. Prvi parametar je *korisnikova poverljivost (UT)* kojim se definiše koliko je korisnik u sistemu pouzdan i koliko mu se može verovati da bi mu se dodelila određena uloga/uloge. Drugi parametar je *stepen poverljivosti zahtevan od uloge (RT)*. Ovaj parametar određuje stepen poverenja koji se zahteva od korisnika da bi mu se dodelila ova uloga. Prilikom definisanja ovih vrednosti na njih mogu da utiču različiti parametri okruženja i korisnika.

Autori su prezentovali algoritam za izračunavanje ova dva parametra koristeći *fuzzy* teoriju skupova. Po izračunavanju UT i RT vrednosti, dodeljivanje uloge korisniku, kao i aktivacija uloge se vrši na osnovu poređenja UT i RT vrednosti. Korisniku može biti dodeljena neka uloga ako je UT korisnika zadovoljava RT uloge koja mu se dodeljuje.

Poglavlje 2

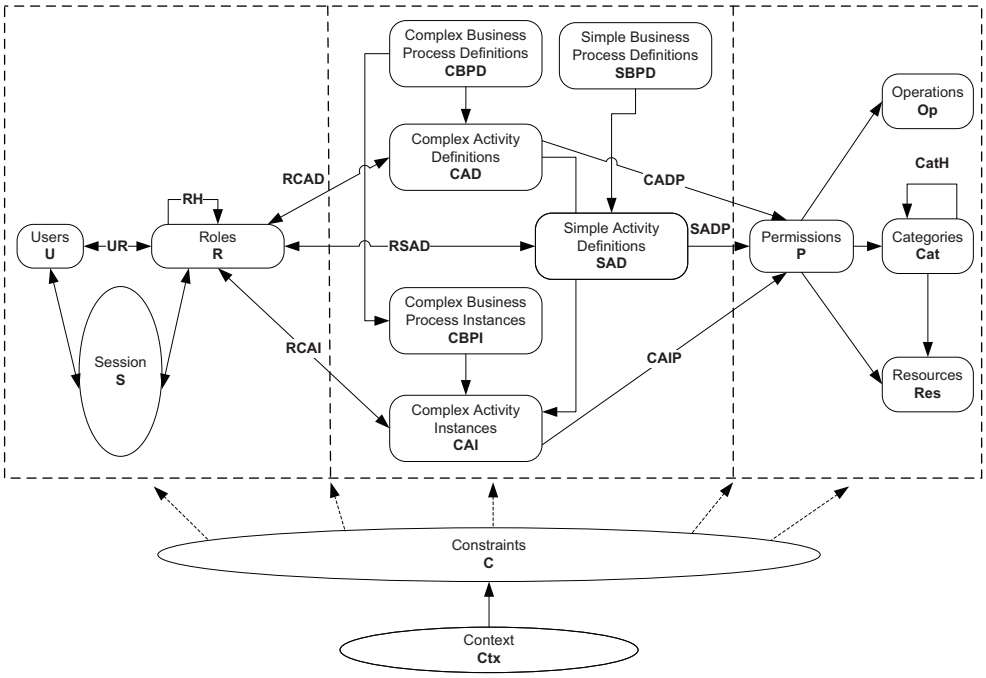
Model kontrole pristupa

2.1 Osnovni koncepti modela

Osnovni koncepti predloženog modela kontekstno zavisne kontrole pristupa u poslovnim sistemima (*COntext-sensetive Business processes Access Control model*, *COBAC*) prikazani su na slici 2.1. Dati model baziran je na standardnom RBAC modelu kontrole pristupa koji je proširen entitetima *poslovnog procesa*, *aktivnostima*, *kontekstom* i *kategorijama resursa*. Razlog za uvođenje entiteta *poslovnog procesa* i *aktivnosti* je da bi se postigla efikasnija primena RBAC baziranog modela za kontrolu pristupa u poslovnim sistemima jer odgovarajući aspekti kontrole pristupa moraju biti definisani za konkretan poslovni proces, odnosno njegove aktivnosti. Pošto postoje slučajevi kada se jedan poslovni proces može u potpunosti izvršiti u okviru jedne korisničke sesije, ali isto tako postoje slučajevi kada je izvršenje procesa distribuirano kroz više korisničkih sesija potreban je mehanizam sprovođenja kontrole pristupa koji ima mogućnost da podrži oba ova slučaja. Uvođenje entiteta *poslovnog procesa* i *aktivnosti* omogućilo je da se odgovarajući segmenti sprovođenja kontrole pristupa vežu za aktivnosti u okviru poslovnog procesa, umesto za tekuću sesiju i na taj način budu nezavisni od broja sesija u kojima se aktivnosti izvršavaju. U datom modelu ulogama se ne dodeljuju privilegije direktno, već su ulogama dodeljene aktivnosti poslovnog procesa koje one imaju pravo da izvrše. Da bi se u okviru aktivnosti mogle izvršiti sve potrebne operacije nad resursima kojima se u okviru njih pristupa njima su dodeljene odgovarajuće privilegije.

Kako je moguće da na celokupnu bezbednost sistema, pa time i na kontrolu pristupa utiču i faktori iz okruženja samog sistema pa i faktori koji čine sam sistem ali ne i eksplicitno model kontrole pristupa, predloženi model proširen je i *kontekstom*. *Kategorizacija* resursa omogućuje definisanje prava pristupa za čitavu kategoriju (grupu) resursa i time potencijalno smanjuje broj prava prava pristupa koje je potrebno definisati.

U narednim odeljcima detaljno je opisan svaki od entiteta kontrole pristupa iz COBAC modela kao i njihove međusobne relacije.



Slika 2.1: Osnovni koncepti COBAC modela

Osnovni pojmovi COBAC modela su:

- U - skup korisnika
- R - skup uloga
- S - skup korisničkih sesija
- $CBPD$ - skup definicija složenih poslovnih procesa
- CAD - skup definicija složenih aktivnosti

- $CBPI$ - skup instanci složenih poslovnih procesa
- CAI - skup instanci složenih aktivnosti
- $SBPD$ - skup definicija jednostavnih poslovnih procesa
- SAD - skup definicija jednostavnih aktivnosti
- P - skup privilegija
- Op - skup operacija
- Res - skup resursa
- Cat - skup kategorija
- C - ograničenja
- Ctx - kontekst
- CC - skup kontekstnih uslova

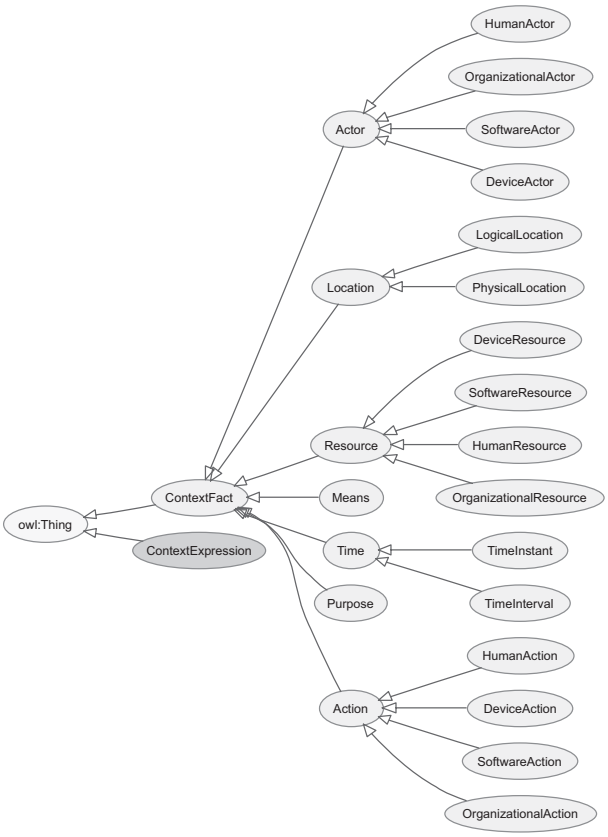
2.2 Model konteksta

Za modelovanje konteksta odabran je ontološki pristup (v. odeljak 1.7.1), pri čemu je kontekst definisan koristeći OWL (*Web Ontology Language*) jezik. Osnovni razlog za korišćenje ontološki baziranog modela je što takav pristup omogućuje formalnu reprezentaciju, bogatstvo informacija i interoperabilnost između različitih sistema. Takođe, ovaj pristup omogućuje da se kontekstni model relativno lako prilagodi i proširi za primenu u različitim sistemima.

Osnovni koncepti ontološkog modela konteksta prikazani su na slici 2.2. Ovaj model definiše tzv. *domensku ontologiju* (*domain ontology*) za kontekst u poslovnim sistemima, kojim su modelovane bazične klase i obeležja konteksta. Za primenu u konkretnom okruženju predloženi ontološki model potrebno je proširiti uvođenjem specijalizacija datih entiteta, uvođenjem novih entiteta kao i novih relacija.

Dve osnovne klase modela konteksta (slika 2.2) su `ContextFact` i `ContextExpression`. Klasom `ContextFact` predstavljene su osnovne činjenice modela, dok klasa `ContextExpression` modeluje kontekstne izraze. Same kontekstne činjenice moguće je klasifikovati u odgovarajuće specijalizacije klase `ContextFact`.

Klasa `Actor` namenjena je za predstavljanje učesnika različitih događaja. Učesnik može biti neka osoba/osobe (`HumanActor`), softverska komponenta (`SoftwareActor`), uređaj (`DeviceActor`) ili nekakav vid organizacije (`OrganizationalActor`). Različite aktivnosti predstavljene su `Action` klasom. Ljudske



Slika 2.2: Model konteksta

aktivnosti predstavljene su klasom `HumanAction`, aktivnosti uređaja sa `DeviceAction`, softverska aktivnost predstavljena je sa `SoftwareAction` klasom, dok je aktivnost organizacije predstavljena sa `OrganizationalAction`. Resursi (klasa `Resource`) predstavljeni u ovom sistemu klasifikuju se u ljudske (`HumanResource`), organizacione (`OrganizationalResource`), resurse uređaja (`DeviceResource`) i softverske (`SoftwareResource`) resurse. Lokacija je predstavljena klasom `Location`, odnosno njezinim specijalizacijama `LogicalLocation` (predstavlja logičku lokaciju) i `PhysicalLocation` (predstavlja fizičku lokaciju). Vremenski faktor predstavljen je `Time` klasom, odnosno specijalizaci-

jama za vremenski trenutak `TimeInstant` i za vremenski interval `TimeInterval`. Različite namene modelovane su klasom `Purpose`, dok su sredstva predstavljena `Means` klasom.

Kontekstni izraz (klasa `ContextExpression`) predstavlja semantičko povezivanje prethodno navedenih koncepata, a zasnovan je na 7 semantičkih dimenzija (relacija). Pet semantičkih dimenzija definisanih u [3, 254] (*"ko"* (*who*), *"šta"* (*what*), *"gde"* (*where*), *"kada"* (*when*) i *"kako"* (*how*)) proširene su sa konceptom *"zašto"* (*why*) kojim je definisana svrha i konceptom *"povezan"* (*related*) kojim je uspostavljena veza između semantički povezanih kontekstnih izraza. Pored toga uvedene su dve specijalizacije *what* koncepta. Specijalizacija *what action* koristi se za uspostavljanje relacija sa aktivnostima, dok se specijalizacija *what resource* koristi za uspostavljanje relacije sa resursima. Relacije za definisanje kontekstnog izraza prikazane su listingom 2.1, dok je kontekstni izraz prikazan listingom 2.2. Kao što se vidi iz listinga 2.2, kontekstni izraz mora da sadrži bar jednu *who* i *what* relaciju, dok su druge opcione.

```
@prefix ctx:  <http://informatika.ftn.uns.ac.rs/cobac/context.owl#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix xsd:  <http://www.w3.org/2001/XMLSchema#>.
@prefix owl:  <http://www.w3.org/2002/07/owl#>.

ctx:hasWhoPart          a owl:ObjectProperty.
ctx:hasWhatPart         a owl:ObjectProperty.
ctx:hasWhatActionPart   a owl:ObjectProperty ;
                          rdfs:subPropertyOf ctx:hasWhatPart.
ctx:hasWhatResourcePart a owl:ObjectProperty ;
                          rdfs:subPropertyOf ctx:hasWhatPart.
ctx:hasWhenPart         a owl:ObjectProperty.
ctx:hasWherePart        a owl:ObjectProperty.
ctx:hasWhyPart          a owl:ObjectProperty.
ctx:hasHowPart          a owl:ObjectProperty.
ctx:isRelatedTo         a owl:ObjectProperty, owl:SymmetricProperty;
```

Listing 2.1: Relacije kontektnog modela

```
ctx:ContextExpression
    a owl:Class;
    owl:equivalentClass[a owl:Class;
        owl:intersectionOf
        ([a owl:Restriction;
            owl:allValuesFrom ctx:Actor;
            owl:onProperty ctx:hasWhoPart]
         [a owl:Restriction;
            owl:onProperty ctx:hasWhoPart;
            owl:someValuesFrom ctx:Actor]
         [a owl:Restriction;
            owl:allValuesFrom ctx:Action;
            owl:onProperty ctx:hasWhatActionPart]
         [a owl:Restriction;
            owl:allValuesFrom ctx:Resource;
            owl:onProperty ctx:hasWhatResourcePart]
         [a owl:Restriction;
            owl:onProperty ctx:hasWhatPart;
            owl:someValuesFrom
            [a owl:Class;
               owl:unionOf (ctx:Action ctx:Resource)]]
         [a owl:Restriction;
            owl:onProperty ctx:hasWhatPart;
            owl:someValuesFrom ctx:Event]
         [a owl:Restriction;
            owl:allValuesFrom ctx:Event;
            owl:onProperty ctx:hasWhatPart]
         [a owl:Restriction;
            owl:allValuesFrom ctx:Means;
            owl:onProperty ctx:hasHowPart]
         [a owl:Restriction;
            owl:allValuesFrom ctx:Time;
            owl:onProperty ctx:hasWhenPart]
         [a owl:Restriction;
            owl:allValuesFrom ctx:Location;
            owl:onProperty ctx:hasWherePart]
         [a owl:Restriction;
            owl:allValuesFrom ctx:Purpose;
            owl:onProperty ctx:hasWhyPart]
         [a owl:Restriction;
            owl:allValuesFrom ctx:ContextExpression;
            owl:onProperty ctx:isRelatedTo])
    ].
```

Listing 2.2: Definicija `ContextExpression` klase

Kontekstni uslov definiše se kao logički izraz koji može da se sastoji od upita za pretraživanje elemenata prethodno definisane kontekstne ontologije, kontekstnih funkcija, kao i od logičkih operatora $\{\neg, \wedge, \vee\}$ i operatora poređenja

$\{<, \leq, >, \geq, =, \neq\}$. Za sve kontekstne funkcije koje su specificirane u okviru određenog modela konteksta mora da postoji odgovarajuća implementacija. Kontekstni uslov u EBNF notaciji dat je listingom 2.3.

```
ContextCondition ::=  Query
                    | Function
                    | ContextCondition BinaryOperator ContextCondition
                    | UnaryLogicalOperator ContextCondition
                    | "("ContextCondition")"
                    | "TRUE"
                    | "FALSE"
                    | String
                    | Number
BinaryOperator ::= ComparsionOperator|LogicalBinaryOperator
ComparsionOperator ::= "<"|"<="|">"|">="|"=="|"!="
LogicalBinaryOperator ::= "&&"|"||"
UnaryLogicalOperator ::= "!"
Query ::= "QUERY {" (String|Function){String|Function} "}"
Function ::= "$$"Name"("[Param{","Param}]")$$"
Param ::= String|Number|Function|Query
```

Listing 2.3: EBNF notacija kontekstnog uslova

Funkcija *evalCond* koja proverava da li je zadati kontekstni uslov zadovoljen pri tekućem stanju konteksta definisana je kao preslikavanje:

$$evalCond : CC \rightarrow \{\top, \bot\}$$

Iz skupa CC izdvajaju se dva specifična uslova, u oznaci: σ ($\sigma = TRUE$) i $\bar{\sigma}$ ($\bar{\sigma} = FALSE$). Uslov σ predstavlja uslov koji je uvek zadovoljen bez obzira na stanje konteksta tj. $evalCond(\sigma) = \top$. Uslov $\bar{\sigma}$ predstavlja suprotnost uslovu σ, odnosno uslov koji nikad nije zadovoljen tj. $evalCond(\bar{\sigma}) = \bot$.

2.3 Model poslovnog procesa

Pojam poslovnog procesa (*business process*) definisan je u [86] kao skup logički povezanih zadataka koji se izvršavaju u cilju postizanja definisanog poslovnog rezultata. U [63] prethodno data definicija proširena je konceptom *uloge* (*role*). Ova ([63]) specifikacija navodi da je poslovni proces skup više povezanih aktivnosti koje zajedno realizuju poslovni cilj u okviru konteksta organizacione strukture kojim su definisane uloge i relacije. Koncept *uloge* definiše obaveze svakog od učesnika (korisnika) i utiče na organizacionu strukturu.

Uloge omogućuju bolje razumevanje načina na koji su realizovane obaveze u organizaciji, dok su korisnicima posredstvom uloga dodeljene aktivnosti koje treba da obavljaju.

Za potrebe modela kontrole pristupa uvedena su sledeća dva koncepta: *složeni poslovni proces (complex business process)* i *jednostavni poslovni proces (simple business process)*. *Složeni poslovni proces*, u skladu sa definicijama datim u [63, 86], čini skup aktivnosti između kojih postoje odgovarajuće relacije uređenja. Kako u okviru organizacije postoje i "relativno jedonstavni" zadaci (npr. formiranje obaveštenja, kreiranje izveštaja, itd.) za koje se može smatrati da direktno nisu deo poslovnih procesa niti je za njihovo izvršavanje neophodan sistem za upravljanje poslovnim tokovima (workflow sistem), za njihovo predstavljanje uveden je *jednostavni poslovni proces* koji se sastoji od samo jedne aktivnosti.

Prava pristupa u predloženom modelu kontrole pristupa moguće je definisati za definicije i instance poslovnih procesa. Prava pristupa definisana za definiciju nekog poslovnog procesa primenjuju se na sve instance tog poslovnog procesa, dok se prava pristupa definisana za instancu primenjuju isključivo na instancu za koju su definisana.

Analizom više različitih poslovnih procesa utvrđeno je da za složene poslovne procese postoji potreba za definisanjem prava pristupa kako za njihove definicije (*Complex Business Process Definitions, CBPD*) tako i za njihove instance (*Complex Business Process Instances, CBPI*), odnosno za definicije (*Complex Activity Definitions, CAD*) i instance (*Complex Activity Instances, CAI*) njihovih aktivnosti. Takođe, uočeno je da je u slučaju jednostavnih poslovnih procesa dovoljno definisati prava pristupa za njihove definicije (*Simple Business Process Definition, SBPD*), odnosno za definiciju njihove aktivnosti (*Simple Activity Definitions, SAD*).

Definicija i-tog složenog poslovnog procesa ($cbpd_i$) definisana je kao par: $cbpd_i = (CAD_i, F_{BPD_i})$, gde je:

- CAD_i - skup definicija aktivnosti koje čine i-ti poslovni proces, tj. $CAD_i = \{cad_{i1}, cad_{i2}, \cdots, cad_{in}\}$, cad_{ij} je definicija j-te kompleksne aktivnosti i-tog poslovnog procesa.
- F_{BPD_i} - relacija kontrole toka kojom je definisan redosled izvršavanja aktivnosti iz skupa CAD_i.

COBAC modelom definisana su tri tipa složene aktivnosti:

- *start* - početna aktivnost poslovnog procesa. Izvršenje aktivnosti ovog tipa inicira startovanje instance poslovnog procesa.
- *end* - krajnja aktivnost poslovnog procesa. Izvršenje aktivnosti ovog tipa predstavlja završetak izvršavanja instance poslovnog procesa.
- *regular* - regularna aktivnost kojom je predstavljen deo poslovne logike implementirane kroz poslovni proces.

Funkcija $typeOf$ određuje tip složene aktivnosti, a definisana je kao preslikavanje $typeOf : CAD_i \rightarrow \{start, regular, end\}$.

Važno je napomenuti da u okviru jednog složenog poslovnog procesa mora da postoji tačno jedna početna aktivnost, tj važi:

$$\forall CAD_i, \exists cad_i \in CAD_i \mid typeOf(cad_i) = start$$
$$\forall cad_i, cad_j \in CAD_i \wedge typeOf(cad_i) = typeOf(cad_{ij}) = start$$
$$\Rightarrow cad_i = cad_j$$

Funkcija $instanceOf_{BP} : CBPI \rightarrow CBPD$ određuje definiciju od koje je nastala odgovarajuća instanca složenog poslovnog procesa.

Funkcija $instanceOf_A : CAI \rightarrow CAD$ određuje definiciju od koje je nastala odgovarajuća instanca složene aktivnosti.

Funkcija kojom se određuje kojoj definiciji složenog poslovnog procesa pripada definicija složene aktivnost definisna je kao preslikavanje $activityDefOf : CAD \rightarrow CBPD$:

$$activityDefOf(cad_i) = cbpd_i \mid cbpd_i = (CAD_i, F_{BPD_i}) \wedge cad_i \in CAD_i$$

k-ta instanca i-tog složenog poslovnog procesa ($cbpi_i^k$) definisana je kao par: $cbpi_i^k = (CAI_i^k, F_{BPI_i^k})$ gde je:

- $instanceOf_{BP}(cbpi_i^k) = cbpd_i$.
- CAI_i^k - skup instanci aktivnosti koje čine k-tu instancu i-tog poslovnog procesa, tj. $CAI_i^k = \{cai_{i1}^k, cai_{i2}^k, \cdot, cai_{in}^k\}$, pri čemu važi $instanceOf_A(cai_{ij}^k) = cad_{ij}$
- $F_{BPI_i^k}$ - relacija kontrole toka kojom je definisan redosled izvršavanja aktivnosti iz skupa CAI_i^k.

Na sličan način kao i funkcija $activityDefOf$ definiše se i $activityInstOf$: $CAI \rightarrow CBPI$ funkcija kojom se određuje instanca složenog poslovnog procesa kojoj pripada odgovarajuća instanca složene aktivnosti:

$$activityInstOf(cai_i) = cbpi_i \mid cbpi_i = (CAI_i, F_{BPI_i}) \wedge cai_i \in CAI_i$$

Kao što je prethodno rečeno, i-ti jednostavni poslovni proces (spd_i) sastoji se od samo jedne aktivnosti (sad_i), tj. $spd_i = (sad_i)$.

Skup svih definicija složenih aktivnosti označen je sa CAD, odnosno:

$$CAD = \bigcup_{i=1}^{n} CAD_i$$

gde je n ukupan broj složenih poslovnih procesa u sistemu.

Sa CAI označen je skup svih instanci složenih aktivnosti, odnosno:

$$CAI = \bigcup_{i=1}^{n} \bigcup_{k=1}^{m_i} CAI_i^k$$

gde je n ukupan broj složenih poslovnih procesa u sistemu, a m_i broj instanci i-tog poslovnog procesa.

Sve jednostavne definicije aktivnosti predstavljene su skupom SAD, odnosno:

$$SAD = \bigcup_{i=1}^{p} SAD_i$$

gde je p broj jednostavnih aktivnosti (jednostavnih poslovnih procesa).

2.4 Model korisnika

Sa U označen je skup svih korisnika u sistemu, dok je sa S označen skup svih korisničkih sesija.

Predikatom $usersSession(u, s)$ proverava se da li sesija $s \in S$ pripada korisniku $u \in U$.

U praksi je identifikovano da na kontrolu pristupa mogu da utiču i odgovarajuća korisnikova obeležja, tj. postoji potreba da prava pristupa budu zavisna od korisnikovih obeležja, odnosno u slučaju RBAC modela da korisniku

budu dodeljene uloge zavisno od njegovih obeležja. Ovaj zahtev u predloženom modelu realizovan je posredstvom konteksta s obzirom da predloženi model konteksta dozovoljava i modelovanje korisnikovih obeležja.

2.5 Model uloga

Skup svih uloga u sistemu označen je sa R.

U predloženom modelu, po uzoru na model opisan u [26, 27, 151], uloge mogu da budu u jednom od sledeća tri stanja:

- *onemogućene* (*disabled*) - Uloge koje korisnik ne može da aktivira u okviru tekuće sesije. Uloge iz ovog stanja mogu da pređu u stanje *omogućeno*.
- *omogućene* (*enabled*) - Uloge koje korisnik može da aktivira u okviru tekuće sesije. Iz ovog stanja uloga može da pređe u *onemogućeno* ili *aktivno* stanje.
- *aktivne* (*active*) - Ovo stanje uloge odnosi se ponaosob za svakog korisnika. Ukoliko korisnik aktivira neku omogućenu ulogu u okviru sesije ona iz stanja *omogućena* prelazi u stanje *aktivna* za tog korisnika. Deaktiviranjem uloge ona prelazi iz *aktivnog* u *omogućeno* stanje samo u slučaju kada je ona bila aktivirana u **jednoj** korisničkoj sesiji tog korisnika, u protivnom i dalje ostaje aktivna jer postoji više korisničkih sesija istog korisnika u kojima je ona aktivirana. Iz *aktivnog* stanja uloga može da pređe u *onemogućeno* stanje ako je zadovoljen uslov za onemogućavanje uloge.

Na to da li će uloga biti u stanju *omogućena* ili *onemogućena* utiče kontekst, pa je shodno tome definicija uloga izmenjena u odnosu na standardni RBAC model i definisana je kao torka: (rn, sc, sct) gde je:

- rn - naziv uloge (*role name*), i
- sc - kontekstni uslov za onemogućavanje/omogućavanje uloge (*state condition*), i
- sct - tip kontekstnog uslova (*state condition type*). Moguće vrednosti su iz skupa $\{dc, ec\}$. Definiše na šta se kontekstni uslov odnosi: na uslov za onemogućavanje uloge (dc) ili na uslov za omogućavanje uloge (ec).

Ukoliko je uslov *sc* zadovoljen uloga je, zavisno od vrednosti *sct*, u *onemogućenom* stanju (ako je $sct = dc$) ili *omogućenom* stanju (ako je $sct = ec$). Ako uslov nije zadovoljen uloga će biti u stanju suprotnom od vrednosti *sct*, tj. ako uslov nije zadovoljen, a definisan je za *omogućavanje* uloge onda će uloga biti u *onemogućenom* stanju i obrnuto.

U specifičnom slučaju *sc* uslov je definisan kao kontekstni uslov koji je uvek zadovoljen, u oznaci: (rn, σ, ec). Ovo praktično znači da nema kontekstnog uticaja na onemogućavanje/omogućavanje uloge. Uloga je uvek *omogućena*.

Za proveru u kom stanju se odgovarajuća uloga nalazi uvedeni su sledeći predikati:

- $disabled(r)$ - proverava da li je uloga onemogućena,
- $enabled(r)$ - proverava da li je uloga omogućena, i
- $active(r, u)$ - proverava da li je uloga r aktivna za korisnika u.

Predikat $disabled(r)$ je tačan ukoliko je uloga r onemogućena pri trenutnom stanju konteksta:

$$\begin{aligned} disabled(r) \Leftarrow & (r.sct = dc \wedge evalCond(r.sc)) \vee \\ & (r.sct = ec \wedge \neg evalCond(r.sc)) \end{aligned}$$

Predikat $enabled(r)$ je tačan ukoliko je uloga r omogućena u trenutnom stanju konteksta:

$$\begin{aligned} enabled(r) \Leftarrow & (r.sct = ec \wedge evalCond(r.sc)) \vee \\ & (r.sct = dc \wedge \neg evalCond(r.sc)) \end{aligned}$$

Predikat $active(r, u)$ je tačan ukoliko postoji bar jedna korisnička sesija korisnika u u kojoj je uloga r aktivirana:

$$active(r, u) \Leftarrow usersSession(u, s) \wedge activeInSession(s, r)$$

gde predikat $activeInSession(s, r)$ proverava da li je uloga r aktivirana u sesiji s korisnika u.

Funkcija $activeRoles : U \rightarrow 2^R$ određuje uloge koje su aktivne u bilo kojoj od sesija zadatog korisnika:

$$activeRoles(u) = \{r \mid active(r, u)\}$$

2.5.1 Dodela uloga korisnicima

Predloženi model omogućuje dva načina dodele uloga korisnicima: *statički* i *dinamički*.

Kod *statičkog* načina, korisniku su dodeljene one uloge za koje postoji definisana relacija dodele između njih i korisnika kome se dodeljuju. Relacija $sRoleAssign(r, u, cc)$ definiše statičku dodelu uloge r korisniku u ukoliko je kontekstni uslov cc zadovoljen.

Osim statičkog načina dodele uloga u praksi je uočena potreba za slučajevima kada ne treba da postoji unapred predefinisana dodela uloga korisnicima, već da se korisnicima dodeljuju određene uloge isključivo ukoliko ti korisnici zadovoljavaju odgovarajuće kriterijume. Ovaj način dodele uloga korisnicima u COBAC modelu podržan je kroz *dinamičku* dodelu. Dinamička dodela uloga omogućuje da bilo kom korisniku bude dodeljena odgovarajuća uloga, ukoliko je uslov za dodelu te uloge zadovoljen.

Relacijom $roleCondAssign(r, rac)$ definiše se uslov $rac \in CC$ (*role assignment condition*) koji treba da je zadovoljen da bi se nekom korisniku dodelila uloga r. Ovaj uslov (rac), kao svoj deo, može da sadrži bilo koji kontekstni entitet uključujući i samog korisnika, pa je na taj način moguća dodela zavisna od korisnika, odnosno njegovih obeležja. Dinamička dodela uloga definisana je sledećim predikatom:

$$dRoleAssign(r, u) \Leftarrow roleCondAssign(r, rac) \wedge evalCond(rac)$$

Korisniku u će biti dodeljena uloga r ukoliko postoji odgovarajući predikat $roleConditionAssign(r, rac)$ kome je kriterijum rac zadovoljen.

Predikatom $isRoleAssigned(u, r)$ proverava se da li je korisniku u dodeljena uloga r bez obzira na nači dodele, tj:

$$\begin{aligned} isRoleAssigned(r, u) \Leftarrow & dRoleAssign(r, u) \vee \\ & (sRoleAssign(r, u, cc) \wedge evalCond(cc)) \end{aligned}$$

Zavisno od toga da li je $cc = \sigma$ postoji *statička relacija dodele bez uticaja konteksta* (*context free static user-role assignment*) i *statička relacija dodele sa uticajem konteksta* (*context influenced static user-role assignment*) u slučaju da je $cc \neq \sigma$.

Predikat $isCFRoleAssigned(r, u)$ proverava da li je korisniku u dodeljena uloga r bez uticaja konteksta, tj. kontekstni uslov je σ (σ kontekstni uslov je uvek zadovoljen):

$$\begin{aligned} isCFRoleAssigned(r, u) \Leftarrow & (roleCondAssign(r, rac) \wedge rac = \sigma) \vee \\ & (sRoleAssign(r, u, cc) \wedge rac = \sigma) \end{aligned}$$

2.5.2 Hijerarhija uloga

Hijerarhija uloga (RH) definisana je kao parcijalno uređenje nad skupom uloga R ($RH \subseteq R \times R$), odnosno kao relacija nasleđivanja, u oznaci $\succeq$, gde ako neka uloga r_1 nasleđuje ulogu r_2 onda važi $r_1 \succeq r_2$.

Predikat $canObtainRole(u, r)$ koji proverava da li je korisniku u moguće dodeliti ulogu r definiše se kao:

$$\begin{aligned} canObtainRole(u, r_j) \Leftarrow & isRoleAssigned(u, r_j) \vee \\ & ((r_i \succeq r_j) \wedge canObtainRole(u, r_i)) \end{aligned}$$

Funkcija $usersRoles : U \rightarrow 2^R$ kojom se određuju sve uloge dodeljene nekom korisniku u prisustvu hijerarhije uloga definisana je na sledeći način:

$$usersRoles(u) = \{r \mid canObtainRole(u, r)\}$$

Predikatom $canCFObtainRole(u, r)$ proverava se da li je korisniku u dodeljena uloga r bez razmatranja konteksta, odnosno kontekstni uslov je σ:

$$\begin{aligned} canCFObtainRole(u, r_j) \Leftarrow & isCFRoleAssigned(u, r_j) \vee \\ & ((r_i \succeq r_j) \wedge canCFObtainRole(u, r_i)) \end{aligned}$$

2.5.3 Dodela aktivnosti ulogama

Svakoj od uloga iz skupa uloga R moguće je dodeliti pravo da izvrši odgovarajuće aktivnosti poslovnih procesa definisanih u sistemu. Shodno tome da se prava pristupa mogu definisati za definicije i instance složenih aktivnosti kao i za definicije jednostavnih aktivnost postoje tri relacije dodele aktivnosti ulogama:

- Izrazom $rcdActivityAssign(r, cad, cc)$ definisana je relacija kojom se dozvoljava ulozi $r \in R$ da izvrši definiciju (odnosno sve instance) složene aktivnosti $cad \in CAD$ ukoliko je kontekstni uslov $cc \in CC$ zadovoljen.
- Izrazom $rciActivityAssign(r, cai, cc)$ definisana je relacija kojom se dozvoljava ulozi $r \in R$ da izvrši tačno određenu instancu složene aktivnosti $cai \in CAI$ ukoliko je kontekstni uslov $cc \in CC$ zadovoljen, pri čemu tip cai aktivnosti ne može biti *start* jer nema smisla dodeljivati privilegije za instancu *start* aktivnosti ($cai \in CAI \wedge instanceOf_A(cai) = cad \wedge typeOf(cad) \neq start$). Tip *start* aktivnosti služi samo da se startuje nova instanca poslovnog procesa.
- Izrazom $rsdActivityAssign(r, sad, cc)$ definisana je relacija kojom se dozvoljava ulozi $r \in R$ da izvrši jednostavnu aktivnost $sad \in SAD$ ukoliko je kontekstni uslov $cc \in CC$ zadovoljen.

Zavisno od toga da li je $cc = \sigma$ postoji *relacija dodele bez uticaja konteksta* (*context free role-activity assignment*) i *relacija dodele sa uticajem konteksta* (*context influenced role-activity assignment*) u slučaju da je $cc \neq \sigma$.

Provera da li je ulozi r dozvoljeno da izvrši odgovarajuću instancu složene aktivnosti $cai \in CAI$ definisana je predikatom:

$$\begin{aligned} canExecCompActInst(r, cai) \Leftarrow & (rciActivityAssign(r, cai, cc) \wedge \\ & evalCond(cc)) \vee \\ & (rcdActivityAssign(r, cad, cc) \wedge \\ & evalCond(cc) \wedge \\ & instanceOf_A(cai) = cad) \end{aligned}$$

Na sličan način definisani su predikati za proveru da li je ulozi r dozvoljeno da izvrši složenu, odnosno jednostavnu aktivnost čija je definicija zadata:

$$canExecCompActDef(r, cad) \Leftarrow rcdActivityAssign(r, cad, cc) \wedge evalCond(cc)$$

$$canExecSimpActDef(r, sad) \Leftarrow rsdActivityAssign(r, sad, cc) \wedge evalCond(cc)$$

Odgovarajuća uloga može da kreira novu instancu složenog poslovnog procesa ako joj je dodeljeno pravo da izvrši *start* tip aktivnosti tog procesa. Funk-

cija $assignedCompProcToStart : R \rightarrow 2^{CAD}$ određuje sve definicije aktivnosti koje su *start* tipa, a dodeljene su navedenoj ulozi:

$$assignedCompProcToStart(r) = \{cad \mid canExecCompActDef(r, cad) \wedge typeOf(cad) = start\}$$

Funkcijom $assignedCompActToExec : R \rightarrow 2^{CAI}$ određuju se instance složenih aktivnosti koje odgovarajuća uloga može da izvrši. Ulozi će biti dozvoljeno da izvrši odgovarajuću instancu složene aktivnosti ukoliko joj je dodeljeno pravo da je izvrši ($canExecCompActInst(r, cai)$) i ako je ta aktivnost na redu da se izvrši, što se proverava sa predikatom $nextToExec$. Ovo omogućuje da se vodi računa o redosledu događaja, tj. da se aktivnosti izvršavaju u redosledu definisanim poslovnim procesom kome pripadaju. Korisnik neće moći izvršiti odgovarajuću instancu aktivnosti sa ulogom koja mu je dodeljena ako po definiciji poslovnog procesa nije vreme da se ta instanca aktivnosti izvrši:

$$assignedCompActToExec(r) = \{cai \mid nextToExec(cai) \wedge canExecCompActInst(r, cai)\}$$

Skup jednostavnih aktivnosti koje uloga može da izvrši određuje se funkcijom $assignedSimpActToExec : R \rightarrow 2^{SAD}$:

$$assignedSimpActToExec(r) = \{sad \mid canExecSimpActDef(r, sad)\}$$

2.6 Model privilegija

U cilju efikasnije administracije resursi se mogu organizovati u kategorije resursa, pri čemu jedan resurs može da bude svrstan u više kategorija. Pripadnost resursa odgovarajućoj kategoriji predstavljena je $assignCat(res, cat)$ relacijom, gde je $res \in Res$ i $cat \in Cat$.

Poput korisničkih uloga i kategorije resursa moguće je hijerarhijski organizovati. Hijerarhija resursa ($CatH \subseteq Cat \times Cat$) predstavlja parcijalno uređenje nad skupom Cat, tj. relacija nasleđivanja, u oznaci $\succeq_{cat}$, gde važi $cat_i \succeq_{cat} cat_j$ ukoliko kategorija cat_i nasleđuje kategoriju cat_j.

Svi resursi koji pripadaju kategoriji cat_i ujedno pripadaju i kategoriji cat_j. Predikatom *belongsToCat* vrši se provera da li resurs pripada odgovarajućoj

kategoriji:

$$belongsToCat(res, cat_j) \Leftarrow assignCat(res, cat_j) \vee (cat_i \succeq_{cat} cat_j \wedge belongsToCat(res, cat_i))$$

Privilegija kojom se dozvoljava da se izvrši neka operacija može da se definiše kako za resurse tako i za kategorije. Privilegija definisana za kategoriju važi i za sve resurse koji pripadaju toj kategoriji kao i za sve podkategorije. Privilegija za resurs definisana je kao uređeni par $rp = (res, op), res \in Res, op \in Op$, dok je privilegija za kategoriju definisana kao $cp = (cat, op), cat \in Cat, op \in Op$. Sa RP označen je skup svih privilegija definisianih za resurse, dok je sa CP označen skup svih privilegija definisanih za kategorije. Skup $P = RP \cup CP$ predstavlja skup svih privilegija u sistemu.

Da bi odgovarajuća aktivnost poslovnog procesa mogla uspešno da se izvrši potrebno joj je dodeliti privilegije koje su joj neophodne da bi se nad potrebnim resursima izvršile zahtevane operacije. U predloženom modelu sledećim izrazima su predstavljene relacije dodele privilegija aktivnostima:

- $cadPermissionAssign(cad, p, cc)$ - definiciji složene aktivnosti $cad \in CAD$) dodeljena je privilegija $p \in P$ ukoliko je kontekstni uslov $cc \in CC$ zadovoljen,
- $caiPermissionAssign(cai, p, cc)$ - instanci složene aktivnosti $cai \in CAI$ dodeljena je privilegija $p \in P$ ukoliko je kontekstni uslov $cc \in CC$ zadovoljen, i
- $sadPermissionAssign(sad, p, cc)$ - definiciji jednostavne aktivnosti $sad \in SAD$ dodeljena je privilegija $p \in P$ ukoliko je kontekstni uslov $cc \in CC$ zadovoljen.

Slično kao i kod dodele aktivnosti ulogama, zavisno od toga da li je $cc = \sigma$ postoji *relacija dodele bez uticaja konteksta* (*context free activity-permission assignment*) i *relacija dodele sa uticajem konteksta* (*context influenced activity-permission assignment*) u slučaju da je $cc \neq \sigma$.

Funkcije kojima se određuju privilegije (definisane za odgovarajuću operaciju) pridružene definiciji i instanci složenih poslovnih procesa kao i definiciji jednostavnih procesa definisane su na sledeći način:

$$cadPermission : CAD \times Op \rightarrow 2^P$$
$$cadPermission(cad, op) = \{p \mid p.op = op \wedge cadPermissionAssign(cad, p, cc)\}$$

$$caiPermission : CAI \times Op \rightarrow 2^P$$
$$caiPermission(cai, op) = \{p \mid p.op = op \wedge caiPermissionAssign(cai, p, cc)\}$$
$$sadPermission : SAD \times Op \rightarrow 2^P$$
$$sadPermission(sad, op) = \{p \mid p.op = op \wedge sadPermissionAssign(sad, p, cc)\}$$

Za privilegiju p_1 se kaže da je sadržana u privilegiji p_2 (u oznaci $p_1 \sqsubseteq p_2$) ako važi da su p_1 i p_2 definisani za (a) istu operaciju i isti resurs ($p_1 = p_2$), ili (b) su definisani za istu operaciju, a kategorija od p_1 je podkategorija kategorije p_2, ili (c) su definisani za istu operaciju, a resurs od p_1 pripada kategoriji od p_2. Ovo se formalno može opisati na sledeći način:

$$\begin{aligned} p_1 \sqsubseteq p_2 \Rightarrow & (p_1 \in RP \wedge p_2 \in RP \wedge p_1.op = p_2.op \wedge p_1.res = p_2.res) \vee \\ & (p_1 \in CP \wedge P_2 \in CP \wedge p_1.op = p_2.op \wedge p_1.cat \succeq_{cat} p_2.cat) \vee \\ & (p_1 \in RP \wedge p_2 \in CP \wedge p_1.op = p_2.op \wedge \\ & \ belongsToCat(p_1.res, p_2.cat)) \end{aligned}$$

Ako neka aktivnost poseduje privilegiju p_2, a da bi izvršila odgovarajuću operaciju nad resursom potrebna joj je privilegija p_1, onda će pristup tom resursu biti odobren ukoliko važi: $p_1 \sqsubseteq p_2$.

2.7 Model ograničenja

COBAC model identifikuje dve klase ograničenja: *statička* i *dinamička*. Klasifikacija je izvršena u odnosu na to kada se ograničenja sprovode. Dok se statička ograničenja primenjuju tokom administracije prava pristupa za poslovne procese, dinamička se primenjuju za vreme izvršavanja poslovnih procesa. Obe klase ograničenja primenljive su isključivo za složene poslovne procese, jer za jednostavne poslovne procese ne postoji potreba za definisanjem ograničenja.

2.7.1 Statička ograničenja

Kako se statička ograničenja definišu i sprovode tokom administracije ona će sprečiti relacije (dodele) koje nisu dozvoljene. U [227] naglašeno je da ona mogu biti veoma restriktivna za poslovna pravila, naročito u manjim organizacijama. Međutim, i dalje se mogu pokazati korisnim u slučajevima kada

poslovna pravila treba da se primenjuju u čitavoj organizaciji i/ili kada treba da budu nepromenljiva tokom vremena. COBAC model identifikuje sledeće tipove statičkih ograničenja:

- Statička separacija obaveza (*Static Separation of Duties, SSoD*),
- Statička separacija obaveza bazirana na korisnicima (*Users based Static Separation of Duties, USSoD*), i
- Statička separacija obaveza bazirana na aktivnostima (*Activity based Static Separation of Duties, ASSoD*).

Sve navedene tipove statičkih ograničenja moguće je definisati za odgovarajuće relacije dodele na koje ne postoji uticaj konteksta (na *context-free* relacije).

Statička separacija obaveza

Statička separacija obaveza sprečava da korisnik istovremeno bude član dve konfliktne uloge. Skup statički konfliktnih uloga definisan je kao $SCR \subseteq R \times R$. Provera da li su dve uloge međusobno konfliktne pri postojanju hijerarhije uloga definisana je predikatom $sRolesInConflict$:

$$\begin{aligned} sRolesInConflict(r_i, r_j) \Leftarrow & (\exists (r_i, r_j) \in SCR \vee \exists (r_j, r_i) \in SCR) \vee \\ & (r_i \succeq r_m \wedge sRolesInConflict(r_m, r_j)) \vee \\ & (r_j \succeq r_n \wedge sRolesInConflict(r_i, r_n)) \end{aligned}$$

SSoD ograničenje formalno se može definisati na sledeći način:

$$r_i, r_j \in R \wedge u \in U \wedge canCFObtainRole(u, r_i) \wedge canCFObtainRole(u, r_j) \Rightarrow \neg sRolesInConflict(r_i, r_j)$$

Funkcija $sConflictRoles : R \rightarrow 2^R$ za određivanje statički konfliktnih uloga neke zadate uloge definisana je kao:

$$sConflictRoles(r_i) = \{r_j \mid sRolesInConflict(r_i, r_j)\}$$

Statička separacija obaveza bazirana na korisnicima

Statička separacija obaveza bazirana na korisnicima predstavlja proširenje prethodnog ograničenja. Naime, postoje slučajevi kada nije dovoljno sprečiti jednom korisniku da bude član dve ili više međusobno konfliktnih uloga, već je to potrebno sprečiti za određenu grupu, na neki način, međusobno povezanih korisnika, tzv. *konfliktnih korisnika*. Ovo ograničenje sprečava da bilo kojim korisnicima iz skupa međusobno konfliktnih korisnika budu dodeljene međusobno konfliktne uloge. Neka je sa $SCU \subseteq U \times U$ označen skup povezanih korisnika, tada važi:

$(u_i, u_j) \in SCU \wedge r_i, r_j \in R \wedge canCFObtain(u_i, r_i) \wedge canCFObtain(u_j, r_j) \Rightarrow \neg sRolesInConflict(r_i, r_j)$

Funkcija $sConflictUsers : U \rightarrow 2^U$ za određivanje statički konfliktnih korisnika za nekog zadatog korisnika definisana je kao:

$$sConflictUsers(u_i) = \{u_j \mid (u_i, u_j) \in SCU \vee (u_j, u_i) \in SCU\}$$

Statička separacija obaveza bazirana na aktivnostima

Statička separacija obaveza bazirana na aktivnostima onemogućuje da dve konfliktne aktivnosti budu izvršene od strane jednog korisnika, odnosno od strane konfliktnih korisnika, pri čemu konfliktne aktivnosti mogu da budu definisane u istom ali i u različitim poslovnim procesima. U statičkoj verziji ovo ograničenje praktično dozvoljava da se konfliktne aktivnosti mogu izvršiti od isključivo strane konfliktnih uloga. Pošto je reč o statičkom ograničenju onda je dozvoljeno da se ono definiše samo za definiciju složene aktivnosti. Ako se sa $SCA \subseteq CAD \times CAD$ označi skup konfliktnih aktivnosti, tada važi:

$((a_i, a_j) \in SCA \wedge r_i, r_j \in R) \wedge$
$(rcdActivityAssign(r_i, a_i, cc_i) \wedge cc_i = \sigma) \wedge$
$(rcdActivityAssign(r_j, a_j, cc_j) \wedge cc_j = \sigma) \Rightarrow sRolesInConflict(r_i, r_j)$

Funkcija $sConflictActivities : CAD \rightarrow 2^{CAD}$ za određivanje statički konfliktnih aktivnosti za neku zadatu aktivnost definisana je kao:

$$sConflictActivities(a_i) = \{a_j \mid (a_i, a_j) \in SCA \vee (a_j, a_i) \in SCA\}$$

2.7.2 Dinamička ograničenja

Dinamička ograničenja sprovode se tokom izvršenja instanci poslovnih procesa. COBAC model omogućuje da se, po potrebi, uz svako dinamičko ograničenje definiše i kontekstni uslov. Ukoliko je uz odgovarajuće ograničenje definisan kontekstni uslov to ograničenje će biti primenjeno samo ako je navedeni kontekstni uslov zadovoljen. Predloženim modelom definisana su sledeća dinamička ograničenja:

- Dinamička separacija obaveza (*Dynamic Separation of Duties, DSoD*),
- Dinamička separacija obaveza bazirana na korisnicima (*Users-based Dynamic Separation of Duties, UDSoD*),
- Dinamička separacija obaveza bazirana na aktivnostima (*Activity-based Dynamic Separation of Duties, ADSoD*),
- Dinamičko povezivanje obaveza (*Dynamic binding of Duties, DBoD*), i
- Dinamičko ograničenje izvršenja (*Dynamic Execution Constraint, DEC*).

Dinamička separacija obaveza

Dinamička separacija obaveza sprečava da korisnik u okviru iste sesije aktivira dve konfliktne uloge. Sa $DCR \subseteq R \times R \times CC$ označen je skup dinamički konfliktnih uloga. Za bilo koji element $(r_i, r_j, cc) \in DCR$ uloge r_i i r_j će biti konfliktne samo ako je uslov cc zadovoljen. Provera da li su dve uloge međusobno konfliktne pri postojanju hijerarhije uloga definisana je predikatom $dRolesInConflict$:

$$\begin{aligned} dRolesInConflict(r_i, r_j) \Leftarrow & (\exists (r_i, r_j, cc) \in DCR \wedge evalCond(cc)) \vee \\ & (\exists (r_j, r_i, cc) \in DCR \wedge evalCond(cc)) \vee \\ & r_i \succeq r_m \wedge dRolesInConflict(r_m, r_j) \vee \\ & r_j \succeq r_n \wedge dRolesInConflict(r_i, r_n) \end{aligned}$$

DSoD ograničenje se formalno može definisati na sledeći način:

$$\begin{aligned} & r_i, r_j \in R \wedge u \in U \wedge s \in S \wedge usersSession(u, s) \wedge \\ & activeInSession(s, r_i) \wedge activeInSession(s, r_j) \Rightarrow \\ & \neg dRolesInConflict(r_i, r_j) \end{aligned}$$

Na sličan način kao i kod statičke varijante definiše se $dConflictRoles : R \rightarrow 2^R$ funkcija koja određuje dinamički konfliktne uloge od zadate uloge:

$$dConflictRoles(r_i) = \{r_j \mid dRolesInConflict(r_i, r_j)\}$$

Algoritam 2.1 opisuje sprovođenje DSoD ograničenja. Ukoliko u skupu uloga ARS postoje dve konfliktne uloge u skladu sa DSoD ograničenjem, jedna od njih se uklanja iz skupa. Prikazani algoritam uvek ukanja drugu ulogu, mada je moguće realizovati modifikaciju algoritma sa dodatnom logikom koja će odlučiti koja od dve uloge treba da se ukloni.

```
NAME: EnforceDSoD
INPUT: ARS ⊆ R - set of user's roles in session
OUTPUT: ARS ⊆ R - input set after DSoD enforcement

for each r_i, r_j ∈ ARS ∧ r_i ≠ r_j do
   if dRolesInConflict(r_i, r_j) then
      removeFromSet(r_j, ARS)
```

Algoritam 2.1: Dinamička separacija obaveza

Dinamička separacija obaveza bazirana na korisnicima

Dinamička separacija obaveza bazirana na korisnicima onemogućuje da dva međusobno konfliktna korisnika aktiviraju određene uloge u okviru svojih sesija, a da te uloge budu međusobno konfliktne. Neka je sa $DCU \subseteq U \times U \times CC$ označen skup dinamički konfliktnih korisnika. Tada, slično kao i u prethodnom slučaju, za bilo koji element $(u_i, u_j, cc) \in DCU$ korisnici u_i i u_j će biti konfliktni samo ako je uslov cc zadovoljen. Provera da li su dva korisnika međusobno konfliktna definisana je predikatom $usersInConflict$:

$$\begin{aligned} usersInConflict(u_i, u_j) \Leftarrow & (\exists (u_i, u_j, cc) \in DCU \wedge evalCond(cc)) \vee \\ & (\exists (u_j, u_i, cc) \in DCU \wedge evalCond(cc)) \end{aligned}$$

UDSoD ograničenje se formalno može definisati na sledeći način:

$$\begin{aligned} & u_i, u_j \in U \wedge r_i, r_j \in R \wedge s_i, s_j \in S \wedge usersInConflict(u_i, u_j) \wedge \\ & usersSession(u_i, s_i) \wedge activeInSession(s_i, r_i) \wedge \\ & usersSession(u_j, s_j) \wedge activeInSession(s_j, r_j) \\ & \Rightarrow \neg dRolesInConflict(r_i, r_j) \end{aligned}$$

Funkcija za određivanje dinamički konfliktnih korisnika $dConflictUsers$: $U \rightarrow 2^U$ definisana je na sledeći način:

$$dConflictUsers(u_i) = \{u_j \mid usersInConflict(u_i, u_j)\}$$

Algoritam 2.2 opisuje sprovođenje UDSoD ograničenja. Za svakog korisnika koji je u konfliktu sa tekućim korisnikom proverava se da li ona/on ima aktiviranu ulogu koja je u konfliktu sa nekom ulogom tekućeg korisnika iz skupa aktivnih uloga ARS. Ako postoji takva uloga onda tekućem korisniku nije dozvoljeno da on aktivira svoju konfliktnu ulogu, te se ona uklanja iz skupa aktivnih uloga ARS.

```
NAME: EnforceUDSoD
INPUT: ARS ⊆ R - set of user's roles in session
u ∈ U - user
OUTPUT: ARS ⊆ R - input set after UDSoD enforcement

CUS := dConflictUsers(u)
for each cu ∈ CUS do
   CURS := activeRoles(cu)
   for each ur ∈ ARS do
      for each cur ∈ CURS do
         if dRolesInConflict(ur, cur) then
            removeFromSet(ur, ARS)
```

Algoritam 2.2: Dinamička separacija obaveza bazirana na korisnicima

Dinamička separacija obaveza bazirana na aktivnostima

Dinamička separacija obaveza bazirana na aktivnostima sprečava da korisnik ili međusobno konfliktni korisnici izvrše dinamički konfliktne aktivnosti. Konfliktne aktivnosti mogu da pripadaju jednom procesu ali i većem broju procesa, tj. podržana su intraprocesna i interprocesna ADSoD ograničenja. ADSoD ograničenje može da se definiše za međusobno konfliktne instance aktivnosti i za međusobno konfliktne definicije aktivnosti. Ukoliko se ograničenje odnosi na instance aktivnosti onda se ono primenjuje samo na instance za koje je definisano. Ako se ograničenje odnosi na definicije aktivnosti onda postoje dve vrste ovog ograničenja:

- *regularno* - ograničenje se primenjuje na sve instance navedenih definicija aktivnost, i

- *interno (intraprocesno intrainstancno)* - važi samo za intraprocesna ograničenja. Ovo ograničenje primenjuje na instance aktivnosti u okviru **iste** instance procesa.

Da bi se ovaj tip dinamičke separacije obaveza mogao definisati neophodno je uvesti *istoriju izvršenja poslovnih procesa*. Istorija izvršenja poslovnih procesa definisana je kao par $(HISTORY, \succeq_H)$, gde $HISTORY$ predstavlja skup koga čine prethodno izvršene instance aktivnosti, a $\succeq_H$ je relacija parcijalnog uređenja nad tim skupom. Ako važi $h_i \succeq_H h_j \wedge h_i, h_j \in HISTORY$ znači da je aktivnost predstavljena elementom h_i izvršena pre aktivnosti predstavljene elementom h_j. Skup $HISTORY$ sastoji se od elemenata definisanih kao torka: $(cbpd, cbpi, cad, cai, u, r)$ gde je:

- $cbpd$ - definicija složenog poslovnog procesa ($cbpd \in CBPD$),
- $cbpi$ - instanca složenog poslovnog procesa ($cbpi \in CBPI \wedge instanceOf_{BP}(cbpi) = cbpd$),
- cad - definicija kompleksne aktivnosti ($activityDefOf(cad) = cbpd$),
- cai - instanca kompleksne aktivnosti ($activityInstOf(cai) = cbpi$),
- u - korisnik koji je izvršio cai ($u \in U$), i
- r - uloga posredstvom koje je korisniku bilo odobreno da izvrši cai ($r \in R$).

Predikatom $wasExecutedInst(cai, u)$ proverava se da li je korisnik u izvršio instancu aktivnosti cai, predikatom $wasExecutedDef(cad, u)$ proverava se da li je korisnik u izvršio bilo koju instancu definicije aktivnosti cad (regularni tip), dok se predikatom $wasExecutedDefInt(cbpi, cad, u)$ proverava da li je korisnik u izvršio instancu definicije aktivnosti cad u instnaci procesa $cbpi$ (interni tip):

$$wasExecutedInst(cai, u) \Leftarrow \exists(cbpd, cbpi, cad, cai, u, r) \in HISTORY$$
$$wasExecutedDef(cad, u) \Leftarrow \exists(cbpd, cbpi, cad, cai, u, r) \in HISTORY$$
$$wasExecutedDefInt(cbpi, cad, u) \Leftarrow \exists(cbpd, cbpi, cad, cai, u, r) \in HISTORY$$

Neka je sa $DCAD \subseteq CAD \times CAD \times CC$ označen skup dinamički konfliktnih definicija aktivnosti za regularni tip pri čemu će aktivnosti cad_i i cad_j iz $(cad_i, cad_j, cc) \in DCAD$ biti konfliktne samo ako je uslov cc zadovoljen.

Provera da li su dve definicije aktivnosti međusobno konfliktne definisana je $actDefInConflict$ predikatom:

$$actDefInConflict(cad_i, cad_j) \Leftarrow (\exists(cad_i, cad_j, cc) \in DCAD \wedge evalCond(cc)) \vee (\exists(cad_j, cad_i, cc) \in DCAD \wedge evalCond(cc))$$

Sada se ADSoD regularno ograničenje u slučaju konfliktnih definicija aktivnosti formalno može definisati na sledeći način:

$$cad_i, cad_j \in CAD \wedge actDefInConflict(cad_i, cad_j) \wedge (\exists h_i \in HISTORY \wedge h_i.cad = cad_i) \wedge (\exists h_j \in HISTORY \wedge h_j.cad = cad_j) \Rightarrow h_i.u \neq h_j.u \wedge \neg usersInConflict(h_i.u, h_j.u)$$

Neka je skup dinamički konfliktnih definicija aktivnosti za interni tip označen sa $DCADI \subseteq CAD \times CAD \times CC$ pri čemu će aktivnosti cad_i i cad_j iz $(cad_i, cad_j, cc) \in DCADI$ biti konfliktne samo ako je uslov cc zadovoljen. Provera da li su dve definicije aktivnosti međusobno konfliktne definisana je $actDefInConflictInt$ predikatom:

$$actDefInConflictInt(cad_i, cad_j) \Leftarrow (\exists(cad_i, cad_j, cc) \in DCADI \wedge evalCond(cc)) \vee (\exists(cad_j, cad_i, cc) \in DCADI \wedge evalCond(cc))$$

Sada se ADSoD interno ograničenje u slučaju konfliktnih definicija aktivnosti formalno može definisati na sledeći način:

$$cad_i, cad_j \in CAD \wedge actDefInConflictInt(cad_i, cad_j) \wedge (\exists h_i \in HISTORY \wedge h_i.cad = cad_i) \wedge (\exists h_j \in HISTORY \wedge h_j.cad = cad_j) \wedge h_i.cbpi = h_j.cbpi \Rightarrow h_i.u \neq h_j.u \wedge \neg usersInConflict(h_i.u, h_j.u)$$

Na sličan način definiše se ADSoD za instance složenih aktivnosti. Neka se sa $DCAI \subseteq CAI \times CAI \times CC$ označi skup dinamički konfliktnih instanci aktivnosti. Predikatom $actInstInConflict$ provera se da li su dve instance aktivnosti međusobno konfliktne:

$$actInstInConflict(cai_i, cai_j) \Leftarrow (\exists(cai_i, cai_j, cc) \in DCAI \wedge evalCond(cc)) \vee (\exists(cai_j, cai_i, cc) \in DCAI \wedge evalCond(cc))$$

ADSoD ograničenje u slučaju konfliktnih instanci aktivnosti formalno je definisano na sledeći način:

$$cai_i, cai_j \in CAI \wedge actInstInConflict(cai_i, cai_j) \wedge$$
$$(\exists h_i \in HISTORY \wedge h_i.cai = cai_i) \wedge (\exists h_j \in HISTORY \wedge h_j.cai = cai_j)$$
$$\Rightarrow h_i.u \neq h_j.u \wedge \neg usersInConflict(h_i.u, h_j.u)$$

Funkcije za određivanje dinamički konfliktnih definicija i instanci složenih aktivnosti definisane su kao preslikavanja:

$$dConflictDefActivity : CAD \rightarrow 2^{CAD}$$
$$dConflictDefActivity(cad_i) = \{cad_j \mid actDefInConflict(cad_i, cad_j)\}$$

$$dConflictDefActivityInt : CAD \rightarrow 2^{CAD}$$
$$dConflictDefActivityInt(cad_i) = \{cad_j \mid actDefInConflictInt(cad_i, cad_j)\}$$

$$dConflictInstActivity : CAI \rightarrow 2^{CAI}$$
$$dConflictInstActivity(cai_i) = \{cai_j \mid actDefInConflict(cai_i, cai_j)\}$$

Sprovođenje regularnog ADSoD ograničenja za definicije aktivnosti nad skupom definicija aktivnosti prikazano je algoritmom 2.3, dok je sprovođenje ADSoD ograničenja za skup instanci aktivnosti prikazano algoritmom 2.4. Kao što je prethodno rečeno na instance se primenjuju ograničenja definisana za same instance ali i ograničenja definisana za njihove definicije (regularna i interna). Rezltat izvršenja oba algoritma je ulazni skup iz koga su uklonjene aktivnosti koje korisnik u skladu sa ADSoD ograničenjem nema pravo da izvrši.

```
NAME: EnforceADSoDDef
INPUT: ADS ⊆ CAD - activity ddefinition set
: u ∈ U - user
OUTPUT: ADS ⊆ CAD - input set after ADSoD enforcement for definitions

US := {u} ∪ dConflictUsers(u)
for each cad ∈ ADS do
   CAS := dConflictDefActivity(cad)
   for each confAD ∈ CAS do
      for each cu ∈ US do
         if wasExecutedDef(confAD, cu) then
            removeFromSet(cad, ADS)
```

Algoritam 2.3: ADSoD ograničenje za definicije - regularno

```
NAME: EnforceADSoDInst
INPUT: AIS ⊆ CAI - activity instance set
: u ∈ U - user
OUTPUT: AIS ⊆ CAI - input set after ADSoD enforcement for instances

US := {u} ∪ dConflictUsers(u)
{Enforce for instances}
for each cai ∈ AIS do
   CAIS := dConflictInstActivity(cai)
   for each confAI ∈ CAIS do
      for each cu ∈ US do
         if wasExecutedInst(confAI, cu) then
            removeFromSet(cai, AIS)
{Enforce for definitions - regular case}
for each cai ∈ AIS do
   CADS := dConflictDefActivity(instanceOf_A(cai))
   for each confAD ∈ CADS do
      for each cu ∈ US do
         if wasExecutedDef(confAD, u) then
            removeFromSet(cai, AIS)
{Enforce for definitions - intra}
for each cai ∈ AIS do
   CADS := dConflictDefActivityInt(instanceOf_A(cai))
   cbpi := activityInstOf(cai)
   for each confAD ∈ CADS do
      for each cu ∈ US do
         if wasExecutedDefInt(cbpi, confAD, u) then
            removeFromSet(cai, AIS)
```

Algoritam 2.4: ADSoD ograničenje za instance

Dinamičko povezivanje obaveza

Dinamičko povezivanje obaveza predstavlja suprotnost prethodnim ograničenjima. Korisnik koji je izvršio jednu aktivnost je u obavezi da izvrši i neku drugu aktivnost. Skup dinamički povezanih aktivnost DBA definisan je kao $DBA = \{(cad_i, cad_j, cc) \mid cad_i, cad_j \in CAD \wedge activityDefOf(cad_i) = activityDefOf(cad_j) \wedge cc \in CC\}$, što praktično znači da COBAC model dozvoljava da se kao povezane aktivnosti mogu definisati samo aktivnosti iz istog poslovnog procesa. Ukoliko je korisnik u određenoj instanci poslovnog procesa izvršio neku povezanu aktivnost onda ga ovaj tip ograničenja obavezuje da u toj istoj instanci izvrši i sve druge aktivnosti koje su povezane sa tom aktivnošću. Slično kao i kod separacije obaveza, aktivnosti cad_i i cad_j će biti povezane ako je kontekstni uslov cc zadovoljen. Predikat $activityInBind$ pro-

verava da li su dve aktivnosti međusobno povezane:

$$activityInBind(cad_i, cad_j) \Leftarrow (\exists(cad_i, cad_j, cc) \in DBA \wedge evalCond(cc)) \vee (\exists(cad_j, cad_i, cc) \in DBA \wedge evalCond(cc))$$

Formalno DBoD ograničenje je predstavljeno na sledeći način:

$$cad_i, cad_j \in CAD \wedge activityInBind(cad_i, cad_j) \wedge (\exists h_i \in HISTORY \wedge h_i.cad = cad_i) \wedge (\exists h_j \in HISTORY \wedge h_j.cad = cad_j) \wedge (h_i.cbpi = h_j.cbpi) \Rightarrow h_i.u = h_j.u$$

Funkcija $dBindActivity : CAD \rightarrow 2^{CAD}$ za određivanje dinamički povezanih aktivnosti definisana je na sledeći način:

$$dBindActivity(cad_i) = \{cad_j \mid activityInBind(cad_i, cad_j)\}$$

Algoritmom 2.5 opisano je sprovođenje DBoD ograničenja nad skupom instanci aktivnosti. Izlaz iz algoritma je ulazni skup AIS iz koga su uklonjne instnace aktivnosti koje treba da izvrši neki drugi korisnik.

```
NAME: EnforceDBoD
INPUT: AIS ⊆ CAI - activity instance set
u ∈ U - user
OUTPUT: AIS ⊆ CID - input set after DBoD enforcement

for each cai ∈ AIS do
   BAS := dBindActivity(cai)
   for each bind ∈ BAS do
      if ¬ wasExecutedInst(bind, u) then
         removeFromSet(cai, AIS)
```

Algoritam 2.5: Dinamičko povezivanje obaveza

Dinamičko ograničenje izvršenja

Dinamičko ograničenje izvršenja onemogućava da korisnik izvrši odgovarajuću aktivnost iako ima pravo da je izvrši posredstvom uloge koja mu je dodeljena. Praktično, ovo ograničenje predstavlja dodatni uslov koji treba da se zadovolji da bi korisniku bilo dozvoljeno da izvrši zahtevanu aktivnost. Kako se ovo ograničenje može definisati za definicije i instance aktivnosti ono je

predstavljeno kao par $(cad, cc), cad \in CAD, cc \in CC$ u slučaju definicije aktivnosti, odnosno kao par $(cai, cc), cai \in CAI, cc \in CC$ u slučaju instance aktivnosti. Sa $DECD \subseteq CAD \times CC$ označen je skup dinamičkih ograničenja izvršenja definisanih za definicije aktivnosti, a sa $DECI \subseteq CAI \times CC$ skup ograničenja definisanih za instance aktivnsoti. Izvršenje odgovarajuće instance aktivnosti, sa stanovišta zadovoljenja definisanih ograničenja izvršenja, je moguće ukoliko su zadovoljena ograničenja izvršenja definisana za tu instancu, odnosno za njezinu definiciju.

Provera ograničenja koja su definisana za definicije aktivnosti realizovana je predikatom $dExecConstDefSatisfied$, a provera ograničenja definisanih za instance predikatom $dExecConstInstSatisfied$:

$$dExecConstDefSatisfied(cad) \Leftarrow \forall(cad, cc) \in DECD \wedge evalCond(cc)$$
$$dExecConstInstSatisfied(cai) \Leftarrow \forall(cai, cc) \in DECI \wedge evalCond(cc)$$

Algoritam 2.6 opisuje sprovođenje DEC ograničenja za definicije nad skupom definicija aktivnosti, a sprovođenje DEC ograničenja za skup instanci aktivnosti prikazan je algoritmom 2.7. Iz ulaznog skupa uklanjaju se one aktivnosti za koje ograničenje izvršenja nije zadovoljeno. Kao što je prethodno rečeno na instance se primenjuju ograničenja definisana za same instance ali i ograničenja definisana za njihove definicije.

NAME: EnforceDECDef
INPUT: $ADS \subseteq CAD$ - activity definition set
OUTPUT: $ADS \subseteq CAD$ - input set after DEC enforcement for definitions

```
for each cad ∈ ADS do
   if ¬ dExecConstDefSatisfied(cad)  then
      removeFromSet(cad, ADS)
```

Algoritam 2.6: Dinamičko ograničenje izvršenja za definicije aktivnosti

```
NAME: EnforceDECInst
INPUT: AIS ⊆ CAI - activity instance set
OUTPUT: AIS ⊆ CAI - input set after DEC enforcement for instances and definitions

for each cai ∈ AIS do
   if ¬ dExecConstInstSatisfied(cai) ∨ ¬ dExecConstDefSatisfied(instanceOf_A(cai))
   then
      removeFromSet(cai, AIS)
```

Algoritam 2.7: Dinamičko ograničenje izvršenja za instance aktivnosti

2.8 Sprovođenje kontrole pristupa

U ovom odeljku opisan je način sprovođenja kontrole pristupa definisan COBAC modelom. Sam proces sprovođenja kontrole pristupa sastoji se od četiri osnovne faze:

- aktivacije uloga,
- kreiranja liste aktivnosti koje korisnik može da izvrši,
- provere da li je korisniku dozvoljeno da izvrši odgovarajuću aktivnost u trenutku kada ona/on inicira njeno izvršavanje, i
- provere da li je dozvoljen pristup resursima kojima se pristupa u okviru izvršavanja aktivnosti.

2.8.1 Promenljivost kontekstnog uslova

Sa stanovišta promenljivosti identifikovane su dve vrste kontekstnih uslova:

- *sporo promenljivi* (*session safe*) kontekstni uslovi su oni uslovi čiji rezultat je nepromenljiv tokom korisničke sesije.
- *često promenljivi* (*activity safe*) kontekstni uslovi su oni uslovi čiji rezultat može da se menja tokom korisničke sesije, ali je nepromenljiv za vreme izvršavanja svake pojedinačne aktivnosti.
- *konstantno promenljivi* (*constantly varying*) kontekstni uslovi su oni uslovi čiji rezultat može da se menja tokom korisničke sesije ili tokom izvršenja aktivnosti.

Da bi se na konzistentan način definisalo sprovođenje kontrole pristupa pretpostavlja se da su kontekstni uslovi navedeni u relacijama dodele uloga

korisnicima kao i kontekstni uslovi za omogućavanje/onemogućavanje uloga *sporo promenljivi*, dok su uslovi u relacijama dodele prava izvršavanja aktivnosti ulogama i relacijama dodele privilegija aktivnostima *često promenljivi*. Uslovi u dinamičkoj separaciji obaveza i dinamičkoj separaciji obaveza baziranoj na korisnicima su *slabo promenljivi*, dok su uslovi u dinamičkoj separaciji obaveza baziranoj na aktivnostima, dinamičkom povezivanju obaveza i dinamičkom ograničenju izvršenja *često promenljivi*.

Na ovaj način uloge dodeljene korisniku u okviru njegove sesije se ne menjaju tokom sesije, već je moguće da se menjaju samo prava, dodeljena tim ulogama, da izvršavaju određene aktivnosti.

2.8.2 Aktivacija uloga

Prvi korak u aktivaciji uloga je učitavanje korisničkih uloga iz kojih se potom uklanjaju *onemogućene* uloge. Sledeća faza je da se od prethodno formiranog skupa korisničkih uloga aktiviraju određene uloge i da se potom sprovedu DSoD i UDSoD ograničenja. Način na koji će se vršiti aktivacija uloga COBAC model ostavlja otvorenim kako bi se zadovoljili zahtevi različitih sistema. Npr. aktivacija može da podrazumeva da korisnik odabere koje uloge želi da aktivira prilikom prijavljivanja na sistem, ali isto tako moguće je da sistem ima internu logiku koja će odlučiti koje uloge treba da se aktiviraju, bez interakcije sa korisnikom po ovom pitanju. Algoritmom 2.8 dat je jednostavan primer aktivacije uloga gde se praktično aktiviraju sve uloge koje su dodeljene korisniku.

```
NAME: ActivateRoles
INPUT: u ∈ U - user
s ∈ S - session of u
OUTPUT: ARS - activated roles
URS := usersRoles(u)
for each r ∈ URS do
   if disabled(r) then
      removeFromSet(r, URS)
ARS := activateAllRolesForSession(URS, s)
ARS := EnforceDSoD(ARS)
ARS := EnforceUDSoD(ARS, u)
```

Algoritam 2.8: Primer aktivacije uloga

2.8.3 Kreiranje liste aktivnosti

Kreiranje liste aktivnosti predstavlja odabir onih aktivnosti koje korisnik ima pravo da izvrši. Formirana lista aktivnosti predstavljena je kao torka $(SCPS, ECAS, ESAS)$ gde:

- $SCPS \subseteq CAD \wedge \forall cad \in SCPS, typeOf(cad) = start$ (*Start Complex Process Set*) predstavlja skup definicija složenih aktivnosti tipa "start" koje korisnik može da izvrši, tj. može da kreira novu instancu složenog poslovnog procesa.
- $ECAS \subseteq CAI$ (*Execute Complex Activity Set*) predstavlja skup instanci složenih poslovnih aktivnosti koje korisnik ima pravo da izvrši.
- $ESAS \subseteq SAD$ (*Execute Simple Activity Set*) predstavlja skup jednostavnih poslovnih aktivnosti koje korisnik ima pravo da izvrši.

Algoritmom 2.9 opisan je postupak kreiranja liste aktivnosti koje u datom trenutku korisnik ima pravo da izvrši.

Formiranje $SCPS$ skupa podrazumeva da se za svaku aktivnu ulogu korisnika učitaju definicije kompleksnih aktivnosti tipa "start" dodeljene tim ulogama, a potom da se nad učitanim skupom sprovede verifikacija ograničenja. U ovom koraku proveravaju se ADSoD i DEC ograničenja definisana za definicije aktivnosti. Od ADSoD ograničenja, sprovode se samo ograničenja **regularnog** tipa, dok se ova ograničenja **internog** tipa ne sprovode. Kako još uvek ne postoji ni jedna izvršena aktivnost u ovoj instanci procesa, pošto "start" tip aktivnosti praktično inicira kreiranje nove instance složenog poslovnog procesa, ne može doći do konflikta internog tipa. Iz sličnog razloga ADSoD i DEC ograničenja definisana za instance nema potrebe sprovoditi jer još uvek ne postoji instanca aktivnosti. Pošto se i DBoD ograničenja odnose na istu instancu poslovnog procesa, a "start" tip aktivnosti je prva aktivnost bilo kog složenog poslovnog procesa za ovaj skup takođe nema potrebe sprovoditi DBoD ograničenja.

Skup $ECAS$ inicijalno se formira tako što u njega ulaze sve instance aktivnosti dodeljene aktivnim korisnikovim ulogama, a koje po redosledu izvršavanja poslovnih procesa treba da se izvrše. Ovako dobijen skup se filtrira tako što se nad njim sprovode ADSoD, DBoD i DEC ograničenja.

Skup $ESAS$ formira se tako što se za aktivne korisnikove uloge učitaju sve definicije jednostavnih poslovnih aktivnosti koje su im dodeljene. Pošto za

jednostavne poslovne aktivnosti COBAC modelom nije predviđeno definisanje ograničenja, onda se sprovođenje dinamičkih ograničenja za ovaj skup i ne sprovodi.

NAME: CreateActivityList
INPUT: $u \in U$ - user
$ARS \subseteq R$ - active roles of user in session
OUTPUT: $(SCPS, ECAS, ESAS)$ - user's activity list

{Create Start Complex Process Set (SCPS)}
$SCPS$:= $\bigcup_{r \in ARS}$ assignedCompProcToStart(r)
$SCPS$:= EnforceADSoDDef($SCPS$, u)
$SCPS$:= EnforceDECDef($SCPS$)

{Create Execute Complex Activity Set (ECAS)}
$ECAS$:= $\bigcup_{r \in ARS}$ assignedCompActToExec(r)
$ECAS$:= EnforceADSoDInst($ECAS$, u)
$ECAS$:= EnforceDBoD($ECAS$, u)
$ECAS$:= EnforceDECInst($ECAS$)

{Create Execute Simple Activity Set (ESAS)}
EAS := $\bigcup_{r \in ARS}$ assignedSimpActToExec(r)

Algoritam 2.9: Kreiranje liste aktivnosti

2.8.4 Provera dozvole za izvršenje aktivnosti

Kada korisnik odabere da izvrši neku aktivnost iz liste ponuđenih aktivnosti, potrebno je još jednom proveriti da li je korisniku dozvoljeno da izvrši tu aktivnost. Razlog za ovu dodatnu proveru je potencijalno postojanje *često promenljivih* kontekstnih uslova, ali i slučaj da je u međuvremenu neki konfliktni korisnik izvršio aktivnost koja je konfliktna sa aktivnošću koju korisnik želi da izvrši. Provera da li je dozvoljeno izvršenje odgovarajuće instance kompleksne aktivnosti predstavljena je algoritmom 2.10. Da bi korisnik mogao da izvrši aktivnost potrebno je da je nekoj od njegovih aktivnih uloga u sesiji dozvoljeno da izvrši tu aktivnost i da pri tome ne bude narušeno ni jedno od ograničenja. Provera da li je ograničenje narušeno svodi se na proveru da li je skup $ECAS$ prazan. Pre verifikacije ograničenja u ovom skupu se nalazi samo instanca aktivnosti koja se želi izvršiti, tako da ako neko ograničenje nije zadovoljeno algoritam za sprovođenje tog ograničenja će ovu instancu aktivnosti

ukloniti iz skupa. Sličan algoritam koristi se i kod provere da li je dozvoljeno izvršenje jednostavnih aktivnosti kao i kreiranje nove instance procesa.

```
NAME: ExecOfComplexActivityAllowed
INPUT: cai ∈ CAI - complex activity instance to execute
u ∈ U - user
ARS ⊆ R - active roles of user in session
OUTPUT: result - result, can user execute taks

result := false
for each r ∈ ARS do
   if canExecCompActInst(r, cai) then
      CAIS := {cai}
      ECAS := EnforceADSoDInst(CAIS, u)
      ECAS := EnforceDBoD(CAIS, u)
      ECAS := EnforceDECInst(CAIS)
      {If cai not removed from CAIS constraints allow execution of cai}
      if ECAS ≠ ∅ then
         result := true
```

Algoritam 2.10: Provera da li je dozvoljeno izvršenje složene aktivnosti

2.8.5 Provera dozvole pristupa resursu

Da bi aktivnost koju korisnik izvršava mogla da pristupi potrebnim resursima u cilju izvršenja zahtevane operacije nad njima potrebno je da aktivnost poseduje odgovarajuće privilegije. Provera da li je instanci složene aktivnosti *cai* dozvoljeno da nad resursom res izvrši operaciju *op* prikazana je algoritmom 2.11. Postupak sličan ovome primenjuje se i kod jednostavnih aktivnosti.

```
NAME: CanExecOperation
INPUT: cai ∈ CAI - complex activity instance
op ∈ Oper - operation
res ∈ Res - resource
OUTPUT: result - result, can operation be executed on resource

result := false
p := (res, op)
RPS := caiPermission(cai, p.op) ∪ cadPermission(instanceOf_A(cai), p.op)
for each rp ∈ RPS do
   if p ⊑ rp ∧ evalCond(rp.cc) then
      result := true
```

Algoritam 2.11: Provera dozvole pristupa resursu

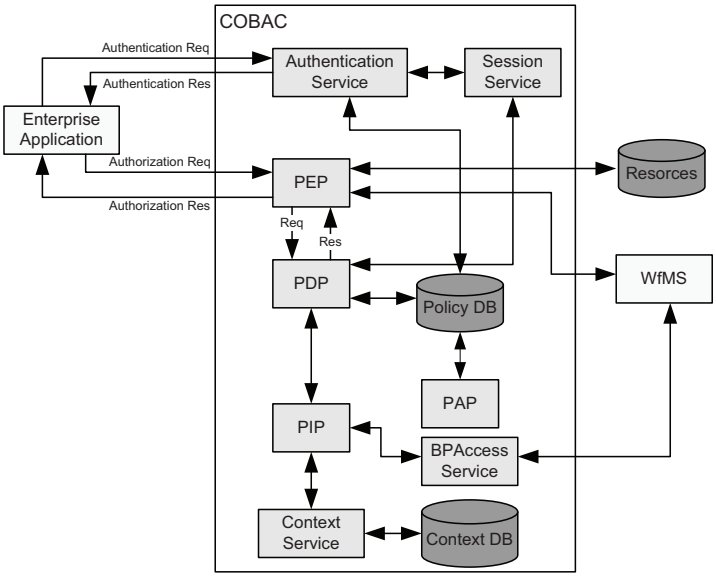
Poglavlje 3

Arhitektura sistema za sprovođenje kontrole pristupa

3.1 Globalna arhitektura sistema

Osnovne komponente arhitekture sistema kojim je realizovan COBAC model kontrole pristupa prikazane su na slici 3.1. *Authentication Service* modul namenjen je za autentifikaciju korisnika. On od klijenata prima zahteve za autentifikaciju, vrši autentifikaciju korisnika i obaveštava klijenta o rezultatu. Posrednik između autentifikacionog i autorizacionog dela sistema predstavlja *Session Service* komponenta, koja sadrži podatke o autentifikovanim korisnicima. *Policy Enforcement Point* (*PEP*) prihvata zahteve od klijenta za izvršenjem odgovarajuće aktivnosti poslovnog procesa, odnosno za pristup odgovarajućem resursu. Po prijemu ovih zahteva PEP formira autorizacioni zahtev koga prosleđuje *Policy Decision Point* (*PDP*) modulu. Ovaj modul na osnovu prava pristupa definisanih u bazi prava pristupa (*Policy DB*) proverava da li je odgovarajući pristup dozvoljen i o tome obaveštava PEP modul. Ukoliko je pristup dozvoljen, PEP prosleđuje zahtev klijenta ka resursu, odnosno ka *workflow* sistemu (kod izvršenja složene aktivnosti). Ako PDP modul, prilikom evaluacije prava pristupa, treba da pristupi podacima iz *workflow* sistema (podacima o složenim poslovnim procesima), odnosno kontekstnim podacima, on im pristupa posredstvom *Policy Information Point* (*PIP*) modula. Ovaj modul podacima iz baze kontekstnih podataka pristupa preko *Context Service*

modula, a podacima iz *workflow*-a pomoću *BPAccess Service* modula. Administracija baze prava pristupa realizovana je posredstvom *Policy Administration Point* (*PAP*) modula.



Slika 3.1: Globalna arhitektura sistema

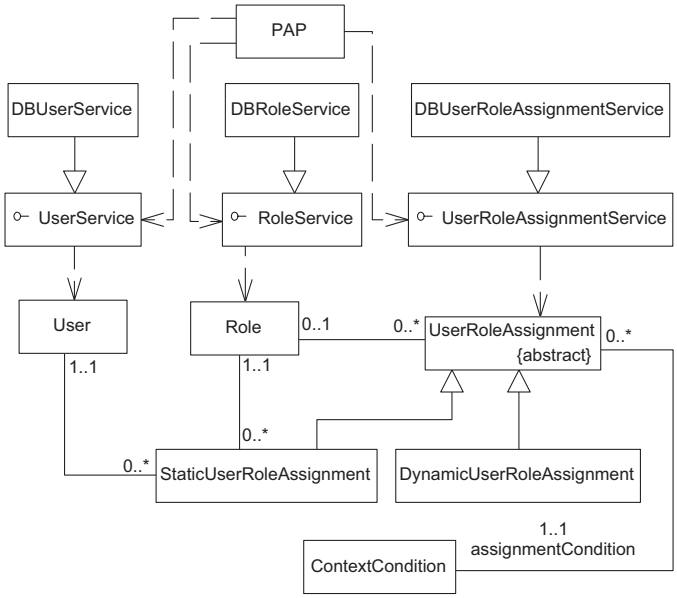
3.2 Podsistem za administraciju

U ovom odeljku opisan je *Policy Administration Point* (*PAP*) modul. Predstavljeni su njegovi podsistemi za administraciju korisnika, uloga, poslovnih procesa sa stanovišta kontrole pristupa, privilegija, i odgovarajućih relacija između navedenih entiteta. Takođe, opisan je i podsistem za administraciju ograničenja definisanih COBAC modelom.

3.2.1 Administracija korisnika i uloga

Klasama na slici 3.2 predstavljen je deo sistema namenjen za administraciju korisnika i uloga. Klasom `User` predstavljeni su korisnici sistema, dok su klasom `Role` opisane uloge u sistemu. Interfejs `UserService` definiše

metode za administraciju korisnika, a interfejs `RoleService` metode za administraciju uloga. Relacija dodele uloga korisnicima predstavljena je `UserRoleAssignment` klasom, odnosno njenim dvema specijalizacijama: `StaticUserRoleAssignment` za statičku relaciju dodele i `DynamicUserRoleAssignment` za dinamičku dodelu uloga korisniku. Zavisnost relacija dodele uloga korisniku od konteksta predstavljena je vezom između `UserRoleAssignment` klase i `ContextCondition` klase koja predstavlja kontekstni uslov. Administracija dodele uloga korisnicima definisana je interfejsom `UserRoleAssignmentService`. Klase `DBUserService`, `DBRoleService` i `DBUserRoleAssignmentService` predstavljaju implementaciju odgovarajućeg interfejsa i omogućuju administraciju navedenih podataka kada su oni smešteni u relacionoj bazi podataka.

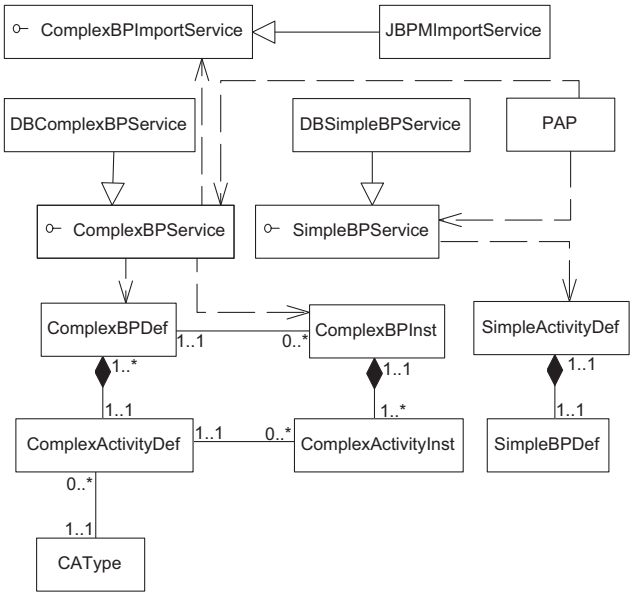


Slika 3.2: Dijagram klasa za administraciju korisnika i uloga

3.2.2 Administracija poslovnih procesa

Dijagramom klasa na slici 3.3 prikazan je deo sistema za administraciju poslovnih procesa, pri čemu se pomoću njih vrši samo administracija potrebna

sa stanovišta kontrole pristupa definisane COBAC modelom, ali ne i kompletna administracija poslovnih procesa. Definicija složenog poslovnog procesa predstavljena je klasom `ComplexBPDef`, dok je njegova instanca predstavljena klasom `ComplexBPInst`. Definicija složene aktivnosti definisana je klasom `ComplexActivityDef`, a njena instanca klasom `ComplexActivityInst`. Tip složene aktivnosti definisan COBAC modelom (*start*, *regular* i *end*) predstavljen je `CAType` klasom. `SympleBPDef` definiše jednostavni poslovni proces, a `SimpleActivityDef` aktivnost jednostavnog poslovnog procesa. Metode za administraciju složenih poslovnih procesa definisane su interfejsom `ComplexBPService`, a interfejsom `SimpleBPService` definisane su metode za administraciju jednostavnih poslovnih procesa. Administracija složenih i jednostavnih poslovnih procesa u slučaju kada se podaci o njima čuvaju u relacionim bazama podataka realizovana je klasama `DBComplexBPService` i `DBSimpleBPService`.



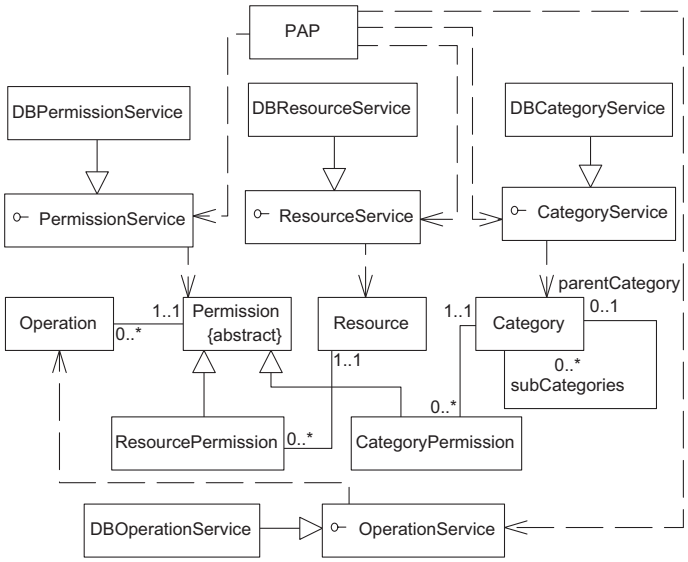
Slika 3.3: Dijagram klasa za administraciju korisnika i uloga

S obzirom da za definisanje poslovnih procesa već postoje različiti jezici i da se podaci o instancama složenih poslovnih procesa čuvaju u bazi podataka *workflow* sistema potrebno je obezbediti način preuzimanja ovih podataka za

potrebe definisanja prava pristupa. Interfejsom `ComplexBPImporterService` definisane su metode za preuzimanje ovih podataka. U slučaju JBoss jBPM *workflow* sistema [131] preuzimanje podataka realizovano je klasom `JBPMImportService`.

3.2.3 Administracija privilegija

Administracija privilegija, pored samih privilegija podrazumeva i administraciju resursa, kategorija kojima resursi pripadaju kao i operacija koje mogu da se izvršavaju nad resursima. Dijagramom klasa na slici 3.4 predstavljene su klase kojima je realizovana administracija ovog dela sistema. Klasom `Category` predstavljene su kategorije resursa opisanih klasom `Resource`. Operacije nad resursima definisane su `Operation` klasom. Privilegije (`Permission` klasa) po COBAC modelu mogu da se definišu na nivou konkretnog resursa (`ResourcePermission`) odnosno na nivou kategorije (`CategoryPermission`). Administracija prethodno navedenih entiteta realizovana je interfejsima `CategoryService`, `ResourceService`, `OperationService` i `PermissionService`, odnosno njihovim implementacijama.

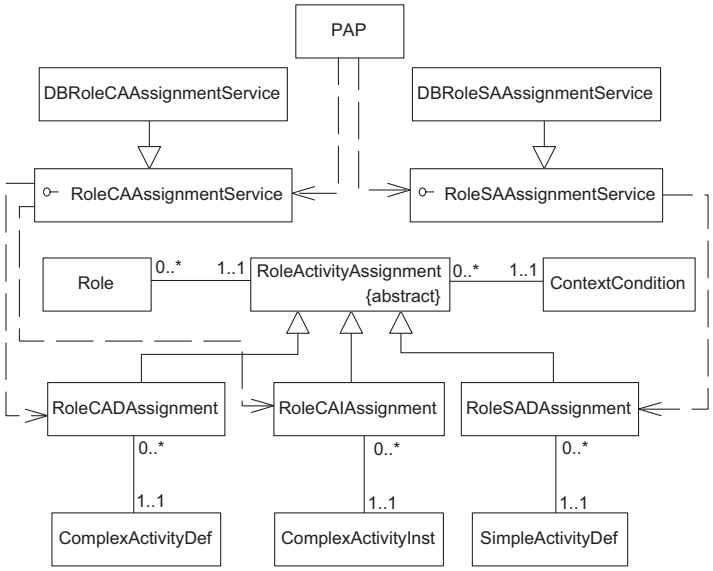


Slika 3.4: Dijagram klasa za administraciju privilegija

3.2.4 Administracija dodele aktivnosti i privilegija

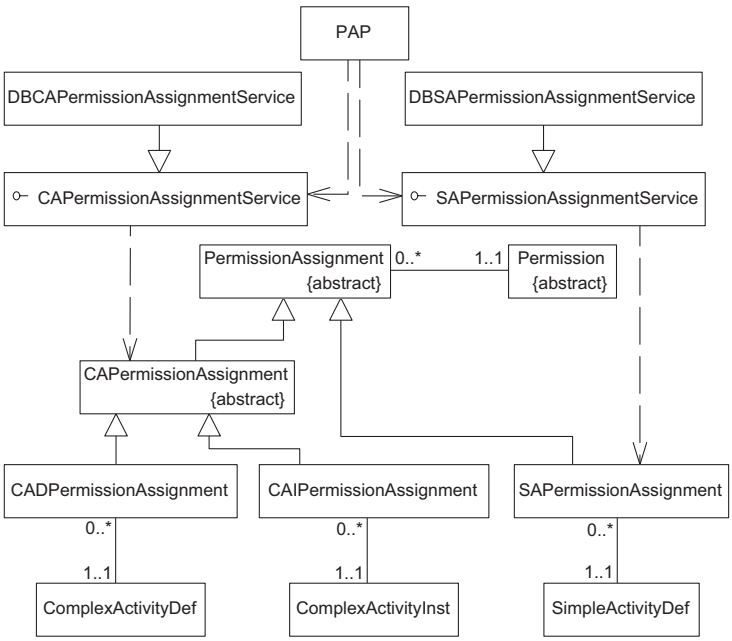
Pod ovim segmentom administracije podrazumeva se dodela aktivnosti ulogama koje imaju pravo izvršenja tih aktivnosti i dodela privilegija aktivnostima kako bi se u okviru njih mogle izvršiti zahtevane operacije nad resursima.

Na slici 3.5 prikazan je dijagram klasa kojima je realizovana dodela aktivnosti ulogama. Klasom `RoleActivityAssignment` predstavljena je relacija dodele aktivnosti ulogama koja je zavisna od konteksta. Specijalizacije ove klase omogućuju definisanje relacije dodele za definiciju složene aktivnosti (`RoleCADAssignment`), instancu složene aktivnosti (`RoleCAIAssignment`) i za definiciju jednostavne aktivnosti (`RoleSADAssignment`). Interfejsima `RoleCAAssignmentService` i `RoleSAAssignmentService` definisane su metode za administraciju relacija dodele aktivnosti ulogama u slučaju složenih, odnosno jednostavnih aktivnosti. Njihove implementacije `DBRoleCAAssignmentService` i `DBRoleSAAssignmentService` omogućuju administraciju u slučaju da su podaci smešteni u relacionoj bazi podataka.



Slika 3.5: Dijagram klasa za administraciju dodele aktivnosti ulogama

Administracija dodele privilegija aktivnostima realizovana je klasama prikazanim na slici 3.6. Osnovna klasa ovog segmenta je `PermissionAssignment` kojom je predstavljena relacija dodele privilegija aktivnostima. Dodela privilegija složenim aktivnostima predstavljena je posredstvom klase `CAPermissionAssignment`, odnosno njenih specijalizacija `CADPermissionAssignment` za definicije aktivnosti i `CAIPermissionAssignment` za instance aktivnosti. Dodela privilegija za jednostavne aktivnosti definisana je klasom `SAPermissionAssignment`. Administracija dodele privilegija realizovana je kroz implementacije `CAPermissionAssignmentService` interfejsa u slučaju složenih aktivnosti i `SAPermissionAssignmentService` interfejsa u slučaju jednostavnih aktivnosti.



Slika 3.6: Dijagram klasa za administraciju dodele privilegija aktivnostima

3.2.5 Administracija ograničenja

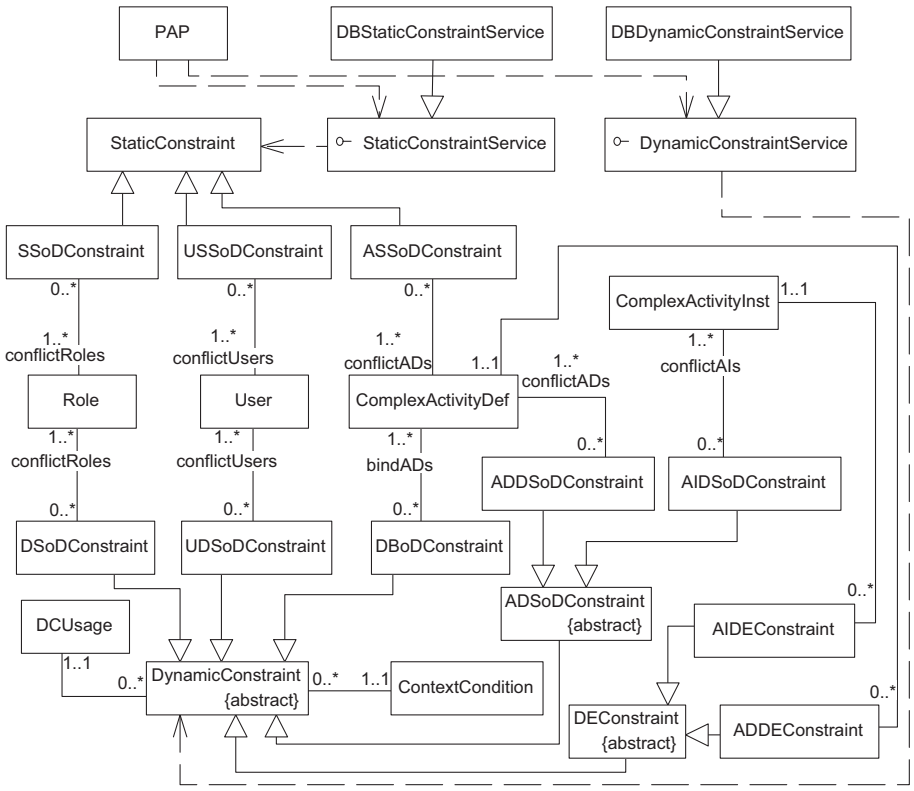
Administracija ograničenja uključuje administraciju statičkih i dinamičkih ograničenja definisanih COBAC modelom. Osnovne klase ovog dela podsistema za administraciju prikazane su dijagramom klasa na slici 3.7. Administracija statičkih ograničenja realizovana je odgovarajućom implementacijom `StaticConstraintService` interfejsa, dok je administracija dinamičkih ograničenja realizovana posredstvom odgovarajuće implementacije `DynamicConstraintService` interfejsa. Sva statička ograničenja predstavljena su kao specijalizacije `StaticConstraint` klase. Klasom `SSoDConstraint` opisana je statička separacija obaveza, klasom `USSoDConstraint` statička separacija obaveza bazirana na korisnicima, a sa `ASSoDConstraint` statička separacija obaveza bazirana na aktivnostima. Dinamička ograničenja realizovana su kao specijalizacije `DynamicConstraint` klase. Ova klasa poseduje relaciju sa kontekstnim uslovom (`ContextCondition`) čime je predstavljen uticaj konteksta na to ograničenje. Za dinamička ograničenja definiše se i mesto njegove primene (klasa `DCUsage`) gde se za odgovarajuće dinamičko ograničenje navodi u kom trenutku treba da se primenjuje (proverava). Moguća su sledeća osnovna mesta primene ograničenja:

- prilikom provere identiteta korisnika, odnosno prilikom aktivacije uloga,
- prilikom provere da li je dozvoljeno startovanje procesa, i
- prilikom provere da li je dozvoljeno izvršavanje instace složene aktivnosti.

Dinamička separacija obaveza definisana je `DSoDConstraint` klasom, dinamička separacija obaveza bazirana na korisnicima definisana je klasom `UDSoDConstraint`, a dinamičko povezivanje obaveza klasom `DBoDConstraint`. Dinamička separacija obaveza bazirana na aktivnostima definiše se za definicije (`ADDSoDConstraint`) i instance aktivnosti (`AIDSoDConstraint`), a isto tako i dinamičko ograničenje izvršenja (klase `ADDEConstraint` i `AIDEConstraint`).

Prilikom dodele uloga korisnicima, odnosno aktivnosti ulogama `PAP` vrši proveru odgovarajućih statičkih ograničenja. Uspostavljanje odgovarajuće dodele će biti dozvoljeno samo ako su sva ograničenja zadovoljena. Takođe, dodavanje novog ograničenja zahteva da sve prethodno definisane dodele ne narušavaju to ograničenje. Na ovaj način, ukoliko se čitava administracija sistema vrši isključivo posredstvom `PAP` komponente, podaci u sistemu (odgo-

varajuće relacije dodele) će uvek biti konzistentni u odnosu na statička ograničenja.



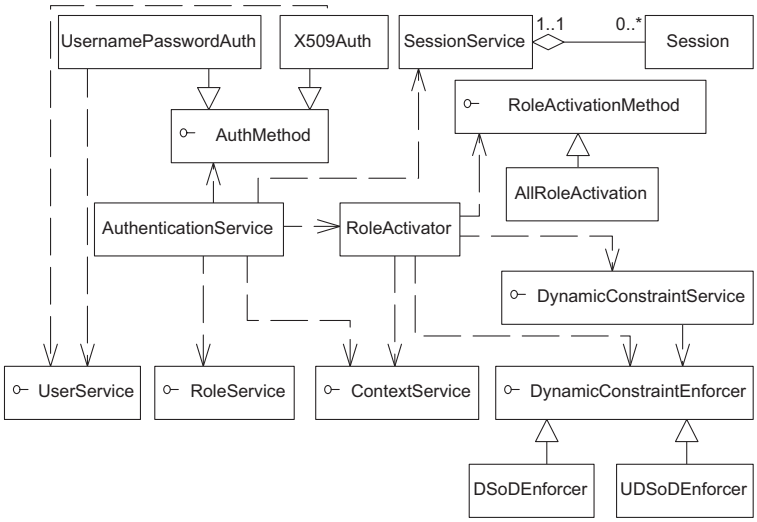
Slika 3.7: Dijagram klasa za administraciju ograničenja

3.3 Podsistem za proveru identiteta

Namena podsistema za proveru identiteta (*authentication*) je da prihvata zahteve klijenata za identifikaciju, izvrši proveru identiteta korisnika i o rezultatu obavesti klijenta. Dijagram klasa podsistema za proveru identiteta prikazan je na slici 3.8. Centralna klasa ovog podsistema je `AuthenticationService` koja predstavlja interfejs za komunikaciju sa ovim delom podsistema. Klasom `AuthMethod` predstavljeni su različiti mehanizmi provere identiteta

korisnika. Proveru identiteta zasnovana na korisničkom imenu i lozinci predstavljena je klasom `UsernamePasswordAuth`, a proveruža identiteta uz oslonac na infrastrukturu javnih ključeva (*Public Key Infrastructure, PKI*) [266] predstavljena je `X509Auth` klasom. Klasa `SessionService` omogućuje pristup sesiji korisnka koja je predstavljena `Session` klasom.

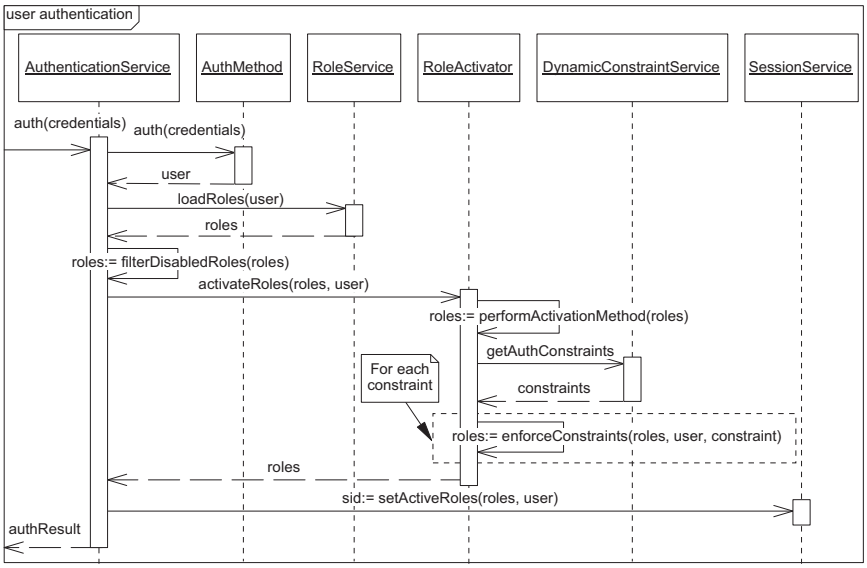
Aktivacija uloga realizuje se posredstvom `RoleActivator` klase. Ova klasa se za aktivaciju oslanja na odgovarajuću implementaciju `RoleActivationMethod` interfejsa. Klasa `AllRoleActivation` omogućuje aktivaciju svih uloga korisnika koje nisu *onemogućene*. Specijalizacije interfejsa `DynamicConstraintEnforcer` omogućuju proveru da li je odgovarajuće dinamičko ograničenje zadovoljeno. Prilikom aktivacije uloga COBAC modelom je definisana provera dinamičke separacije obaveza (klasa `DSoDEnforcer`) i dinamičke separacije obaveza bazirane na korisnicima (klasa `UDSoDEnforcer`).



Slika 3.8: Dijagram klasa podisistema za autentifikaciju

Postupak provere identiteta korisnika i aktivacije njegovih uloga prikazan je dijagramom sekvenci na slici 3.9. Pristigli zahtev za identifikaciju, `AuthenticationService` prosleđuje odgovarajućem mehanizmu za proveru identiteta (`AuthMethod`) koji vraća podatke o identfikovanom korisniku u slučaju uspešne identifikacije. Za identifikovanog korisnika posredstvom `RoleService`-a se uči-

tavaju uloge koje su mu dodeljene, a iz kojih se potom uklanjaju sve one uloge koje su *onemogućene*, tj. uloge koje se ne mogu aktivirati. Sledeći korak predstavlja aktivacija uloga koja se obavlja posredstvom `RoleActivator`-a. `RoleActivator` u okviru svoje metode `performActivationMethod` konsultuje odgovarajući mehanizam za aktivaciju uloga (odgovarajuću implementaciju `RoleActivationMethod` interfejsa) i potom nad aktiviranim ulogama sprovodi proveru potrebnih ograničenja. Ovim ograničenjima pristupa se posredstvom `DynamicConstraintService` komponente. Rezultat sprovođenja ograničenja je da su iz skupa aktiviranih uloga uklonjene sve one uloge koje narušavaju bilo koje od učitanih ograničenja. Poslednji korak je da se sa `SessionService` komponentom u sesiju korisnika postave njegove aktivne uloge i da se klijentu prosledi odgovor o rezultatu identifikacije.



Slika 3.9: Dijagram sekvenci provere identiteta korisnika

3.4 Podsistem za autorizaciju

Namena podsistema za autorizaciju je da proveri da li je korisniku dozvoljeno izvršavanje odgovarajuće operacije i da mu omogući njeno izvršavanje u

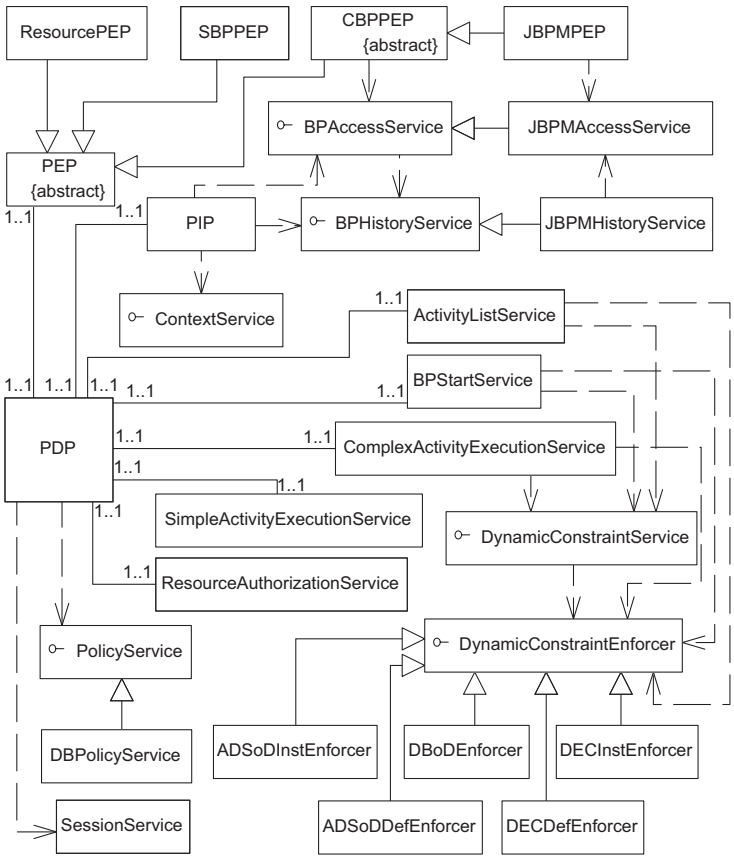
skladu sa definisanim pravima pristupa. Osnovne klase podsistema za autorizaciju prikazane su dijagramom klasa na slici 3.11.

Zahtev za izvršenjem odgovarajuće operacije nad zahtevanim resursom upućuje se određenoj specijalizaciji `PEP` klase. Predloženom arhitekturom predviđene su tri specijalizacije `ResourcePEP` za pristup resursima, `SBPPEP` za izvršenje jednostavne aktivnosti i `CBPPEP` za izvršenje odgovarajuće složene aktivnosti. U slučaju JBoss jBPM *workflow*-u koristi se `JBPMPEP` klasa. Prilikom pristupa konkretnom *workflow* sistemu `CBPPEP` koristi odgovarajuću implementaciju `BPAccessService` interfejsa. Za JBoss jBPM *workflow* razvijena je klasa `JBPMAccessService`.

Pre nego što `PEP` izvrši zahtevanu operaciju on se obraća `PDP`-u koji treba da proveri da li je izvršenje zahtevane operacije dozvoljeno. `PDP` korisničkoj sesiji pristupa posredstvom `SessionService`-a, a pravima pristupa smeštenim u bazu prava pristupa posredstvom odgovarajuće implementacije `PolicyService` interfejsa. Kreiranje liste aktivnosti koje korisnik može da izvrši `PDP` realizuje posredstvom `ActivityListService` klase. Za proveru da li je dozvoljeno startovanje složenog poslovnog procesa `PDP` koristi `BPStartService` klasu, a za proveru da li je dozvoljeno izvršenje instance složene aktivnosti koristi se `ComplexActivityExecutionService` klasa. Dozvolu za izvršenje jednostavne aktivnosti `PDP` proverava uz oslonac na `SimpleActivityExecutionService`, a provera dozvole za izvršenje zahtevane operacije nad navedenim resursom se vrši pomoću `ResourceAuthorizationService` klase. Pristup potrebnim dinamičkim ograničenjima koja je potrebno proveriti u ovoj fazi vrši se pomoću `DynamicConstraintService`-a. Provera ograničenja dinamičkog povezivanja obaveza realizovana je klasom `DBoDEnforcer`. Verifikacija ograničenja dinamičke separacije obaveza bazirane na aktivnostima realizovana je klasom `ADSoDDefEnforcer` za definicije aktivnosti i `ADSoDInstEnforcer` za instance aktivnosti. Za proveru dinamičkog ograničenja izvršenja definisane su klase `DecDefEnforcer` za definicije aktivnosti i `DecInstEnforcer` za instance aktivnosti.

Ukoliko `PDP` odnosno neki od njegovih podsistema treba da pristupi kontekstnim podacima ili podacima *workflow*-a on to može ostvariti posredstvom `PIP` klase. `PIP` kontekstnim podacima pristupa posredstvom `ContextService` interfejsa, dok podacima iz *workflow*-a pristupa posredstvom odgovarajuće im-

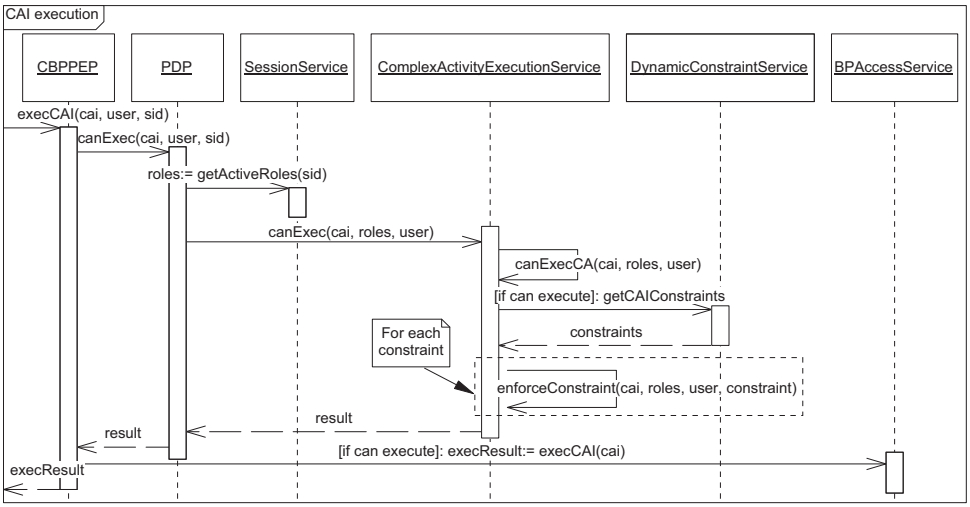
plementacije `BPAccessService`-a. Pristup istoriji izvršenja poslovnih procesa može se ostvariti preko impelmentacije `BPHistoryService` interfejsa.



Slika 3.10: Dijagram klasa podsistema za autorizaciju

Dijagramom sekvenci na slici 3.11 opisan je postupak prilikom izvršenja instance složene aktivnosti. Po prijemu zahteva za izvršenje `CBPPEP` konsultuje `PDP` radi provere da li je dozvoljeno izvršenje navedene aktivnosti. Aktivnim ulogama korisnika `PDP` pristupa posredstvom `SessionService`-a. Proveru da li je dozvoljeno izvršenje složene aktivnosti realizuje klasa `ComplexActivityExecutionService`. Ona u okviru svoje metode `canExecCAI` proverava da li je na osnovu definisanih prava pristupa dozvoljeno izvršenje zahtevane ak-

tivnosti, a potom učitava definisana ograničenja i proverava da li će neko od ograničenja biti narušeno ukoliko korisnik izvrši zahtevanu aktivnost. Krajnji rezultat odluke prosleđuje se nazad `BPPEP`-u koji, ako je dozvoljeno izvršenje posredstvom `BPAccessService` izvršava zahtevanu aktivnost.



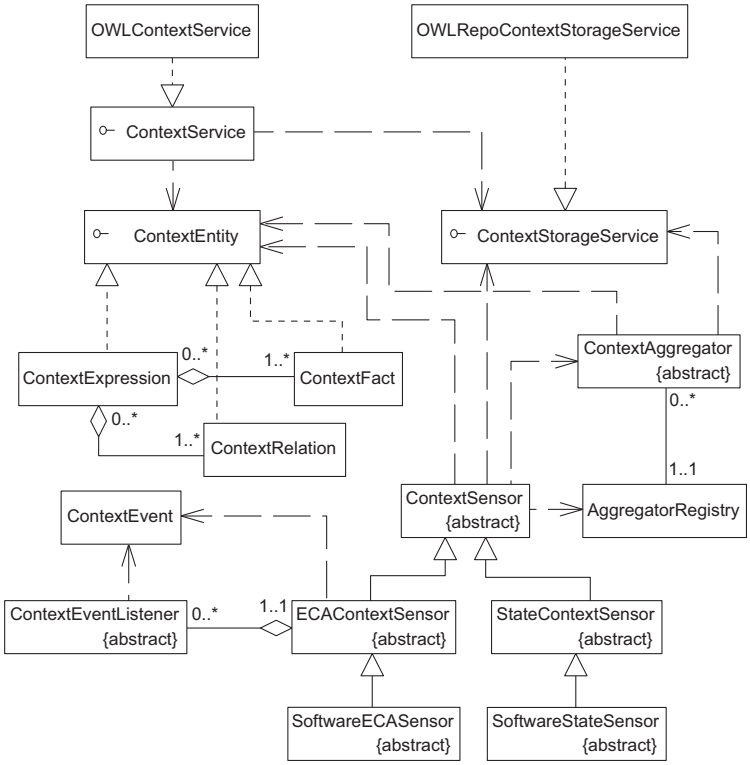
Slika 3.11: Dijagram sekvenci autorizacije za izvršenje složene aktivnosti

3.5 Kontekstni podsistem

Osnovne klase kontekstnog podsistema i njihove međusobne relacije prikazane su dijagramom klasa na slici 3.12. U osnovi, kontekstni podsistem se sastoji iz dela koji obezbeđuje kontekstno zavisnu kontrolu pristupa i dela koji obezbeđuje akviziciju konteksta.

Ostatak sistema za sprovođenje kontrole pristupa kontekstnom podsistemu posredstvom odgovarajuće implementacije `ContextService` interfejsa. U slučaju korišćenja OWL (*Web Ontology Language*) jezika [195] za predstavljanje konteksta koristi se `OWLContextService` klasa. Kontekstni entiteti definisani su odgovarajućom implementacijom `ContextEntity` interfejsa. Kontekstna činjenica predstavljena je `ContextFact` klasom, kontekstna relacija sa `ContextRelation`, a kontekstni izraz sa `ContextExpression` klasom.

Akvizicija kontekstnih podataka vrši se posredstvom *kontekstnih senzora* predstavljenih klasom `ContextSensor`. Akviziciju je moguće vršiti praćenjem promena u stanju nekog sistema (`StateContextSensor`) ili posredstvom odgovarajućih događaja, odnosno ECA (*Event Condition Action*) pravila [14, 92] (`ECAContextSensor`). U slučaju akvizicije posredstvom događaja, klasom `ContextEvent` predstavljen je odgovarajući događaj, a klasom `ContextEventListener` osluškivač (*listener*) za odgovarajući događaj. Akvizcija informacija iz softverskih komponenti realizuje se klasama koje su specijalizacija `SoftwareECASensor`, odnosno `SoftwareStateSensor` klase.

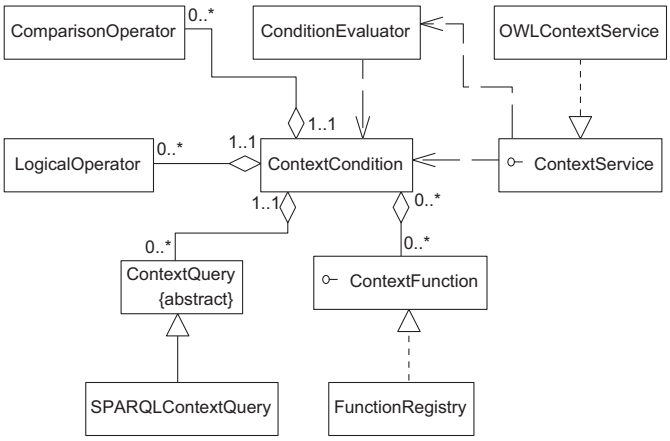


Slika 3.12: Dijagram klasa kontekstnog podsistema

Prikupljene kontekstne podatke kontekstni senzor može da smesti u odgovarajuće kontekstno skladište posredstvom `ContextStoregeService`-a, odnosno da prikupljene informacije prosledi odgovarajućim *kontekstnim agregatorima*

(`ContextAggregator`) koji na osnovu prikupljenih kontekstnih podataka formiraju izvedene kontekstne podatke (npr. kontekstne izraze ili nove kontekstne činjenice) i njih smeštaju u kontekstno skladište. Kontekstni senzor posredstvom `AggregatorRegistry`-a dobija informaciju kojim sve kontekstnim agregatorima treba da prosledi prikupljene podatke.

Kontekstni uslovi koji se koriste za definisanje prava pristupa realizovani su kroz `ContextCondition` klasu prikazanu na slici 3.13. Upit za kontektstne podatke modelovan je klasom `ContextQuery`. Klasa `SPARQLContextQuery` predstavlja implementaciju `ContextQuery`-a koja omogućuje pretragu kontekstnih podataka uz oslonac na SPARQL [236] jezik. Kontekstne funkcije predstavljene implementacijom `ContextFunction` interfejsa. Klasa `FunctionRegistry` predstavlja registar kontekstnih funkcija. Logički operatori u kontekstnom uslovu predstvljeni su `LogicalOperator` klasom, a operatori poređenja `ComparisonOperator` klasom.



Slika 3.13: Dijagram klasa kontekstnog uslova

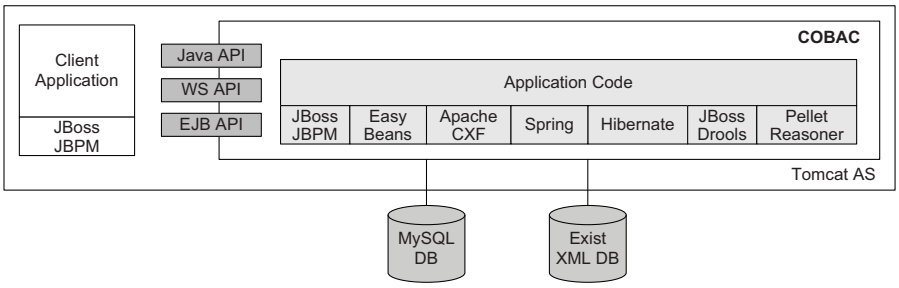
3.6 Specifikacija tehnologija za implementaciju prototipa sistema

Implementacija prototipa navedenog sistema izvedena je upotrebom programskog jezika Java. Prototip je implementiran prema modelu datom u po-

glavlju 2 i arhitekturi prezentovanoj u ovom poglavlju. Sistem je u potpunosti realizovan korišenjem tehnologija otvorenog koda (*open source*).

Prototipska implementacija je realizovana tako da se može koristiti na dva načina. Jedan je da se koristi kao modul klijentske aplikacije kojoj se pristupa preko definisanih interfejsa (Java API). Drugi način korišćenja prototipske implementacije je da ona bude posebna aplikacija koja se izvršava u okviru aplikativnog servera, a klijent joj pristupa preko odgovarajućih web servisa (WS API) ili preko EJB komponenti (EJB API). U drugom slučaju potrebno je da se aplikacija izvršava u okvirnu aplikativnog servera, poput *Apache Tomcat* [252] servera.

Na slici 3.14 prikazane su osnovne tehnologije korišćene prilikom realizacije prototipa sistema. Čitav sistem realizovan je uz oslonac na *Spring* [237] razvojno okruženje. Pristup bazi podataka vrši se posredstvom *Hibernate* [124] alata za objektno-relaciono mapiranje. Za pristup i izvršavanje upita nad kontekstom (predstavljenog OWL dokumentom) primenjen je *Pellet Reasoner* [200]. Implementacija kontekstnih ECA senzora izvršena je pomoću *JBoss Drools* [93] alata. Web servisi za pristupu sistemu realizovani su pomoću *Apache CXF* [80] biblioteke, a EJB komponente posredstvom *Easy Beans* [94] biblioteke. Ukoliko su poslovni procesi u sistemu realizovani pomoću *JBoss jBPM* [131] sistema za upravljanje poslovnim tokovima prototipska implementacija se oslanja i na *JBoss jBPM* biblioteku da bi mogla pristupati zahtevanim podacima u okviru procesa.



Slika 3.14: Tehnologije za implementaciju prototipa

Podaci definisani COBAC modelom smešteni su u odgovarajućoj bazi podataka. Kontekstni podaci (podaci definisani modelom konteksta) smešteni su

u XML bazi, poput *Exist* [96] XML baze, dok su ostali podaci smešteni u relacionoj bazi podataka, poput *MySQL* [175] baze.

Poglavlje 4

Verifikacija sistema na postupku izbora u nastavna zvanja

Potvrda koncepta COBAC modela i identifikacija prednosti i nedostataka u njegovom korišćenju izvodi se na bazi analize, dizajna i implementacije potpuno funkcionalnog segmenta informacionog sistema u cilju podrške određenom poslovnom procesu. Bez umanjenja opštosti odabran je poslovni proces definisan internim dokumentom sistema kvaliteta Q2.LO.07 Fakulteta tehničkih nauka koji opisuje postupak izbora u nastavno zvanje [245]. Kao posledica složenosti odabranog procesa verifikacija se sastoji iz više koraka:

- analiza procesa,
- specifikacija resursa i operacija nad resursima,
- specifikacija korisničkih uloga,
- analiza zahteva kontrole pristupa,
- specifikacija modela konteksta,
- specifikacija prava pristupa i ograničenja, i
- specifikacija kontekstnih modula.

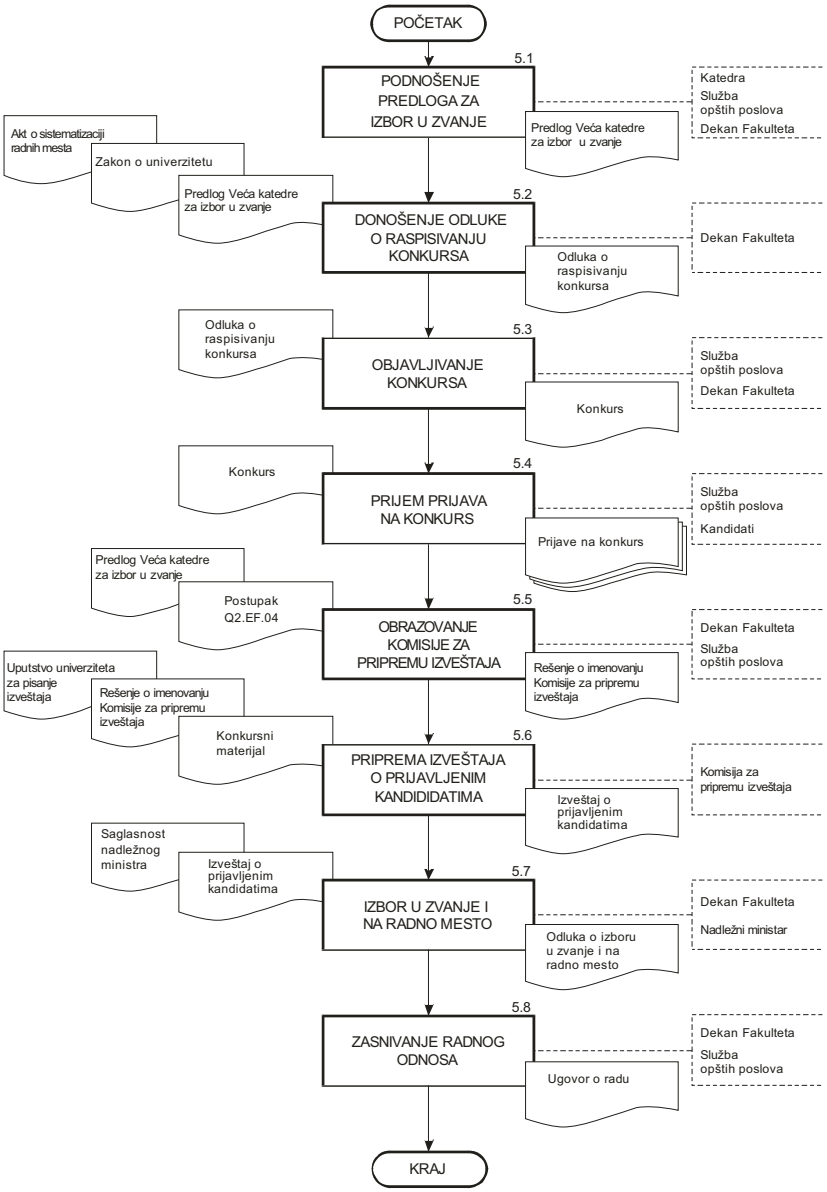
4.1 Analiza procesa

Dijagram toka procesa (slika 4.1) opisan je internim dokumentom sistema kvaliteta pod oznakom Q2.LO.07 [245]. Za svaki od pojedinačnih koraka dijagrama toka vezan je propratni tekst koji često otkriva njihovu složeniju se-

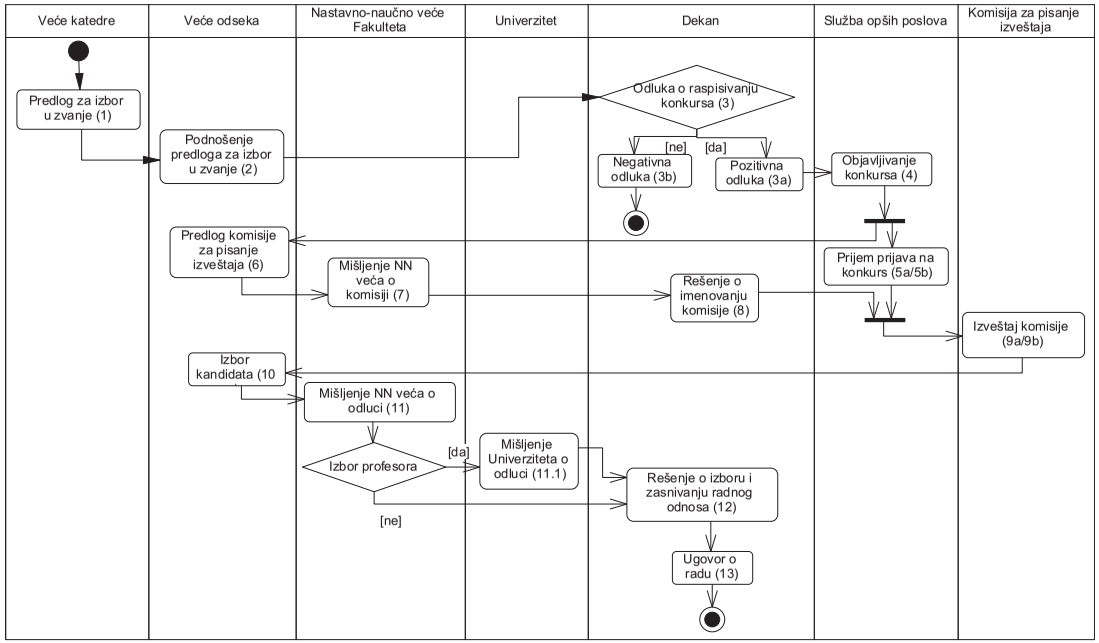
mantiku. Relativno visok nivo apstrakcije pojedinih koraka (korak 5.5 – Obrazovanje komisije za pripremu izveštaja) kao i bogata semantička struktura (korak 5.6 – Priprema izveštaja o prijavljenim kandidatima) navode na zaključak da dati opis toka procesa ne može direktno da se iskoristi u softverskoj implementaciji.

Detaljnija analiza pojedinih koraka dovodi do konkretnijeg opisa procesa prikazanog na slici 4.2. Sa slike može da se uoči prisustvo paralelizma koje se ogleda u mogućnosti istovremenog prikupljanja prijava na konkurs (korak 5a/5b) i postupka imenovanja komisije za ocenu kandidata (koraci 6, 7 i 8). Postupak obrazovanja komsije za pripremu izveštaja je složen i sastoji se iz sekvence koraka u kojima učestvuju različite institucije poput Veća odseka, Nastavno-naučnog veća Fakulteta i Dekana. Postupak izbora u nastavna zvanja je najvećim delom unifikovan bez obzira da li se kandidat bira u saradnička (saradnik u nastavi, asistent) ili nastavnička (docent, vanredni profesor, redovni profesor) zvanja. Ukoliko se kandidat bira u nastavničko zvanje potrebno je obezbediti i saglasnost Univerziteta o izabranom kandidatu.

Opis pojednih aktivnosti procesa prikazanog na slici 4.2, dokumenti i korisničke uloge koje u tim aktivnostima učestvuju dat je u tabelama 4.1 - 4.15.



Slika 4.1: Dijagram toka procesa Postupka izbora izbora u nastavno zvanje



Slika 4.2: Dijagram aktivnosti za poslovni proces Postupak izbora u nastavna zvanja

Aktivnost	Predlog za izbor u zvanje
Opis	Veće katedre inicira proces izbora u nastavno zvanje. Šef katedre popunjava i potpisuje dokumet *Predlog veća katedre za izbor u zvanje*. Po formiranju dokumenta označava da je aktivnost završena.
Dokumenti	Predlog veća katedre za izbor u zvanje
Uloge	Šef katedre

Tabela 4.1: Aktivnost *Predlog za izbor u zvanje*

Aktivnost	Podnošenje predloga za izbor u zvanje
Opis	Na osnovu predloga katedre Veće odseka podnosi dekanu predlog za izbor u zvanje. Šef odseka popunjava i potpisuje dokument *Predlog veća odseka za izbor u zvanje*. Po formiranju dokumenta označava da je aktivnost završena.
Dokumenti	Predlog veća odseka za izbor u zvanje
Uloge	Šef odseka

Tabela 4.2: Aktivnost *Podnošenje predloga za izbor u zvanje*

Aktivnost	Odluka o raspisivanju konkursa
Opis	Dekan odlučuje da li se prihvata predlog za izbor u nastavno zvanje ili ne. Zavisno od odluke, kreira se dokument *Odluka o raspisivanju konkursa* ili *Odluka o odbijanju predloga za izbor u zvanje*. Deo sadržaja kreiranog dokumenta inicijalno se popunjava sadržajem iz dokumenata *Predlog veća katedre/odseka za izbor u zvanje*. Dekan popunjava i potpisuje dokument i na kraju označava da je aktivnost završena.
Dokumenti	Odluka o raspisivanju konkursa, Odluka o odbijanju predloga za izbor u zvanje, Predlog veća katedre/odseka za izbor u zvanje
Uloge	Dekan

Tabela 4.3: Aktivnost *Odluka o raspisivanju konkursa*

Aktivnost	Objavljivanje konkursa
Opis	U slučaju pozitivne odluke dekana o raspisivanju konkursa, služba opštih poslova raspisuje konkurs, tj. kreira se dokument *Raspisan konkurs*. Delovi ovog dokumenta će inicijalno biti popunjeni sadržajem iz dokumenta *Odluka o raspisivanju konkursa*. Po popunjavanju dokumenta član službe opštih poslova ga potpisuje i označava da je aktivnost završena.
Dokumenti	Raspisan konkurs, Odluka o raspisivanju konkursa
Uloge	Član službe opštih poslova

Tabela 4.4: Aktivnost *Objavljivanje konkursa*

Aktivnost	Prijem prijava na konkurs
Opis	Služba opštih poslova unosi pristigle prijave na konkurs. Za svaku pristiglu prijavu kreira se po jedan dokument *Prijava na konkurs nastavnika*, odnosno *Prijava na konkurs saradnika* (zavisno od biranog zvanja). Uslov za prelazak u sledeću fazu procesa je da je istekao vremenski period predviđen za prijavu na konkurs i da je služba opštih poslova unela sve pristigle prijave na konkurs.
Dokumenti	Prijava na konkurs nastavnika/Prijava na konkurs saradnika, Raspisan konkurs
Uloge	Član službe opštih poslova

Tabela 4.5: Aktivnost *Prijem prijava na konkurs*

Aktivnost	Predlog komisije za pisanje izveštaja
Opis	Veće odseka predlaže komisiju za pisanje izveštaja o prijavljenim kandidatima. Na osnovu ovog predloga šef odseka popunjava dokument *Predlog veća odseka za komisiju za pisanje izveštaja*. Delovi ovog dokumenta će inicijalno biti popunjeni sadržajem iz dokumenta *Predlog veća odseka/katedre za izbor u zvanje*. Po formiranju dokumenta šef odseka ga potpisuje i potom označava da je aktivnost završena.
Dokumenti	Predlog veća odseka za komisiju za pisanje izveštaja, Predlog veća katedre za izbor u zvanje
Uloge	Šef odseka

Tabela 4.6: Aktivnost *Predlog komisije za pisanje izveštaja*

Aktivnost	Mišljenje NN veća Fakulteta o komisiji
Opis	NN veće Fakulteta daje mišljenje o predloženoj komisiji. Dekan na osnovu mišljenja NN veća popunjava dokument *Mišljenje NN veća Fakulteta o komisiji* i potom ga potpisuje. Delovi ovog dokumenta su inicijalno popunjeni sadržajem iz dokumenata *Raspisan konkurs* i *Predlog veća odseka za komisiju za pisanje izveštaja*. Po potpisivanju dokumenta dekan označava da je aktivnost završena.
Dokumenti	Mišljenje NN veća Fakulteta o komisiji, Raspisan konkurs, Predlog veća odseka za komisiju za pisanje izveštaja
Uloge	Dekan

Tabela 4.7: Aktivnost *Mišljenje NN veća Fakulteta o komisiji*

Aktivnost	Rešenje o imenovanju komisije
Opis	Rešenje o imenovanju komisije donosi dekan. Popunjava se dokument *Rešenje o imenovanju komisije*. Delovi dokumenta su inicijalno popunjeni sadržajem dokumenta *Mišljenje NN veća Fakulteta o komisiji*. Dekan potpisuje ovaj dokument i označava da je aktivnost završena.
Dokumenti	Rešenje o imenovanju komisije, Mišljenje NN veća Fakulteta o komisiji
Uloge	Dekan

Tabela 4.8: Aktivnost *Rešenje o imenovanju komisije*

Aktivnost	Izveštaj komsije
Opis	Komisija formira dokument koji predstavlja izveštaj komisije o prijavljenim kandidatima (*Izveštaj komisije za nastavnika* ili *Izveštaj komisije za saradnika*). Delovi ovog dokumenta inicijalno su popunjeni sadržajem prijava na konkurs kandidata kao i sadržajem dokumenata *Raspisan konkurs* i *Rešenje o imenovanju komisije*. Formirani dokument potpisuju predsednik i ostali članovi komisije. Kada su svi članovi komisije potpisali dokument predsednik komisije označava kraj aktivnosti.
Dokumenti	Izveštaj komisije za nastavnika/Izveštaj komisije za saradnika, Prijava na konkurs nastavnika/Prijava na konkurs saradnika, Raspisan konkurs, Rešenje o imenovanju komisije
Uloge	Član komisije, Predsednik komisije

Tabela 4.9: Aktivnost *Izveštaj komsije*

Aktivnost	Izbor kandidata
Opis	Na osnovu izveštaja komisije veće odseka donosi odluku o izboru kandidata odseka. Šef odseka popunjava i potpisuje dokument *Predlog veća odseka o izabranom kandidatu*. Delovi ovog dokumenta inicijalno su popunjeni sadržajem dokumenta *Rešenje o imenovanju komisije*. Po formiranju dokumenta šef odseka označava da je aktivnost okončana.
Dokumenti	Predlog veća odseka o izabranom kandidatu, Izveštaj komisije za nastavnika/Izveštaj komisije za saradnika, Rešenje o imenovanju komisije
Uloge	Šef odseka

Tabela 4.10: Aktivnost *Izbor kandidata*

Aktivnost	Mišljenje NN veća Fakulteta o odluci
Opis	NN veće fakulteta donosi mišljenje o izabranom kandidatu. Na osnovu ovog mišljenja kreira se dokument dokument *Mišljenje NN veća Fakulteta o izboru*, čiji delovi su inicijalno popunjeni sadržajem dokumenta *Predlog veća odseka o izabranom kandidatu*. Po kreiranju dokumenta dekan ga potpisuje i označava da je aktivnost završena.
Dokumenti	Mišljenje NN veća Fakulteta o izboru, Predlog veća odseka o izabranom kandidatu
Uloge	Dekan

Tabela 4.11: Aktivnost *Mišljenje NN veća Fakulteta o odluci*

Aktivnost	Mišljenje Univerziteta o odluci
Opis	U slučaju da je birano zvanje docent, vanredni ili redovni profesor, Univerzitet (odgovarajući organ) daje mišljenje o izabranom kandidatu. Na osnovu ovog mišljenja rektor popunjava i potpisuje dokument *Mišljenje Univerziteta o izboru*. Delovi ovog dokumenta će inicijalno biti popunjen sadržajem dokumenta *Predlog veća odseka o izabranom kandidatu*. Nakon što je dokument popunjen i potpisan rektor označava da je aktivnost završena.
Dokumenti	Mišljenje Univerziteta o izboru, Predlog veća odseka o izabranom kandidatu
Uloge	Rektor

Tabela 4.12: Aktivnost *Mišljenje Univerziteta o odluci*

Aktivnost	Rešenje o izboru i zasnivanju radnog odnosa
Opis	Rešenje o izboru u zvanje i zasnivanju radnog odnosa donosi dekan na osnovu pozitivnog mišljenja NN veća Fakulteta, odnosno mišljenja Univerziteta. Dekan popunjva dokument *Rešenje o izboru u zvanje i zasnivanju radnog odnosa*, čiji delovi su inicijalno popunjen sadržajem dokumenata *Rešenje o imenovanju komisije* i *Mišljenje NN veća Fakulteta o izboru*. Po popunjavanju ovog dokumenta dekan ga potpisuje i označava da je aktivnost završena.
Dokumenti	Rešenje o izboru u zvanje i zasnivanju radnog odnosa, Rešenje o imenovanju komisije, Mišljenje NN veća Fakulteta o izboru, Mišljenje Univerziteta o izboru
Uloge	Dekan

Tabela 4.13: Aktivnost *Rešenje o izboru i zasnivanju radnog odnosa*

Aktivnost	Formiranje ugovora o radu
Opis	Na osnovu rešenja o izboru u zvanje i zasnivanju radnog odnosa dekan formira dokument *Ugovor o radu* i potpisuje ga. Delovi dokument se inicijalno popunjavaju sadržajem dokumenta *Rešenje o izboru u zvanje i zasnivanju radnog odnosa*. Na kraju dekan označava da je aktivnost završena.
Dokumenti	Ugovor o radu, Rešenje o izboru u zvanje i zasnivanju radnog odnosa
Uloge	Dekan

Tabela 4.14: Aktivnost *Formiranje ugovora o radu*

Aktivnost	Potpisivanje ugovora o radu
Opis	Ugovor o radu kojeg je formirao i potpisao dekan sada potpisuje i izabrani kandidat, koji potom označava da je aktivnost završena. Završetak ove aktivnosti predstavlja i završetak procesa izbora u nastavno zvanje.
Dokumenti	Ugovor o radu
Uloge	Nastavnik/Saradnik (izabrani kandidat)

Tabela 4.15: Aktivnost *Potpisivanje ugovora o radu*

4.2 Specifikacija resursa i operacija nad resursima

Za svaki od koraka u dijagramu toka koji opisuje postupak izbora u nastavno zvanje (slika 4.1) vezani su popratni dokumenti. Prema dokumentu Q2.LO.07 [245] većina pratećih dokumenata je u slobodnom formatu (osim dokumenta koji predstavljaju *Izveštaj komisije o prijavljenim kandidatima* i *Ugovor o radu* čiju formu i sadržaj određuju uputstva Univerziteta i Zakon o radnim odnosima Republike Srbije, tim redosledom). Kako je analiza procesa *Postupka izbora u nastavno zvanje* pokazala da je njegova softverska implementacija složenija od one opisane u dokumentu Q2.LO.07 tako je broj dokumenata koji prate postupak veći. Na osnovu prikupljenih primeraka dokumenata koji prate pojedine korake procesa opisanog na slici 4.2 definisan je skup dokumenata prikazan u tabeli 4.16.

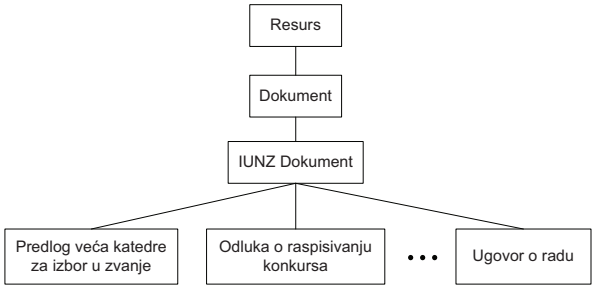
Šifra	Naziv dokumenta
1	Predlog veća katedre za izbor u zvanje
2	Predlog veća odseka za izbor u zvanje
3d	Odluka o raspisivanju konkursa
3n	Odluka o odbijanju predloga za izbor u zvanje
4	Raspisan konkurs
5a/5b	Prijava na konkurs nastavnika/saradnika
6	Predlog veća odseka za komisiju za pisanje izveštaja
7	Mišljenje NN veća Fakulteta o komisiji
8	Rešenje o imenovanju komisije
9a/9b	Izveštaj komisije za nastavnika/saradnika
10	Predlog veća odseka o izabranom kandidatu
11	Mišljenje NN veća Fakulteta o izboru
11.1	Mišljenje Univerziteta o izboru
12	Rešenje o izboru u zvanje i zasnivanju radnog odnosa
13	Ugovor o radu

Tabela 4.16: Dokumenti koji prate proces

S obzirom da je reč o poslovnom procesu koji je dokument orijentisan, resursi za koje je potrebno definisati prava pristupa su dokumenti koji su navedeni u tabeli 4.16. Na slici 4.3 prikazana je hijerarhija kategorija resursa. Praktično za svaki tip dokumenta koji se javlja u navedenom poslovnom procesu definisana je odgovarajuća kategorija.

Analizom svih aktivnosti poslovnog procesa *Postupak izbora u nastavna zvanja* utvrđeno je da se nad dokumentima izvršavaju sledeće operacije:

- čitanje dokumenta,
- kreiranje dokumenta,
- brisanje dokumenta,
- ažuriranje dokumenta, i
- digitalno potpisivanje dokumenta.



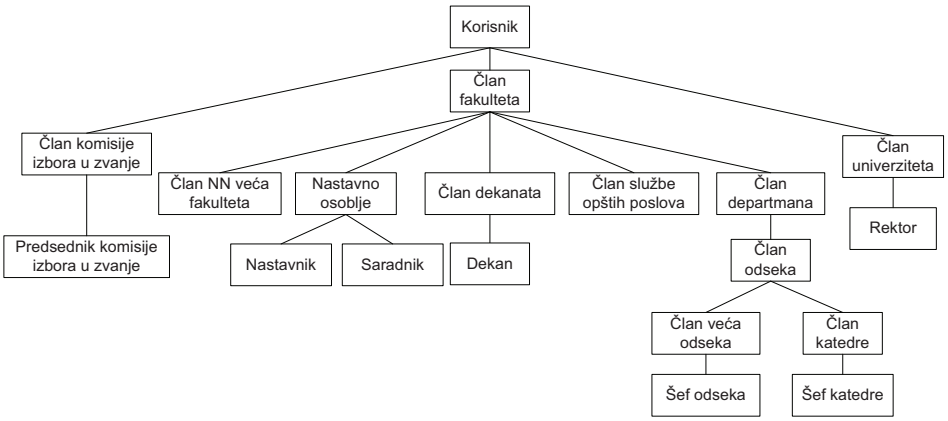
Slika 4.3: Kategorije resursa

4.3 Specifikacija korisničkih uloga

Analizom svih korisnika koji učestvuju u procesu izbora u nastavna zvanja formirana je hijerarhija korisničkih uloga prikazana na slici 4.4. Iako je COBAC modelom omogućeno višestruko nasleđivanje uloga, specificirana hijerarhija uloga koristi isključivo jednostruko nasleđivanje jer takva forma hijerarhije znatno pojednostavljuje administraciju u odnosu na slučaj da je primenjena višestruka hijerarhija uloga. Uvođenje jednostruke hijerarhije uloga u ovom slučaju nije narušilo ni jedan aspekt kontrole pristupa definisan kroz COBAC model.

Dalja analiza uloga utvrdila je da za postojeće uloge ne postoji potreba za definisanjem uslova omogućavanja/onemogućavanja (v. odeljak 2.5), odnosno sve uloge su uvek omogućene, tj. za sve uloge važi: $sc = \sigma, sct = ec$.

Takođe, daljom analizom korisničkih uloga koje se javljaju u procesu izbora u nastavna zvanja uočeno je da pored statički dodeljenih uloga korisnicima potrebno je omogućiti i dinamičku dodelu uloga (v. odeljak 2.5.1). Kako u aktivnosti *Predlog komisije za pisanje izveštaja* (tabela 4.5) veće odseka predlaže članove komisije, uloge *Član komisije izbora u zvanje* odnosno *Predsednik komisije izbora u zvanje* se dinamički dodeljuju izabranim članovima komisije. Ove uloge nije prihvatljivo statički dodeliti korisnicima jer nije moguće unapred znati da li će neko biti član komisije ili ne.



Slika 4.4: Hijerarhija korisničkih uloga

4.4 Analiza zahteva kontrole pristupa

Za svaku od prethodno navedenih aktivnosti *Postupka izbora u nastavna zvanja* potrebno je izanalizirati zahteve kontrole pristupa i na osnovu toga definisati odgovarajuća prava pristupa i ograničenja. Odnosno, potrebno je određenim korisničkim ulogama dodeliti pravo izvršavanja pojedinih aktivnosti. Isto tako potrebno je dodeliti odgovarajuće privilegije aktivnostima kako bi se u okviru njih mogle izvršavati potrebne operacije nad specificiranim dokumentima.

Na osnovu prethodno pomenutog može se uočiti da je za navedeni poslovni proces potrebno definisati značajan broj prava pristupa. U cilju preglednosti i razumljivosti, a bez umanjenja opštosti i značaja u ovom odeljku će biti data samo analiza pojedinih (reprezentativnih) zahteva kontrole pristupa.

Zahtev 4.1 *Da bi dekan mogao da donese odluku o raspisivanju konkursa (v. tabelu 4.3), ulozi* Dekan *treba dodeliti pravo za izvršavanje aktivnosti* Odluka o raspisivanju konkursa*. Samoj aktivnosti je potrebno dodeliti privilegije za čitanje, kreiranje, ažuriranje, brisanje i potpisivanje dokumenata* Odluka o raspisivanju konkursa *i* Odluka o odbijanju predloga za izbor u zvanje*. Takođe, ovoj aktivnosti je potreno dodeliti i privilegiju za čitanje dokumenta* Predlog veća katedre za izbor u zvanje*, na osnovu koga dekan odlučuje o raspisivanju konkursa.*

Zahtev 4.2 *Ovaj slučaj demonstrira upotrebu kontekstnih uslova. Uloge* Član komisije izbora u zvanje *i* Predsednik komisije izbora u zvanje *nije prikladno statički dodeliti korisnicima, već je potrebno da se one dodele onim korisnicima koji su tokom sprovođenja postupka izbora u zvanja imenovani za članove komisije, odnosno za predsednika komisije. Fromirana komisija za izbor u zavanje ima pravo da izvrši aktivnost* Izveštaj komisije *(v. tabelu 4.9), ali isključivo u okviru instance procesa u kojoj su imenovani za članove komisije. Da bi članovi komisije mogli da formiraju izveštaj potrebno je da imaju privilegije za čitanje, kreiranje, ažuriranje, brisanje i potpisivanje dokumenata* Izveštaj komisije za nastavnika/saradnika *kao i privilegiju za čitanje dokumenata* Prijava na konkurs nastavnika/saradnika, Raspisan konkurs *i* Rešenje o imenovanju komisije *koji pripadaju toj instanci procesa.*

Zahtev 4.3 *Kako bi se sprečio sukob interesa osoba koja je odabrana na osnovu izveštaja komisije ne sme da bude i član te iste komisije.*

4.5 Specifikacija modela konteksta

Da bi se sva potrebna prava pristupa i ograničenja za dati proces mogla definisati neophodno je proširiti model konteksta, predstavljen u odeljku 2.2, sa elementima specifičnim sa stanovišta bezbednosti za slučaj poslovnog procesa *Postupak izbora u nastavna zvanja*. U ovom odeljku predstavljen je deo proširenja modela konteksta neophodan za definisanje prava pristupa i ograničenja čija je analiza izvršena u odeljku 4.4.

Analizom zahteva kontrole pristupa uočeni su sledeći slučajevi (elementi) koji treba da se realizuju posredstvom konteksta:

- potrebno je znati kojoj instanci procesa pripada odgovarajući dokument,
- potrebno je znati članove komisije u okviru svake instance procesa.

Za njihovo predstavljanje kroz model konteksta uvedeni su entiteti opisani u nastavku ovog odeljka.

Korisnici sistema predstavljeni su `Korisnik` klasom (listing 4.1) definisanom kao specijalizacija `HumanResource` i `HumanActor` klasa iz osnovnog on-

tološkog modela prikazanog u odeljku 2.2. Jedinstveni identifikator korisniku moguće je dodeliti posredstvom relacije `imaUID`.

```
@prefix ctx:  <http://informatika.ftn.uns.ac.rs/cobac/context.owl#>.
@prefix iunz: <http://informatika.ftn.uns.ac.rs/cobac/iunz-context.owl#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix xsd:  <http://www.w3.org/2001/XMLSchema#>.
@prefix owl:  <http://www.w3.org/2002/07/owl#>.

iunz:Korisnik a owl:Class;
      rdfs:subClassOf ctx:HumanResource, ctx:HumanActor.
iunz:imaUID a owl:DatatypeProperty;
      rdfs:domain iunz:Korisnik;
      rdfs:range xsd:string.
```

Listing 4.1: Kontekstni model korisnika

Opisani proces predstavljen je klasom `IUNZProces` (listing 4.2) koja je realizovana kao specijalizacija `SoftwareActor` i `SoftwareResource` klasa. Relacija `imaPID` omogućuje dodelu jedinstvenog identifikatora svakoj instanci klase `IUNZProces`.

```
@prefix ctx:  <http://informatika.ftn.uns.ac.rs/cobac/context.owl#>.
@prefix iunz: <http://informatika.ftn.uns.ac.rs/cobac/iunz-context.owl#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix xsd:  <http://www.w3.org/2001/XMLSchema#>.
@prefix owl:  <http://www.w3.org/2002/07/owl#>.

iunz:IUNZProces a owl:Class;
     rdfs:subClassOf ctx:SoftwareActor, ctx:SoftwareResource.
iunz:imaPID a owl:DatatypeProperty;
     rdfs:domain iunz:IUNZProces;
     rdfs:range xsd:string.
```

Listing 4.2: Kontekstni model procesa

Dokumenti u navedenom procesu predstavljeni su klasom `IUNZDokument` (listing 4.3), odnosno njenim specijalizacijama za svaki od tipova dokumenta koji su definisani u tabeli 4.16. Relacijom `imaDID` moguće je instancama dokumenata dodeliti jedinstveni identifikator.

```
@prefix ctx:  <http://informatika.ftn.uns.ac.rs/cobac/context.owl#>.
@prefix iunz: <http://informatika.ftn.uns.ac.rs/cobac/iunz-context.owl#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix xsd:  <http://www.w3.org/2001/XMLSchema#>.
```

```
@prefix owl:  <http://www.w3.org/2002/07/owl#>.

iunz:IUNZDokument a owl:Class;
     rdfs:subClassOf ctx:SoftwareActor, ctx:SoftwareResource.

iunz:PredlVKZaIUNZ a owl:Class; rdfs:subClassOf iunz:IUNZDokument.
iunz:OdlORaspKonk a owl:Class; rdfs:subClassOf iunz:IUNZDokument.
iunz:OdlOOdbijanjuKonk a owl:Class;rdfs:subClassOf iunz:IUNZDokument.
iunz:RaspisanKonkurs a owl:Class; rdfs:subClassOf iunz:IUNZDokument.
iunz:PrijavaNaKonk a owl:Class; rdfs:subClassOf iunz:IUNZDokument.
iunz:PrijNaKonkNast a owl:Class; rdfs:subClassOf iunz:PrijavaNaKonk.
iunz:PrijNaKonkSarad a owl:Class; rdfs:subClassOf iunz:PrijavaNaKonk.
iunz:PredVOZaKomisiju a owl:Class; rdfs:subClassOf iunz:IUNZDokument.
iunz:MisljenjeNNVFOKomisiji a owl:Class; rdfs:subClassOf iunz:IUNZDokument.
iunz:ResOImenKomisije a owl:Class; rdfs:subClassOf iunz:IUNZDokument.
iunz:IzvestKom a owl:Class; rdfs:subClassOf iunz:IUNZDokument.
iunz:IzvestKomZaNast a  owl:Class; rdfs:subClassOf iunz:IzvestKom.
iunz:IzvestKomZaSarad a owl:Class; rdfs:subClassOf iunz:IzvestKom.
iunz:PredlogVOOIzabKand a owl:Class; rdfs:subClassOf iunz:IUNZDokument.
iunz:MisljenjeNNVFOIzboru a owl:Class; rdfs:subClassOf iunz:IUNZDokument.
iunz:MisljenjeNNVUOIzboru a owl:Class; rdfs:subClassOf iunz:IUNZDokument.
iunz:ResenjeOIzboru a owl:Class; rdfs:subClassOf iunz:IUNZDokument.
iunz:UgovorORadu a owl:Class; rdfs:subClassOf iunz:IUNZDokument.

iunz:imaDID a owl:DatatypeProperty;
      rdfs:domain iunz:IUNZDokument;
      rdfs:range xsd:string.
```

Listing 4.3: Kontekstni model dokumenata

Osim navedenih entiteta, za potrebe definisanja kontekstnih uslova u pravima pristupa, uvedena je i klasa `ProcesIUNZCE` (lisitng 4.4) koja predstavlja specijalizaciju kontekstnog izraza za navedeni proces.

```
@prefix ctx:  <http://informatika.ftn.uns.ac.rs/cobac/context.owl#>.
@prefix iunz: <http://informatika.ftn.uns.ac.rs/cobac/iunz-context.owl#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix xsd:  <http://www.w3.org/2001/XMLSchema#>.
@prefix owl:  <http://www.w3.org/2002/07/owl#>.

iunz:ProcesIUNZCE a owl:Class ;
     rdfs:subClassOf ctx:ContextExpression.
```

Listing 4.4: Specijalizacija kontekstnog izraza

Predstavljanje pripadnosti dokumenta odgovarajućoj instanci procesa realizovano je kontekstnim izrazom `DokumentiProcesaIUNZCE` (lisitng 4.5). Za

definisanje ovog izraza uvedene su relacije `dokumentJe` i `pripadaProcesu`. Relacija `dokumentJe` predstavlja specijalizaciju `hasWhoPart` relacije, a njome se dodeljuje dokument ovom kontekstnom izrazu. Relacija `pripadaProcesu` realizovana je kao specijalizacija `hasWhatResourcePart` relacije. Ovom relacijom se uspostavlja veza između kontekstnog izraza i odgovarajućeg procesa.

```
@prefix ctx:  <http://informatika.ftn.uns.ac.rs/cobac/context.owl#>.
@prefix iunz: <http://informatika.ftn.uns.ac.rs/cobac/iunz-context.owl#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix xsd:  <http://www.w3.org/2001/XMLSchema#>.
@prefix owl:  <http://www.w3.org/2002/07/owl#>.

iunz:dokumentJe a owl:ObjectProperty;
     rdfs:subPropertyOf ctx:hasWhoPart.
iunz:pripadaProcesu a owl:ObjectProperty;
      rdfs:subPropertyOf ctx:hasWhatResourcePart.

iunz:DokumentiProcesaIUNZCE a owl:Class ;
      rdfs:subClassOf iunz:ProcesIUNZCE;
      owl:equivalentClass [a owl:Class;
          owl:intersectionOf([a owl:Restriction ;
                              owl:allValuesFrom iunz:IUNZDokument;
                              owl:onProperty iunz:dokumentJe]
                             [a owl:Restriction;
                              owl:onProperty iunz:dokumentJe;
                              owl:someValuesFrom iunz:IUNZDokument]
                             [a owl:Restriction;
                              owl:allValuesFrom iunz:IUNZProces;
                              owl:onProperty iunz:pripadaProcesu]
                             [a owl:Restriction;
                              owl:onProperty iunz:pripadaProcesu;
                              owl:someValuesFrom iunz:IUNZProces])
                          ] .
```

Listing 4.5: Opis pripadnosti dokumenta procesu

Povezanost članova komisije sa odgovarajućom instancom procesa predstavljeno je kontekstnim izrazom `KomisijaProcesaIUNZCE` (lisitng 4.6). Za modelovanje `KomisijaProcesaIUNZCE` kontekstnog izraza definisane su relacije `clanKomisijeJe`, `predesdnikKomisijeJe` i `zaProces`. Relacija `clanKomisijeJe` (realizovana kao specijalizacija `hasWhoPart` relacije) služi za dodelu člana komisije kontektsnom izrazu, dok je njenom specijalizacijom `predsednikKomisijeJe` predstavljena dodela predsednika komisje kontekstnom izrazu. Pomoću relacije `zaProces` moguće je dodeliti odgovarajući proces ovom kontekstnom izrazu.

```
@prefix ctx:  <http://informatika.ftn.uns.ac.rs/cobac/context.owl#>.
@prefix iunz: <http://informatika.ftn.uns.ac.rs/cobac/iunz-context.owl#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix xsd:  <http://www.w3.org/2001/XMLSchema#>.
@prefix owl:  <http://www.w3.org/2002/07/owl#>.

iunz:clanKomisijeJe a owl:ObjectProperty;
     rdfs:subPropertyOf ctx:hasWhoPart.
iunz:predesdnikKomisijeJe a owl:ObjectProperty;
     rdfs:subPropertyOf iunz:clanKomisijeJe;
iunz:zaProces a owl:ObjectProperty;
     rdfs:subPropertyOf ctx:hasWhatResourcePart.

iunz:KomisijaProcesaIUNZCE a owl:Class;
      rdfs:subClassOf iunz:ProcesIUNZCE;
      owl:equivalentClass[a owl:Class;
          owl:intersectionOf([a owl:Restriction;
                              owl:allValuesFrom iunz:Korisnik;
                              owl:onProperty iunz:clanKomisijeJe]
                             [a owl:Restriction;
                              owl:onProperty iunz:clanKomisijeJe;
                              owl:someValuesFrom iunz:Korisnik]
                             [a owl:Restriction;
                              owl:cardinality 1;
                              owl:onProperty iunz:predesdnikKomisijeJe]
                             [a owl:Restriction;
                              owl:allValuesFrom iunz:IUNZProces;
                              owl:onProperty iunz:zaProces]
                             [a owl:Restriction;
                              owl:onProperty iunz:zaProces;
                              owl:someValuesFrom iunz:IUNZProces])
                         ].
```

Listing 4.6: Opis pripadnosti članova komisije procesu

4.6 Specifikacija prava pristupa i ograničenja

U ovom odeljku data su prava pristupa i ograničenja kojim su realizovani zahtevi prethodno analizirani u odeljku 4.4.

Primer prava pristupa kojim se omogućuje da dekan donese odluku o raspisivanju konkursa (zahtev 4.1) predstavljen je listingom 4.7.

```
/*Dodela aktivnosti ulozi*/
CAD_1 - definicije aktivnosti postupka izvora u nastavna zvanja
cad ∈ CAD_1 ∧ cad = Odluka o raspisivanju konkursa
r ∈ R ∧ r.rn =Dekan
cc ∈ CC ∧ cc = σ

rcdActivityAssign(r, cad, cc)

/*Dodela privilegija aktivnosti*/
cat_1 ∈ Cat ∧ cat_1 = Predlog veća katedre za izbor u zvanje
cat_2 ∈ Cat ∧ cat_2 = Odluka o raspisivanju konkursa

op_r ∈ Op ∧ op_r = čitaj dokument
op_c ∈ Op ∧ op_c = kreiraj dokument
op_u ∈ Op ∧ op_u = ažuriraj dokument
op_d ∈ Op ∧ op_d = briši dokument
op_s ∈ Op ∧ op_s = potpiši dokument

p_1r ∈ CP ∧ p_1r = (cat_1, op_r)
p_2r ∈ CP ∧ p_2r = (cat_2, op_r)
p_2c ∈ CP ∧ p_2c = (cat_2, op_c)
p_2u ∈ CP ∧ p_2u = (cat_2, op_u)
p_2d ∈ CP ∧ p_2d = (cat_2, op_d)
p_2s ∈ CP ∧ p_2s = (cat_2, op_s)

cadPermissionAssign(cad, p_1r, cc)
cadPermissionAssign(cad, p_2r, cc)
cadPermissionAssign(cad, p_2c, cc)
cadPermissionAssign(cad, p_2u, cc)
cadPermissionAssign(cad, p_2d, cc)
cadPermissionAssign(cad, p_2s, cc)
```

Listing 4.7: Pravo pristupa za aktivnost *Odluka o raspisivanju konkursa*

Pravo pristupa za zahtev 4.2 može se definisati na dva načina:

Prva varijanta bi bila da se pravo pristupa definiše za instancu aktivnosti, što bi značilo da ono treba da se dinamički kreira, pre izvršenja aktivnosti *Izveštaj komisije*, odnosno po okončanju aktivnosti koje joj prethode (v. sliku 4.2).

Drugi način za definisanje prava pristupa za prethodni slučaj je da se ono ne kreira dinamički. Kod ovog načina pravo pristupa se definiše za definiciju aktivnosti, a kontekstnim uslovom će se dozvoliti da samo članovi komisije

izabrani u okviru konkretne instance procesa imaju pravo da izvrše instancu aktivnosti *Izveštaj komisije* u okviru te instance procesa.

Slučaj kada se komisiji dozovoljava da izvrši određenu instancu aktivnosti *Izveštaj komisije* predstavljen je pravom pristupa prikazanim listingom 4.8.

Uloga *Član komisije izbora u zvanje* biće dodeljena svim korisnicima koji zadovoljavaju kontekstni uslov cc_r, tj. svi korisnici koji su imenovani za člana komisije u bilo kojoj instanci ovog poslovnog procesa. Kontekstni uslov sadrži upit koji proverava da li u kontekstnim podacima postoji proces za koga je tekući korisnik imenovan za člana komisije. Identifikator tekućeg korisnika preuzima se posredstvom konteksnte funkcije `$$currentUserID()$$`.

Ulozi *Član komisije izbora u zvanje* dodeljeno je pravo da izvrši instancu aktivnosti *Izveštaj komisije*, pri čemu je dodat kontekstni uslov cc_{ra} koji zahteva da korisnik sa ulogom *Član komisije izbora u zvanje* mora biti član komisije baš u okviru te instance procesa da bi mogao da izvrši instancu te aktivnosti. Upit od koga se sastoji cc_{ra} sadrži kontekstne funkcije `$$currentUserID()$$` i `$$currentProcessID()$$` koje vraćaju vrednost identifikatora korisnika, odnosno vrednost identifikatora instance tekućeg procesa.

Privilegije dodeljene instanci aktivnosti *Izveštaj komisije* definisane su za potrebne dokumente kreirane u okviru instance procesa kome ta instanca aktivnosti pripada.

```
/*Dinamička dodela uloga korisniku*/
CAI_1^k - instance aktivnosti k-te instance postupka izvora u nastavna zvanja
cai ∈ CAI_1^k ∧ cai = Izveštaj komisije
r ∈ R ∧ r.rn =Član komisije izbora u zvanje
cc_r ∈ CC
cc_r = QUERY {
        PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
        PREFIX iunz: <http://informatika.ftn.uns.ac.rs/cobac/iunz-context.owl#>.
        ASK ?X
         WHERE { ?X rdf:type iunz:KomisijaProcesaIUNZCE.
                 ?X izun:clanKomisijeJe ?Y.
                 ?Y izun:imaUID $$currentUserID()$$ }
        }
roleCondAssign(r, cc_r)

/*Dodela aktivnosti ulozi*/
cc_ra ∈ CC
cc_ra = QUERY {
         PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
         PREFIX iunz: <http://informatika.ftn.uns.ac.rs/cobac/iunz-context.owl#>.
         ASK ?X
```

```
          WHERE { ?X rdf:type iunz:KomisijaProcesaIUNZCE.
                  ?X izun:clanKomisijeJe ?Y.
                  ?Y izun:imaUID $$currentUserID()$$
                  ?X izun:zaProces ?Z.
                  ?Z izun:imaPID $$currentProcessID()$$ }
        }
rciActivityAssign(r, cai, cc_ra)

/*Dodela privilegija aktivnosti*/
doc_1 ∈ Res ∧ doc_1 = Prijava na koknurs za nastavnika/saradnika
                      (za k-tu instancu procesa)
doc_2 ∈ Res ∧ doc_2 = Raspisan konkurs
                      (za k-tu instancu procesa)
doc_3 ∈ Res ∧ doc_3 = Rešenje o imenovanju komisija
                      (za k-tu instancu procesa)
doc_4 ∈ Res ∧ doc_4 = Izveštaj komisije za nastavnika/saradnika
                      (za k-tu instancu procesa)

op_r ∈ Op ∧ op_r = čitaj dokument
op_c ∈ Op ∧ op_c = kreiraj dokument
op_u ∈ Op ∧ op_u = ažuriraj dokument
op_d ∈ Op ∧ op_d = briši dokument
op_s ∈ Op ∧ op_s = potpiši dokument

p_1r ∈ RP ∧ p_1r = (doc_1, op_r)
p_2r ∈ RP ∧ p_2r = (doc_2, op_r)
p_3r ∈ RP ∧ p_3r = (doc_3, op_r)
p_4r ∈ RP ∧ p_4r = (doc_4, op_r)
p_4c ∈ RP ∧ p_4c = (doc_4, op_c)
p_4u ∈ RP ∧ p_4u = (doc_4, op_u)
p_4d ∈ RP ∧ p_4d = (doc_4, op_d)
p_4s ∈ RP ∧ p_4s = (doc_4, op_s)

cc_p ∈ CC ∧ cc = σ

caiPermissionAssign(cai, p_1r, cc_p)
caiPermissionAssign(cai, p_2r, cc_p)
caiPermissionAssign(cai, p_3r, cc_p)
caiPermissionAssign(cai, p_4r, cc_p)
caiPermissionAssign(cai, p_4c, cc_p)
caiPermissionAssign(cai, p_4u, cc_p)
caiPermissionAssign(cai, p_4d, cc_p)
caiPermissionAssign(cai, p_4s, cc_p)
```

Listing 4.8: Pravo pristupa za instancu aktivnosti *Izveštaj komisije*

Drugi način za definisanje prava pristupa za prethodni slučaj je da se ono ne kreira dinamički. Kod ovog načina pravo pristupa se definiše za definiciju aktivnosti, a kontekstnim uslovom cc_{ra} se dozvoljava da samo članovi komisije

izabrani u okviru konkretne instance procesa imaju pravo da izvrše instancu aktivnosti *Izveštaj komisije* u okviru te instance procesa.

Listingom 4.8 prikazan je primer prava pristupa specificiranog za definiciju aktivnosti *Izveštaj komisije*. Uloga *Član komisije izbora u zvanje* biće dodeljena korisnicima ukoliko je kontekstni uslov cc_r zadovoljen. Da bi korisnik sa ovom ulogom mogao da izvrši navedenu aktivnost potrebno je i da kontekstni uslov cc_{ra} bude zadovoljen. Ovim uslovom praktično se zahteva da korisnik mora biti član komisije baš u okviru instance procesa koga trenutno izvršava.

Aktivnosti su dodeljene privilegije za izvršavanje odgovarajućih operacija nad kategorijama dokumenata, dok je kontekstnim uslovom cc_p privilegija sužena samo na dokumenta koja pripadaju konkretnoj instanci procesa. Kada se proverava da li je u okviru aktivnosti dozvoljeno izvršenje zahtevane operacije nad određenim dokumentom ovim kontekstnim uslovom će se proveriti da li taj dokument pripada tekućoj instanci procesa. Idenfitikator dokumenta kome se želi pristupiti predstavlja vrednost funkcije `$$currentDocumentID()$$`.

```
/*Dinamička dodela uloga korisniku*/
CAD_1 - definicije aktivnosti postupka izvora u nastavna zvanja
cad ∈ CAD_1 ∧ cad = Izveštaj komisije
r ∈ R ∧ r.rn =Član komisije izbora u zvanje
cc_r ∈ CC
cc_r = QUERY {
        PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
        PREFIX iunz: <http://informatika.ftn.uns.ac.rs/cobac/iunz-context.owl#>.
        ASK ?X
         WHERE { ?X rdf:type iunz:KomisijaProcesaIUNZCE.
                 ?X izun:clanKomisijeJe ?Y.
                 ?Y izun:imaUID $$currentUserID()$$ }
        }
roleCondAssign(r, cc_r)

/*Dodela aktivnosti ulozi*/
cc_ra ∈ CC
cc_ra = QUERY {
         PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
         PREFIX iunz: <http://informatika.ftn.uns.ac.rs/cobac/iunz-context.owl#>.
         ASK ?X
          WHERE { ?X rdf:type iunz:KomisijaProcesaIUNZCE.
                  ?X izun:clanKomisijeJe ?Y.
                  ?Y izun:imaUID $$currentUserID()$$.
                  ?X izun:zaProces ?Z.
                  ?Z izun:imaPID $$currentProcessID()$$ }
        }
rcaActivityAssign(r, cad, cc_ra)
```

```
/*Dodela privilegija aktivnosti*/
cat1 ∈ Cat ∧ cat1 = Prijava na koknurs za nastavnika/saradnika
cat2 ∈ Cat ∧ cat2 = Raspisan konkurs
cat3 ∈ Cat ∧ cat3 = Rešenje o imenovanju komisij
cat4 ∈ Cat ∧ cat4 = Izveštaj komisije za nastavnika/saradnika

opr ∈ Op ∧ opr = čitaj dokument
opc ∈ Op ∧ opc = kreiraj dokument
opu ∈ Op ∧ opu = ažuriraj dokument
opd ∈ Op ∧ opd = briši dokument
ops ∈ Op ∧ ops = potpiši dokument

p1r ∈ CP ∧ p1r = (cat1, opr)
p2r ∈ CP ∧ p2r = (cat2, opr)
p3r ∈ CP ∧ p3r = (cat3, opr)
p4r ∈ CP ∧ p4r = (cat4, opr)
p4c ∈ CP ∧ p4c = (cat4, opc)
p4u ∈ CP ∧ p4u = (cat4, opu)
p4d ∈ CP ∧ p4d = (cat4, opd)
p4s ∈ CP ∧ p4s = (cat4, ops)

ccp ∈ CC
ccp = QUERY {
        PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
        PREFIX iunz: <http://informatika.ftn.uns.ac.rs/cobac/iunz-context.owl#>.
        ASK ?X
         WHERE { ?X rdf:type iunz:DokumentiProcesaIUNZCE.
                 ?X izun:pripadaProcesu ?Y
                 ?Y iyun.imadPID $$currentProcessID()$$
                 ?Z izun:dokumentJe ?Z.
                 ?Z izun:DID $$currentDocumentID()$$ }
        }

cadPermissionAssign(cad, p1r, ccp)
cadPermissionAssign(cad, p2r, ccp)
cadPermissionAssign(cad, p3r, ccp)
cadPermissionAssign(cad, p4r, ccp)
cadPermissionAssign(cad, p4c, ccp)
cadPermissionAssign(cad, p4u, ccp)
cadPermissionAssign(cad, p4d, ccp)
cadPermissionAssign(cad, p4s, ccp)
```

Listing 4.9: Pravo pristupa za definiciju aktivnosti *Izveštaj komisije*

Jedno moguće rešenje za sprečavanje sukoba interesa navedenog u zahtevu 4.3 je da se aktivnosti *Izveštaj komisije* i *Potpisivanje ugovora o radu* proglase međusobno konfliktnim čime se sprečava da ih izvršava isti korisnik, tj. da se definiše odgovarajuće ADSoD ograničenje. Primer ADSoD ograničenja za ovaj

slučaj prikazan je listingom 4.10. Kako ovo ograničenje treba da važi za sve instance procesa izbora u zvanje onda je ono definisano na nivou definicija aktivnosti, a pošto ograničenje treba da važi na nivou instance procesa (korisnik može biti odabrani kandidat i član komsije u različitim instancama procesa) onda je reč o internom tipu ADSoD ograničenja za definicije aktivnosti. Za ovaj konkretan slučaju nije potrebno nikakvo dodatno kontekstno ograničenje da bi navedene definicije aktivnosti bile konfliktne.

```
/*Definisanje ADSoD ograničenja */
CAD1 - definicije aktivnosti postupka izvora u nastavna zvanja
cad1 ∈ CAD1 ∧ cad1 = Izveštaj komisije
cad2 ∈ CAD1 ∧ cad2 = Potpisivanje ugovora o radu
cc ∈ CC ∧ cc = σ
(cad1, cad2, cc) ∈ DCADI
```

Listing 4.10: Primer ADSoD ograničenja

U prethodno navedenim pravima pristupa nije razmatrano vreme u koje se izvršava određena aktivnost, kao ni lokacija sa koje se izvršava. Ukoliko bi se analizom rizika sa stanovišta bezbednosti utvrdilo da su i ovi faktori bitni onda bi se i oni mogli definisati posredstvom odgovarajućeg kontekstnog uslova.

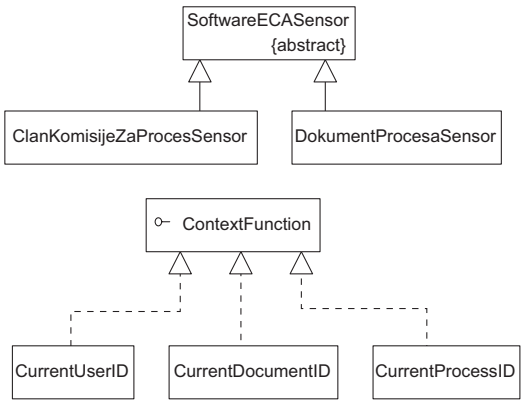
4.7 Specifikacija kontekstnih modula

Da bi se mogla definisati prava pristupa navedena u odeljku 4.6 i na osnovu njih sprovoditi kontrola pristupa neophodno je obezbediti module za akviziciju kontekstnih informacija predstavljenih u odeljku 4.5 kao i module za realizovanje kontekstnih funkcija korišćenih za definisanje kontekstnih uslova u tim pravima pristupa.

Klasa `ClanKomisijeZaProcesSensor` (slika 4.5) predstavlja ECA senzor namenjen za formiranje konteksnog izraza kojim je predstavljeno članstvo korisnika u komisiji za izbor u zvanje za odgovarajuću instancu proces. Na sličan način klasa `DokumentProcesaSensor` (slika 4.5) formira kontekstne izraze kojima je opisana pripadnost dokumenata konkretnoj instanci procesa.

Klasama `CurrentUserID`, `CurrentDocumentID` i `CurrentProcessID` (slika 4.5) modelovane su funkcije potrebne za definisanje kontekstnih uslova. Funkcija predstavljena klasom `CurrentUserID` određuje identifikator tekućeg korisnika. Klasa `CurrentDocumentID` omogućuje da se odredi identifikator tekućeg

dokumenta (dokumenta kome se pristupa), a funkcijom koja je predstavljena klasom `CurrentProcessID` moguće je odrediti identifikator instance procesa za koji se trenutno sprovodi kontrola pristupa.



Slika 4.5: Dijagram klasa kontekstnih modula

Poglavlje 5

Verifikacija sistema na postupku obrade bibliografskih zapisa

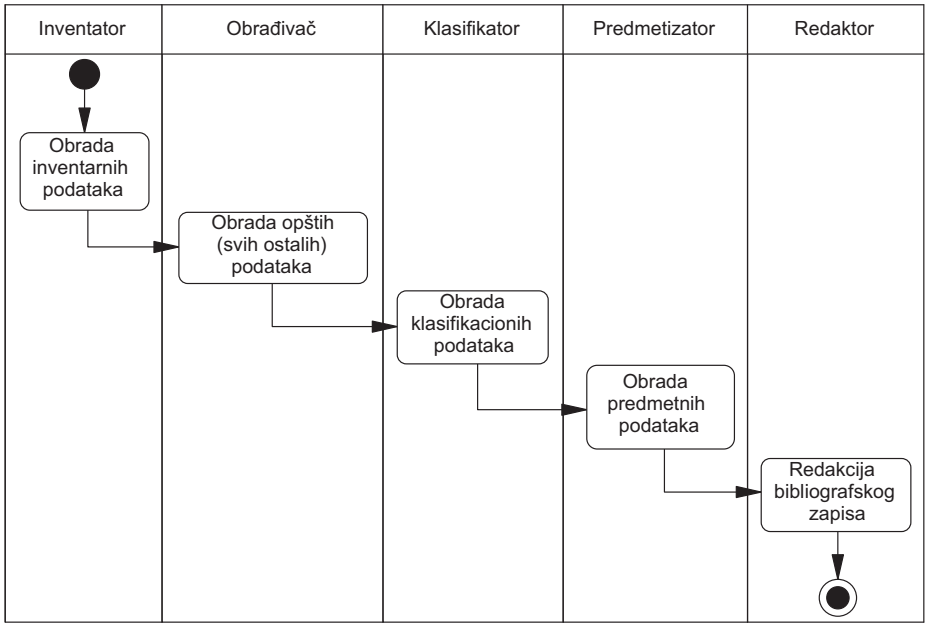
U drugoj studiji slučaja prikazana je primena COBAC modela za kontrolu pristupa poslovnog procesa u bibliotekama u okviru koga se vrši obrada bibliografskih zapisa. Odabran je realni proces obrade bibliografskih zapisa realizovan korišćenjem BISIS [127] bibliotečkog informacionog sistema u Biblioteci grada Beograda [24].

Za razliku od procesa predstavljenog u prethodnom poglavlju gde je težište kontrole pristupa bilo na nivou procesa, ovaj proces karakteriše težište kontrole pristupa na nivou resursa (dokumenata), tj. bibliografskih zapisa, pošto je potrebno sprovoditi kontrolu pristupa nad delovima bibliografskih zapisa. Cilj ove studije slučaja je da pokaže da se COBAC prototip može iskoristiti u ovakvim i sličnim slučajevima, gde se COBAC koristi za kontrolu pristupa na nivou procesa, a neki drugi sistem, u ovom slučaju XXACF (v. odeljak 1.4.9), se koristi za kontrolu pristupa na nivou bibliografskih zapisa. Rezultati ovog istraživanja objavljeni su u radu [234].

5.1 Analiza procesa

Na slici 5.1 prikazan je dijagram aktivnosti procesa *Obrada bibliografskih zapisa*. Proces kreiranja zapisa započinje *Inventator* koji je zadužen za unos inventarnih podataka, potom *Obrađivač* unosi opšte podatke vezane za publika-

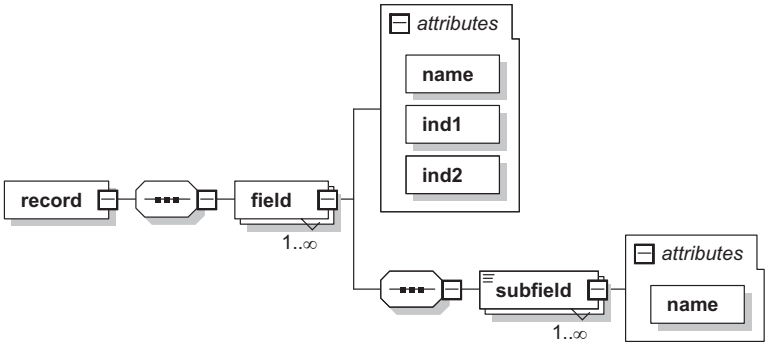
ciju koja se obrađuje. *Klasifikator* je zadužen za unos klasifikacionih podataka, dok *Predmetizator* vrši obrada podataka koji se odnose na predmet (temu) publikacije. Funkcija *Redaktora* je da pregleda podatke unete u prethodnim aktivnostima i izvrši ispravke ukoliko su one potrebne. Prezentovani proces obrade bibliografskih zapisa identičan je za sve tipove publikacija čiji zapisi se unose.



Slika 5.1: Dijagram aktivnosti za poslovni proces Obrada bibliografskih zapisa

5.2 Specifikacija resursa i operacija nad resursima

U ovom slučaju resursi za koje se kontrola pristupa sprovodi su bibliografski zapisi. Za potrebe sprovođenja kontrole pristupa ovi zapisi su predstavljeni kao XML dokumenti definisani u skladu sa XML šemom razvijenom za MARC bazirane standarde (slika 5.2).



Slika 5.2: XML šema za predstavljanje MARC baziranih bibliografskih zapisa

Operacije koje se izvršavaju nad bibliografskim zapisima, a za koje se sprovodi kontrola pristupa su:

- čitanje zapisa,
- pretraga zapisa,
- kreiranje zapisa,
- brisanje zapisa, i
- ažuriranje zapisa.

5.3 Specifikacija korisničkih uloga

Analizom svih korisnika koji učestvuju u procesu formiranja bibliografskih zapisa formirana je hijerarhija korisničkih uloga prikazana na slici 5.3. Na prvom nivou definisana je korenska uloga *BISIS Korisnik*, dok je na sledećem nivou definisana uloga *Katalogizator* kojom su predstavljeni bibliotekari koji učestvuju u postupku obrade bibliografskih zapisa.

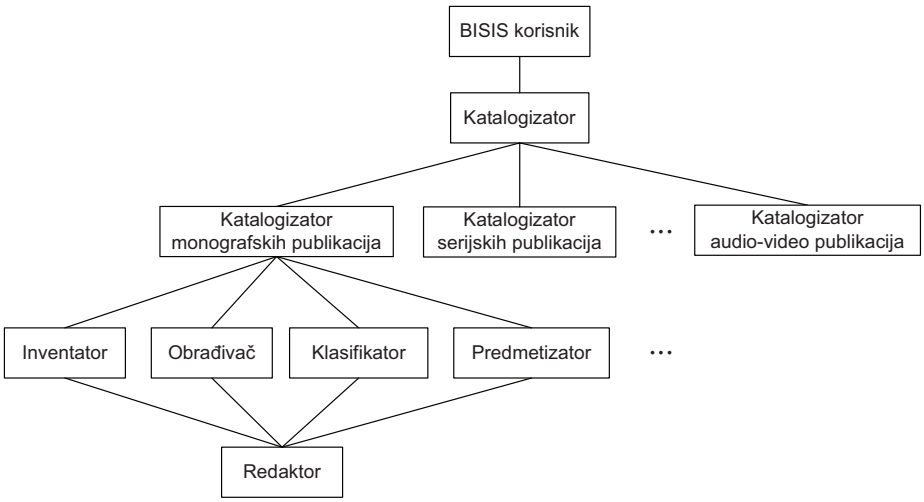
Pošto u predstavljenoj studiji slučaja različiti korisnici vrše katalogizaciju različitih tipova publikacija, definisane su specijalizacije *Katalogizator* uloge: *Katalogizator monografskih publikacija*, *Katalogizator serijskih publikacija*, *Katalogizator audio-video publikacija*, itd. Uloge na ovom nivou su zadužene za katalogizaciju publikacija odgovarajućeg tipa.

Svaka od prethodno navedenih specijalizacija *Katalogizator* uloge kao podređene uloge ima uloge navedene u opisu procesa na slici 5.1: *Inventator*, *Obrađivač*, *Klasifikator* i *Predmetizator*. Na slici 5.3 prikazane su ove uloge za

ulogu *Katalogizator monografskih publikacija*, medutim ista hijerarhija postoji i za ostale specijalizacije *Katalogizator* uloge.

Na najvišem hijerarhijskom nivou definisana je uloga *Redaktor*. Korisnik kome je dodeljena ova uloga ima pravo da obrađuje sve delove bibliografskog zapisa odgovarajućeg tipa publikacije i da ispravi eventualne greške. Stoga uloga *Redaktor* nasleđuje uloge na prethodnom nivou i time poseduje privilegije koje su dodeljene ulogama na tom nivou, tako da nema potrebe za eksplicitnim definisanjem privilegija za *Redaktor* ulogu.

Dalja analiza uloga utvrdila je da za postojeće uloge ne postoji potreba za definisanjem uslova omogućavanja/onemogućavanja (v. odeljak 2.5), odnosno sve uloge su uvek omogućene, tj. za sve uloge važi: $sc = \sigma, sct = ec$.



Slika 5.3: Hijerarhija korisničkih uloga

5.4 Specifikacija prava pristupa

U ovom odeljku predstavljena su prava pristupa za ulogu *Katalogizator monografskih publikacija*, odnosno za njene pod uloge. Prava pristupa za ostale poduloge *Katalogizator* uloge su slična prezentovanim, izuzev što su definisana za drugi tip publikacije.

5.4.1 Specifikacija prava pristupa COBAC modela

Kao što je rečeno COBAC model u ovom slučaju iskorišćen je za sprovođenje kontrole pristupa na nivou procesa, odnosno aktivnosti procesa. Primeri prava pristupa kojima se dozvoljava ulogama *Inventator*, *Obrađivač*, *Klasifikator*, *Predmetizator* i *Redaktor* da izvrše definicije aktivnosti navedene na slici 5.1 predstavljeni su listingom 5.1.

```
/*Dodela aktivnosti ulozi*/
CAD_1 - definicije aktivnosti obrade bibliografskih zapisa
cad_1 ∈ CAD_1 ∧ cad_1 = Obrada inventarnih podataka
cad_2 ∈ CAD_1 ∧ cad_2 = Obrada opštih podataka
cad_3 ∈ CAD_1 ∧ cad_3 = Obrada klasifikacionih podataka
cad_4 ∈ CAD_1 ∧ cad_4 = Obrada predmetnih podataka
cad_5 ∈ CAD_1 ∧ cad_5 = Obrada bibliografskog zapisa

r_1 ∈ R ∧ r_1.rn =Inventator
r_2 ∈ R ∧ r_2.rn =Obrađivač
r_3 ∈ R ∧ r_3.rn =Klasifikator
r_4 ∈ R ∧ r_4.rn =Predmetizator
r_5 ∈ R ∧ r_5.rn =Redaktor

cc ∈ CC ∧ cc = σ

rcdActivityAssign(r_1, cad_1, cc)
rcdActivityAssign(r_2, cad_2, cc)
rcdActivityAssign(r_3, cad_3, cc)
rcdActivityAssign(r_4, cad_4, cc)
rcdActivityAssign(r_5, cad_5, cc)
```

Listing 5.1: Prava pristupa za izvršenje aktivnosti procesa obrade bibliografskih zapisa

5.4.2 Specifikacija prava pristupa XXACF modela

Sva prava pristupa u ovom slučaju definišu se na nivou čitavog zapisa, odnosno na nivou polja ili potpolja i ista su za različite instance dokumenata. Ovo znači da je prava pristupa potrebno definisati na nivou šeme dokumenta, odnosno na nivou odgovarajućeg elementa jer će prava pristupa definisana na nivou šeme dokumenta biti primenjena na sve instance te šeme dokumenta [228, 229, 231, 232].

Svi resursi u definisanim privilegijama identifikovani su XPath izrazima kojima su selektovana odgovarajuća polja/potpolja u XML formi MARC bibliografskih zapisa.

Kako poduloge *Katalogizator* uloge imaju pravo da čitaju i pretražuju sve zapise, privilegije za čitanje i pretragu za čitav zapis dodeljene su *Katalogizator* ulozi (tabela 5.1). Sve njezine poduloge će naslediti ovu privilegiju.

Uloga	Katalogizator
Operacije	čitanje, pretraga
Propagacija	*smer*: niz hijerarhiju, *nivo*: neograničen
Tip	dozvola pristupa
Resursi	/record

Tabela 5.1: Privilegija za čitanje za ulogu *Katalogizator*

Sve naredno definisane privilegije dodeljene ulozi *Katalogizator monografskih publikacija* i njezinim podulogama primenjuju se na bibliografske zapise tipa monografske publikacije. Tip publikacije zapisan je kao vrednost potpolja *001c*. XPath izraz koji proverava da li vrednost potpolja *001c* sadrži vrednost *m* definisan je kao *MON* alias u tabeli 5.2. Svaki resurs u privilegijama dodeljenih podulogama uloge *Katalogizator monografskih publikacija* sadrži *MON* izraz kao XPath predikat. Na ovaj način ovim privilegijama će biti obuhvaćeni jedino zapisi koji predstavljaju monografske publikacije.

Inventator može da kreira, briše i ažurira polja *001* i *996* za monografski tip publikacije. Pored toga, pošto ova uloga treba da kreira i novi zapis potrebno joj je za tu operaciju dodeliti odgovarajuću privilegiju. Privilegija u tabeli 5.2 dozvoljava kreiranje elementa */record*, ali pošto je nivo propagacije 0 ona ne dozvoljava kreiranje polja/potpolja. Privilegije za polja *001* i *996* data su u tabeli 5.3.

Uloga	Inventator
Operacije	kreiranje
Propagacija	*smer*: niz hijerarhiju, *nivo*: 0
Tip	dozvola pristupa
Resursi	MON=exists(field[@name='001']/subfield[name='c' and text()='m']) /record[MON]

Tabela 5.2: Privilegija za kreiranje zapisa za ulogu *Inventator*

Uloga	Inventator
Operacije	kreiranje, brisanje, ažuriranje
Propagacija	*smer*: niz hijerarhiju, *nivo*: neograničen
Tip	dozvola pristupa
Resursi	/record[MON]/field[@name='001' or @name='996']

Tabela 5.3: Privilegije za polja *001* i *996* za ulogu *Inventator*

Primer 5.1 Inventator *želi da kreira novi zapis sa poljima/potpoljima navedenim u listingu 5.2. Privilegija u tabeli 5.2 će dozvoliti kreiranje novog zapisa, dok će privilegije u tabeli 5.3 dozvoliti kreiranje samo polja* 001 *i* 996*, dok potpolje* 102a *neće biti kreirano jer ova uloga nema privilegiju potrebnu za njegovo kreiranje. Rezultat kreiranja zapisa nakon sprovođenja kontrole pristupa dat je u listingu 5.3.*

```
001 ## $7ba$ai$ba$cm$d0
102 ## $ausa
996 0# $dM-20111$f000020111
```

Listing 5.2: Polja/potpolja koja *Inventator* želi da kreira

```
<record>
  <field name="001" ind1=" " ind2=" ">
    <subfield name="7">ba</subfield>
    <subfield name="a">i</subfield>
    <subfield name="b">a</subfield>
    <subfield name="c">m</subfield>
    <subfield name="d">0</subfield>
  </field>
```

```
  <field name="996" ind1="0" ind2=" ">
    <subfield name="d">M-20111</subfield>
    <subfield name="f">000020111</subfield>
  </field>
</record>
```

Listing 5.3: Zapis koji je kreiran od strane *Inventator*-a

Uloga *Obrađivač* može dodavati, brisati i ažurirati sva polja/potpolja za monografski tip publikacije osim polja/potpolja za koja su zadužene uloge *Inventator*, *Klasifikator* i *Predmetizator*. Ovaj zahtev može da se realizuje sa dve vrste privilegija, jednom koja odobrava pristup i drugom koja ga zabranjuje. Privilegije u tabeli 5.4 dozvoljavaju ulozi *Obrađivač* da kreira, briše i ažurira sva polja i potpolja zapisa, dok privilegije u tabeli 5.5 zabranjuju ovoj ulozi da kreira, briše i ažurira sva polja i potpolja za koje je zadužena jedna od preostale tri poduloge *Katalogizator* uloge.

Na polja/potpolja *001*, *996*, *675a* i *600-610* će biti primenjena oba tipa privilegija, međutim, kako privilegije u tabeli 5.5 imaju specifičniji (detaljniji) resurs, ovim poljima/potpoljima će biti zabranjen pristup [228, 229, 231].

Uloga	Obrađivač
Operacije	kreiranje, brisanje, ažuriranje
Propagacija	*smer*: niz hijerarhiju, *nivo*: neograničen
Tip	dozvola pristupa
Resursi	/record[MON]/field

Tabela 5.4: Privilegije koje dozvoljavaju obradu za ulogu *Obrađivač*

Uloga	Obrađivač
Operacije	kreiranje, brisanje, ažuriranje
Propagacija	*smer*: niz hijerarhiju, *nivo*: neograničen
Tip	zabrana pristupa
Resursi	/record[MON]/field[@name='001' or @name='996'] /record[MON]/field[@name='675']/subfield[@name='a'] /record[MON]/field[@name>='600' and @name<='610']

Tabela 5.5: Privilegije koje zabranjuju obradu za ulogu *Obrađivač*

Primer 5.2 *Korisnik sa ulogom* Obrađivač *želi da modifikuje zapise u listingu 5.3 sa poljima/potpoljima navedenim u listingu 5.4. Korisnik želi da kreira određena polja/potpolja, ali takode želi i da ažurira potpolje* 996f *i obriše potpolje* 0017*. Privilegije u tabeli 5.4 će dozvoliti kreiranje, ažuriranje i brisanje čitavog zapisa, međutim privilegije u tabeli 5.5 zabranjuju navedene operacije za* 001*,* 996*,* 675a *i* 600-610 *polja/potpolja. Stoga će korsinik biti u mogućnosti samo da doda navedena polja/potpolja, ali neće moći izvršiti ažuriranje potpolja* 996f *i brisanje* 0017*. Novi zapis nakon izvršenih modifikacija i sprovođenja kontrole pristupa dat je u listingu 5.5.*

```
ADD FIELDS/SUBFIELDS
010 ## $a0-8053-7133-8
100 ## $d1996$hscr
101 0# $aeng
102 ## $ausa
105 ## $aa
200 0# $aConcepts of Programming Languages$fRobert W. Sebesta
210 ## $aReading [etc.]$cAddison-Wesley Publishing Company$d1996
215 ## $axv, 634 str.$cilustr.$d24 cm
700 #1 $4070$aSebesta$bRobert W.
992 ## $bcrsale9701-old

UPDATE SUBFIELD
996 ## $f0001701982
DELETE SUBFIELD
001 ## $7ba
```

Listing 5.4: Polja/potpolja koja *Obrađivač* želi da obrađuje

```
<record>
  <field name="001" ind1=" " ind2=" ">
    <subfield name="7">ba</subfield>
    <subfield name="a">i</subfield>
    <subfield name="b">a</subfield>
    <subfield name="c">m</subfield>
    <subfield name="d">0</subfield>
  </field>
  <field name="010" ind1=" " ind2=" ">
    <subfield name="a">0-8053-7133-8</subfield>
  </field>
  <field name="100" ind1=" " ind2=" ">
    <subfield name="b">d</subfield>
    <subfield name="c">1996</subfield>
    <subfield name="h">scr</subfield>
  </field>
```

```
  <field name="101" ind1="0" ind2=" ">
    <subfield name="a">eng</subfield>
  </field>
  <field name="102" ind1=" " ind2=" ">
    <subfield name="a">usa</subfield>
  </field>
  <field name="105" ind1=" " ind2=" ">
    <subfield name="a">a</subfield>
  </field>
  <field name="200" ind1="0" ind2=" ">
    <subfield name="a">Concepts of Programming Languages</subfield>
    <subfield name="f">Robert W. Sebesta</subfield>
  </field>
  <field name="210" ind1=" " ind2=" ">
    <subfield name="a">Reading [etc.]</subfield>
    <subfield name="c">Addison-Wesley Publishing Company</subfield>
    <subfield name="d">1996</subfield>
  </field>
  <field name="215" ind1=" " ind2=" ">
    <subfield name="a">xv, 634 p.</subfield>
    <subfield name="c">ilustr.</subfield>
    <subfield name="d">24 cm</subfield>
  </field>
  <field name="700" ind1=" " ind2="1">
    <subfield name="a">Sebesta</subfield>
    <subfield name="b">Robert W.</subfield>
    <subfield name="4">070</subfield>
  </field>
  <field name="992" ind1=" " ind2=" ">
    <subfield name="b">crsale9701-old</subfield>
  </field>
  <field name="996" ind1="0" ind2=" ">
    <subfield name="d">M-20111</subfield>
    <subfield name="f">000020111</subfield>
  </field>
</record>
```

Listing 5.5: Zapis nakon obrade *Obrađivač*-a

Uloga *Klasifikator* može da kreira, briše i ažurira potpolje *675a*. Da bi mogla da kreira navedeno potpolje *Klasifikator* uloga treba da ima privilegiju i za kreiranje polja *675* kao i privilegije za navedene operacije za potpolje *675a*. Ove privilegije date su u tabelama 5.6 i 5.7.

Uloga	Klasifikator
Operacije	kreiranje
Propagacija	*smer*: niz hijerarhiju, *nivo*: 0
Tip	dozvola pristupa
Resursi	/record[MON]/field[@name='675']

Tabela 5.6: Privilegija za kreiranje polja *675* za ulogu *Klasifikator*

Uloga	Klasifiaktor
Operacije	kreiranje, brisanje, ažuriranje
Propagacija	*smer*: niz hijerarhiju, *nivo*: neograničen
Tip	dozvola pristupa
Resursi	/record[MON]/field[@name='675']/subfield[@name='a']

Tabela 5.7: Privilegije za obradu potpolja za ulogu *Klasifikator*

Primer 5.3 *Ako uloga* Klasifikator *doda univerzalnu decimalnu klasifikaciju (UDK) sa vrednošću UDK=514.7 u zapis kreiran od strane* Obrađivača *formira se zapis dat u listingu 5.6. Kako uloga* Klasifikator *ima pravo da kreira potpolje* 675a *(UDK), kontrola pristupa će dozvoliti kreiranje navedenog potpolja.*

```
<record>
  <field name="001" ind1=" " ind2=" ">
    <subfield name="7">ba</subfield>
    <subfield name="a">i</subfield>
    <subfield name="b">a</subfield>
    <subfield name="c">m</subfield>
    <subfield name="d">0</subfield>
  </field>
  <field name="010" ind1=" " ind2=" ">
    <subfield name="a">0-8053-7133-8</subfield>
  </field>
  <field name="100" ind1=" " ind2=" ">
    <subfield name="b">d</subfield>
    <subfield name="c">1996</subfield>
    <subfield name="h">scr</subfield>
  </field>
  <field name="101" ind1="0" ind2=" ">
    <subfield name="a">eng</subfield>
  </field>
  <field name="102" ind1=" " ind2=" ">
    <subfield name="a">usa</subfield>
  </field>
```

```
  <field name="105" ind1=" " ind2=" ">
    <subfield name="a">a</subfield>
  </field>
  <field name="200" ind1="0" ind2=" ">
    <subfield name="a">Concepts of Programming Languages</subfield>
    <subfield name="f">Robert W. Sebesta</subfield>
  </field>
  <field name="210" ind1=" " ind2=" ">
    <subfield name="a">Reading [etc.]</subfield>
    <subfield name="c">Addison-Wesley Publishing Company</subfield>
    <subfield name="d">1996</subfield>
  </field>
  <field name="215" ind1=" " ind2=" ">
    <subfield name="a">xv, 634 p.</subfield>
    <subfield name="c">ilustr.</subfield>
    <subfield name="d">24 cm</subfield>
  </field>
  <field name="675" ind1=" " ind2=" ">
    <subfield name="a">514.7</subfield>
  </field>
  <field name="700" ind1=" " ind2="1">
    <subfield name="a">Sebesta</subfield>
    <subfield name="b">Robert W.</subfield>
    <subfield name="4">070</subfield>
  </field>
  <field name="992" ind1=" " ind2=" ">
    <subfield name="b">crsale9701-old</subfield>
  </field>
  <field name="996" ind1="0" ind2=" ">
    <subfield name="d">M-20111</subfield>
    <subfield name="f">000020111</subfield>
  </field>
</record>
```

Listing 5.6: Zapis nakon obrade *Klasifikator*-a

Uloga *Predmetizator* ima privilegije za kreiranje, brisanje i ažuriranje svih polja/potpolja između *600* i *610* (tabela 5.8).

Uloga	Predmetizator
Operacije	kreiranje, brisanje, ažuriranje
Propagacija	*smer*: niz hijerarhiju, *nivo*: neograničen
Tip	dozvola pristupa
Resursi	/record[MON]/field[@name>='600' and @name<='610']

Tabela 5.8: Privilegije za obradu polja za ulogu *Predmetizator*

Primer 5.4 *Ako uloga* Predmetizator *doda potpolje* 610a=Osnovne tehnike programiranja *u prethodno kreiran zapis formiraće se zapis predstavljen u listingu 5.7.*

```
<record>
  <record>
  <field name="001" ind1=" " ind2=" ">
    <subfield name="7">ba</subfield>
    <subfield name="a">i</subfield>
    <subfield name="b">a</subfield>
    <subfield name="c">m</subfield>
    <subfield name="d">0</subfield>
  </field>
  <field name="010" ind1=" " ind2=" ">
    <subfield name="a">0-8053-7133-8</subfield>
  </field>
  <field name="100" ind1=" " ind2=" ">
    <subfield name="b">d</subfield>
    <subfield name="c">1996</subfield>
    <subfield name="h">scr</subfield>
  </field>
  <field name="101" ind1="0" ind2=" ">
    <subfield name="a">eng</subfield>
  </field>
  <field name="102" ind1=" " ind2=" ">
    <subfield name="a">usa</subfield>
  </field>
  <field name="105" ind1=" " ind2=" ">
    <subfield name="a">a</subfield>
  </field>
  <field name="200" ind1="0" ind2=" ">
    <subfield name="a">Concepts of Programming Languages</subfield>
    <subfield name="f">Robert W. Sebesta</subfield>
  </field>
  <field name="210" ind1=" " ind2=" ">
    <subfield name="a">Reading [etc.]</subfield>
    <subfield name="c">Addison-Wesley Publishing Company</subfield>
    <subfield name="d">1996</subfield>
  </field>
  <field name="215" ind1=" " ind2=" ">
    <subfield name="a">xv, 634 p.</subfield>
    <subfield name="c">ilustr.</subfield>
    <subfield name="d">24 cm</subfield>
  </field>
  <field name="610" ind1=" " ind2=" ">
    <subfield name="a">Osnovne tehnike programiranja</subfield>
  </field>
  <field name="675" ind1=" " ind2=" " >
    <subfield name="a">514.7</subfield>
  </field>
```

```
  <field name="700" ind1=" " ind2="1">
    <subfield name="a">Sebesta</subfield>
    <subfield name="b">Robert W.</subfield>
    <subfield name="4">070</subfield>
  </field>
  <field name="992" ind1=" " ind2=" ">
    <subfield name="b">crsale9701-old</subfield>
  </field>
  <field name="996" ind1="0" ind2=" ">
    <subfield name="d">M-20111</subfield>
    <subfield name="f">000020111</subfield>
  </field>
</record>
```

Listing 5.7: Zapis nakon obrade *Predmetizator*-a

Poglavlje 6

Zaključak

Predmet istraživanja disertacije pripada oblasti kontrole pristupa (*access control*). Istraživanja u ovoj oblasti pre svega razmatraju na koji način korisnici mogu pristupiti resursima računarskog sistema. Kontrola pristupa predstavlja fundamentalni sigurnosni mehanizam koji je danas u upotrebi.

Pomoću sistema za upravljanje poslovnim tokovima (*workflow*) moguća je implementacija složenih poslovnih procesa koji se susreću u realnim slučajevima. Realizacija kontrole pristupa za takve sisteme može da zavisi i od značajnog broja različitih faktora koji se razlikuju od procesa do procesa. Mogući način za rešavanja kontrole pristupa u ovakvim uslovima je uspostavljanje kontekstno zavisne kontrole pristupa koja omogućuje da različiti faktori utiču na mehanizam sprovođenja kontrole pristupa. Model konteksno zavisnog sistema namenjenog za sprovođenje kontrole pristupa u poslovnim sistemima predstavlja osnovnu temu ove disertacije.

Sistematizacija dostupne literature iz ove oblasti izvšena je u prvom poglavlju. Dat je opis osnovnih pojmova i principa u kontroli pristupa, a potom je predstavljen model kontrole pristupa zasnovan na korisničkim ulogama (*Role Based Access Control, RBAC*). Takođe, analizirana je njegova primena u različitim sistemima. Zatim su predstavljeni različiti aspekti proširenja RBAC modela koji se susreću u literaturi. Na kraju poglavlja analiziran je uticaj konteksta na modele kontrole pristupa.

Problemu kontekstno zavisne kontrole pristupa u sistemima za upravljanje poslovnim procesima posvećen je relativno mali broj istaživanja, imajući u

vidu ukupan broj objavljenih rezultata koji se odnose na kontrolu pristupa u poslovnim sistemima. Postojeći dostupni rezultati u ovoj oblasti uglavnom definišu modele kontrole pristupa koji parcijalno razmatraju uticaj konteksta na definisanje i sprovođenje kontrole pristupa.

Drugo poglavlje predstavlja centralni deo disertacije. U njemu je definisan model kontekstno zavisne kontorle pristupa u poslovnim sistemima (*COntext-sensetive Business processes Access Control model*, *COBAC*). Prezentovani model baziran je na standardnom RBAC modelu kontrole pristupa koji je proširen entitetima *poslovnog procesa*, *aktivnostima*, *kontekstom* i *kategorijama resursa*. Baziranjem COBAC modela na RBAC modelu iskorišćene su sve prednosti RBAC modela u odnosu na ostale modele kontrole pristupa, kao i brojna iskustva u primeni RBAC modela navedena u literaturi. Uvođenjem entiteta *poslovnog procesa* i *aktivnosti* omogućeno je efikasnije definisanje i sprovođenje kontrole pristupa za poslovne procese. S obzirom da je moguće da na kontrolu pristupa utiču i faktori iz okruženja samog sistema pa i faktori koji čine sam sistem ali ne i eksplicitno model kontrole pristupa, predloženi model proširen je i *kontekstom*. *Kategorizacija* resursa omogućuje definisanje prava pristupa za čitavu kategoriju (grupu) resursa i time potencijalno smanjuje broj prava prava pristupa koje je potrebo definisati.

Formalna specifikacija modela obuhvata osnovne entitete modela i njihove međusobne relacije, kao i različita ograničenja definisana COBAC modelom. Pored toga, formalnom specifikacijom predstavljen je i model sprovođenja kontrole pristupa.

Za modelovanje konteksta odabran je ontološki pristup jer korišćenje ovakvog pristupa omogućuje formalnu reprezentaciju, bogatstvo informacija i interoperabilnost između različitih sistema. Osim toga, ovim pristupom je model konteksta moguće relativno lako prilagoditi različitim slučajevima korišćenja.

Softverska arhitektura sistema predstavljena je u trećem poglavlju. Arhitekturom su specificirane osnovne komponente sistema i njihove međusobne veze. COBAC sistem se može dekomponovati u četiri podsistema:

- podsistem za administraciju,
- podsistem za proveru identiteta,
- podsistem za autorizaciju, i
- kontekstni podsistem.

Na kraju trećeg poglavlja prikazano je okruženje implementacije prototipa kojim je realizovan COBAC model.

Četvrto poglavlje sadrži verifikaciju prikazanog modela za konkretan sistem — Postupak izbora u nastavna zvanja na Fakultetu tehničkih nauka u Novom Sadu. Izvršena je analiza datog postupka na osnovu koje je formiran dijagram ovog poslovnog procesa. Specificirani su resursi koji se javljaju u navedenom procesu kao i operacije koje se nad njima izvršavaju. Takođe, formirana je hijerarhija korisničkih uloga koje učestvuju u ovom poslovnom procesu. Prezentovana je analiza zahteva sa stanovišta kontrole pristupa na osnovu koje je formiran model konteksta, prava pristupa i dodatni kontekstni moduli potrebni za akviziciju konteksta i definisanje kontekstno zavisnih prava pristupa.

Peto poglavlje demonstrira primenu COBAC modela na realnom poslovnom procesu u bibliotekama – Obrada bibliografskih zapisa. Za razliku od procesa predstavljenog u četvrtom poglavlju gde je težište kontrole pristupa bilo na nivou procesa, poslovni proces u ovom poglavlju karakteriše težište kontrole pristupa na nivou resursa, tj. bibliografskih zapisa, pošto je potrebno sprovoditi kontrolu pristupa nad delovima bibliografskih zapisa. Cilj ove studije slučaja je da pokaže da se COBAC protip može iskoristiti u ovakvim i sličnim slučajevima, gde se COBAC koristi za kontrolu pristupa na nivou procesa, a neki drugi namenski sistem se koristi za kontrolu pristupa na nivou delova resursa.

Na osnovu detaljne analize dostupne literature, date u prvom poglavlju, može se zaključiti da postojeći modeli kontrole pristupa imaju neke od sledećih nedostataka u slučaju njihove primene u poslovnim sistemima:

- ne podržavaju uticaj konteksta ili je taj uticaj ograničen,
- nisu pogodni za efikasnu primenu u poslovnim sistemima jer ne identifikuju sve bezbedonosne aspekte poslovnih procesa,
- ne mogu se lako prilagoditi za različite poslovne sisteme, i
- nije moguće definisanje kako jednostavnih tako i složenih zahteva kontrole pristupa.

Osnovni doprinos disertacije dat je COBAC modelom koji prevazilazi navedene nedostatke. COBAC model karakterišu sledeće osobine:

- Baziran je na RBAC modelu koji je proširen konceptom poslovnog procesa i aktivnosti. Korišćenjem ova dva koncepta moguće je direktno povezati određene segmente kontrole pristupa sa aktivnostima umesto da se povezuju sa tekućom sesijom korisnika i na ovaj način obezbediti efikasnije definisanje prava pristupa i ograničenja za različite poslovne sisteme. Pored toga, korišćenje poslovnog procesa i aktivnosti obezbeđuje nezavisnost odgovarajućih segmenata kontrole pristupa od broja sesija u kojima se aktivnosti izvršavaju. Kako se u COBAC model poslovni proces posmatra kao sekvenca aktivnosti koje se izvršavaju, ulogama su dodeljene aktivnosti koje one imaju pravo da izvrše, dok su privilegije (za izvršenje operacija nad resursima) potrebne za izvršenje aktivnosti dodeljene aktivnostima. Na ovaj način obezbeđena je preciznija kontrola najmanjih privilegija.
- Prava pristupa mogu biti definisana na nivou određene instance procesa (instance aktivnosti) i na taj način zadovoljiti različite sigurnosne zahteve koje različite instance procesa mogu da imaju. Sa druge strane, zahtevanje da se za svaku instancu procesa eksplicitno definišu prava pristupa značajno komplikuje proces definisanja autorizacija, stoga COBAC model podržava definisanje prava pristupa i na nivou definicije procesa (definicije aktivnosti). Prava pristupa definisana na nivou definicije procesa primenjuju se na sve instance tog procesa.
- Modelom je definisano više statičkih i dinamičkih ograničenja potrebnih za efikasno sprovođenje kontrole pristupa u sistemima za upravljanje poslovnim tokovima.
- Kako realni poslovni procesi mogu da imaju sofisticirane autorizacione zahteve koji mogu da zavise od više različitih faktora iz okruženja, relacije u COBAC modelu proširene su kontetkstno zavisnim uslovima. Na ovaj način obezbeđena je podrška za kontekstno zavisnu kontrolu pristupa. Pored ovoga, proširena je definicija uloge sa kontekstnim informacijama.
- Predloženi model konteksta razvijen je koristeći ontologije kako bi se na efikasan način mogle opisati različite kontekstne informacije, obezbediti semantička interoperabilnosti između različitih kontekstno zavisnih sistema i relativno jednostavno proširiti kontekst za konkretni sulučaj korišćenja.
- Resursi u COBAC modelu mogu biti hijerarhijski organizovani, što može da utiče na smanjenje broja privilegija koje je potrebno definisati.

Prikazana prototipska implementacija koja ispunjava ciljeve u pogledu funkcionalnosti postavljene pred COBAC predstavlja potvrdu praktične vrednosti predloženog modela.

Predloženi COBAC model ne razmatra kvalitet kontekstnih informacija, poput tačnosti, pouzdanosti i sl., što u određenim situacijama može biti od velikog značaja i uticaja na kontrolu pristupa. Jedan od daljih pravaca razvoja modela bilo bi da se prilikom definisanje prava pristupa i sprovođenja kontrole pristupa razmatra i kvalitet kontekstnih informacija. Kako danas postoji sve veći broj interoganizacionih poslovnih procesa varijanta COBAC modela prilagođena distribuiranom okruženju bi bila još jedan od mogućih pravaca daljeg razvoja. Uključenje DRM-a (*Digital Rights Management*) može biti intresantan pravac u proširenju ovog modela.

Bibliografija

[1] Diala Abi Haidar, Nora Cuppens-Boulahia, Frederic Cuppens, and Herve Debar. An extended RBAC profile of XACML. In *SWS '06: Proceedings of the 3rd ACM workshop on Secure web services*, pp. 13–22, New York, NY, USA, 2006. ACM.

[2] Gregory D. Abowd, Anind K. Dey, Peter J. Brown, Nigel Davies, Mark Smith, and Pete Steggles. Towards a better understanding of context and context-awareness. In *HUC '99: Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*, pp. 304–307, London, UK, 1999. Springer-Verlag.

[3] Gregory D. Abowd, Elizabeth D. Mynatt, and Tom Rodden. The human experience. *IEEE Pervasive Computing*, 1(1):48–57, 2002.

[4] Ross Ackland, Kerry L. Taylor, Laurent Lefort, Mark A. Cameron, and Joel Rahman. Semantic service integration for water resource management. In Yolanda Gil, Enrico Motta, V. Richard Benjamins, and Mark A. Musen, editors, *International Semantic Web Conference*, volume 3729 of *Lecture Notes in Computer Science*, pp. 816–828. Springer, 2005.

[5] Raman Adaikkalavan and Sharma Chakravarthy. Active authorization rules for enforcing role-based access control and its extensions. In *ICDEW '05: Proceedings of the 21st International Conference on Data Engineering Workshops*, p. 1197, Washington, DC, USA, 2005. IEEE Computer Society.

[6] Gail-Joon Ahn and Hongxin Hu. Towards realizing a formal RBAC model in real systems. In *SACMAT '07: Proceedings of the 12th ACM symposium on Access control models and technologies*, pp. 215–224, New York, NY, USA, 2007. ACM.

[7] Gail-Joon Ahn and Ravi Sandhu. The RSL99 language for role-based separation of duty constraints. In *RBAC '99: Proceedings of the fourth ACM workshop on Role-based access control*, pp. 43–54, New York, NY, USA, 1999. ACM.

[8] Gail-Joon Ahn and Ravi Sandhu. Role-based authorization constraints specification. *ACM Trans. Inf. Syst. Secur.*, 3(4):207–226, 2000.

[9] Gail-Joon Ahn, Ravi Sandhu, Myong Kang, and Joon Park. Injecting RBAC to secure a web-based workflow system. In *RBAC '00: Proceedings of the fifth ACM workshop on Role-based access control*, pp. 1–10, New York, NY, USA, 2000. ACM.

[10] Mohammad A. Al-Kahtani and Ravi Sandhu. A model for attribute-based user-role assignment. In *ACSAC '02: Proceedings of the 18th Annual Computer Security Applications Conference*, p. 353, Washington, DC, USA, 2002. IEEE Computer Society.

[11] Mohammad A. Al-Kahtani and Ravi Sandhu. Induced role hierarchies with attribute-based RBAC. In *SACMAT '03: Proceedings of the eighth ACM symposium on Access control models and technologies*, pp. 142–148, New York, NY, USA, 2003. ACM.

[12] Muhammad Alam, Michael Hafner, and Ruth Breu. Constraint based role based access control (CRBAC) for restricted administrative delegation constraints in the SECTET. In *PST '06: Proceedings of the 2006 International Conference on Privacy, Security and Trust*, pp. 1–5, New York, NY, USA, 2006. ACM.

[13] Muhammad Alam, Michael Hafner, and Ruth Breu. A constraint based role based access control in the SECTET a model-driven approach. In *PST '06: Proceedings of the 2006 International Conference on Privacy, Security and Trust*, pp. 1–13, New York, NY, USA, 2006. ACM.

[14] J. J. Alferes, F. F Banti, and A. Brogi. An event-condition-action logic programming language. In *Logics in Artificial Intelligence*, volume 4160/2006, pp. 29–42. Springer Berlin Heidelberg, 2006.

[15] Sinan Si Alhir. *Learning UML*. O'Reilly, 2003.

[16] American National Standard for Information Technology (ANSI) ANSI INCITS 359-2004, International Committee for Information Technology Standards (INCITS). Role-based access control. Standard, 2004. http://csrc.nist.gov/groups/SNS/rbac.

[17] Elisa Bertino Arjmand Samuel, Arif Ghafoor. A framework for specification and verification of generalized spatio-temporal role based access control model. Technical report, 2007.

[18] V Atluri and W Huang. An authorization model for workflows. In *In Proceedings of the 4th European Symposium on Research in Computer Security*, pp. 44–64. Springer-Verlag, 1996.

[19] Vijayalakshmi Atluri and Wei-Kuang Huang. A Petri net based safety analysis of workflow authorization models. *Journal of Computer Security*, 8(2,3):209–240, 2000.

[20] Jean Bacon, Ken Moody, and Walt Yao. A model of OASIS role-based access control and its support for active security. *ACM Trans. Inf. Syst. Secur.*, 5(4):492–540, 2002.

[21] Prasanna H. Bammigatti and P. R. Rao. Generic WA-RBAC: role based access control model for web applications. In *ICIT '06: Proceedings of the 9th International Conference on Information Technology*, pp. 237–240, Washington, DC, USA, 2006. IEEE Computer Society.

[22] Yubin Bao, Jie Song, Daling Wang, Derong Shen, and Ge Yu. A role and context based access control model with UML. In *International Conference for Young Computer Scientists*, volume 0, pp. 1175–1180, Los Alamitos, CA, USA, 2008. IEEE Computer Society.

[23] R. Baxter, M. Cameron, J. Colton, C. M. O'Keefe, R. Sparks, K. Taylor, and U. Srinivasan. An Architecture for health data integration. Technical report, 2003.

[24] Biblioteka grada Beograda. http://www.bgb.rs/.

[25] András Belokosztolszki, David M. Eyers, and Ken Moody. Policy contexts: controlling information flow in parameterised RBAC. In *POLICY '03: Proceedings of the 4th IEEE International Workshop on Policies for Distributed Systems and Networks*, pp. 99–110, Washington, DC, USA, 2003. IEEE Computer Society.

[26] Elisa Bertino, Piero Andrea Bonatti, and Elena Ferrari. TRBAC: a temporal role-based access control model. In *RBAC '00: Proceedings of the fifth ACM workshop on Role-based access control*, pp. 21–30, New York, NY, USA, 2000. ACM.

[27] Elisa Bertino, Piero Andrea Bonatti, and Elena Ferrari. TRBAC: A temporal role-based access control model. *ACM Trans. Inf. Syst. Secur.*, 4(3):191–233, 2001.

[28] Elisa Bertino, Barbara Carminati, and Elena Ferrari. Access control for XML documents and data. *Information Security Technical Report*, 9(3):19–34, 2004.

[29] Elisa Bertino, Silvana Castano, and Elena Ferrari. Securing XML documents with Author-X. *IEEE Internet Computing*, 05(3):21–31, 2001.

[30] Elisa Bertino, Barbara Catania, Maria Luisa Damiani, and Paolo Perlasca. GEO-RBAC: a spatially aware RBAC. In *SACMAT '05: Proceedings of the tenth ACM symposium on Access control models and technologies*, pp. 29–37, New York, NY, USA, 2005. ACM.

[31] Elisa Bertino, Barbara Catania, Elena Ferrari, and Paolo Perlasca. A logical framework for reasoning about access control models. *ACM Trans. Inf. Syst. Secur.*, 6(1):71–127, 2003.

[32] Elisa Bertino and Elena Ferrari. Secure and selective dissemination of XML documents. *ACM Trans. Inf. Syst. Secur.*, 5(3):290–331, 2002.

[33] Elisa Bertino, Elena Ferrari, and Vijay Atluri. The specification and enforcement of authorization constraints in workflow management systems. *ACM Trans. Inf. Syst. Secur.*, 2(1):65–104, 1999.

[34] Elisa Bertino, Elena Ferrari, and Vijayalakshmi Atluri. A flexible model supporting the specification and enforcement of role-based authorizations in workflow management systems. In *In ACM Workshop on Role-based Access Control*, pp. 1–12, New York, NY, USA, 1997. ACM.

[35] Elisa Bertino, Bhavani Thuraisingham, Michael Gertz, and Maria Luisa Damiani. Security and privacy for geospatial data: concepts and research directions. In *SPRINGL '08: Proceedings of the SIGSPATIAL ACM GIS 2008 International Workshop on Security and Privacy in GIS and LBS*, pp. 6–19, New York, NY, USA, 2008. ACM.

[36] H. Bestougeff, J. E. Dubois, and B. (Eds.) Thuraisingham. *Heterogeneous information exchange and organizational hubs.* Springer-Verlag New York, Inc., New York, NY, USA, 2002.

[37] C. Bettini, S. Jajodia, X. Wang, and D. Wijesekera. Obligation monitoring in policy management. In *POLICY '02: Proceedings of the 3rd International Workshop on Policies for Distributed Systems and Networks (POLICY'02)*, p. 2, Washington, DC, USA, 2002. IEEE Computer Society.

[38] Claudio Bettini, Sushil Jajodia, X. Sean Wang, and Duminda Wijesekera. Provisions and obligations in policy management and security applications. In *VLDB '02: Proceedings of the 28th international conference on Very Large Data Bases*, pp. 502–513. VLDB Endowment, 2002.

[39] Rafae Bhatti, Elisa Bertino, and Arif Ghafoor. A trust-based context-aware access control model for web-services. In *ICWS '04: Proceedings of the IEEE International Conference on Web Services*, pp. 184–191, Washington, DC, USA, 2004. IEEE Computer Society.

[40] Rafae Bhatti, Elisa Bertino, and Arif Ghafoor. A trust-based context-aware access control model for web-services. *Distributed and Parallel Databases*, 18(1):83–105, 2005.

[41] Rafae Bhatti, Elisa Bertino, Arif Ghafoor, and James B.D. Joshi. XML-based specification for web services document security. *Computer*, 37(4):41–49, 2004.

[42] Rafae Bhatti, Arif Ghafoor, Elisa Bertino, and James B. D. Joshi. X-GTRBAC: an XML-based policy specification framework and architecture for enterprise-wide access control. *ACM Trans. Inf. Syst. Secur.*, 8(2):187–227, 2005.

[43] Rafae Bhatti, James B.D. Joshi, Elisa Bertino, and Arif Ghafoo. Access control in dynamic XML-based web-services with X-RBAC. In *1st International Conference on Web Services*, 2003.

[44] Piero Bonatti, Sabrina De Capitani di Vimercati, and Pierangela Samarati. An algebra for composing access control policies. *ACM Trans. Inf. Syst. Secur.*, 5(1):1–35, 2002.

[45] Grady Booch, James Rumbaugh, and Ivar Jacobson. *The unified modeling language user guide*. Addison Wesley, 1998.

[46] Reinhardt A. Botha and Jan H. P. Eloff. Access control in document-centric workflow systems – an agent-based approach. *Computers & Security*, 20(6):525 – 532, 2001.

[47] Reinhardt A. Botha and Jan H. P. Eloff. Designing role hierarchies for access control in workflow systems. In *COMPSAC '01: Proceedings of the 25th International Computer Software and Applications Conference on Invigorating Software Development*, pp. 117–122, Washington, DC, USA, 2001. IEEE Computer Society.

[48] Reinhardt A. Botha and Jan H. P. Eloff. Separation of duties for access control enforcement in workflow environments. *IBM Systems Journal*, 40(3):666–682, 2001.

[49] Reinhardt A. Botha and Jan H.P. Eloff. A Framework for access control in workflow environments. *Information Management and Computer Security*, 9(3):126–133, 2001.

[50] Patrick Brézillon and Ghita Kouadri Mostéfaoui. Context-based security policies: a new modeling approach. In *PERCOMW '04: Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops*, p. 154, Washington, DC, USA, 2004. IEEE Computer Society.

[51] T. Buchholz, A. Kupper, and M. Schiffers. Quality of context: what it is and why we need it. In *Proceedings of the Workshop of the HP Open View University Association (HPOVUA 2003)*, 2003.

[52] Changwoo Byun and Seog Park. Two phase filtering for XML access control. In *Secure Data Management*, pp. 115–130. Springer, 2006.

[53] Paolina Centonze, Gleb Naumovich, Stephen J. Fink, and Marco Pistoia. Role-Based access control consistency validation. In *ISSTA '06: Proceedings of the 2006 international symposium on Software testing and analysis*, pp. 121–132, New York, NY, USA, 2006. ACM.

[54] Guanling Chen and David Kotz. A Survey of context-aware mobile computing research. Technical report, Hanover, NH, USA, 2000.

[55] H. Chen, T. Finin, and A. Joshi. Using OWL in a pervasive computing broker. In *Workshop on Ontologies in Agent Systems*, 2003.

[56] Hong Chen and Ninghui Li. Constraint generation for separation of duty. In *SACMAT '06: Proceedings of the eleventh ACM symposium on Access control models and technologies*, pp. 130–138, New York, NY, USA, 2006. ACM.

[57] Shu-Ching Chen, Mei-Ling Shyu, and Na Zhao. SMARXO: towards secured multimedia applications by adopting RBAC, XML and object-relational database. In *MULTIMEDIA '04: Proceedings of the 12th annual ACM international conference on Multimedia*, pp. 432–435, New York, NY, USA, 2004. ACM.

[58] Damian G. Cholewka, Reinhardt A. Botha, and Jan H. P. Eloff. A context-sensitive access control model and prototype implementation. In *Proceedings of the IFIP TC11 Fifteenth Annual Working Conference on Information Security for Global Information Infrastructures*, pp. 341–350, Deventer, The Netherlands, The Netherlands, 2000. Kluwer, B.V.

[59] Lorenzo Cirio, Isabel F. Cruz, and Roberto Tamassia. A role and attribute based access control system using semantic web technologies. In *On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops*, pp. 1256–1266, New York, NY, USA, 2007. Springer-Verlag.

[60] David D. Clark and David R. Wilson. Evolution of a model for computer integrity. In *Workshop on Data Integrity*, 1989.

[61] James Clark. XSL Transformations (XSLT) 1.0. W3C Recommendation, 1999. http://www.w3.org/TR/xslt.

[62] Jan De Clercq. *Windows server 2003 security infrastructures: core security features*. Digital Press, Newton, MA, USA, 2004.

[63] Workflow Management Coalition. Workflow management coalition the workflow reference model. TCOO- 1003, 1994.

[64] Eve Cohen, Roshan K. Thomas, William Winsborough, and Deborah Shands. Models for coalition-based access control (CBAC). In *SACMAT '02: Proceedings of the seventh ACM symposium on Access control models and technologies*, pp. 97–106, New York, NY, USA, 2002. ACM.

[65] Alessandro Colantonio, Roberto Di Pietro, and Alberto Ocello. A cost-driven approach to role engineering. In *SAC '08: Proceedings of the 2008 ACM symposium on Applied computing*, pp. 2129–2136, New York, NY, USA, 2008. ACM.

[66] Open GIS Consortium. Open GIS simple features specification for SQL. Revision 1.1. OGC specification, 1999.

[67] Open GIS Consortium. The open GIS abstract specification. topic 1: feature geometry (ISO 19107 spatial schema). Version 5. OGC specification, 2001.

[68] Open GIS Consortium. Open GIS geography markup language (GML) implementation specification. Version 3.00. OGC specification, 2003.

[69] Antonio Corradi, Rebecca Montanari, and Daniela Tibaldi. Context-based access control for ubiquitous service provisioning. *Proceedings of the 28th Annual International Computer Software and Applications Conference (COMPSAC)*, 1:444–451, 2004.

[70] Antonio Corradi, Rebecca Montanari, and Daniela Tibaldi. Context-based access control management in ubiquitous environments. *Proceedings of the 3rd IEEE International Symposium on Network Computing and Applications(NCA)*, 00:253–260, 2004.

[71] Michael J. Covington, Wende Long, Srividhya Srinivasan, Anind K. Dev, Mustaque Ahamad, and Gregory D. Abowd. Securing context-aware applications using environment roles. In *Proceedings of the 6th ACM Symposium on Access Control Models and Technologies (SACMAT)*, pp. 10–20, New York, NY, USA, 2001. ACM.

[72] Michael J. Covington, Matthew J. Moyer, and Mustaque Ahamad. Generalized role-based access control for securing future applications. In *Proceedings of the 23rd National Information Systems Security Conference (NISSC)*, Baltimore, MD, USA, 2000.

[73] Edward J. Coyne. Role engineering. In *RBAC '95: Proceedings of the first ACM Workshop on Role-based access control*, p. 4, New York, NY, USA, 1996. ACM.

[74] Jason Crampton. Specifying and enforcing constraints in role-based access control. In *SACMAT '03: Proceedings of the eighth ACM symposium on Access control models and technologies*, pp. 43–50, New York, NY, USA, 2003. ACM.

[75] Jason Crampton. Applying hierarchical and role-based access control to XML documents. In *SWS '04: Proceedings of the 2004 workshop on Secure web service*, pp. 37–46, New York, NY, USA, 2004. ACM.

[76] Jason Crampton. A reference monitor for workflow systems with constrained task execution. In *SACMAT '05: Proceedings of the tenth ACM symposium on Access control models and technologies*, pp. 38–47, New York, NY, USA, 2005. ACM.

[77] Jason Crampton. Applying hierarchical and role-based access control to XML documents. *Science and System Engineering*, 21(5):325–338, 2006.

[78] Isabel F. Cruz, Rigel Gjomemo, Benjamin Lin, and Mirko Orsini. A location aware role and attribute based access control system. In *GIS '08: Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems*, pp. 1–2, New York, NY, USA, 2008. ACM.

[79] Xiutao Cui, Yuliang Chen, and Junzhong Gu. Ex-RBAC: an extended role based access control model for location-aware mobile collaboration system. In *ICIMP '07: Proceedings of the Second International Conference on Internet Monitoring and Protection*, p. 36, Washington, DC, USA, 2007. IEEE Computer Society.

[80] Apache CXF. The Apache Software Foundation. http://cxf.apache.org/.

[81] Ernesto Damiani, Sabrina De Capitani di Vimercati, Stefano Paraboschi, and Pierangela Samarati. A fine-grained access control system for XML documents. *ACM Trans. Inf. Syst. Secur.*, 5(2):169–202, 2002.

[82] Ernesto Damiani, Pierangela Samarati, Sabrina De Capitani di Vimercati, and Stefano Paraboschi. Controlling access to XML documents. *IEEE Internet Computing*, 5(6):18–28, 2001.

[83] Maria Luisa Damiani, Elisa Bertino, Barbara Catania, and Paolo Perlasca. GEO-RBAC: a spatially aware RBAC. *ACM Trans. Inf. Syst. Secur.*, 10(1):2, 2007.

[84] Maria Luisa Damiani, Elisa Bertino, and Claudio Silvestri. Approach to supporting continuity of usage in location-based access control. In *FTDCS '08: Proceedings of the 2008 12th IEEE International Workshop on Future Trends*

of Distributed Computing Systems, pp. 199–205, Washington, DC, USA, 2008. IEEE Computer Society.

[85] Maria Luisa Damiani and Claudio Silvestri. Towards movement-aware access control. In *SPRINGL '08: Proceedings of the SIGSPATIAL ACM GIS 2008 International Workshop on Security and Privacy in GIS and LBS*, pp. 39–45, New York, NY, USA, 2008. ACM.

[86] Thomas H. Davenport and James E. Short. The new industrial engineering: information technology and business process redesign. *Sloan Management Review*, 31(4):11–27, 1990.

[87] Steven Dawson, Shelly Qian, and Pierangela Samarati. Providing security and interoperation of heterogeneous systems. *Distrib. Parallel Databases*, 8(1):119–145, 2000.

[88] Renato de Freitas Bulcao Neto and Maria da Graca Campos Pimentel. Toward a domain-independent semantic model for context-aware computing. In *Proceedings of the 3rd Latin American Web Congress (LA-WEB)*, pp. 61–70, Washington, DC, USA, 2005. IEEE Computer Society.

[89] M. A. C. Dekker, J. G. Cederquist, J. Crampton, and S. Etalle. Extended privilege inheritance in RBAC. In *ASIACCS '07: Proceedings of the 2nd ACM symposium on Information, computer and communications security*, pp. 383–385, New York, NY, USA, 2007. ACM.

[90] A.K. Dey. *Providing architectural support for building context-aware applications*. PhD thesis, Georgia Institute of Technology, Georgia, US, 2000.

[91] Anind K. Dey. Understanding and using context. *Personal Ubiquitous Comput.*, 5(1):4–7, 2001.

[92] Robert B. Doorenbos. *Production matching for large learning systems*. PhD thesis, Pittsburgh, PA, USA, 1995.

[93] JBoss Drools. JBoss Community. http://www.jboss.org/drools/.

[94] EasyBeans EJB. OW2 Consortium. http://www.easybeans.org/.

[95] Sareh Sadat Emami, Morteza Amini, and Saadan Zokaei. A context-aware access control model for pervasive computing environments. *Proceedings of the IEEE International Conference on Intelligent Pervasive Computing (IPC)*, 0:51–56, 2007.

[96] eXist-db Open Source Native XML Database. The SourceForge. http://exist.sourceforge.net/.

[97] Glenn Faden. RBAC in UNIX administration. In *RBAC '99: Proceedings of the fourth ACM workshop on Role-based access control*, pp. 95–101. ACM Press, 1999.

[98] Fujun Feng, Chuang Lin, Dongsheng Peng, and Junshan Li. A trust and context based access control model for distributed systems. In *10th IEEE International Conference on High Performance Computing and Communications*, volume 0, pp. 629–634, Los Alamitos, CA, USA, 2008. IEEE Computer Society.

[99] Xu Feng, Xie Jun, Huang Hao, and Xie Li. Context-aware role-based access control model for web services. *Grid and Cooperative Computing – GCC 2004 Workshops, International Workshop on Information Security and Survivability for Grid*, 3252:430–436, 2004.

[100] Eduardo B. Fernandez, Maria M. Larrondo-Petrie, and Alvaro E. Escobar. Contexts and context-based access control. *Proceedings of the 3rd International Conference on Wireless and Mobile Communications (ICWMC)*, 0:73–78, 2007.

[101] D. Ferraiolo, D. Gilbert, and N. Lynch. An examination of federal and commercial access control policy needs. In *Proc. of the NIST-NCSC Nat. (U.S.) Comp. Security Conference*, pp. 107–116, 1993.

[102] D. Ferraiolo and R. Kuhn. Role-based access control. In *In Proceedings of National Computer Security Conference*, 1992.

[103] David Ferraiolo, Janet Cugini, and Richard Kuhn. Role-based access control (RBAC): features and motivations. In *In Proceedings of 11th Annual Computer Security Application Conference*, pp. 241–248, 1995.

[104] David F. Ferraiolo, John F. Barkley, and D. Richard Kuhn. A role-based access control model and reference implementation within a corporate intranet. *ACM Trans. Inf. Syst. Secur.*, 2(1):34–64, 1999.

[105] David F. Ferraiolo, D. Richard Kuhn, and Ramaswamy Chandramouli. *Role-based access control*. Artech Hous, 2003.

[106] David F. Ferraiolo, Ravi Sandhu, Serban Gavrila, D. Richard Kuhn, and Ramaswamy Chandramouli. Proposed NIST standard for role-based access control. *ACM Transactions on Information and System Security (TISSEC)*, 4(3):224–274, 2001.

[107] José Bringel Filho and Hervé Martin. Using context quality indicators for improving context-based access control in pervasive environments. In *EUC '08: Proceedings of the 2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing*, pp. 285–290, Washington, DC, USA, 2008. IEEE Computer Society.

[108] Simone Fischer Hubner. *IT-security and privacy: design and use of privacy-enhancing security mechanisms*. Springer-Verlag New York, Inc., New York, NY, USA, 2001.

[109] Organisation for Economic Co-operation and Development. OCED guidelines on the protection of privacy and transborder flows of personal data. Guidlines, 1980. http://www.oecd.org/.

[110] Mario Frank, David Basin, and Joachim M. Buhmann. A class of probabilistic models for role engineering. In *CCS '08: Proceedings of the 15th ACM conference on Computer and communications security*, pp. 299–310, New York, NY, USA, 2008. ACM.

[111] Adam Freeman and Allen Jones. *Programming .NET security*. O'Reilly & Associates, Inc., Sebastopol, CA, USA, 2003.

[112] Eric Freudenthal, Tracy Pesin, Lawrence Port, Edward Keenan, and Vijay Karamcheti. dRBAC: distributed role-based access control for dynamic coalition environments. In *International Conference on Distributed Computing Systems*, volume 0, p. 411, Los Alamitos, CA, USA, 2002. IEEE Computer Society.

[113] Irini Fundulaki and Sebastian Maneth. Formalizing XML access control for update operations. In *SACMAT '07: Proceedings of the 12th ACM symposium on Access control models and technologies*, pp. 169–174, New York, NY, USA, 2007. ACM.

[114] Alban Gabillon. An authorization model for XML databases. In *SWS '04: Proceedings of the 2004 workshop on Secure web service*, pp. 16–28, New York, NY, USA, 2004. ACM.

[115] Lijun Gao, Lu Zhang, and Lei Xu. Access control scheme for workflow. In *ICCET '09: Proceedings of the 2009 International Conference on Computer Engineering and Technology*, pp. 215–217, Washington, DC, USA, 2009. IEEE Computer Society.

[116] Christos K. Georgiadis, Ioannis Mavridis, George Pangalos, and Roshan K. Thomas. Flexible team-based access control using contexts. In *SACMAT '01: Proceedings of the sixth ACM symposium on Access control models and technologies*, pp. 21–27, New York, NY, USA, 2001. ACM.

[117] V.D. Gligor, S.I. Gavrila, and D. Ferraiolo. On the formal definition of separation-of-duty policies and their composition. In *IEEE Symposium on Security and Privacy, 1998. Proceedings*, pp. 172–183, May 1998.

[118] Robert Goldblatt. *Logics of time and computation*. Center for the Study of Language and Information, Stanford, CA, USA, 1992.

[119] Li Gong and Xiaolei Qian. Computational issues in secure interoperation. *IEEE Trans. Softw. Eng.*, 22(1):43–52, 1996.

[120] Google. Gogle Maps API. API. http://code.google.com/apis/maps/.

[121] The Open Group. Single Sign-On. http://www.opengroup.org/security/sso/.

[122] Shen Haibo and Hong Fan. A context-aware role-based access control model for web services. *Proceedings of the IEEE International Conference on e-Business Engineering (ICEBE)*, 0:220–223, 2005.

[123] Weili Han, Junjing Zhang, and Xiaobo Yao. Context-sensitive access control model and implementation. *Proceedings of the 5th International Conference on Computer and Information Technology (CIT)*, 0:757–763, 2005.

[124] Hibernate relational persistence. JBoss Community. http://www.hibernate.org/.

[125] R. Housley, W. Ford, W. Polk, and D. Solo. Internet X.509 public key infrastructure certificate and CRL profile. IETF RFC 2459, 1999. http://www.ietf.org/rfc/rfc2459.txt.

[126] Wei-Kuang Huang and Vijayalakshmi Atluri. SecureFlow: a secure web-enabled workflow management system. In *RBAC '99: Proceedings of the fourth ACM workshop on Role-based access control*, pp. 83–94, New York, NY, USA, 1999. ACM.

[127] PMF i FTN Univerzitet u Novom Sadu. Bibiliotečki informacioni sistem BISIS, 1992. http://bisis.uns.ac.rs.

[128] IBM-SAP. WS-BPEL extension for people (BPEL4People). Joint white paper, 2007. http://www.ibm.com/developerworks/webservices/library/specification/ws-bpel4people/.

[129] Keith Irwin, Ting Yu, and William H. Winsborough. Enforcing security properties in task-based systems. In *SACMAT '08: Proceedings of the 13th ACM symposium on Access control models and technologies*, pp. 41–50, New York, NY, USA, 2008. ACM.

[130] Trent Jaeger and Jonathon E. Tidswell. Practical safety in flexible access control models. *ACM Trans. Inf. Syst. Secur.*, 4(2):158–190, 2001.

[131] JBoss jBPM (java business process management). JBoss Community. http://www.jboss.org/jbpm.

[132] Dirk Jonscher. Extending access control with duties - realized by active mechanisms. In *Results of the Sixth Working Conference of IFIP Working Group 11.3 on Database Security on Database security, VI : status and prospects*, pp. 91–111, New York, NY, USA, 1993. Elsevier Science Inc.

[133] James B. D. Joshi, Walid G. Aref, Arif Ghafoor, and Eugene H. Spafford. Security models for web-based applications. *Communications of the ACM*, 44(2):38–44, 2001.

[134] James B. D. Joshi, Elisa Bertino, and Arif Ghafoor. Hybrid role hierarchy for generalized temporal role based access control model. In *COMPSAC '02: Proceedings of the 26th International Computer Software and Applications Conference on Prolonging Software Life: Development and Redevelopment*, pp. 951–956, Washington, DC, USA, 2002. IEEE Computer Society.

[135] James B D Joshi, Elisa Bertino, and Arif Ghafoor. Temporal hierarchies and inheritance semantics for GTRBAC. In *SACMAT '02: Proceedings of the seventh ACM symposium on Access control models and technologies*, pp. 74–83, New York, NY, USA, 2002. ACM.

[136] James B. D. Joshi, Elisa Bertino, and Arif Ghafoor. An analysis of expressiveness and design issues for the generalized temporal role-based access control model. *IEEE Trans. Dependable Secur. Comput.*, 2(2):157–175, 2005.

[137] James B. D. Joshi, Elisa Bertino, Arif Ghafoor, and Yue Zhang. Formal foundations for hybrid hierarchies in GTRBAC. *ACM Trans. Inf. Syst. Secur.*, 10(4):1–39, 2008.

[138] James B. D. Joshi, Basit Shafiq, Arif Ghafoor, and Elisa Bertino. Dependencies and separation of duty constraints in GTRBAC. In *SACMAT '03: Proceedings of the eighth ACM symposium on Access control models and technologies*, pp. 51–64, New York, NY, USA, 2003. ACM.

[139] Myong H. Kang, Brian J. Eppinger, and Judith N. Froscher. Tools to support secure enterprise computing. In *ACSAC '99: Proceedings of the 15th Annual Computer Security Applications Conference*, p. 143, Washington, DC, USA, 1999. IEEE Computer Society.

[140] Myong H. Kang, Joon S. Park, and Judith N. Froscher. Access control mechanisms for inter-organizational workflow. In *SACMAT '01: Proceedings of the sixth ACM symposium on Access control models and technologies*, pp. 66–74, New York, NY, USA, 2001. ACM.

[141] Vassilis Kapsalisa, Loukas Hadellisb, Dimitris Karelisb, and Stavros Koubiasc. A dynamic context-aware access control architecture for e-services. *Computers & Security*, 25(7):507–521, 2006.

[142] Axel Kern, Andreas Schaad, and Jonathan Moffett. An administration concept for the enterprise role-based access control model. In *SACMAT '03: Proceedings of the eighth ACM symposium on Access control models and technologies*, pp. 3–11, New York, NY, USA, 2003. ACM.

[143] Axel Kern and Claudia Walhorn. Rule support for role-based access control. In *SACMAT '05: Proceedings of the tenth ACM symposium on Access control models and technologies*, pp. 130–138, New York, NY, USA, 2005. ACM.

[144] Kyu II Kim, Won Gil Choi, Eun Ju Lee, and Ung Mo Kim. RBAC-based access control for privacy protection in pervasive environments. In *ICUIMC '09: Proceedings of the 3rd International Conference on Ubiquitous Information Management and Communication*, pp. 255–259, New York, NY, USA, 2009. ACM.

[145] Guangqian Kong and Jianshi Li. Research on RBAC-based separation of duty constraints. *Journal of Information and Computing Science*, 2(3):235–240, 2007.

[146] Zora Konjović, Goran Sladić, and Branko Milosavljević. Kontrola pristupa XML dokumentima. In *YUNG Info 2005*, p. 14, 2005.

[147] Michiharu Kudo and Satoshi Hada. XML document security based on provisional authorization. In *CCS '00: Proceedings of the 7th ACM conference on Computer and communications security*, pp. 87–96. ACM Press, 2000.

[148] D. Richard Kuhn. Mutual exclusion of roles as a means of implementing separation of duty in role-based access control systems. In *RBAC '97: Proceedings of the second ACM workshop on Role-based access control*, pp. 23–30, New York, NY, USA, 1997. ACM.

[149] Arun Kumar, Neeran Karnik, and Girish Chafle. Context sensitivity in role-based access control. *SIGOPS Oper. Syst. Rev.*, 36(3):53–66, 2002.

[150] Peter Lamb, Robert Power, Gavin Walker, and Michael Compton. Role-based access control for data service integration. In *SWS '06: Proceedings of the 3rd ACM workshop on Secure web services*, pp. 3–12, New York, NY, USA, 2006. ACM.

[151] Usman Latif, James B. D. Joshi, Elisa Bertino, and Arif Ghafoor. A generalized temporal role-based access control model. *IEEE Trans. on Knowl. and Data Eng.*, 17(1):4–23, 2005.

[152] Ninghui Li, Ziad Bizri, and Mahesh V. Tripunitara. On mutually-exclusive roles and separation of duty. In *CCS '04: Proceedings of the 11th ACM conference on Computer and communications security*, pp. 42–51, New York, NY, USA, 2004. ACM.

[153] Ninghui Li, Ji-Won Byun, and Elisa Bertino. A critique of the ANSI standard on role-based access control. *IEEE Security and Privacy*, 5(6):41–49, 2007.

[154] Ninghui Li and Ziqing Mao. Administration in role-based access control. In *ASIACCS '07: Proceedings of the 2nd ACM symposium on Information, computer and communications security*, pp. 127–138, New York, NY, USA, 2007. ACM.

[155] Ninghui Li, Mahesh V. Tripunitara, and Ziad Bizri. On mutually exclusive roles and separation-of-duty. *ACM Trans. Inf. Syst. Secur.*, 10(2):5, 2007.

[156] Y. Li, J. Hong, and J. Landay. ContextMap: modeling scenes of the real world for context-aware computing. In *5th Int. Conf. on Ubiquitous Computing (UbiComp2003)*, 2003.

[157] Ramiro Liscano and Kaining Wang. A context-based delegation access control model for pervasive computing. In *AINAW '07: Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops*, pp. 44–51, Washington, DC, USA, 2007. IEEE Computer Society.

[158] Duen-Ren Liu, Mei-Yu Wu, and Shu-Teng Lee. Role-based authorizations for workflow systems in support of task-based separation of duty. *Journal of Systems and Software*, 73(3):375–387, 2004.

[159] Peng Liu and Zhong Chen. An extended RBAC model for web services in business process. In *CEC-EAST '04: Proceedings of the E-Commerce Technology for Dynamic E-Business, IEEE International Conference*, pp. 100–107, Washington, DC, USA, 2004. IEEE Computer Society.

[160] Douglas L. Long, Julie Baker, and Francis Fung. A prototype secure workflow server. In *ACSAC '99: Proceedings of the 15th Annual Computer Security Applications Conference*, p. 129, Washington, DC, USA, 1999. IEEE Computer Society.

[161] Zhenxing Luo, Nuermaimaiti Heilili, and Zuoquan Lin. A flexible applicable RBAC model and its administration. In *DEXA '07: Proceedings of the 18th International Conference on Database and Expert Systems Applications*, pp. 192–196, Washington, DC, USA, 2007. IEEE Computer Society.

[162] Mingchao Ma and Steve Woodhead. Constraint-enabled distributed rbac for subscription-based remote network services. In *CIT '06: Proceedings of the Sixth IEEE International Conference on Computer and Information Technology*, p. 160, Washington, DC, USA, 2006. IEEE Computer Society.

[163] Zohar Manna and Amir Pnueli. *The temporal logic of reactive and concurrent systems*. Springer-Verlag New York, Inc., New York, NY, USA, 1992.

[164] Zohar Manna and Amir Pnueli. *Temporal verification of reactive systems: safety.* Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[165] Sun Microsystems. Java Enterprise Edition 5 - JEE. Specification, 2006. http://java.sun.com/javaee/.

[166] Sun Microsystems. Java authentication and autorisation service - JAAS. Specification, 2007.

[167] Gerome Miklau and Dan Suciu. Controlling access to published data using cryptography. In *VLDB '2003: Proceedings of the 29th international conference on Very large data bases*, pp. 898–909. VLDB Endowment, 2003.

[168] N. H. Minsky and A. D. Lockman. Ensuring integrity by adding obligations to privileges. In *Proceedings of the 8th International Conference on Software Engineering*, pp. 92–102, 1985.

[169] Till Mossakowski, Michael Drouineaud, and Karsten Sohr. A temporal-logic extension of role-based access control covering dynamic separation of duties. In *Proceedings of the 10th international symposium on temporal representation and fourth international conference on temporal logic*, p. 8, Washington, DC, USA, 2003. IEEE Computer Society.

[170] G. Kouadri Mostéfaoui. Security in pervasive environments, what's next? In *In the proceedings of the 2003 International Conference on Security and Management(SAM'03)*, pp. 93–96. CSREA Press, 2003.

[171] Ghita Kouadri Mostéfaoui and P. Brézillon. A generic framework for context-based distributed authorizations. In *In 4th International and Interdisciplinary Conference on Modeling and Using Context (Context'03)*, pp. 204–217, Berlin, DE, 2003. SpringerLink.

[172] Ghita Kouadri Mostéfaoui and Patrick Brézillon. Modeling context-based security policies with contextual graphs. In *PERCOMW '04: Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops*, p. 28, Washington, DC, USA, 2004. IEEE Computer Society.

[173] Ghita Kouadri Mostéfaoui and Pasquier J. Deterministic context-based security policies: an object-oriented approach. In *In the proceedings of the ACIS 4th International Conference on software Engineering Artificial Intelligence, Networking and Parallel/distributed Computing (SNPD'03)*, pp. 160–165, Mt. Pleasant, MI, USA, 2003. International Association for Computer and Information Science.

[174] Matthew J. Moyer and Mustaque Ahamad. Generalized role-based access control. *Distributed Computing Systems, International Conference on*, 0:391–398, 2001.

[175] MySQL database. MySQL. http://www.mysql.com/.

[176] Salma Najar, Oumaima Saidani, Manuele Kirsch-Pinheiro, Carine Souveyet, and Selmin Nurcan. Semantic representation of context models: a framework for analyzing and understanding. In *CIAO '09: Proceedings of the 1st Workshop on Context, Information and Ontologies*, pp. 1–10, New York, NY, USA, 2009. ACM.

[177] Michael J. Nash and Keith R. Poland. Some conundrums concerning separation of duty. *Security and Privacy, IEEE Symposium on*, 0:201, 1990.

[178] Gleb Naumovich and Paolina Centonze. Static analysis of role-based access control in J2EE applications. *SIGSOFT Softw. Eng. Notes*, 29(5):1–10, 2004.

[179] Gustaf Neumann and Mark Strembeck. Design and implementation of a flexible RBAC-service in an object-oriented scripting language. In *CCS '01: Proceedings of the 8th ACM conference on Computer and Communications Security*, pp. 58–67, New York, NY, USA, 2001. ACM.

[180] Gustaf Neumann and Mark Strembeck. A scenario-driven role engineering process for functional RBAC roles. In *SACMAT '02: Proceedings of the seventh ACM symposium on Access control models and technologies*, pp. 33–42, New York, NY, USA, 2002. ACM.

[181] Gustaf Neumann and Mark Strembeck. An approach to engineer and enforce context constraints in an RBAC environment. In *SACMAT '03: Proceedings of the 8th ACM symposium on Access control models and technologies*, pp. 65–79, New York, NY, USA, 2003. ACM.

[182] Qun Ni, Alberto Trombetta, Elisa Bertino, and Jorge Lobo. Privacy-aware role based access control. In *SACMAT '07: Proceedings of the 12th ACM symposium on Access control models and technologies*, pp. 41–50, New York, NY, USA, 2007. ACM.

[183] OASIS. Core and hierarchical role based access control (RBAC) profile of XACML v2.0. OASIS specification, 2005. http://www.oasis-open.org/committees/xacml/.

[184] OASIS. Security assertion markup language, version 2.0. OASIS specification, 2005. http://saml.xml.org/.

[185] OASIS. XACML 2.0 core: extensible access control markup language (XACML) version 2.0. OASIS specification, 2005. http://www.oasis-open.org/committees/xacml.

[186] OASIS. Web services business process execution language (WSBPEL) version 2.0. OASIS specification, 2007. www.oasis-open.org/committees/wsbpel.

[187] Department of Defense National Computer Security Center. Trusted computer system evaluation criteria (Orange book), 1985. 5200.28-STD.

[188] United State Department of Health. Health insurance portability and accountability act of 1996. US Act, 1996. http://www.hhs.gov/ocr/hipaa.

[189] Sejong Oh and Seog Park. Task-role-based access control model. In *DEXEA '00, Proceedings of the 9th International Workshop on Database and Expert Systems Applications*, pp. 264–273, Berlin, DE, 2000. Springer.

[190] Sejong Oh and Seog Park. An integration model of role-based access control and activity based access control using task. In *Proceedings of the IFIP TC11/ WG11.3 Fourteenth Annual Working Conference on Database Security*, pp. 355–360, Deventer, The Netherlands, The Netherlands, 2001. Kluwer, B.V.

[191] Sejong Oh and Seog Park. Task-role-based access control model. *Information Systems*, 28(6):533–562, 2003.

[192] Sejong Oh and Ravi Sandhu. A model for role administration using organization structure. In *SACMAT '02: Proceedings of the seventh ACM symposium on Access control models and technologies*, pp. 155–162, New York, NY, USA, 2002. ACM.

[193] U.S. Senate Committee on Banking Housing and Urban Affairs. Information regarding the gramm-leach-bliley act. US Act, 1999. http://banking.senate.gov/conf/.

[194] Oracle. Oracle documentation. www.oracle.com/technology/documentation.

[195] Web Ontology Language (OWL). W3C. http://www.w3.org/2004/OWL/.

[196] Leon Pan and Chang N. Zhang. A criterion-based role-based multilayer access control model for multimedia applications. In *ISM '06: Proceedings of the Eighth IEEE International Symposium on Multimedia*, pp. 145–152, Washington, DC, USA, 2006. IEEE Computer Society.

[197] Jaehong Park and Ravi Sandhu. The UCONABC usage control model. *ACM Trans. Inf. Syst. Secur.*, 7(1):128–174, 2004.

[198] Joon S. Park, Ravi Sandhu, and Gail-Joon Ahn. Role-based access control on the web. *ACM Trans. Inf. Syst. Secur.*, 4(1):37–71, 2001.

[199] Mi Park, Mi Sug Gu, and Keun Ho Ryu. Context information model using ontologies and rules based on spatial object. In *Advanced Intelligent Computing Theories and Applications. With Aspects of Contemporary Intelligent Computing Techniques*, volume 2, pp. 107–114. Springer Berlin Heidelberg, 2007.

[200] Pellet: OWL 2 reasoner for Java. Clark and Parsia. http://clarkparsia.com/pellet/.

[201] Huanchun Peng, Jun Gu, and Xiaojun Ye. Dynamic purpose-based access control. In *ISPA '08: Proceedings of the 2008 IEEE International Symposium on Parallel and Distributed Processing with Applications*, pp. 695–700, Washington, DC, USA, 2008. IEEE Computer Society.

[202] Stephen Perelson, Reinhardt Botha, and Jan Eloff. Separation of duty administration. *SACJ/SART - South African Computer Journal*, 1(27):64–69, 2001.

[203] Stephen Perelson and Reinhardt A. Botha. Conflict analysis as a means of enforcing static separation of duty requirements in workflow environments. *South African Computer Journal*, pp. 212–216, 2000.

[204] Charles Eric Pigeot, Yann Gripay, Marian Scuturici, and Jean Marc Pierson. Context-sensitive security framework for pervasive environments. In *ECUMN '07: Proceedings of the Fourth European Conference on Universal Multiservice Networks*, pp. 391–400, Washington, DC, USA, 2007. IEEE Computer Society.

[205] Smithi Piromruen and James B. D. Joshi. An RBAC framework for time constrained secure interoperation in multi-domain environments. In *WORDS '05: Proceedings of the 10th IEEE International Workshop on Object-Oriented Real-Time Dependable Systems*, pp. 36–48, Washington, DC, USA, 2005. IEEE Computer Society.

[206] Marco Pistoia, Nataraj Nagaratnam, Larry Koved, and Anthony Nadalin. *Enterprise Java 2 security: building secure and robust J2EE applications*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 2004.

[207] Naizhen Qi, Michiharu Kudo, Jussi Myllymaki, and Hamid Pirahesh. A function-based access control model for XML databases. In *CIKM '05: Proceedings of the 14th ACM international conference on Information and knowledge management*, pp. 115–122, New York, NY, USA, 2005. ACM.

[208] Xiaolei Qian and Teresa F. Lunt. A MAC policy framework for multilevel relational databases. *IEEE Trans. on Knowl. and Data Eng.*, 8(1):3–15, 1996.

[209] Indrakshi Ray and Manachai Toahchoodee. A spatio-temporal role-based access control model. In *In Proceedings of the 21st Annual IFIP WG 11.3 Working Conference on Data and Applications Security*, pp. 211–226, Berlin, Heidelberg, 2007. Springer-Verlag.

[210] Indrakshi Ray and Manachai Toahchoodee. A spatio-temporal access control model supporting delegation for pervasive computing applications. In *TrustBus '08: Proceedings of the 5th international conference on Trust, Privacy and Security in Digital Business*, pp. 48–58, Berlin, Heidelberg, 2008. Springer-Verlag.

[211] Rosslin John Robles, Min-Kyu Choi, Sang-Soo Yeo, and Tai-hoon Kim. Application of role-based access control for web environment. In *UMC '08: Proceedings of the 2008 International Symposium on Ubiquitous Multimedia Computing*, pp. 171–174, Washington, DC, USA, 2008. IEEE Computer Society.

[212] Patrick Roder, Omid Tafreschi, and Claudia Eckert. History-based access control for XML documents. In *ASIACCS '07: Proceedings of the 2nd ACM symposium on Information, computer and communications security*, pp. 386–388, New York, NY, USA, 2007. ACM.

[213] Haio Roeckle, Gerhard Schimpf, and Rupert Weidinger. Process-oriented approach for role-finding to implement role-based security administration in a large industrial organization. In *RBAC '00: Proceedings of the fifth ACM workshop on Role-based access control*, pp. 103–110, New York, NY, USA, 2000. ACM.

[214] James Rumbaugh, Ivar Jacobson, and Grady Booch. *The unified modeling language reference manual*. Addison Wesley, 1999.

[215] Arjmand Samuel, Arif Ghafoor, and Elisa Bertino. Context-aware adaptation of access-control policies. *IEEE Internet Computing*, 12(1):51–54, 2008.

[216] Ravi Sandhu. Role activation hierarchies. In *RBAC '98: Proceedings of the third ACM workshop on Role-based access control*, pp. 33–40, New York, NY, USA, 1998. ACM.

[217] Ravi Sandhu, Venkata Bhamidipati, and Qamar Munawer. The ARBAC97 model for role-based administration of roles. *ACM Trans. Inf. Syst. Secur.*, 2(1):105–135, 1999.

[218] Ravi Sandhu and Qamar Munawer. The ARBAC99 model for administration of roles. In *ACSAC '99: Proceedings of the 15th Annual Computer Security Applications Conference*, p. 229, Washington, DC, USA, 1999. IEEE Computer Society.

[219] Ravi S Sandhu. Transaction control expressions for separation of duties. In *In Fourth Annual Computer Security Application Conference*, pp. 282–286, 1988.

[220] Bill Schilit, Norman Adams, and Roy Want. Context-aware computing applications. In *Proc of IEEE Workshop on Mobile Computing Systems and Applications*, pp. 85–91, Washington, DC, USA, 1994. IEEE Computer Society.

[221] Basit Shafiq, James B. D. Joshi, Elisa Bertino, and Arif Ghafoor. Secure interoperation in a multidomain environment employing RBAC policies. *IEEE Trans. on Knowl. and Data Eng.*, 17(11):1557–1577, 2005.

[222] Basit Shafiq, Arjmand Samuel, Elisa Bertino, and Arif Ghafoor. Technique for optimal adaptation of time-dependent workflows with security constraints. In *ICDE '06: Proceedings of the 22nd International Conference on Data Engineering*, pp. 119–121, Washington, DC, USA, 2006. IEEE Computer Society.

[223] Basit Shafiq, Arjmand Samuel, and Halima Ghafoor. A GTRBAC based system for dynamic workflow composition and management. In *Proceedings of the 8th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC)*, pp. 284–290, Los Alamitos, CA, USA, 2005. IEEE Computer Society.

[224] Chaowang Shang, Zongkai Yang, Qingtang Liu, and Chengling Zhao. A context based dynamic access control model for web service. In *International Conference on Embedded and Ubiquitous Computing, IEEE/IFIP*, volume 2, pp. 339–343, Los Alamitos, CA, USA, 2008. IEEE Computer Society.

[225] Won Bo Shim and Seog Park. Implementing web access control system for the multiple web servers in the same domain using RBAC concept. In *ICPADS '01: Proceedings of the Eighth International Conference on Parallel and Distributed Systems*, pp. 768–773, Washington, DC, USA, 2001. IEEE Computer Society.

[226] Dongwan Shin, Gail-Joon Ahn, Sangrae Cho, and Seunghun Jin. On modeling system-centric information for role engineering. In *SACMAT '03: Proceedings of the eighth ACM symposium on Access control models and technologies*, pp. 169–178, New York, NY, USA, 2003. ACM.

[227] Richard Simon and Mary Ellen Zurko. Separation of duty in role-based environments. In *CSFW '97: Proceedings of the 10th IEEE workshop on Computer Security Foundations*, p. 183, Washington, DC, USA, 1997. IEEE Computer Society.

[228] Goran Sladić. Kontrola pristupa XML dokumentima zasnovana na korisničkim ulogama. Master's thesis, Fakultet tehničkih nauka, Novi Sad, 2006.

[229] Goran Sladić. *Kontrola pristupa XML dokumentima.* Zadužbina Andrejevic, Beograd, Srbija, 2008.

[230] Goran Sladić, Branko Milosavljević, and Zora Konjović. Kontrola pristupa XML dokumentima. *Info M*, 15-16:53–60, 2005.

[231] Goran Sladić, Branko Milosavljević, and Zora Konjović. Extensible Access Control Model for XML Document Collections. In *ICETE SECRYPT: Proceedings of the 2nd International Conference on Security and Cryptography*, pp. 373–380. INSTICC, 2007.

[232] Goran Sladić, Branko Milosavljević, and Zora Konjović. Implementacija kontrole pristupa u workflow sistemima baziranih na dokumentima uz oslonac na XXACF sistem. In *Zbornik radova YU Info 2007*, p. 6, Beograd, Srbija, 2007. Informaciono društvo Srbije.

[233] Goran Sladić, Branko Milosavljević, and Gordana Milosavljević. Kontrola pristupa dokumentima bazirana na XACML standardu. In *Zbornik radova YUInfo 2010*, p. 6, Beograd, Srbija, 2010. Informaciono društvo Srbije.

[234] Goran Sladić, Branko Milosavljević, Dušan Surla, and Zora Konjović. Flexible Access Control Framework for MARC Records. *The Electronic Library*, X:25p, 201X (*accepted for publication*).

[235] Karsten Sohr, Michael Drouineaud, Gail-Joon Ahn, and Martin Gogolla. Analyzing and managing role-based access control policies. *IEEE Trans. on Knowl. and Data Eng.*, 20(7):924–939, 2008.

[236] SPARQL query language for RDF. W3C. http://www.w3.org/TR/rdf-sparql-query/.

[237] Spring framework. Spring Source Community. http://www.springsource.org/.

[238] Scott D. Stoller, Ping Yang, C R. Ramakrishnan, and Mikhail I. Gofman. Efficient policy analysis for administrative role based access control. In *CCS '07: Proceedings of the 14th ACM conference on Computer and communications security*, pp. 445–455, New York, NY, USA, 2007. ACM.

[239] Thomas Strang and Claudia Linnhoff-Popien. A context modeling survey. In *In: Workshop on Advanced Context Modelling, Reasoning and Management, UbiComp 2004 - The Sixth International Conference on Ubiquitous Computing, Nottingham/England*, 2004.

[240] Mark Strembeck and Gustaf Neumann. An integrated approach to engineer and enforce context constraints in RBAC environments. *ACM Trans. Inf. Syst. Secur.*, 7(3):392–427, 2004.

[241] Yuqing Sun, Xiangxu Meng, and Fang Yin. A novel approach for role hierarchies in flexible RBAC workflow. In *EDOC '06: Proceedings of the 10th IEEE International Enterprise Distributed Object Computing Conference*, pp. 488–491, Washington, DC, USA, 2006. IEEE Computer Society.

[242] Yuqing Sun and Peng Pan. PRES: a practical flexible RBAC workflow system. In *ICEC '05: Proceedings of the 7th international conference on Electronic commerce*, pp. 653–658, New York, NY, USA, 2005. ACM.

[243] Hassan Takabi, Morteza Amini, and Rasool Jalili. Separation of duty in role-based access control model through fuzzy relations. In *IAS '07: Proceedings of the Third International Symposium on Information Assurance and Security*, pp. 125–130, Washington, DC, USA, 2007. IEEE Computer Society.

[244] Hassan Takabi, Morteza Amini, and Rasool Jalili. Trust-based user-role assignment in role-based access control. In *ACS/IEEE International Conference on Computer Systems and Applications*, volume 0, pp. 807–814, Los Alamitos, CA, USA, 2007. IEEE Computer Society.

[245] Fakultet tehničkih nauka. Postupak izboran u nastavna zvanja. Dokument sistema kvaliteta, 2007.

[246] Roshan K. Thomas. Team-based access control (TMAC): a primitive for applying role-based access controls in collaborative environments. In *RBAC '97: Proceedings of the second ACM workshop on Role-based access control*, pp. 13–19, New York, NY, USA, 1997. ACM.

[247] Roshan K. Thomas and Ravi S. Sandhu. Task-based authorization controls (TBAC): a family of models for active and enterprise-oriented autorization management. In *Proceedings of the IFIP TC11 WG11.3 11th International Conference on Database Securty XI*, pp. 166–181, London, UK, UK, 1998. Chapman & Hall, Ltd.

[248] Bhavani Thuraisingham. Security issues for federated database systems. *Comput. Secur.*, 13(6):509–525, 1994.

[249] Bhavani Thuraisingham. Data warehousing, data mining, and security. In *Database Security Conference*, 1996.

[250] Bhavani Thuraisingham and Srinivasan Iyer. Extended RBAC - based design and implementation for a secure data warehouse. In *ARES '07: Proceedings of the The Second International Conference on Availability, Reliability and Security*, pp. 821–828. IEEE Computer Society, 2007.

[251] Manachai Toahchoodee, Indrakshi Ray, Kyriakos Anastasakis, Geri Georg, and Behzad Bordbar. Ensuring spatio-temporal access control for real-world applications. In *SACMAT '09: Proceedings of the 14th ACM symposium on Access control models and technologies*, pp. 13–22, New York, NY, USA, 2009. ACM.

[252] Apache Tomcat. The Apache Software Foundation. http://tomcat.apache.org/.

[253] Anand R. Tripathi, Devdatta Kulkarni, and Tanvir Ahmed. A specification model for context-based collaborative applications. *Pervasive Mob. Comput.*, 1(1):21–42, 2005.

[254] Khai N. Truong, Gregory D. Abowd, , and Jason A. Brotherton. Who, what, when, where, how: design issues of capture & access applications. In *Ubicomp 2001: Ubiquitous Computing*, pp. 209–224, New York, NY, USA, 2001. Springer-Verlag.

[255] Sofia K. Tzelepi, Dimitrios K. Koukopoulos, and George Pangalos. A flexible content and context-based access control model for multimedia medical image database systems. In *MM and Sec '01: Proceedings of the 2001 workshop on Multimedia and security*, pp. 52–55, New York, NY, USA, 2001. ACM.

[256] Jaideep Vaidya, Vijayalakshmi Atluri, Qi Guo, and Nabil Adam. Migrating to optimal RBAC with minimal perturbation. In *SACMAT '08: Proceedings of the 13th ACM symposium on Access control models and technologies*, pp. 11–20, New York, NY, USA, 2008. ACM.

[257] Wil van der Aalst and Kees van Hee. *Workflow management: models, methods, and systems*. The MIT Press, 2002.

[258] Jacques Wainer, Paulo Barthelmess, and Akhil Kumar. W-RBAC - a workflow security model incorporating controlled overriding of constraints. *International Journal of Cooperative Information Systems*, 12(4):455–485, 2003.

[259] Baoyi Wang and Shaomin Zhang. An organization and task based access control model for workflow system. In *Advances in Web and Network Technologies, and Information Management*, pp. 485–490, Berlin, DE, 2007. SpringerLink.

[260] Xiao Hang Wang, Da Qing Zhang, Tao Gu, and Hung Keng Pung. Ontology based context modeling and reasoning using OWL. In *PERCOMW '04: Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops*, p. 18, Washington, DC, USA, 2004. IEEE Computer Society.

[261] Xin Wang, Yanchun Zhang, and Hao Shi. Access control for human tasks in service oriented architecture. In *ICEBE '08: Proceedings of the 2008 IEEE*

International Conference on e-Business Engineering, pp. 455–460, Washington, DC, USA, 2008. IEEE Computer Society.

[262] Jos Warmer and Anneke Kleppe. *The object constraint language: getting your models ready for MDA*. Addison Wesley, Second edition edition, 2003.

[263] Janice Warner and Vijayalakshmi Atluri. Inter-instance authorization constraints for secure workflow management. In *SACMAT '06: Proceedings of the eleventh ACM symposium on Access control models and technologies*, pp. 190–199, New York, NY, USA, 2006. ACM.

[264] Ruben Wolf, Thomas Keinz, and Markus Schneider. A model for context-dependent access control for web-based services with role-based approach. *Proceedings of the 14th IEEE International Workshop on Database and Expert Systems Applications (DEXA)*, 00:209–214, 2003.

[265] Roosdiana Wonohoesodo and Zahir Tari. A role based access control for web services. In *SCC '04: Proceedings of the 2004 IEEE International Conference on Services Computing*, pp. 49–56, Washington, DC, USA, 2004. IEEE Computer Society.

[266] IETF-PKIX working group. Public key infrastructure. IETF, 1995. http://www.ietf.org/dyn/wg/charter/pkix-charter.html.

[267] Wei Xu, Jun Wei, Yu Liu, and Jing Li. SOWAC: a service-oriented workflow access control model. In *COMPSAC '04: Proceedings of the 28th Annual International Computer Software and Applications Conference*, pp. 128–134, Washington, DC, USA, 2004. IEEE Computer Society.

[268] Hanbing Yao, Heping Hu, Baohua Huang, and Ruixuan Li. Dynamic role and context-based access control for grid applications. In *Proceedings of the 6th International Conference on Parallel and Distributed Computing Applications and Technologies (PDCAT)*, pp. 404–406, Washington, DC, USA, 2005. IEEE Computer Society.

[269] Lin Yao, Xiangwei Kong, and Zichuan Xu. A task-role based access control model with multi-constraints. In *NCM '08: Proceedings of the 2008 Fourth International Conference on Networked Computing and Advanced Information Management*, pp. 137–143, Washington, DC, USA, 2008. IEEE Computer Society.

[270] Walt Yao, Ken Moody, and Jean Bacon. A model of OASIS role-based access control and its support for active security. In *SACMAT '01: Proceedings of the sixth ACM symposium on Access control models and technologies*, pp. 171–181, New York, NY, USA, 2001. ACM.

[271] S.S. Yau, Yisheng Yao, and V. Banga. Situation-aware access control for service-oriented autonomous decentralized systems. In *Proceedings otf the Autonomous Decentralized Systems, 2005. ISADS*, pp. 17–24, April 2005.

[272] Dana Zhang, Kotagiri Ramamohanarao, and Tim Ebringer. Role engineering using graph optimisation. In *SACMAT '07: Proceedings of the 12th ACM symposium on Access control models and technologies*, pp. 139–144, New York, NY, USA, 2007. ACM.

[273] Guangsen Zhang and Manish Parashar. Dynamic context-aware access control for grid applications. In *Proceedings of the 4th International Workshop on Grid Computing (GRID)*, pp. 101–108, Washington, DC, USA, 2003. IEEE Computer Society.

[274] Li Zhang, Lili Luo, Liyong Zhang, Tiesuo Geng, and Zongge Yue. Task-role-based access control in application on MIS. In *Proceedings of the 2006 IEEE Asia-Pacific Conference on Services Computing (APSCC)*, pp. 153–159, Washington, DC, USA, 2006. IEEE Computer Society.

[275] Wendong Zhang and Kaiji Zhang. A role-based workflow access control model. In *ETCS '09: Proceedings of the 2009 First International Workshop on Education Technology and Computer Science*, pp. 1136–1139, Washington, DC, USA, 2009. IEEE Computer Society.

[276] Xinwen Zhang, Francesco Parisi-Presicce, Ravi Sandhu, and Jaehong Park. Formal model and policy specification of usage control. *ACM Trans. Inf. Syst. Secur.*, 8(4):351–387, 2005.

[277] Yue Zhang and James B. D. Joshi. SARBAC07: a scoped administration model for RBAC with hybrid hierarchy. In *IAS '07: Proceedings of the Third International Symposium on Information Assurance and Security*, pp. 149–154, Washington, DC, USA, 2007. IEEE Computer Society.

[278] Gansen Zhao, David Chadwick, and Sassa Otenko. Obligations for role based access control. In *AINAW '07: Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops*, pp. 424–431, Washington, DC, USA, 2007. IEEE Computer Society.

[279] Hui Zhao, Zhiyi Fang, Peng Xu, Lianyu Zhao, Jin Liu, and Tianyang Wang. An improved role-based workflow access control model. *Information Technology: New Generations, Third International Conference on*, 0:551–556, 2008.

Biografija

Goran Sladić je rođen 19.02.1979. godine u Kninu. Srednju elektrotehničku školu "Mihajlo Pupin" završio je u Novom Sadu sa odličnim uspehom. Na odsek za Elektrotehniku i računarstvo Fakulteta tehničkih nauka u Novom Sadu upisao se 1997. godine. Studije na smeru Računarstvo i upravljanje sistemima, usmerenje Računarske nauke i informatika, završio je sa prosečnom ocenom 8,46. Diplomski rad odbranio je 30.09.2002. godine iz predmeta "Veštačka inteligencija" sa ocenom 10. Tema diplomskog rada je bila "Arhitektura i komponente bezbednosnog segmenta hijerarhijskog agentskog okruženja".

Na poslediplomske studije Fakulteta tehničkih nauka – smer Računarstvo i automatika, usmerenje Računarske nauke i informatika, upisao se školske 2002/03. godine i položio sve ispite predviđene planom i programom sa prosečnom ocenom 10. Magistarski rad pod nazivom "Proširivi sistem za kontrolu pristupa XML dokumentima zasnovanu na korisničkim ulogama" odbranio je 07.09.2006. godine.

U periodu od 2002. do 2004. bio je angažovan na projektima i u nastavi na Fakultetu tehničkih nauka. U zvanje asistenta-pripravnika na Fakultetu tehničkih nauka izabran je 2004. godine, a u zvanje asistenta 2007. Trenutno drži vežbe iz predmeta "Sistemi elektronskog plaćanja" na master studijama i vežbe iz predmeta "XML i Web servisi", "Bezbednost u sistemima elektronskog poslovanja" na osnovnim akademskim studijama. U toku studija i rada na Fakultetu objavio je 29 radova, jednu monografiju i koautor je dva softverska projekta, sa ukupnim koeficijentom kompetencije $M = 38,5$. Pored toga, učestvovao na 11 naučnih i stručnih projekata. Oblasti interesovanja su bezbednost, upravljanje dokumentima, XML tehnologije i elektronsko poslovanje.

Oženjen je, ima jednog sina i živi u Novom Sadu. Od stranih jezika govori engleski jezik.

Univerzitet u Novom Sadu
Fakultet tehničkih nauka

Ključna dokumentacijska informacija

Redni broj:
RBR

Identifikacioni broj:
IBR

Tip dokumentacije: TD	monografska dokumentacija
Tip zapisa: TZ	tekstualni štampani dokument
Vrsta rada: VR	doktorska disertacija
Autor: AU	mr Goran Sladić
Mentor: MN	prof. dr Branko Milosavljević, vanr. prof.
Naslov rada: NR	Model kontekstno zavisne kontrole pristupa u poslovnim sistemima
Jezik publikacije: JP	srpski (latinica)
Jezik izvoda: JI	srpski (latinica) / engleski
Zemlja publikovanja: ZP	Srbija

Uže geografsko područje: Vojvodina
UGP

Godina: 2011
GO

Izdavač: autorski reprint
IZ

Mesto i adresa: Fakultet tehničkih nauka,
MA Trg. D. Obradovića 6, Novi Sad

Fizički opis rada: *broj poglavlja/strana/referenci/ tabela/slika/grafikona/priloga*
FO 6/208/279/25/24/0/0

Naučna oblast: računarske nauke i informatika
NO

Naučna disciplina: bezbednost informacionih sistema
ND

Ključne reči: kontrola pristupa, kontekst, RBAC
PO poslovni tokovi

UDK:

Čuva se: Biblioteka Fakulteta tehničkih nauka,
ČU Trg D. Obradovića 6, Novi Sad

Važna napomena:
VN

Datum prihvatanja od strane NN veća:
DP

Datum odbrane:
DO

Izvod:
IZ

Kontrola pristupa odnosno autorizacija, u širem smislu, razmatra na koji način korisnici mogu pristupiti resursima računarskog sistema i na koji način ih koristiti. Ova disertacija se bavi problemima kontrole pristupa u poslovnim sistemima. Tema disertacije je formalna specifikacija modela kontekstno zavisne kontrole pristupa u poslovnim sistemima koji je baziran na RBAC modelu kontrole pristupa. Uvođenjem kontekstno zavisne kontrole pristupa omogućeno je definisanje složenijih prava pristupa koje u postojećim modelima kontrole pristupa za poslovne sisteme nije bilo moguće realizovati ili bi njihova realizacija bila komplikovana. Dati model primenljiv je u različitim poslovnim sistemima, a podržava definisanje prava pristupa kako za jednostavne tako i za složene poslovne tokove. Sistem je verifikovan na dva realna poslovna procesa pomoću razvijenog prototipa. Prikazana prototipska implementacija koja ispunjava ciljeve u pogledu funkcionalnosti postavljene pred sistem predstavlja potvrdu praktične vrednosti predloženog modela.

Komisija:
KO

predsednik: prof. dr Dušan Surla, prof. emeritus,
Prirodno-matematički fakultet, Novi Sad

član: prof. dr Zora Konjović, red. prof.,
Fakultet tehničkih nauka, Novi Sad

član: prof. dr Milan Milosavljević, red. prof.,
Elektrotehnički fakultet, Beograd

član: prof. dr Milan Vidaković, vanr. prof.,
Fakultet tehničkih nauka, Novi Sad

član: prof. dr Branko Milosavljević, vanr. prof., mentor
Fakultet tehničkih nauka, Novi Sad

University of Novi Sad
Faculty of Technical Sciences

Keyword Documentation

Accession number:
ANO

Identification number:
INO

Document type: DT	monograph publication
type of record: TR	textual printed material
Contents code: CC	doctoral dissertation
Author: AU	Goran Sladić, MSc
Menthor: MN	Branko Milosavljević, PhD, assoc. prof.
Title: TI	Context Sensitive Access Control Model for Business Processes
Lanugage of text: LT	Serbian (latin)
Lanugage of abstract: LA	Serbian (latin) / English
Country of publication: CP	Serbia

Locality of publication: Vojvodina
LP

Publication year: 2011
PY

Publisher: author's reprint
PU

Publishing place: Faculty of Technical Sciences,
PP Trg. D. Obradovića 6, Novi Sad

Physical description: *num. of chapters/pages/references/ tables/pictures/graphs/appendix*
PD 6/208/279/25/24/0/0

Scientific field: computer sciences and informatics
SF

Scientific discipline: information security
SD

Subject/keywords: access control, context, RBAC, workflow
SKW

UDC:

Holding data: Library of Faculty of Technical Sciences,
HD Trg D. Obradovića 6, Novi Sad

Note:
N

Accepted by scientific board on:
ASB

Defended:
DE

Abstract: Access control is concerned with the way in which users can access to
AB resources in the computer system. This dissertation focuses on problems of access control for business processes. The subject of the dissertation is a formal specification of the RBAC-based context sensitive access control model for business processes. By using a context-sensitive access control it is possible to define more complex access control policies whose implementation in existing access control models for business processes is not possible or is very complicated. The given model is applicable in different business systems, and supports the definition of access control policies for both simple and complex business processes. The model's prototype is verified by two case studies on real business processes. The presented prototype implementation represents a proof of the proposed model's practical value.

Thesis defend board:
DB

president: prof. Dušan Surla, PhD,
Faculty of Science, Novi Sad
member: prof. Zora Konjović, PhD,
Faculty of Technical Sciences, Novi Sad
member: prof. Milan Milosavljević, PhD,
Faculty of Electrical Engineering, Belgrade
member: prof. Milan Vidaković, PhD,
Faculty of Technical Sciences, Novi Sad
member: prof. Branko Milosavljević, PhD, menthor
Faculty of Technical Sciences, Novi Sad