

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

Pod'akovanie:

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

Úvod	1
Analýza	3
1.1 Workflow management systém	4
1.1.1 Dopyt po WfMS	5
1.1.2 Výhody použitia WfMS	5
1.1.3 Riadenie zdrojov vo WfMS	5
1.2 Petriho siete	5
1.2.1 Implementovanie Petriho sietí na workflow management systém	6
1.3 Riadenie prístupu	6
1.3.1 Princípy bezpečného dizajnu	7
1.3.2 Model Bell-LaPadula	8
1.3.3 Separation of duties (SoD)	9
1.3.4 DAC	9
1.3.5 MAC	9
1.4 RBAC	10
1.4.1 Popis modelu	10
1.4.2 Dizajn	10
1.4.3 Porovnanie RBAC s DAC a MAC	13
1.5 Výhody RBAC	13
Opis riešenia	14
2.1 Špecifikácia požiadaviek	14
2.1.1 Základný opis aplikácie	14
2.1.2 Funkcia rolí v aplikácii	15
2.1.3 Referencie	15
2.1.4 Určenie požiadaviek	16
2.2 Návrh	17
2.2.1 Model fungovania aplikácie	17
2.2.2 Životný cyklus	17
2.3 Implementácia	17

2.3.1	Popis architektúry	17
2.3.2	Priradovanie užívateľou k roliam	18
2.3.3	Riadenie právomocí	19
2.3.4	Datový model	20
2.3.5	Ukladanie rolí a referencií k Petriho sieti	21
2.3.6	Používateľské rozhranie	21
2.4	Overenie riešenia	21
Záver		22

Zoznam skratiek

WFMS – Workflow management system

RBAC – Role-based access control

CSS - Cascading Style Sheets

XML - Extensible Markup Language

Zoznam obrázkov

1.1	smerovacie konštrukcie	4
1.2	Framework RBAC96 podľa Sandhu a spol.	11
1.3	Vzťahy RBAC podľa [2]	12
1.4	Rozšírené vzťahy RBAC podľa [2]	12
1.5	Príklad hierarchie rolí	12
2.6	Model RBAC v aplikácii	18
2.7	Model RBAC v aplikácii	19
2.8	Právomoci rolí v sieti	20

Úvod

Systém vývoja aplikačných programov sa veľmi rýchlo mení. So stúpajúcou zložitou 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 funkcionálnosť od aplikácie. Oddelenie 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 systémom je správa zdrojov. Treba určiť kto môže a naopak kto nesmie pristupovať ku konkrétnym prostriedkom, aby sa zachovala dôvernosc informácií a nemohol byť narušený systém. Vo WFSM je najpoužívanější 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 ďalšie 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 diskretných 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 managementu 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 priradovanie 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 zadať 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.

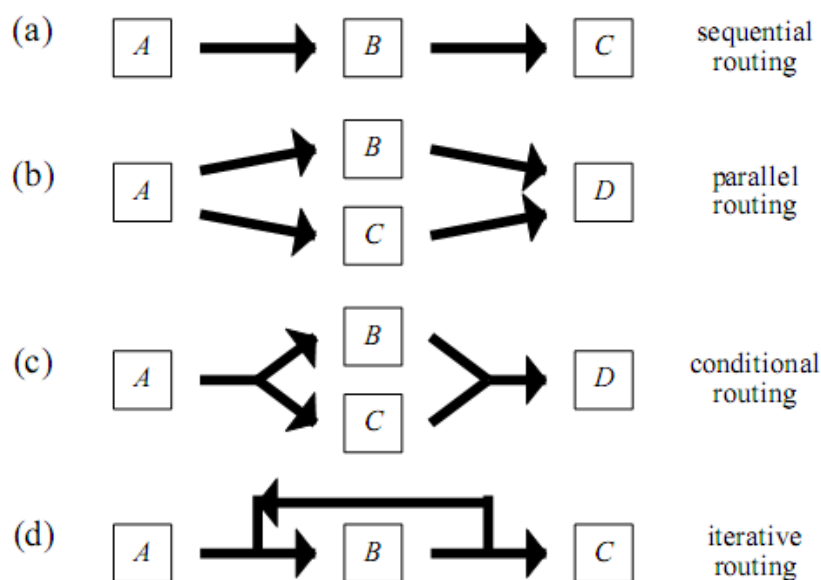
Možným nástrojom na implementovanie WfMS sa naskytuje použitie Petriho sietí. Petriho siete sú jasne formálne a matematicky definované, pričom ich modelovanie je jednoduché a intuitívne. Medzi hlavné výhody môžeme považovať grafické znázornenie, ktoré uľahčuje pochopenie princípov a fungovania tohto 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 výhod,

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

1.1 Workflow management systém

Workflow management systém je termín pre automatizáciu firemných procesov, v priebehu ktorého informácia putuje z jednej aktivity na ďalšiu, od jedného účastníka k druhému za určitých zadefinovaných pravidiel. Cieľom WfMs je zabezpečiť kontrolu a koordináciu nad procesmi, ktoré zahŕňujú ľudskú prácu v organizovanom prostredí. Dôležitým faktorom pri simulácii daných procesov je zabezpečiť korektné poradie vykonávaných úloh.



Obr. 1.1: Smerovacie konštrukcie vo workflow systémoch podľa [4]

Vo WfMS je možné medzi úlohami použiť smerovacie konštrukcie zobrazené na obrázku :

- 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

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 častokrát spúšťané samotnými zákazníkmi, ale nie je to pravidlo. Cieľom workflow management systému je efektívne zvládať jednotlivé prípady. Workflow proces je časť workflow management systému zameraný na spracovanie podobných prípadov. Každý workflow proces pozostáva z úloh , ktoré sú za sebou radené v špecifickom poradí. Workflow process definition zabezpečuje , ktorá úloha sa má spustiť a v akom poradí. Veľa prípadov sa môže vykonať s rovnakou postupnosťou. Pri umývaní auta značky Volkswagen postupujeme rovnako ako pri umývaní auta značky Škoda. Jednotlivé úlohy (tasks) sa teda môžu vykonávať súčasne vo viacerých prípadoch. Úloha, ktorá sa vykonáva v konkrétnom prípade, sa nazýva „working item“. Väčšina úloh (working items) je spúšťaná konkrétnymi zdrojmi. [4]

1.1.1 Dopyt po WfMS

TODO

1.1.2 Výhody použitia WfMS

TODO

1.1.3 Riadenie zdrojov vo WfMS

Pod pojmom zdroj rozumieme jednotku, ktorá je určená na vykonávanie úlohy. Môžeme si pod tým predstaviť počítač (server, tlačiareň, fax ...) alebo človeka ako pracovnú jednotku. V business prostredí väčšina 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 role si môžeme predstaviť akúkoľvek pracovnú pozíciu a pod samotnými pracovnými zdrojmi samotného zamestnanca. Táto klasifikácia umožňuje uľahčenie pridelovania právomocí na spúšťanie jednotlivých úloh. Zároveň tak jednotlivé zdroje môžu byť priradené k rôznym roliam a zabráni sa tým problému s nulovou referenciou. Ak by sme priradili jednotlivú úlohu ku konkrétnemu zdroju, nastali by problémy po vymazaní konkrétneho zdroja a samotný proces by nebolo možné ukončiť.

1.2 Petriho siete

//TODO Definícia [4] :

Petriho sieť orientovaný bipartitný graf s dvoma typmi uzlov nazývaných miesta (places) a prechody (transitions). Uzly sú spojené cez orientované hrany (arcs). Spojenia medzi uzlami rovnakého typu nie sú povolené.

//TODO

Petriho siete slúžia na matematické modelovanie a simuláciu procesov. Ich hlavná výhoda spočíva v jednoduchosti a grafickej reprezentácii.

. V Petriho sieti sa používajú najmä miesta, prechody a tokeny. Miesta určujú stav, v akom sa proces nachádza, prechody slúžia ako prostriedok, ako sa medzi nimi pohybovať. Logiku systému dotvárajú tokeny a hrany. Tokeny určujú, v akom stave sa systém práve nachádza. Podľa typu a násobnosti hrany vieme, aké množstvo tokenov je potrebné, aby sa spustil prechod na danú hranu nadväzujúci.

1.2.1 Implementovanie Petriho sietí na workflow management systém

Implementovanie Petriho sietí na workflow management systém je veľmi jednoduché. Na úrovni procesov sa určuje, aké úlohy budú za sebou nasledovať a v akom poradí. Modelovanie workflow procesu v Petriho sieti je priamočiare. Úlohy sú reprezentované pomocou prechodu, podmienky sú modelované prostredníctvom miest a jednotlivé prípady sú zaznačené pomocou tokenov. Ak chceme Petriho sieť namapovať na workflow systém, musíme zabezpečiť dve konkrétne požiadavky:

a) PN má dve špeciálne miesta: miesto „i“ a miesto „o“, ktoré reprezentujú počiatočný a koncový stav b) ak pridáme prechod t do PN, ktorý spája miesto „i“ a „o“, potom je PN silno spojená a to znamená, že pre každý pár uzlov x a y existuje priama cesta od x do y .

Na obrázku referencia môžeme vidieť jednoduchú implementáciu smerovacích konštrukcií na Petriho sieťach. Prvá Petriho sieť (a) zobrazuje jednoduché sekvenčné spúšťanie prechodov. V danej sieti môžeme spúšťať prechody len jednotlivo, v určenom poradí za sebou. Druhá Petriho sieť (b) predstavuje podmienené spúšťanie úloh. V danej sieti máme na výber z dvoch prechodov, ktoré je možné spustiť. Tretia Petriho sieť znázorňuje paralelné vykonanie úloh. Po spustení prechodu sa začnú vykonávať dva procesy nezávisle, pričom na konci sa spoja dokopy. Štvrtá Petriho sieť predstavuje iteračnú konštrukciu, ktorá umožní vrátiť token na predchádzajúci stav.

1.3 Riadenie prístupu

V nasledovnej sekcii rozoberieme princíp a pravidlá riadenia prístupu podľa [2]. Riadenie prístupu, známe pod pojmom autorizácia je koncept, ktorý používame od čias,

kedy ľudia pocitovali 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 dopĺňujú. 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, či 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 situáciách považovať programy, tlačiarne, a iné aktívne entity
- operácia - je aktívny proces vyvolaný subjektom. Staršie modely riadenia prístupu, ktoré sa striktne zaoberali tokom informácií (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 operácií- 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 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 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 princíp 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 najmenej 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. Princíp tohto modelu je jednoduchý a matematicky jasne zadefinovaný. Určuje 3 pravidiel:

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 "Access Control Matrix", čiže sa dá vyhľadať v matici podľa používateľa a objektu, aké má daný používateľ nad objektom právomoci.

1.3.3 Separation of duties (SoD)

1.3.4 DAC

Podľa [1] DAC predstavuje riadenie prístupu k objektu založené na identite subjektu, skupiny alebo ich kombinácie. Subjekt, ktorý vlastní určité práva je schopný ich predávať akémukoľvek ďalšiemu subjektu.

1.3.5 MAC

V počítačovej bezpečnosti MAC (Mandatory access control) predstavuje typ riadenia, v ktorom môžu iba administrátori spravovať a priradovať prístupové práva. Nad každým objektom administrátor definuje politiku prístupu pre každý subjekt samostatne.

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 [3] 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.

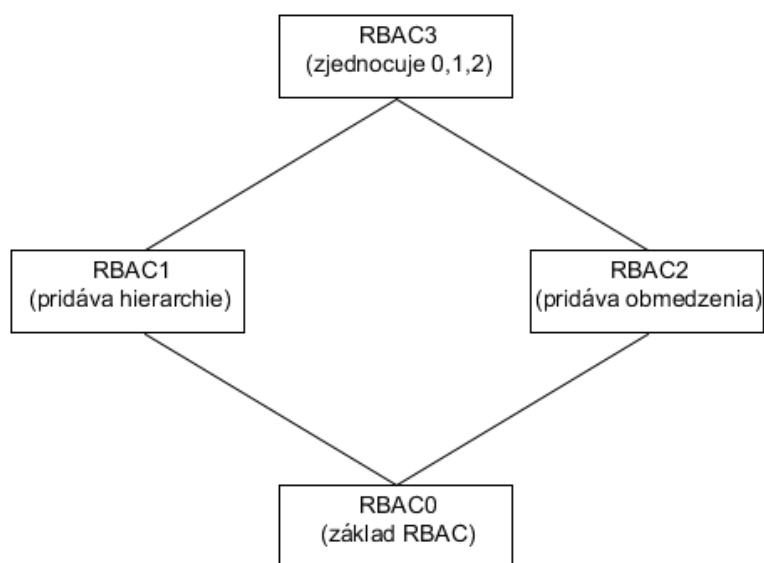
1.4.2 Dizajn

V jednej organizácii môže byť viacero rolí, ktoré vykonávajú rôzne funkcie. Práva na vykonávanie určitých operácií sú priradené týmto roliam. Jednotliví zamestnanci a členovia firmy sú priradený k roliam, čím nepriamo získavajú tieto právomoci. Vďaka tomu, že tieto právomoci nie sú priamo priradené konkrétnym členom, ale roliam, administrácia právomocí pre konkrétného 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 oprávnené 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 [ref] 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 [ref] predstavili framework

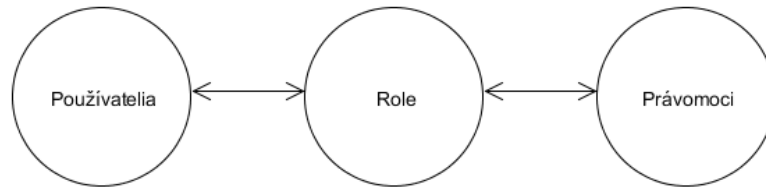
RBAC96 založený na RBAC a rozložil tak RBAC do štyroch koncepčných modelov. Ponúkli tak riešenie pre komerčnú implementáciu RBAC. Základný model RBAC0 predstavuje jadro, v ktorom sú obsiahnuté všetky 3 spomenuté pravidlá. Nadväzujúci model RBAC1 pridáva ku základnému modelu hierarchický systém, čím umožňuje 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).



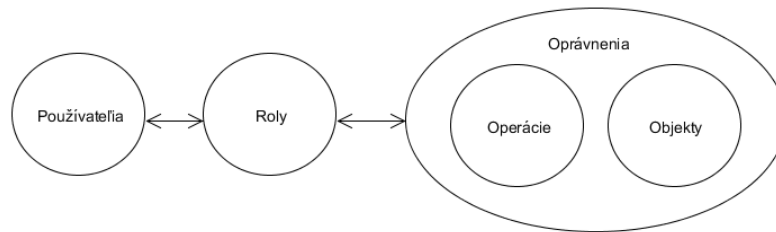
Obr. 1.2: Framework RBAC96 podľa Sandhu a spol.

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 roľám. Obrázok 1.3 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 roľám a užívateľov k roľám, čím podporuje princíp minimálnych oprávnení. Každé oprávnenie predstavuje kombináciu objektov a operácii. Celý systém oprávnenia na základe môžeme vidieť na obrázku 1.4.



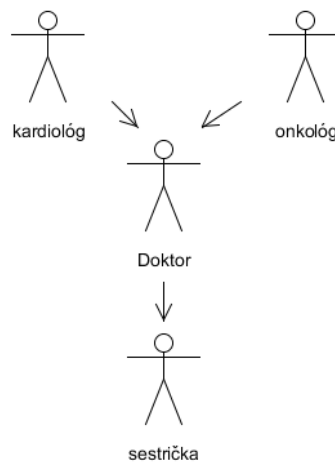
Obr. 1.3: Vzťahy RBAC podľa [2]



Obr. 1.4: Rozšírené vzťahy RBAC podľa [2]

Hierarchické štruktúry

Mnoho rolí 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.



Obr. 1.5: Príklad hierarchie rol

Obmedzenia

Do modelu RBAC je možné pridať mnohé 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. —
TODO—

1.4.3 Porovnanie RBAC s DAC a MAC

TODO

1.4.4 Výhody RBAC

TODO

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, zaškrtnuté 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 pre použiť 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, upravitel'né 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 funkcionality nedokážu zabezpečiť. V našej aplikácii je nutné ju explicitne zdefinovať. Do našej aplikácie sme použili systém referencií na prechody v Petriho sieti. Do nášho projektu sme implementovali dva typy referencií : 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 referenciou 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 referencií ...

2.1.4 Určenie požiadaviek

Aplikačným výstupom tejto bakalárskej práce je web stránka, ktorej účelom je vytvoriť jednoduché a intuitívne aplikačné rozhranie pre priradzovanie 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 podsekciiu. 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 podsekcii. 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 implementáciu 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 ilustrácií 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 stránky sa zaregistrujeme a prihlásime. Vytvoríme si firmu

2.2.2 Životný cyklus

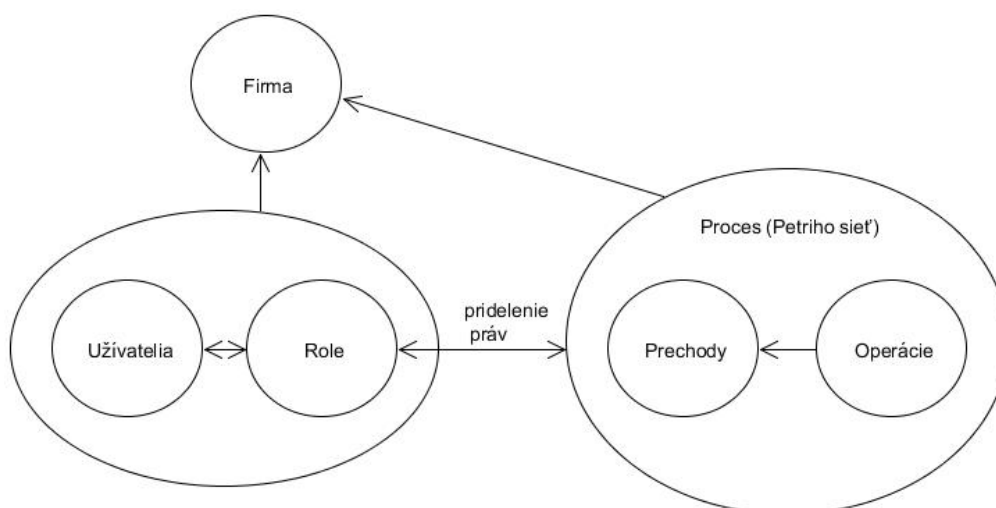
2.3 Implementácia

V tejto časti sa zameriame na podrobný opis implementácie modulu na správu a priradovanie 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 priradovanie 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. 2.6 môžeme zreteľne vidieť dve nezávislé časti fungovania workflow systému. V ľavej časti ilustrácie vidíme sekciu, ktorá sa zaoberá pridelovaním používateľov k roliam, zatiaľ čo pravá časť znázorňuje proces samotný. Vo firme je vďaka tomu zabezpečené, aby sa procesy mohli vytvárať nezávisle od užívateľov. Priradenie práv je zabezpečené väzbou medzi rolami a procesmi. Pre správnu funkcionálnosť 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.

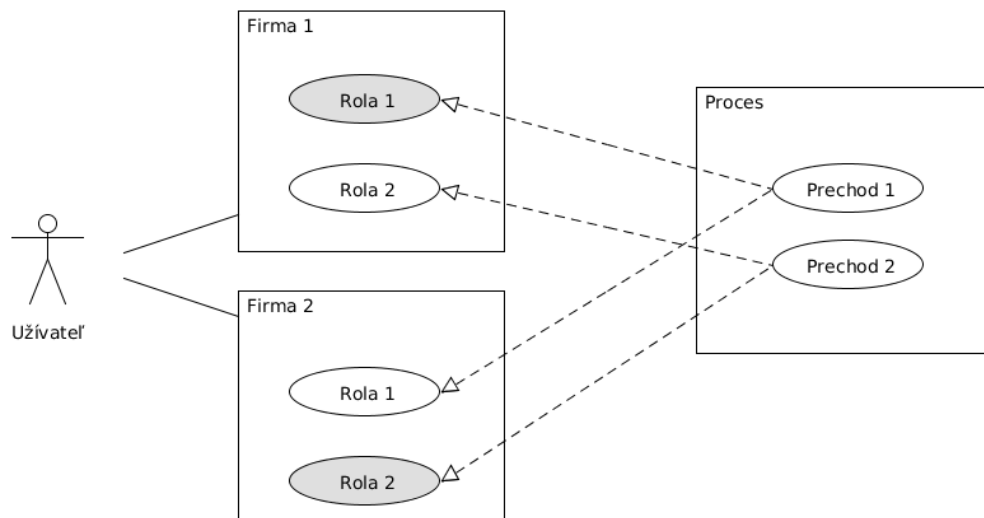


Obr. 2.6: Model RBAC v aplikácii

2.3.2 Priradenie užívateľov 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 2.7 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. 2.7: 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.

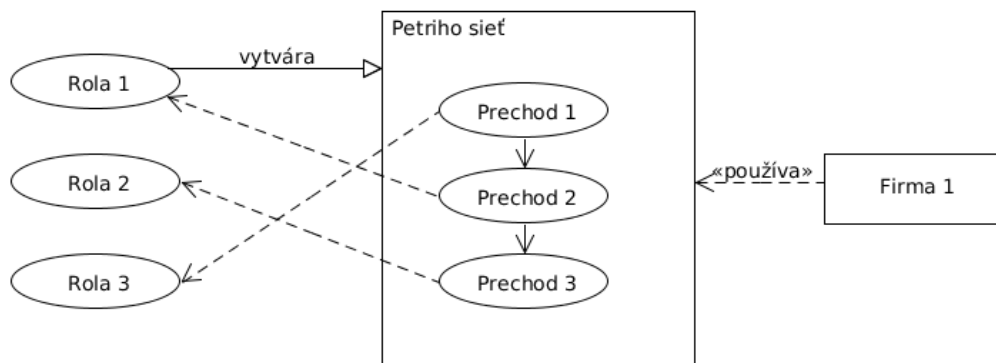
Priradenie práv k roliam

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. 2.8: 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 referencií: **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 referencií 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 jednotliví užívatelia, ich prihlasovacie a osobné údaje. V tabuľke USERS_X_FIRM priradíme užívateľov ku firmám a v tabuľke USERS_X_ROLE priradíme 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 referencií 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] Hartmut Ehrig, Gabriel Juhás, Julia Padberg, and Grzegorz Rozenberg (Eds.). *Unifying Petri Nets: Advances in Petri Nets*. Springer, 2003.
- [2] David F. Ferraiolo, D. Richard Kuhn, and Ramaswamy Chandramouli. *ROLE-BASED ACCESS CONTROL*. Artech Print on Demand; 2 edition, 2007.
- [3] 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.
- [4] 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.
- [5] 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