Slovenská technická univerzita
Fakulta elektrotechniky a informatiky

Evidenčné číslo: FEI-5382-72557

Workflow management rolí a užívateľov

Bakalárska práca

2016
Pavol Martiš

Slovenská technická univerzita v Bratislave
Fakulta elektrotechniky a informatiky

Evidenčné číslo: FEI-5382-72557

Workflow management rolí a užívateľov

Bakalárska práca

Študijný program:	Aplikovaná informatika
Študijný odbor:	9.2.9 aplikovaná informatika
Školiace pracovisko:	Ústav informatiky a matematiky
Školiteľ:	prof. RNDr. Gabriel Juhás, PhD.

Bratislava, 2016
Pavol Martiš

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE

Fakulta elektrotechniky a informatiky

Evidenčné číslo: FEI-5382-72557

Workflow manažment systém – server manažmentu rolí a užívateľov

Bakalárska práca

Študijný program: Aplikovaná informatika
Študijný odbor: 9.2.9. aplikovaná informatika
Školiace pracovisko: Ústav informatiky a matematiky
Vedúci záverečnej práce: prof. RNDr. Gabriel Juhás, PhD.

Bratislava 2016 **Pavol Martiš**

Poďakovanie:

Abstrakt

Slovenský abstrakt v rozsahu 100-500 slov, jeden odstavec. Abstrakt stručne sumarizuje výsledky práce. Mal by byť pochopiteľný pre bežného informatika. Nemal by teda využívať skratky, termíny alebo označenie zavedené v práci, okrem tých, ktoré sú všeobecne známe.

Kľúčové slová: užívateľ, rola , workflow, Petriho sieť, RBAC , workflow

Abstract

Abstract in the English language (translation of the abstract in the Slovak language).

Keywords:

Obsah

Zoznam skratiek

WFMS – Workflow management system
RBAC – Role-based access control
CSS - Cascading Style Sheets
XML - Extensible Markup Language

Zoznam obrázkov

Úvod

Systém vývoja aplikačných programov sa veľmi rýchlo mení. So stúpajúcou zložitosťou počítačových systémov a zvyšovaním počtu užívateľov sa vynára potreba rozložiť aplikácie do viacerých navzájom nezávislých modulov. Najprv sa vytvorili databázy , ktoré umožnili oddeliť dáta od aplikácie. Následne sa v 80-ych rokoch analogickým spôsobom oddelilo užívateľské rozhranie od samotnej aplikácie. Vznikla tak MVC (model-view-controller) architektúra. Ďalším stupňom vývoja je potreba oddeliť všeobecnú funkcionalitu od aplikácie. Odčlení sa tak procesná, aplikačná a dátová časť. Túto myšlienku poskytuje workflow management systém (WFMS).

Workflow management systém umožňuje ľahko oddeliť procesy od vizuálnej a dátovej časti, vďaka čomu je upravovanie a znovupoužitie daných procesov jednoduchšie. Je možné si teda predstaviť, že dve rôzne firmy v rovnakej oblasti by využívali rovnaké procesy, pričom by mali osobitnú databázovú aj vizuálnu stránku.

Dôležitou otázkou pri tvorbe workflow management systémou je správa zdrojov. Treba určiť kto môže a naopak kto nesmie pristupovať ku konkrétnym prostriedkom, aby sa zachovala dôvernosť informácii a nemohol byť narušený systém. Vo WFSM je najpoužívanejší spôsob správa zdrojov na základe systému rolí Role-based access control (RBAC). Tento model sa využíva kvôli tomu, že je založený na organizačnej štruktúre podnikov a poskytuje efektívne riadenie a údržbu právomocí. Vo WFSM je potreba oddeliť tvorbu procesov od jej používania. RBAC dokáže zabezpečiť previazanie medzi procesmi a konkrétnymi používateľmi. Okrem toho má RBAC ďaľšie výhody ako napríklad uľahčenie administrácie priradením užívateľov k jednotlivým roliam v porovnaní so správou práv pre každého jednotlivého užívateľa osobitne.

Jedna z možností ako modelovať WFMS je prostredníctvom Petriho sietí. Petriho sieť je matematický nástroj na modelovanie a simulovanie diskrétnych procesov. Hlavné výhody Petriho sietí sú:

- formálna sémantika – proces špecifikovaný Petriho sieťou má precíznu definíciu
- grafické zobrazenie – Petriho sieť je grafický jazyk. Dôsledkom tohto Petriho siete sú intuitívne a ľahko pochopiteľné, preto sú vhodné aj pri komunikácii s koncovými užívateľmi.

- expresivita – Petriho siet podporuje všetky primitíva potrebné pre modelovanie workflow procesov. Keďže stavy sú reprezentované explicitne, umožňujú modelovanie závislostí a implicitné voľby.

- analýza – Umožňujú overiť vlastnosti (bezpečnosť, invariantnosť, deadlocky, atď.) a vyčísliť výkonové merania (čas odozvy, čas čakania, podiel obsadenosti, atď.)

Cieľom tejto práce je na základe štúdia a analýzy workflow managemantu a Petriho sietí vytvoriť modul na správu rolí a užívateľov pre webovú aplikáciu, ktorá bude poskytovať WFMS na základe Petriho sietí.

Bakalárska práca pozostáva z 2 hlavných kapitol. Prvá kapitola je zameraná na analýzu workflow systému, Petriho siete a riadenie prístupu vo WFMS. Kapitola sa bližšie zameriava na možné prístupy

Druhá kapitola je zameraná na opis riešenia celej aplikácie, pričom bližšie rozoberá návrh a implementáciu modulu na priraďovanie rolí k užívateľom a jej nadväznosti k ostatným častiam aplikácie.

Analýza

V biznis sektore je neustála potreba monitorovať a kontrolovať firemné procesy. Čím je podniková štruktúra hlbšia, procesy komplexnejšie a narastá počet zamestnancov, tým stúpa náročnosť riadenia a manažmentu podniku. Komplexné projekty sú len zriedkavo vypracované podľa pôvodného časového plánu. Toto meškanie je často spôsobené zlou koordináciou pracovných procesov. Vo firemných procesoch jednotlivé úlohy za sebou nasledujú v určitom poradí, ktoré musí byť prísne dodržané. Aj meškanie drobnej úlohy dokáže dominovým efektom zastaviť vykonanie nasledovných úloh. V konečnom dôsledku tak malé zaváhanie môže spôsobiť obrovské oneskorenie a finančné straty. Vzniká tak potreba jasne vymedziť jednotlivé kroky práce a minimalizovať administráciu a možné chyby. Riešenie tohto problému je možné riešiť využitím Workflow management systémov. WfMS ponúkajú možnosť explicitne zadefinovať firemné procesy, pričom dokážu automaticky zabezpečiť správnosť vykonávania jednotlivých úloh v korektnom poradí. Vo firemnom prostredí tak nastane poriadok a zjednoduší sa administrácia. Hlavnou nevýhodou WfMS sú náklady na samotnú implementáciu. Väčšina WfMS sa vyvíja pre potreby konkrétneho podniku, pričom náklady na ich naprogramovanie presahujú finančnú kapacitu stredných a drobných firiem. Na trhu sa tak objavujú generické WfMS systémy, ktoré dokážu zabezpečiť využívanie procesov viacerými organizáciami súčasne.

Možným nástrojom na implementovanie WfMS sa naskytuje použitie Petriho sietí. Petriho siete sú jasne formálne a matematicky definované, pričom ich modelovanie je jednoduché a intuitívne. Medzi hlavné výhody môžeme považovať grafické znázornenie, ktoré uľahčuje pochopenie princípov a fungovania tohto modelu. Ak pripojíme fakt, že adaptácia Petriho sietí na WfMS nie je komplikovaná , môžeme tak Petriho siete považovať za vhodný nástroj na modelovanie WfMS.

Petriho siete sú dobrým nástrojom na modelovanie procesov vo WfMS, nedokážu nám však samy o sebe zabezpečiť riadenie prístupových práv. Pre potreby WfMS systémov je najpoužívanejšia metóda riadenia rolí RBAC. Tento model nám ponúka lepšiu alternatívu k bežne používaným systémom DAC (data access control) a MAC (mandatory access control). Zásadný rozdiel medzi RBAC a ostatnými modelmi spočíva v oddelení prístupových práv od samotných používateľov. To zabezpečuje možnosť namodelovať proces nezávisle od používateľskej časti. Okrem toho má mnoho výhôd,

ktoré pramenia z toho, že tento model je zhodný s tým, ktorý je vo firmách prirodzene zaužívaný.

1.1 Workflow management systém

V každom podniku je zaužívaná určitá logistika, nastavenie procesov na základe ktorých prebieha celé riadenie a manažment práce. Cieľom workflow manažmentu (WfM) je zaistiť aby bola táto logistika správne vykonávaná, čo znamená určiť kto má v danom momente spustiť akú úlohu. S narastajúcou veľkosťou podniku alebo zložitosťou procesov, manažment riadenia príde do bodu, kedy je len veľmi ťažko ovládateľný. Naskytuje sa tak potreba, aby sa celý manažment zautomatizoval. Vzniká tak workflow management systém (WfMS).Workflow Management Coalition definoval termín WfMS ako systém, ktorý kompletne určí, manažuje a vykonáva workflow cez softvér, ktorý má určené vykonávanie jednotlivých úloh na základe počítačovej reprezentácie workflow logiky. [9]

Workflow systémy sú založené na jednotlivých prípadoch (cases). Ako jednotlivý prípad si môžeme predstaviť konkrétnu požiadavku, ako je napríklad založenie si účtu, objednávka, spísanie závete a podobne. Jednotlivé prípady sú často krát spúšťané samotnými zákazníkmi, ale nie je to pravidlo. Hoci takýchto prípadov môže vzniknúť v systéme mnoho, ich priebeh je riadený za pomoci jednotnej workflow logiky. Každý prípad je unikátny a má konečnú životnosť. Úloha, ktorá sa vykonáva v konkrétnom prípade sa nazýva pracovaná jednotka (working item). Spôsob vykonávania jednotlivých úloh (pracovných jednotiek) v prípade je určený prostredníctvom definície workflow procesov. [9]

Každý prípad vo WfMS má svoj začiatok a koniec. Medzitým sa proces nachádza v určitom stave, ktorý je definovaný prostredníctvom:

- architektúry daného prípadu - určuje nám ako sa môže daný prípad vyvíjať.
- doposiaľ splnených podmienok - zistíme tak aktuálnu "polohu", v akej sa daný prípad nachádza
- hodnoty atribútov v danom prípade - atribúty nám definujú isté podmienky. Predstavme si prípad šetrenia peňazí do pokladničky. Do pokladničky budeme vkladať peniaze, dokiaľ suma v pokladničke nepresiahne určitú hodnotu. V našom prípade sú atribútom peniaze

1.1.1 Úloha

Úloha je základnou logickou jednotkou workflow systému. Ako taká je nedeliteľná a v systéme sa musí vždy vykonať ako celok. V prípade neúspechu úlohy tak treba úlohu spustiť celú odznova. Je však možné rozlišovať medzi spustením a ukončením úlohy.

1.1.2 Proces

Proces definuje spôsob, akým sa vykonajú určité úlohy v konkrétnom prípade. Proces môže byť vo viacerých prípadoch rovnaký ale môže sa takisto odlišovať. Tieto odlišnosti v procese môžu byť napríklad spôsobené smerovacími konštrukciami, za pomoci ktorých je prípad namodelovaný. V prípade vybavovanie pôžičky môže byť žiadosť pre Ferka zamietnutá, zatiaľ čo taká istá žiadosť bude pre Janka schválená.

1.1.3 Smerovacie konštrukcie

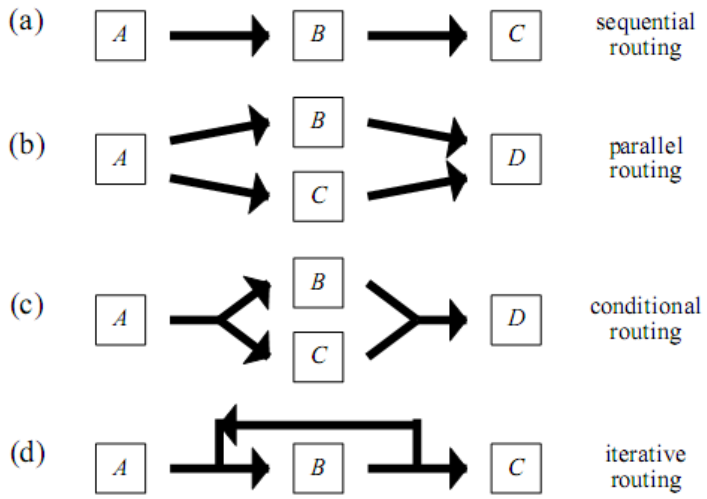
Postupnosť vykonávania úloh v systéme je určená na základe smerovania prípadu. V jednotlivých vetvách určujeme pre každú úlohu smerovacie konštrukcie, čím umožňujeme vytvoriť pre prípad odlišné životné cykly. Využívame štyri základné:

- sekvenčné - úlohy sa vykonávajú za sebou v poradí,v akom nasledujú
- paralelné – úlohy sa vykonávajú paralelne nezávisle na sebe za pomoci AND-splitov a AND-joinov.
- podmienené – vykonávanie úloh závisí od definovaných podmienok. Realizácia sa vykonáva za pomoci OR-splitov a OR-joinov
- iteračné- vykonanie jednej alebo viacerých úloh viackrát za sebou

1.1.4 Výhody použitia WfMS

Workflow management systém ponúka možnosť oddeliť procesnú časť, čím sa zabezpečia nasledovné výhody:

- zabezpečí sa tak jednotná funkcionalita manažmentu a oddelí sa od zvyšku systému. Tradične sa táto funkcionalita rozprestiera v celom systéme. Vďaka tejto separácii je možné rovnakú funkcionalitu použiť na viac ako jednu úlohu.
- aplikácie sú nezávislé od manažmentu , čo umožňuje upravovať logiku aj po implementovaní systému
- k logickej vrstve je možné pridávať rôzne aplikácie a rozšíriť tak funkcionalitu systému



Obr. 1.1: Smerovacie konštrukcie vo workflow systémoch podľa [10]

- definovanie a návrh procesu je možné diagnostikovať a vopred tak zabrániť nežiaducim chybám
- z pohľadu manažmentu je možné v konkrétnom prípade identifikovať, v akom stave sa prípad nachádza. Celý proces sa tak ľahšie monitoruje. Dá sa zistiť, kto má v danom prípade a čase čo vykonávať. Celé vykonávanie procesu je pod drobnohľadom, čím sa urýchli čas vykonania celého procesu. V prípade problému vo vykonávaní prípadu, je možné rýchlo identifikovať zdroj

1.1.5 Riadenie zrojov vo WfMS

Pod pojmom zdroj rozumieme jednotku, ktorá je určená na vykonávanie úlohy. Môžme si pod tým predstaviť počítač (server, tlačiareň, fax ...) alebo človeka ako pracovnú jednotku. V biznis prostredí väčšinu zdrojov tvoria jednotliví pracovníci a zákazníci, nie je to však pravidlo.

Jednotlivé zdroje sa môžu zoskupovať do skupín s podobnou charakteristikou. Ak majú rovnakú funkcionálnu charakteristiku, nazývame túto skupinu rola. Ako príklad roly si môžeme predstaviť akúkoľvek pracovnú pozíciu a pod samotnými pracovnými zdrojmi samotného zamestnanca. Pridelenie právomocí zdrojom sa udeľuje nepriamo pomocou rol. Týmto spôsobom sa oddelí návrh procesu od jednotlivých používateľov. Taktiež sa tým uľahčí administrácia a zabráni sa deadlockom. Ak by sme totiž priradili úlohu ku konkrétnemu zdroju, nastali by problémy po vymazaní daného zdroja a samotný proces by nebolo možné ukončiť.

1.2 Petriho siete

1.2.1 História

Petriho siete vymyslel nemecký matematik Carl Adam Petri už ako 13 ročný na modelovanie chemických procesov. Neskôr ich popísal vo svojej dizertačnej práci "Communication with Automata"[?] v roku 1962. Časom sa však Petriho siete menili a dopĺňali. Dnes už podľa [1] poznáme viacero definícii Petriho sietí.

1.2.2 Všeobecná definícia Petriho siete

Petriho sieť je podľa zvláštny typ orientovaného grafu spolu so začiatočným stavom (začiatočným značkovaním) - PN = (N, M_0). Obsahuje dva typy uzlov nazývanými: *place* (miesto) a *transition* (prechod). Tieto uzly môžu byť spolu prepojené prostredníctvom orientovanej hrany (arc). Spojenie medzi dvoma uzlami rovnakého typu nie je povolené.

Graf Petriho siete obsahuje:

- miesta - označované krúžkom
- prechody - označené obdĺžnikom
- hrany - spájajú jednotlivé uzly. Hrany môžu mať určitú násobnosť. Poznáme tieto typy hrán:
  - hrana z miesta do prechodu
  - hrana z prechodu do miesta
- váha hrany - reprezentuje násobnosť hrany
- značkovanie - priraďuje každému miestu počet značiek (tokenov) - m-rozmerný vektor nezáporných celých M, M(p)- počet značiek v mieste p

Formálna definícia

Pre formálne opísanie Petriho sietí použijeme definíciu z .
Petriho sieť je orientovaný bipartitný graf reprezentovaný päticou (P,T,F,W,m_0), kde:

P = $\{p_1, ..., p_n\}$ - konečná neprázdna množina miest;
T = $\{t_1, ..., t_n\}$ - konečná neprázdna množina prechodov;
$P \cap T = \emptyset, P \cup T \neq \emptyset$;
$F \subseteq (PxT) \cup (TxP)$ - množina hrán (toková relácia);
$W : F \to N \cup \{0\}$ - váhová funkcia ;

$m_0 : P \rightarrow N \cup \{0\}$ - počiatočné značkovanie;

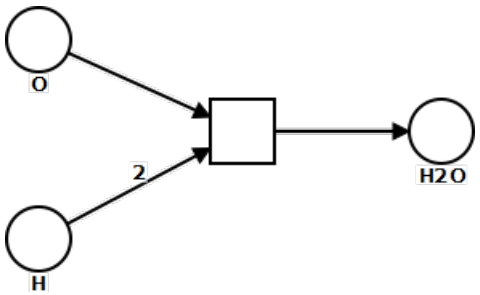
1.2.3 Pravidlá pre spúšťanie prechodov

1. Prechod je aktivovaný , ak každé z jeho vstupných miest obsahuje aspoň w(p,t) značiek

2. Spustenie prechodu t spôsobí zrušenie w(p,t) značiek v každom vstupnom mieste p prechodu t a pridanie w(t,p) značiek do každého výstupného miesta p prechodu t;

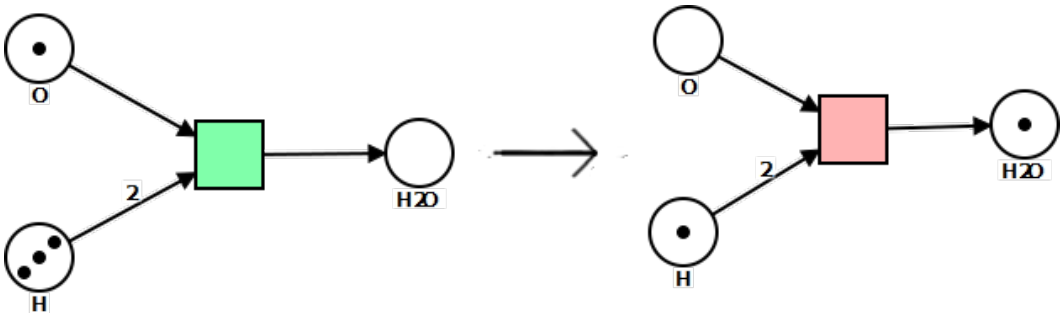
1.2.4 Príklad

Na obrázku 1.2 môžeme vidieť jednoduchú Petriho sieť ktorá znázorňuje spojenie dvoch prvkov vodíka a kyslíka na vytvorenie vody. Máme tri miesta. Dve z nich nám predstavujú jednotlivé prvky a tretie predstavuje molekulu vody H_20. Medzi nimi je prechod ktorý v tejto sieti znázorňuje spojenie dvoch prvkov do jednej molekuly. Medzi miestom O a prechodom máme jednoduchú hranu, zatiaľ čo z miesta H do prechodu máme hranu násobnosti dva. Jednotlivé násobnosti znázorňujú, koľko tokenov (prvkov) treba skonzumovať z každého miesta, aby bolo možné prechod spustiť. Na druhej strane vidíme jednoduchú hranu z prechodu do miesta H_20, ktorá nám značí, že po skonzumovaní značiek z miest O a H sa vytvorí práve jedna značka v mieste H_20 (vyprodukuje sa práve jedna molekula).



Obr. 1.2: Petriho sieť na "vytvorenie" vody

Pre spustenie tohto procesu je potrebné aby bola v mieste "O" minimálne jedna značka a v mieste "H" značiek minimálne dve. Po spustení prechodu sa v miestach vstupujúcich do prechodu odoberie počet tokenov daný násobnostou hrany. Naopak vo výstupnom mieste sa značky podľa násobnosti hrany pridajú pridajú.



Obr. 1.3: Spustenie siete s jedným žetónom v mieste "O" a troma žetónmi v mieste "H"

1.2.5 Implementovanie Petriho sietí na workflow management systém

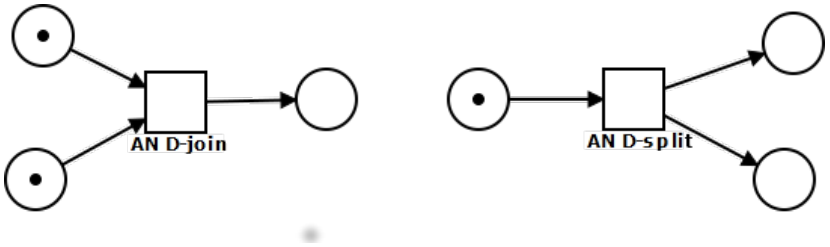
V minulosti sa objavilo niekoľko systémov , ktoré sa snažili implementovať WfMS, ale po čase narazili na problémy, lebo ich architektúra nespĺňala náležité požiadavky. Mnohé z nich nedokázali súčasne implementovať paralelné a alternatívne smerovanie, ostatné narazili na problémy, keď ich aplikácia modelovala WfMS len pomocou udalostí (pri vykonaní úlohy sa systém nedostal do určitého stavu, ale na udalosť nasledovalo okamžité spustenie ďalšej udalosti). W.M.P. van der Aalst v [8] vysvetlil 3 hlavné výhody použitia Petriho sietí na modelovanie workflow management systému:

- formálna sémantika spolu s grafickou reprezentáciu
- stavové modelovanie namiesto udalostného - toto riešenie umožňuje vidieť stav v akom sa proces nachádza a takisto rozlišuje medzi povolením a vykonaním úlohy
- analýza – vďaka tomu, že sú Petriho siete dobre matematicky popísané, umožňujú dobre analyzovať workflow sieť, čo umožní získať štatistické dáta, prípadne zamedziť problémom

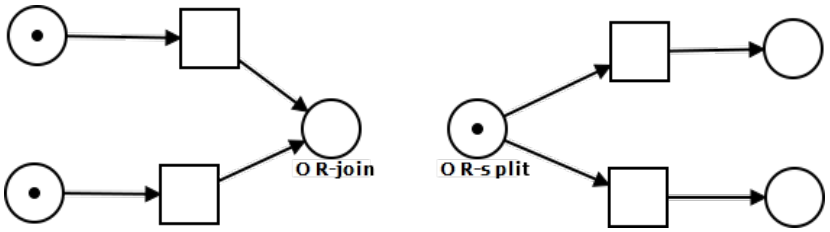
Mapovanie WfMS na Petriho siete je možné ľahko ukázať. Úlohy sú mapované na prechody. Závislosti sa mapujú prostredníctvom miest a hrán. Jednotlivé prípady môžu byť odlíšené rôznou farbou tokenov v rozšírených Petriho sieťach. Stav, v akom sa prípad nachádza definuje rozloženie tokenov Petriho sieť, ktorá modeluje WfMS sa nazýva WorkFlow sieť (WF-sieť). Aby Petriho sieť bola súčasne WF-sieťou, musí spĺňať tieto podmienky

- PN má dve špeciálne miesta: miesto "I" a miesto "O", ktoré reprezentujú počiatočný a koncový stav
- ak pridáme prechod "t" do PN , ktorý spája miesto "I" a "O", potom je PN silno spojená. To znamená, že pre každý pár uzlov X a Y existuje priama cesta od X do Y.

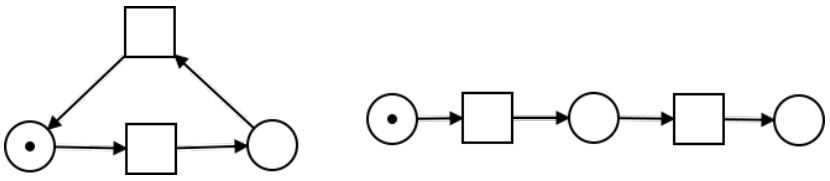
Smerovacie konštrukcie



Obr. 1.4: AND konštrukcia v Petriho sieti



Obr. 1.5: OR konštrukcia v Petriho sieti



Obr. 1.6: Iteračné (vľavo) a sekvenčné(vpravo) konštrukcie v Petriho sieti

1.3 Riadenie prístupu

V nasledovnej sekcii rozoberieme princíp a pravidlá riadenia prístupu podľa [3]. Riadenie prístupu, známe pod pojmom autorizácia je koncept, ktorý používame od čias, kedy ľudia pociťovali potrebu chrániť svoje veci. Hlavnou myšlienkou je určiť oprávnenia tak, aby k chránenému objektu mohli pristupovať iba zdroje, ktoré patria do určitej množiny. Tento pojem je ľahké pomýliť si s autentifikáciou. Autentifikácia je proces určenia, či identita používateľa nie je falošná. V skutočnosti sa autentifikácia a autorizácia doplňujú. Správna autorizácia závisí od presnej autentifikácie . Každý internetový používateľ rozdiel dobre pochopí na príklade zadávania hesla. Ak sa chce Janko prihlásiť do systému , musí zadať svoje používateľské meno janko96 a k nemu priradiť určené heslo. Tento proces poznáme pod pojmom autentifikácia. Zatiaľ čo autentifikácia zisťuje , kto ste, autorizácia určuje, čo môžete spraviť. Nad každým objektom

definuje, ci máte k nemu práva alebo nie. Na to aby bola autorizácia realizovaná je potrebný určitý typ vzťahu medzi používateľovým ID a systémovými zdrojmi. Jednou z možností je priradiť k objektu zoznam oprávnených používateľov, alebo naopak uložiť zoznam objektov ku ktorým môže pristupovať používateľ s daným ID. Tento vzťah je bližšie definovaný konkrétnym modelom pre riadenie prístupu. Na to, aby sme správne pochopili rôzne modely, potrebujeme si ako prvé definovať tieto pojmy:

- používateľ - pojem odkazuje na človeka, ktorý komunikuje s počítačovým systémom. Vo väčšine systémov je dovolené, aby jeden používateľ mal priradených viacero prihlasovacích ID a zároveň tieto ID môžu byť súčasne aktívne
- session - inštancia komunikácie medzi používateľom a systémom
- subjekt - počítačový proces, ktorý je spustený na základe práv používateľa
- objekt - môže byť zdroj , ku ktorému pristupujeme v počítačovom systéme. Nahliadame naň zväčša ako na pasívnu entitu, ktoré obsahujú informácie. Nie je to však pravidlo. Ako objekt môžeme v určitých situáciach považovať programy, tlačiarne, a iné aktívne entity
- operácia - je aktívny proces vyvolaný subjektom. Staršie modely riadenia prístupu , ktoré sa striktne zaoberali tokom informácii (čítanie a zapisovanie) používali pojem subjekt na všetky aktívne procesy. Model RBAC však vyžaduje vymedzenie týchto dvoch pojmov. V prípade bankomatu, po prihlásení používateľa prostredníctvom PIN, riadiaci program pod oprávneniami používateľa je subjekt. Tento subjekt môže spustiť viacero operácii- výber z účtu, dočasný vklad, výpis z účtu a iné
- oprávnenia - sú povolenia na vykonanie určitej akcie v systéme

1.3.1 Princípy bezpečného dizajnu

V roku 1975 Salzer a Schroeder [6] popísali ochranné mechanizmy, ktoré musí dobrý dizajn prístupových práv obsahovať. V skutočnosti však niektoré z nich siahajú do 19-teho storočia, kedy ich popísal Auguste Kerchoffs.

Minimálne oprávnenia : Hlavnou požiadavkou autorizácie je potreba vymedziť používateľovi minimálne oprávnenia, teda také, ktoré mu vymedzia prístup iba k úlohám, ktoré potrebuje na vykonávanie svojej práce. Tým sa predíde problémom, kedy jednotlivý používateľ môže svojou aktivitou vykonať nežiaduce operácie. Je potrebné si uvedomiť, že že práva používateľa sa môžu meniť v závislosti od času, úlohy alebo samotnej funkcie.

Jednoduchosť mechanizmu Dizajn by mal byť dostatočne zrozumiteľný , aby bolo možné dokázať jeho správne fungovanie. Taktiež by mal byť dostatočne jednoduchý, aby prípadné bolo možné chyby ľahko identifikovať a opraviť.

Protiporuchové princípy Prístupové oprávnenia by mali byť založené na inklúzii. Subjekt má mať explicitne definované práva. Na začiatku by mali byť jeho práva k objektu nulové. Tento princíp zabezpečí v prípade poruchy mechanizmu, aby bol zamietnutý prístup oprávnenému ako aj neoprávnenému prístupu.

Pravidelná autorizácia Každá požiadavka na prístup k objektu by mala požadovať autorizáciu subjektu. Žiadne ukladanie výsledkov prístupu by nemalo byť povolené.

Verejne známy príncíp autorizácie (Kerckhoffov princíp) Bezpečnosť autorizácie by nemala závisieť od utajenia princípu. Ak je dizajn správny, jeho odhalenie by nemalo narušiť bezpečnosť.

Oddelenie právomocí Ak je to možné, ochranný mechanizmus by mal závisieť od čo najviac nezávislých podmienok. Príkladom môže byť vyžiadanie spolupráce dvoch nezávislých entít.

Mechanizmus najmenšej právomoci V dizajne by sa malo minimalizovať zdieľanie viacerými používateľmi. Implementácia tohto princípu vyžaduje fyzické alebo logické oddelenie systémov

Používateľská akceptácia Ochranný systém by mal byť pre používateľov transparentný a ľahký na používanie. Užívateľ by nemal byť nútený sa odhlásiť a znova prihlásiť k vykonaniu bežných úloh

1.3.2 Model Bell-LaPadula

V 70-tych rokoch vznikla potreba formálne popísať model, ktorý bude spĺňať bezpečnostné požiadavky pre potreby viacúrovňovej bezpečnostnej politiky pre Ministerstvo Obrany USA. V roku 1973 tak vznikol model Bell-LaPadula [1] . Princíp tohto modelu je jednoduchý a matematicky jasne zadefinovaný. Určuje 3 pravidlá:

1. Prvé pravidlo definuje,že subjekt nesmie čítať dokumenty, pokiaľ pre ne nedosahuje dostatočnú úroveň bezpečnostnej klasifikácie. Používateľ teda nie je oprávnený čítať dokumenty, ktoré majú vyššiu úroveň bezpečnosti. Napríklad používateľ s klasifikáciou *secret* môže čítať súbory s bezpečnosťou *secret* a *regular*, ale nesmie čítať dokumenty s úrovňou *Top secret*

2. Druhé pravidlo definuje, že subjekt nie je oprávnený zapisovať do objektov s nižšou úrovňou bezpečnosti. Toto pravidlo zabezpečuje aby nebolo možné neúmyselne kompromitovať informácie do nižšej bezpečnostnej úrovne. Naopak zapisovanie do vyššej úrovne bezpečnosti je povolené. Môžeme si napríklad predstaviť, že chceme poslať tajný dopis prezidentovi.

3. Tretie pravidlo pridáva extra úroveň bezpečnosti nezávislú od ostatných. Zabezpečuje aby bolo možné rozdeliť riadenie prístupu na základe toho, kto sa snaží pristupovať k akému objektu. Napríklad úradník z ministerstva vnútra môže mať prístup k niektorým dokumentom s klasifikáciou *Top secret* , ale nesmie mať prístup ku každému takémuto dokumentu, ako napríklad dokument s pozíciu jadrových hlavíc. Pridáva sa tak takzvaný Äccess Control Matrix", čiže sa dá vyhľadať v matici podľa používateľa a objektu, aké má daný používateľ nad objektom právomoci.

1.4 RBAC

1.4.1 Popis modelu

Vo WfMS sa ako najlepší model riadenia prístupu preukázal systém rolí RBAC (role-based access control). RBAC poskytuje abstraktný a všeobecný model. Sám o sebe však neposkytuje striktné riešenia bezpečnosti prístupu a nie je ani priamo zadefinované ako musí byť tento model implementovaný. V [5] je RBAC popísaný ako model , ktorý dokáže zabezpečiť schopnosť vizualizovať a centrálne riadiť práva používateľov. Zároveň poskytuje možnosť zadefinovať, vymedziť a kontrolovať prístupové práva medzi používateľom a rolou, rolou a rolou alebo rolou a právomocami. RBAC ako model neurčuje konkrétnu politiku riadenia a môže implementovať tradičný princíp riadenia ako DAC, MAC a iné.

Výhodou tohto modelu je možnosť priradiť viacerým používateľom rovnaké práva na základe určitých spoločných vlastností. Vo WfMS je táto vlastnosť veľmi cenená pokiaľ chceme oddeliť návrh procesu od používateľského riešenia.

1.4.2 Dizajn

V jednej organizácii môže byť viacero rol, ktoré vykonávajú rôzne funkcie. Práva na vykonávanie určitých operácii sú priradené týmto rolám. Jednotliví zamestnanci a členovia firmy sú priradení k rolám, čím nepriamo získavajú tieto právomoci. Vďaka tomu, že tieto právomoci nie sú priamo priradené konkrétnym členom, ale rolám, administrácia právomocí pre konkrétneho člena sa zjednodušuje na zaradenie člena do určitej

roly. V roku 1992 NIST sformulovala základné požiadavky do všeobecného modelu. Definovali sa 3 pravidlá:

1. Priradenie k role : subjekt je oprávnený vykonať operáciu jedine v prípade, že je priradený k roli. Autentifikácia ako taká sa v RBAC nepovažuje za takúto operáciu. Pre všetky ostatné operácie v RBAC architektúre je však potrebné aby mal subjekt priradenú rolu

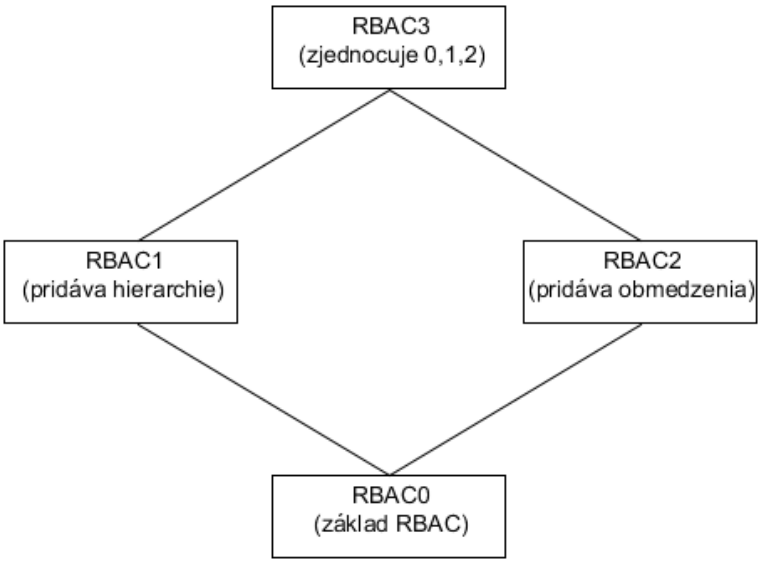
2. Autorizácia role: Užívateľ môže vystupovať iba pod takou rolou, ku ktorej je oprávnene priradený.

3. Autorizácia operácie: subjekt s aktívnou rolou je oprávnený spúšťať iba také operácie, pre ktoré má daná rola oprávnenia ich spúšťať. Spolu s prvým a druhým pravidlom sa tak zabezpečí, že subjekt môže vykonávať iba povolené operácie

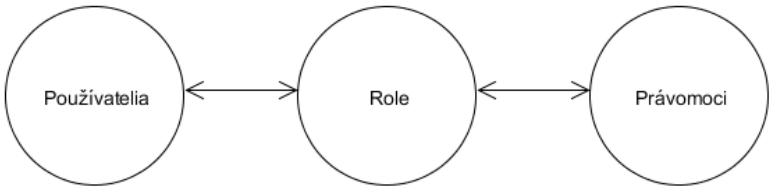
V nasledujúcej publikácii [4] NIST rozobrala RBAC viac do detailov , navrhla prídavné funkcionality a zahrnula špecifické formy vzťahov , tak aby spĺňali požiadavky na oddelenie právomocí (SoD). V roku 1996 Sandhu a spol. [7] predstavili framework RBAC96 založený na RBAC a rozložil tak RBAC do štyroch koncepčných modelov. Ponúkli tak riešenie pre komerčnú implementáciu RBAC. Základný model RBAC0 predstavuje jadro,v ktorom sú obsiahnuté všetky 3 spomenuté pravidlá. Nadväzujúci model RBAC1 pridáva ku základnému modelu hierarchický systém, čím umožňuje pre jednotlivú rolu zdediť právomoci inej roly. Celý systém sa tak dokáže sprehľadniť a ak je jedna rola nadradená iným, stačí, aby bol používateľ priradený k nadradenej role a automaticky dostane právomoci všetkých podradených rolí. RBAC2 pridáva obmedzenia ako napríklad oddelenie právomocí (ak chceme aby dve úlohy nemohol vykonať rovnaký používateľ z rovnakej role).

Jadro RBAC

Jadro RBAC , taktiež značené ako RBAC0 poskytuje základ pre riadenie prístupu na základe rol. Rozlišuje päť základných administratívnych prvkov: používateľov, roly a oprávnenia, pričom oprávnenia pozostávajú z operácii aplikovaných na objekty. Základ RBAC pozostáva z definovania oprávnení pre roly a používatelia sú priradení k roliam a tým nadobúdajú právomoci patriace rolám. Obrázok 1.8 ukazuje vzťah užívateľov, rolí a oprávnení. Dvojitá šípka značí vzťah m:n. Teda napríklad jeden používateľ môže byť priradený k viacerým roliam a zároveň jedna rola môže pozostávať z mnohých používateľov. Tento princíp predstavuje flexibilné riešenie priradenia práv rolám a užívateľov k rolám, čím podporuje princíp minimálnych oprávnení. Každé oprávnenie predstavuje kombináciu objektov a operácii. Celý systém oprávnenia na základe môžeme vidieť na obrázku 1.9.



Obr. 1.7: Framework RBAC96 podľa Sandhu a spol.



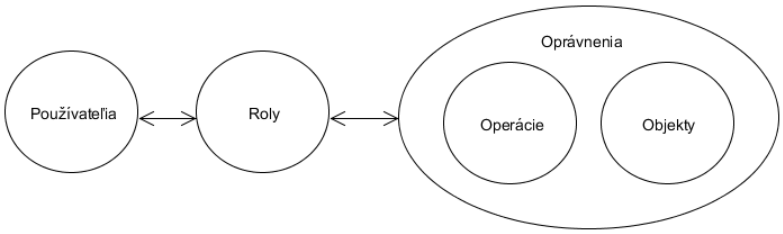
Obr. 1.8: Vzťahy RBAC podľa [3]

Hierarchické štruktúry

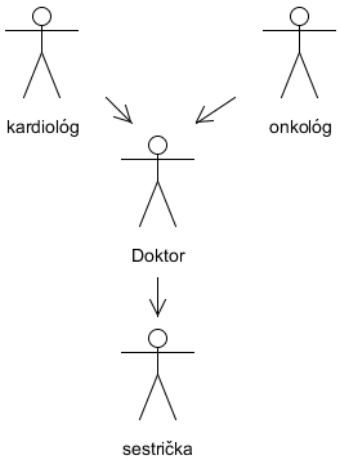
Mnoho rol v má v procesoch prelínajúce sa právomoci. Používatelia patriaci odlišným rolám môžu byť oprávnení vykonávať rovnaké operácie. Hierarchia rol zabezpečí prirodzené rozšírenie pre automatické splnomocňovanie nadradených rol. Môžeme si predstaviť napríklad juniorské a seniorské pozície v organizácii. Seniorská pozícia by mala mať obsahovať oprávnenia juniorskej, pričom môže zahŕňať oprávnenia, ktoré juniorskej pozícii neprináležia. V opačnom smere to nie je možné. Pre realizáciu hierarchickej štruktúry sa často zavádza relácia čiastočného usporiadania, ktorá verne znázorní štruktúru organizácie.

Obmedzenia

Do modelu RBAC je možné pridať ďalšie obmedzenia, ktoré bližšie definujú politiku prístupu práv. Medzi tie základné patrí vylúčenie práv. V niektorých prípadoch chceme zamedziť aby používateľ mohol pristupovať súčasne k dvom objektom. [?] definuje vylúčenie práv ako rozdelenie zodpovednosti pre citlivé údaje, tak aby žiaden jednotlivec



Obr. 1.9: Rozšírené vzťahy RBAC podľa [3]



Obr. 1.10: Príklad hierarchie rol

nebol schopný narušiť bezpečnosť dát. Poznáme dve základné kategórie: statické a dynamické vylúčenie práv. Najľahšie ich rozlíšime podľa toho , v akom čase ich v systéme zavádzame. Statické vylúčenie práv sa určuje na začiatku a vopred zamedzí jednému používateľovi súčasne byť priradený v rolách, ktoré sa vzájomne vylučujú. Pri dynamickom vylúčení práv sa vylúčenie práv aplikuje v čase, keď je používateľ prihlásený v systéme. Tento typ vylúčenia je slabšou formou obmedzenia a dovoľuje jednému používateľovi zastávať dve vylučujúce sa role, avšak nemôže používať obidve v samotnej používateľskej session.

1.4.3 Porovnanie RBAC s alternatívnymi metódami riadenia

Okrem modelu RBAC existuje na trhu mnoho alternatívnych riešení riadenia prístupu. Zatiaľ čo RBAC rieši politiku prístupu na základe organizačnej štruktúry , ostatné metódy sa viac zameriavajú na bezpečnosť prístupu z pohľadu používateľa alebo administrátora. Zabezpečenie práv je definované až k používateľom. V tejto časti si ukážeme tri časté modely: access control list (ACL), DAC (discretionary access control) a MAC (mandatory access control) .

RBAC	Štandartné systémy riadenia
• Rola nemusí obsahovať žiadnych používateľov • Rola sa ľahko mapuje na štruktúru organizácie • Prístupové práva sa mapujú na roly • Používateľom sú priradené role	• Prístupové práva sú naviazané na používateľov • Používatelia sú rozdelení do skupín • Skupina je pomenovaná množina používateľov, práv a ďalších skupín, ktorá spravidla obsahuje aspoň dvoch používateľov • Skupiny sa neviažu na štruktúru organizácie

Tabuľka 1.1: Porovnanie RBAC so štandartými prístupmi riadenia

ACL

Tento prístup definuje ku každému objektu zoznam používateľov a ich právomocí. Administrátor v systéme ľahko vyhľadá kto má k akému objektu aké právomoci. Problém nastáva , ak chceme zistiť všetky právomoci , ktoré prináležia konkrétnemu používateľovi. Ešte väčšie komplikácie nastávajú pri potrebe upraviť alebo vymazať používateľovi nejaké právomoci, pretože treba prejsť všetky možné objekty v danom systéme.

voliteľné riadenie prístupu DAC

Podľa [] DAC predstavuje riadenie prístupu k objektu založené na identite subjektu, skupiny alebo ich kombinácie. Subjekt, ktorý vlastní určité práva je schopný ich predávať ďalšiemu subjektu. DAC teda necháva jednotlivým používateľom možnosť prideľovať a odoberať prístupové práva. Tento prístup však môže byť často nežiadúci, ak chceme zamedziť šíreniu dát medzi neoprávnených používateľov. DAC bližšie nešpecifikuje právomoci pre konkrétneho používateľa. Akonáhle môže používateľ k objektom pristupovať, môže dáta zmeniť alebo rozšíriť práva pre neautorizovaného používateľa.(Sandhu and Samarati, 1994)

povinné riadenie prístupu MAC

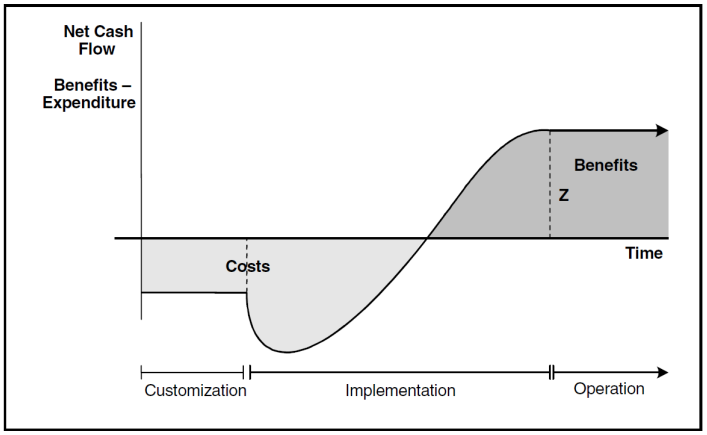
Ako riešenie pre hlavný problém DAC architektúry, povinné riadenie prístupu MAC definuje hierarchické úrovne bezpečnosti. Vychyľuje sa tak od štandartných prístupov a implementuje MLS (multilevel security). Iba administrátori sú oprávnení spravovať a

priraďovať prístupové práva. Ku každému objektu ako aj subjektu sa priradí určitá bezpečnostnú úroveň. Riadenie prístupu je následne automaticky riadené pomocou dvoch pravidiel. Prvé z nich povoľuje čítať v rovnakej a nižšej bezpečnostnej úrovni, zatiaľ čo druhé zabezpečuje zapisovanie do rovnakej alebo vyššej bezpečnostnej úrovne.

1.4.4 Výhody RBAC

V podnikovej oblasti prináša RBAC model revolúciu pre riadenie prístupu. Tento systém výrazne zjednodušuje administratívu, čo v konečnom dôsledku zvyšuje celkovú produktivitu a znižuje náklady. Všeobecne platí, že čím viac používateľov sa v systéme vyskytuje, tým je výhodnejšie a efektívnejšie použiť tento systém. Systém RBAC sa ukazuje ako atraktívna náhrada alternatívnych modelov. V [5] NIST popísala ekonomickú návratnosť zavedenia RBAC do systému. Môžeme vidieť (obr. 1.11), že návratnosť investície rastie s časom používanie. Hlavné výhody RBAC sú

- rýchla administrácia - najťažšie pri systéme RBAC je definovať na začiatku jej štruktúru. Neskôr pri pridávaní nových užívateľov nie je zväčša potrebné vytvárať novú rolu, ale iba priradiť nového používateľa do už existujúcich rol. Pri zmene alebo odstránení používateľa z firmy nevznikajú problémy s odstraňovaním a mazaním právomocí
- zrozumiteľnosť - zväčša sa na definovanie rol používajú termíny blízke človeku ako lekár, manažér, programátor. Takéto riadenie práv je teda pre človeka prirodzenejšie ako ostatné zaužívané metódy. Ľudia tak nevynaložia veľa úsilia na samotné študovanie administrácie rol.
- nové možnosti bezpečnosti - statické a dynamické vylúčenie práv



Obr. 1.11: Ekonomická návratnosť RBAC podľa [5]

Opis riešenia

2.1 Špecifikácia požiadaviek

V nasledujúcej sekcii opíšeme základný model fungovania aplikácie, analyzujeme požiadavky a naznačíme spôsob jej realizácie. Pre lepšie pochopenie aplikácie si pomôžeme jednoduchým príkladom procesu, na ktorom vyniknú hlavné výhody nášho systému. V rámci nášho systému sa špeciálne zameriame na model riadenia systému pomocou rolí.

2.1.1 Základný opis aplikácie

Zistili sme, že Workflow management systému je silný nástroj, ktorý umožní používateľovi oddeliť a jasne znázorniť procesnú časť od aplikácie. Tento systém ponúka mnohé výhody. Jednou z možností ako WfMS implementovať je použitím Petriho sietí. Petriho siete sú jasne formálne definované a matematicky overené. Ako základ pre fungovanie našej aplikácie sme sa preto rozhodli použiť práve Petriho siete. Pri aplikácii Petriho siete na WfMS prechody predstavujú úlohy, ktoré za sebou nasledujú v určitom poradí. Tieto úlohy môžeme rôzne definovať. Príklad takejto úlohy je napríklad vypísanie daňového priznania, schválenie žiadosti alebo iné. V našej aplikácii bude ku každému prechodu v sieti možné priradiť formulár, ktorý pokryje väčšinu bežných požiadaviek. Pre vypísanie daňového priznania sa teda bude dať jednoducho vypísať formulár, ktorý bude obsahovať potrebné políčka ako krátka odpoveď, dlhá odpoveď, zaškrtávacie políčka prípadne výber z viacerých možností. Každú úlohu musí niekto spustiť a vykonať. Je treba teda definovať prístup riadenia v aplikácii. Pre tieto účely sme zvolili systém riadenia za pomoci rolí, ktorý sa ukázal ako flexibilné riešenie našich požiadaviek. Ku každému prechodu v procese bude okrem formuláru pridelená rola ktorá môže daný prechod spustiť.

Naznačili sme spôsob vytvárania biznis procesov prostredníctvom Petriho sietí. Aby však daná aplikácia mohla fungovať ako plnohodnotné WfMS potrebujeme aplikačné rozhranie, ktoré bude nad vytvoreným procesom zabezpečovať jeho správne fungovanie. Naša aplikácia bude poskytovať webové rozhranie, ktoré umožní užívateľom používať náš systém bez nutnosti sťahovať dodatočný softvér. Takisto sa tým vyhneme problémom spojených s kompatibilitou odlišných operačných systémov. Na strane servera

budeme využívať kombináciu jazyka PHP s databázovým systémom MySQL. Na strane klienta využijeme framework Jquery, ktorý je rýchly a nenáročný na pamäť.

Kľúčovou výhodou WfMS je možnosť oddeliť proces od aplikačnej časti. Veľa firiem používa rovnaké alebo veľmi podobné procesy. Predstavme si dve rôzne pekárne. V zjednodušenom pohľade môžeme povedať, že pre obidve pekárne platí rovnaký proces výroby chleba. Jediné v čom sa tieto pekárne odlišujú je zamestnanecká štruktúra. Náš systém by umožnil obidvom pekárňam použiť rovnaký proces. Ak je možné prepoužiť rovnaké procesy pre viacero firiem, vzniká tu možnosť procesy predávať. V našej aplikácii je preto dôležitou súčasťou portál, ktorý okrem poskytnutia tvorby procesov a jeho následného používania, poskytne možnosť vyhľadávať a prepoužívať procesy, ktoré už boli vytvorené.

2.1.2 Funkcia rolí v aplikácii

V biznis procesoch väčšinu úloh vykonávajú fyzické osoby. V každej firme je niekoľko zamestnancov, ktorí majú rôzne právomoci.Zároveň však môžeme vidieť ich neustálu fluktuáciu. Zamestnanci do firmy prichádzajú a aj odchádzajú. Takisto sa stáva, že zamestnanec zmení pozíciu vo firme a dostane tak nové právomoci. Pre management riadenia takéhoto systému nám nestačí využitie klasických modelov, prípadne systém na správu takéhoto systému by bol vysoko nákladný. Z týchto dôvodov väčšina workflow management systémov využíva model systému prístupu na základe rolí RBAC. Pre potreby našej aplikácie preto využijeme základy tohoto modelu. V každom procese priradíme rolu na samotný proces, aby sme vedeli určiť, kto môže spustiť nový prípad. Zároveň v Petriho sieti definujeme ku každému prechodu jednu rolu, ktorá môže daný prechod spustiť. Samotné právomoci danej role nad prechodom sú dané určené vo dátovej časti.Vo formulári ku prechodu sa dajú políčka nastaviť ako povinné, upraviteľné a viditeľné. To nám zabezpečí ekvivalent k právomociam read, write.

2.1.3 Referencie

Role v procese zabezpečia prenos prístupových práv z úloh na užívateľa. V rámci jednej role si môžeme predstaviť skupinu užívateľov, ktorí majú určité spoločné prístupové práva vo firme. V niektorých procesoch však treba zabezpečiť, aby dve od seba závislé úlohy, mohol spustiť len ten samý užívateľ. Samotné role túto funkcionalitu nedokážu zabezpečiť . V našej aplikácii je nutné ju explicitne zadefinovať. Do našej aplikácie sme použili systém referencii na prechody v Petriho sieti. Do nášho projektu sme implementovali dva typy referencii : referencia na prechod a referencia na prvého užívateľa z role.

Najskôr opíšeme referenciu na prechod. Referencia na prechod zabezpečí, aby ten samý užívateľ ktorý spustí prechod, na ktorý odkazuje referencia, bol jediný oprávnený

na spustenie prechodu s touto referenciou. Jednoduchým príkladom je ak rozložíme jeden zložitý prechod , ktorý musí vykonať ten samý užívateľ na dve menšie. Predstavme si napríklad podpísanie tlačiva. Ku tlačivu treba podpísať aj jeho kópiu. Náš prvý prechod predstavuje podpísanie tlačiva a druhý prechod je priradený k podpísaniu kópie tlačiva. Podpísanie kópie obsahuje v sebe referenciu na prvý prechod s podpísaním originálneho tlačiva. V praxi to znamená, že obidve tlačivá musí podpísať tá istá osoba.

V procese však môžu existovať úlohy, ktoré sa v určitom prípade nikdy nevykonajú. Príkladom v Petriho sieti je podmienené vykonanie prechodov na základe OR-splitu. V prípade, že by sme namodelovali proces s referenciu na takýto prechod, pri vykonávaní procesu by sme sa dostali do stavu, kedy by nikto daný prechod nemohol spustiť a tým pádom by nebolo možné proces ukončiť. Takýto stav by bol nežiadúci a spôsobil by mnoho problémov. Druhým problémom ktorý vzniká je, že v procese vopred nevieme určiť poradie vykonávania jednotlivých úloh. Chceme aby prechod1 aj prechod2 spustil rovnaký človek z role. Nevieme však v akom poradí budú prechody za sebou nasledovať. Z týchto dôvodov sme sa rozhodli pridať "referenciu na prvého užívateľa z role". Táto referencia rieši obidva problémy zároveň. Prechod, ktorý bude označený touto referenciou , bude môcť spustiť jedine užívateľ, ktorý v procese prvýkrát spustil prechod pod takou rolou akú má proces s referenciou. V prípade, že taká neexistuje, znamená to, že dosiaľ nebol spustený žiaden prechod, ktorý by obsahoval danú rolu. V tomto prípade bude môcť prechod spustiť ktorýkoľvek užívateľ, ktorý je priradený k danej role. Jednou z alternatívnych riešení bolo vytvoriť zásobník referencii ...

2.1.4 Určenie požiadaviek

Aplikačným výstupom tejto bakalárskej práce je web stránka, ktorej účelom je vytvoriť jednoduché a intuitívne aplikačné rozhranie pre priraďovanie užívateľov k roliam. V aplikácii by sa mali dať vyhľadať všetci užívatelia vo firme. Pre rýchlejšie vyhľadávanie a prehľadné údaje bude možné užívateľov zotriediť do kategórie podľa rolí. Užívatelia, ktorí nemajú priradenú žiadnu rolu, budú mať vlastnú podsekciu. Vďaka tomu bude mať firma každého užívateľa pod kontrolou, aby mal priradenú minimálne jednu rolu. Užívatelia sa budú dať zoradiť podľa ich id, mena, priezviska alebo e-mailovej adresy. V každej podsekcii umožníme vyhľadávať konkrétneho užívateľa na základe vstupu . Tento vstup sa porovná s id,menom, priezviskom aj emailom každého užívateľa z vybranej podsekcie. Niektoré firmy majú vo svojej štruktúre niekoľko desiatok zamestnancov. Každý užívateľ sa bude dať jednotne aj skupinovo priradiť k požadovanej role. Zároveň bude možné osobitne spravovať role pre konkrétneho člena firmy. Ďalšou funkcionalitou aplikácie je manažment rolí vo firme. Do firmy bude možné vložiť nové role, prípadne niektoré role odstrániť. Pri odstránení role z firmy, treba dať pozor, aby sme nemohli odstrániť role ktoré firma práve používa na vykonávanie vlastných procesov.

Takýmto spôsobom bude možné do firmy pridať role z xml súboru vygenerované v aplikácii na vytváranie a prideľovanie rolí k procesom od Kristiána Stroku. Túto funkcionalitu bude možné pri integrácii plne odstrániť. Zároveň má aplikácia poskytovať rozhranie, prostredníctvom ktorého bude možné výstup od Kristiána Stroku možné uložiť do databázy.

2.2 Návrh

V nasledujúcej kapitole podrobne opíšeme fungovanie celého systému, pričom sa bližšie zameriame na konkrétnu imolementáciu a v krátkosti vysvetlíme jednotlivé časti systému, tak ako sme si ich rozdelili. Na začiatku na obrázku popíšeme model celého systému. Následne sa pozrieme na príklad systému , ktorý bude možné za pomoci našej aplikácie vytvoriť. V rámci tohto príkladu za pomoci ilustrácii a opisu bližšie rozoberieme jednotlivé časti celej aplikácie. Na konci si ukážeme možné scénare a životné cykly.....

2.2.1 Model fungovania aplikácie

Schvaľovanie bakalárskych prác

Pre bližšie porozumenie fungovania aplikácie si ukážeme príklad schvaľovania bakalárskej práce. Najprv si definujeme celý proces vrátane osôb ktoré sa daného procesu zúčastňujú. Následne si ukážeme príklad na vytvorenie procesu a jeho spracovanie v aplikácii. Celý proces ilustrujeme obrázkami —-TODO—

Najprv si celý proces predstavíme z pohľadu ——TODO—— . Na stránke úvodnej stránke sa zaregistrujeme a prihlásime. Vytvoríme si firmu .... .

2.2.2 Životný cyklus

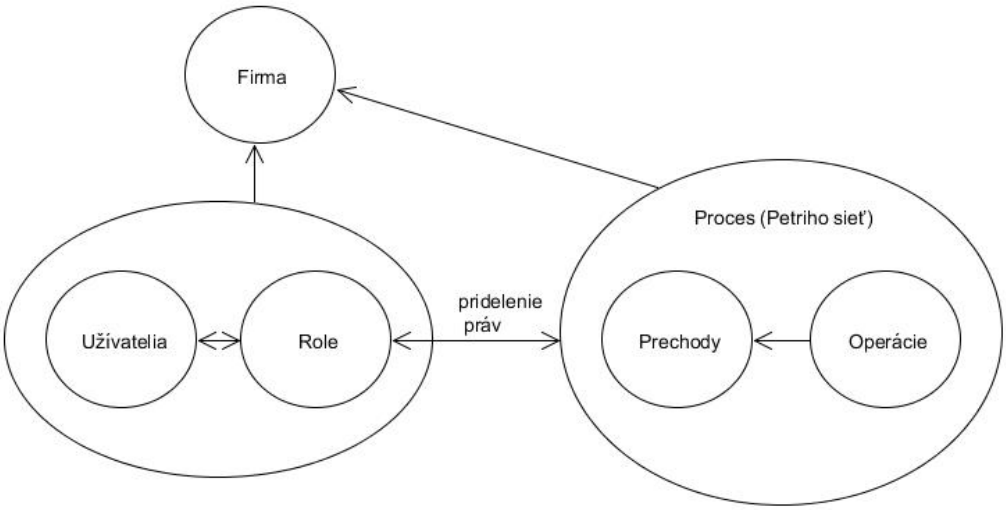
2.3 Implementácia

V tejto časti sa zameriame na podrobný opis implementácie modulu na správu a priraďovanie rolí k užívateľom. Podrobne si vysvetlíme fungovanie rolí v systéme, vysvetlíme databázový model a popíšeme celkovú architektúru a spôsob implementácie rolí v našom systéme. Rozoberieme možné bezpečnostné riziká a spôsob ich riešenia.

2.3.1 Popis architektúry

Základnou myšlienkou RBAC architektúry je odstrániť priame priraďovanie práv k užívateľom. Tento výsledok sa zabezpečí pridaním medzikroku a teda rolí medzi sa-

motných užívateľov a ich právomoci .Samotná implementácia tejto architektúry závisí od konkrétneho systému a jeho architektúry. Naša aplikácia sa zameriava na vytvorenie WfMS za pomoci Petriho sietí. Ako prvé si definujeme základnú architektúru. Na Obr. 2.12 môžeme zreteľne vidieť dve nezávislé časti fungovania workflow systému. V ľavej časti ilustrácie vidíme sekciu, ktorá sa zaoberá prideľovaním používateľou k roliam, zatiaľ čo pravá časť znázorňuje proces samotný. Vo firme je vďaka tomu zabezpečené, aby sa procesy mohli vytvárať nezávisle od užívateľov. Priraďovanie práv je zabezpečené väzbou medzi rolami a procesmi. Pre správnu funkcionalitu je však potrebné, aby firma mala priradené tie role, ktoré sú použité v jednotlivých procesoch, ktoré firma využíva. Definovanie prístupových práv je zabezpečené nad samotnými prechodmi v sieti.

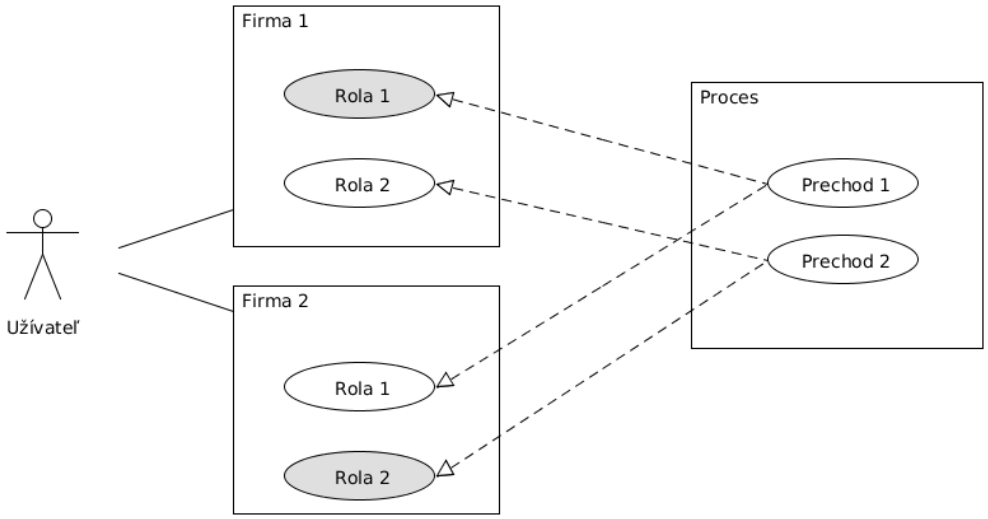


Obr. 2.12: Model RBAC v aplikácii

2.3.2 Priraďovanie užívateľou k roliam

V našej aplikácii nechceme aby bol užívateľ viazaný len na jednu firmu. Chceme aby pod rovnakým účtom mohol figurovať vo viacerých firmách, prípadne mal možnosť si založiť vlastnú. Preto je potrebné, aby sa užívatelia neviazali len na samotnú rolu. V aplikácii bude väzba užívateľa na rolu závislá od konkrétnej firmy. V každej firme bude môcť administrátor, užívateľ s právami na riadenie rolí, mať možnosť priradiť užívateľa ku konkrétnej role, ktorá je vo firme obsiahnutá. Samotný užívateľ môže byť tým pádom priradený vo viacerých firmách, pričom v každej firme bude mať iné práva. Na obrázku 2.13 môžeme vidieť zjednodušený model mapovania právomocí užívateľa prostredníctvom systému rolí. Šedé pozadie v roli znamená, že užívateľ je k roli priradený. V rovnakom procese vidíme, že užívateľ, ktorý môže vo firme 1 spustiť prechod 1 , nie

je oprávnený vykonať prechod 1 aj vo firme 2, pretože v nej nemá priradenú rolu. Vo firme 2 môže spustiť iba prechod 2.



Obr. 2.13: Model RBAC v aplikácii

2.3.3 Riadenie právomocí

Zadefinovali sme si ako sa v aplikácii mapujú užívatelia na role. V nasledujúcej časti si bližšie definujeme pravidlá pre spúšťanie prechodov v sieti, rovnako ako aj samotné právomoci ktoré daná rola v procese nadobudne.

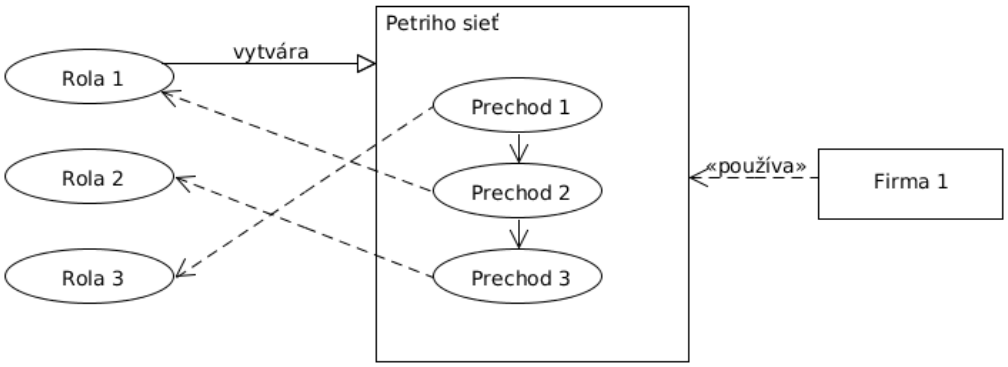
Priradenie práv k roliam

Priradenie práv k roliam je definované nepriamo prostredníctvom jednotlivých prechodov. Hlavnou úlohou role v procese je zadefinovať práva na vytváranie nových prípadov a takisto určiť právomoci na spúšťanie prechodov v procese. Schopnosť spustiť nový prechod je však vymedzená referenciami, návrhom Petriho siete a stave v akom sa tokeny momentálne nachádzajú.

Definovanie prístupových práv ku prechodu

Osoba môže v jednotlivom prípade spustiť prechod, ak spĺňa nasledovné požiadavky:

1. prechod je spustiteľný
2. osoba je priradená k roli, ktorej daný prechod prináleží
3. osoba spĺňa požiadavky referencie



Obr. 2.14: Právomoci rolí v sieti

Spustiteľnosť prechodu je zadefinovaná prostredníctvom Petriho siete. Prechod v sieti je spustiteľný len vtedy, ak každé miesto vstupujúce do prechodu obsahuje minimálne toľko tokenov, aká je násobnosť hrany medzi daným miestom a prechodom. V aplikácii máme dva typy referencii: **referenciu na prechod** a **referenciu na prvú rolu** . Ak má prechod nastavenú referenciu na prechod, v databáze sa porovná používateľove id s id užívateľa , ktorý spustil referovaný prechod. V prípade, že tieto dáta súhlasia, užívateľ je oprávnený spustiť prechod.

Ak má prechod nastavenú referenciu na prvého užívateľa, overí sa ,či sa v procese už daná rola nevyskytla. Ak prechod s danou rolou v danom prípade ešte nebol spustený, užívateľ je oprávnený tento prechod spustiť. V opačnom prípade je potrebné overiť id užívateľa s užívateľom ktorý ako prvý spustil prechod s touto rolou. Ak sa zistí zhoda, užívateľ môže daný prechod spustiť.

Operácie nad prechodom

Práva, ktoré rola nad prechodom získa sú zadefinované operáciami v konkrétnom prechode. Tieto operácie sa určia pri vytváraní formulára k danému prechodu. Vo formulári je možné nastaviť aké dáta budú užívateľovi prístupné, či ich bude môcť užívateľ iba zobrazovať, alebo aj editovať. Takisto sa definuje, ktoré údaje je potrebné vyplniť, aby mohol užívateľ tento prechod dokončiť.

2.3.4 Datový model

Dátová časť našej aplikácie je implementovaná v databáze MySQL5.1. Na začiatku je potrebné definovať návrh tabuliek pre uloženie rolí a referencii ku konkrétnej sieti. Druhým krokom je návrh databázového modelu, ktorý bude umožňovať konkrétnym užívateľom vo firme priradiť rolu. Pre tieto účely sme si definovali nasledovné tabuľky: ROLES,TRANSITIONS_X_ROLE , ROLES_START_CASES, REFEREN-

CES, USERS, USERS_X_FIRM, USERS_X_ROLE Do tabuľky ROLES budeme ukladať role a ich názov. Názov role sme zvolili ako unikátny atribút, kvôli tomu aby bolo uľahčené ukladanie role do databázy. TODOOOOO . V tabuľke TRANSITIONS_X_ROLE sa viažu role na prechody v petriho sieti. Každý prechod bude mať v tejto tabuľke presne určené , ktorá roľa ho môže spustiť. V Tabuľke ROLES_START_CASES priradíme k petriho sieti rolu, ktorá môže spustiť nový prípad. V tabuľke REFERENCES definujeme , či má prechod v sieti referenciu. Atribút value typu boolean v tejto tabuľke definuje, či má prechod referenciu. Ak má referenciu, v atribúte referenced_transition_id sa skontroluje,či obsahuje referenciu na konkrétny prechod. V prípade , že tento atribút nemá hodnotu NULL, odkazuje tento atribút na primárny kľúč referencovaného prechodu. V tabuľke USERS budú uložený jednotlivý užívatelia, ich prihlasovacie a osobné údaje. V tabuľke USERS_X_FIRM priraďujeme užívateľov ku firmám a v tabuľke USERS_X_ROLE priraďujeme užívateľom role v jednotlivých firmách. V tejto tabuľke sme zvolili ako primárny kľúč kombináciu cudzích kľúčov odkazujúcich na id užívateľa, role a firmy .

2.3.5 Ukladanie rolí a referencii k Petriho sieti

2.3.6 Používateľské rozhranie

2.4 Overenie riešenia

TODO

Záver

Na záver už len odporúčania k samotnej kapitole Záver v bakalárskej práci podľa smernice : „V závere je potrebné v stručnosti zhrnúť dosiahnuté výsledky vo vzťahu k stanoveným cieľom. Rozsah záveru je minimálne dve strany. Záver ako kapitola sa nečísluje.“

Všimnite si správne písanie slovenských úvodzoviek okolo predchádzajúceho citátu, ktoré sme dosiahli príkazmi `\glqq` a `\grqq`.

Literatúra

[1] E. Bell and L. J. La Padula. *dopln.* dopln, 1973.

[2] Hartmut Ehrig, Gabriel Juhás, Julia Padberg, and Grzegorz Rozenberg (Eds.). *Unifying Petri Nets: Advances in Petri Nets.* Springer, 2003.

[3] David F. Ferraiolo, D. Richard Kuhn, and Ramaswamy Chandramouli. *ROLE-BASED ACCESS CONTROL.* Artech Print on Demand; 2 edition, 2007.

[4] J. Cugini Ferraiolo, D. F. and D. R. Kuh. *Role-Based Access Control dopln, dopln.*

[5] Michael P. Gallaher, Alan C. O'Connor, and Brian Kropp. The Economic Impact of Role-Based Access Control. *National Institute of Standards and Technology, 2002.*

[6] J. H. Saltzer and M. D. Schroeder. The Protection of Information in Computer Systems. *zmen toto este, 1975.*

[7] R. a kolektív Sandhu. Role-Based Access Control Models. *IEEE Computer, 1996.*

[8] W. M. Van Der Aalst. Three good reasons for using a petri-net-based workflow management system. *MIT press., 1996.*

[9] Wil Van Der Aalst and Kees Max Van Hee. Workflow management: models, methods, and systems. *MIT press., dopln rok.*

[10] W.M.P. van der Aalst. The Application of Petri Nets to Workflow Management. *Department of Mathematics and Computing Science, Eindhoven University of Technology, 1998.*

[11] Shengli Wu, University of Georgia Amit ShethAffiliated with LSDIS Lab, John Miller, and Zongwei Luo. Authorisation and Access Control of Application Data in Workflow Systems, *volume 18. Kluwer Academic Publishers, 2002.*

Literatúra

[1] MOLINA H. G. - ULLMAN J. D. - WIDOM J., 2002, Database Systems, Upper Saddle River : Prentice-Hall, 2002, 1119 s., Pearson International edition, 0-13-098043-9