

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



## ZADANIE BAKALÁRSKEJ PRÁCE

Študent: **Pavol Martiš**  
ID študenta: 72557  
Študijný program: Aplikovaná informatika  
Študijný odbor: 9.2.9. aplikovaná informatika  
Vedúci práce: prof. RNDr. Gabriel Juhás, PhD.

Názov práce: **Workflow manažment systém – server manažmentu rolí a užívateľov**

Špecifikácia zadania:

Cieľom je navrhnuť a implementovať časť workflow manažment systému – server manažmentu rolí a užívateľov – vo forme webovej aplikácie.  
Aplikácia umožňuje registrovanie a správu užívateľov vrátane priradovania rolí a užívateľov.

Úlohy:

1. Naštudujte problematiku aplikácie petriho sietí vo workflow managemente.
2. Implementujte webovú aplikáciu server manažmentu rolí a užívateľov.
3. V prípade potreby integrujte aplikáciu s ďalšími časťami workflow manažment systému.
4. Otestujte a vyhodnoťte efektívnosť a použiteľnosť aplikácie v praxi.

Riešenie zadania práce od: 21. 09. 2015

Dátum odovzdania práce: 20. 05. 2016

**Pavol Martiš**  
študent



**prof. RNDr. Otokar Grošek, PhD.**  
vedúci pracoviska

**prof. RNDr. Gabriel Juhás, PhD.**  
garant študijného programu

# Vyhlásenie autora

Podpísaný Pavol Martiš čestne vyhlasujem, že som Bakalársku prácu Workflow manažment systém – server manažmentu rolí a užívateľov vypracoval na základe poznatkov získaných počas štúdia a informácií z dostupnej literatúry uvedenej v práci.

Prácu som vypracoval pod vedením Prof. RNDr. Gabriela Juhása, PhD.

V Bratislave dňa 15.05.2016

.....

podpis autora

# Pod'akovanie

Chcel by som poďakovať všetkým, ktorí mi akýmkoľvek spôsobom pomohli pri spracovaní tejto bakalárskej práce. Moje poďakovanie patrí najmä vedúcemu práce, prof. RNDr. Gabrielovi Juhásovi, PhD., za vedenie a celému tímu študentov, ktorí sa spoločne podieľali na tvorbe Workflow manažment systému.

Osobitné poďakovanie patrí mojej rodine, ktorá ma psychicky podporovala a kamarátovi Jakubovi Slávikovi za motiváciu k práci.

# SÚHRN

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE  
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Študijný program:	Aplikovaná informatika
Autor:	Pavol Martiš
Bakalárska práca :	Workflow manažment systém - server manaž- mentu rolí a užívateľov
Vedúci záverečnej práce :	prof. RNDr. Gabriel Juhás, PhD.
Miesto a rok predloženia práce :	Bratislava 2016

Práca sa zaoberá návrhom riešenia jednej z častí Workflow manažment systému, a to konkrétne serverovej časti rolí a používateľov. Na začiatku práce je analyzovaný workflow manažment, workflow manažment systém, Petriho siete a metódy riadenia prístupu, pričom analyzujeme hlavne metódu na základe rolí role-based access control (RBAC) so zameraním na jej architektúru a výhody oproti štandardným modelom. Následne sa práca sústreďuje na celkový opis fungovania tímovej aplikácie Workflow manažment systému a jej architektúry. V tejto časti sa bližšie rozoberajú jednotlivé časti systému a poukazuje sa na význam rolí a referencií v aplikácii. Na záver je venovaná pozornosť návrhu a implementácii serverovej časti rolí a používateľov. Opisuje sa v nej riešenie aplikácie na správu rolí pre používateľov vo firme a rozhranie pre ukladanie rolí a referencií do databázy. Na konci tejto časti sa testuje navrhnuté riešenie a rozoberajú sa možné vylepšenia do budúcnosti.

**Kľúčové slová:** workflow manažment, workflow manažment systém, Petriho siete, server manažmentu rolí, rola, riadenie prístupu

# ABSTRACT

SLOVAK UNIVERSITY OF TECHNOLOGY IN BRATISLAVA

FACULTY OF ELECTRICAL ENGINEERING AND INFORMATION TECHNOLOGY

Study Programme :	Applied Informatics
Author :	Pavol Martiš
Bachelor Thesis :	Workflow management system - server of role management and organizational structure
Supervisor :	prof. RNDr. Gabriel Juhás, PhD.
Place and year of submission :	Bratislava 2016

This thesis deals finding solution to the server part of roles and structure management in the complex of Workflow management system. In the beginning, there is an analyse of workflow management, workflow management system, Petri nets and methods of access control, especially focusing on the architecture of Role-based access control-RBAC, comparing it to standard access models. In the middle section, the thesis contains an overview of the team project Workflow management system, its architecture and the purpose of roles and references in it. This section individually characterizes each part of complex application. The last part of the thesis describes the design of the server part of mapping users into roles and implementation of the application to manage users and roles in firms. In the end, there are tests of implemented solution and the suggestions for future development.

**Keywords:** workflow management, workflow management system, Petri nets, roles, access control, management of roles

# Obsah

<b>Úvod</b>	<b>1</b>
<b>Analýza</b>	<b>2</b>
1.1 Workflow management systém . . . . .	3
1.1.1 Úloha . . . . .	4
1.1.2 Proces . . . . .	4
1.1.3 Smerovacie konštrukcie . . . . .	4
1.1.4 Výhody použitia WfMS . . . . .	5
1.1.5 Riadenie zdrojov vo WfMS . . . . .	5
1.2 Petriho siete . . . . .	5
1.2.1 História . . . . .	5
1.2.2 Definícia Petriho siete . . . . .	6
1.2.3 Príklad . . . . .	7
1.2.4 Implementovanie Petriho sietí na Workflow management systém	8
1.3 Riadenie prístupu . . . . .	10
1.3.1 Princípy bezpečného dizajnu . . . . .	11
1.3.2 Model Bell-LaPadula . . . . .	12
1.4 RBAC . . . . .	13
1.4.1 Popis modelu . . . . .	13
1.4.2 Dizajn . . . . .	13
1.4.3 Porovnanie RBAC s alternatívnymi metódami riadenia . . . . .	16
1.4.4 Výhody RBAC . . . . .	17
<b>Opis riešenia - Workflow systém</b>	<b>19</b>
2.1 Špecifikácia požiadaviek . . . . .	19
2.1.1 Základný opis aplikácie . . . . .	19
2.1.2 Funkcia rolí v aplikácii . . . . .	19
2.1.3 Referencie . . . . .	21
2.2 Návrh . . . . .	22
2.2.1 Procesný portál - Martin Kováčik . . . . .	22
2.2.2 Procesný editor - Tomáš Priboj . . . . .	23



2.2.3	Editor formulárov a dátového modelu - Ján Polaček . . . . .	24
2.2.4	Editor manažmentu rolí a organizačnej štruktúry- Kristián Stroka	25
2.2.5	Procesný server - Anna Demeterová . . . . .	26
2.2.6	Server formulárov a dátového modelu- Jakub Kováčik . . . . .	27
<b>Opis riešenia - Server manažmentu rolí a používateľov</b>		<b>28</b>
3.1	Určenie požiadaviek . . . . .	28
3.2	Návrh . . . . .	29
3.2.1	Popis architektúry . . . . .	29
3.3	Implementácia . . . . .	32
3.3.1	Modul na priradovanie používateľov k rolám . . . . .	32
3.3.2	Ukladanie rolí a referencií z editoru do databázy . . . . .	35
3.3.3	Dátový model . . . . .	36
3.4	Testovanie riešenia . . . . .	38
3.4.1	Používateľská skúsenosť . . . . .	38
3.4.2	Google testovanie . . . . .	39
3.4.3	Bezpečnosť . . . . .	41
3.4.4	Možné rozšírenia . . . . .	41
<b>Záver</b>		<b>43</b>
<b>Zoznam príloh</b>		<b>i</b>
	Štruktúra elektronického nosiča . . . . .	ii

# Zoznam skratiek

WfM - Workflow management  
WfMS – Workflow management system  
RBAC – Role-based access control  
MVC - Model-view-controller  
DAC - Discretionary Access Control  
MAC - Mandatory access control  
ACL - Access control list  
NIST - National Institute of Standards and Technology  
MLS - Multilevel security  
SVG - Scalable Vector Graphics  
XML - Extensible Markup Language  
JSON - JavaScript Object Notation  
CSS - Cascading Style Sheets  
PHP - Hypertext Preprocessor  
WWW - World wide web  
ID - identifikačné číslo

# Zoznam obrázkov

1.1	Smerovacie konštrukcie vo workflow systémoch podľa [2]	4
1.2	Petriho sieť na "vytvorenie" vody	7
1.3	Spustenie siete s jedným žetónom v mieste O a tromi žetónmi v mieste H	8
1.4	AND konštrukcia v Petriho sieti	9
1.5	XOR konštrukcia v Petriho sieti	9
1.6	Sekvenčné a iteračné konštrukcie v Petriho sieti	9
1.7	Model Bell-LaPadula	12
1.8	Vzťahy RBAC podľa [4]	13
1.9	Framework RBAC96 podľa Sandhu a spol. [9]	14
1.10	Rozšírené vzťahy RBAC podľa [4]	15
1.11	Príklad hierarchie rolí	15
1.12	Ekonomická návratnosť RBAC podľa [3]	18
2.1	Model Workflow systému	20
2.2	Príklad priradenia rolí k prechodom	20
2.3	Referencia na prechod	21
2.4	Referencia na prvého z role	22
2.5	Procesný portál	23
2.6	Procesný editor	24
2.7	Editor formulárov a dátového modelu	25
2.8	Vytváranie formulára	25
2.9	Editor manažmentu rolí a organizačnej štruktúry	26
2.10	Procesný server	27
2.11	Server formulárov a dátového modelu- Jakub Kováčik	27
3.1	Model RBAC v aplikácii	29
3.2	Mapovanie používateľov na roly vo firme	30
3.3	Právomoci rolí v sieti	31
3.4	Právomoci definované v prechodoch	32
3.5	Prideľovanie používateľov k rolám	32
3.6	Výber akcie pre označených používateľov	33
3.7	Správa rolí pre konkrétneho používateľa	33
3.8	Možnosti pre konkrétneho používateľa priamo z tabuľky	34

3.9	Administrácia rolí vo firme . . . . .	34
3.10	Priradenie právomocí rolám . . . . .	36
3.11	Priradenie používateľov do rol vo firmách . . . . .	37
3.12	Hlavné tabuľky pre prácu s rolami . . . . .	38
3.13	Mapa zobrazujúca hlavné body pozornosti používateľa . . . . .	39
3.14	Hlavné záujmové body používateľa . . . . .	39
3.15	Výsledky rýchlosti pre mobil . . . . .	40

## Zoznam tabuliek

1.1	Porovnanie RBAC so štandardnými prístupmi riadenia . . . . .	17
-----	--	----

# Úvod

Zámerom bakalárskej práce je navrhnúť a implementovať server manažmentu rolí vhodný pre komplexný *Workflow manažment systém*, ktorý by fungoval na princípe Petriho sietí.

Workflow manažment systém ponúka možnosť automatizovať procesy, ktoré sa riadia určitou logickou postupnosťou. Vo firemnej oblasti dokáže zrýchliť a zefektívniť vykonávanie určitej logistiky. WfMS oddeľuje procesnú časť od aplikačnej a dátovej časti, vďaka čomu sa zjednodušuje návrh, úprava a implementovanie procesov.

Dôležitou otázkou pri vytváraní WfMS je správa zdrojov. Treba určiť, kto môže a naopak, kto nesmie pristupovať ku konkrétnym prostriedkom, aby sa zachovala dôvernosť informácií a nemohol byť narušený systém. Vo WFSM je najpoužívanejší spôsob správa zdrojov založená na základe systému rolí.

Bakalárska práca pozostáva z troch hlavných kapitol. V prvej kapitole rozoberáme problematiku workflow manažmentu, Petriho sietí a riadenia prístupu. Sústreďujeme sa najmä na riadenie prístupu na základe rolí RBAC - Role Based Access Control, ktoré je najviac využívané vo Workflow systémoch, a to najmä vďaka tomu, že je založený na organizačnej štruktúre podnikov a poskytuje efektívne riadenie a údržbu právomocí.

V druhej kapitole prezentujeme návrh aplikácie Workflow manažment systému a jej architektúru. Systém pozostáva z viacerých častí, ktoré dokopy utvárajú komplexný celok. Každú časť podrobne vysvetľujeme s ohľadom na vzájomnú integráciu.

V tretej kapitole sa zaoberáme implementáciou serveru manažmentu rolí a používateľov. V tejto časti WfMS rozoberáme serverové riešenie riadenia prístupu na základe rolí. Aplikácia poskytuje dve hlavné funkcionality - mapovanie používateľov na roly vo firme a komunikačné rozhranie pre ukladanie rolí a referencií do databázy. Na konci tejto kapitoly testujeme efektívnosť a funkcionality aplikácie.

Na základe praktickej časti a jej testovania sme prišli na viaceré možné vylepšenia, ktoré bude možné aplikovať v budúcom vývoji Workflow manažment systému.

# 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 malé zaváhania môže spôsobiť obrovské oneskorenie a finančné straty. Vzniká preto 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 zdefinovať firemné procesy, pričom dokážu automaticky zabezpečiť správnosť vykonávania jednotlivých úloh v korektnom poradí. Vo firemnom prostredí nastane poriadok a zjednoduší sa administrácia. Hlavnou nevýhodou WfMS sú náklady na samotnú implementáciu. Väčšina WfMS sa vyvíja so zameraním na potreby konkrétneho podniku, pričom náklady na ich naprogramovanie presahujú finančnú spôsobilosť stredných a drobných podnikov. Na trhu sa postupne objavujú generické WfMS systémy, ktoré dokážu zabezpečiť využívanie procesov viacerými organizáciami súčasne.

Možným nástrojom na implementovanie WfMS sa ukazuje 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 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

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 ktorú úlohu. S narastajúcou veľkosťou podniku alebo zložitou procesov, manažment riadenia príde do bodu, kedy je len veľmi ťažko ovládateľný. Naskytuje sa preto potreba, aby sa celý manažment zautomatizoval. Následne teda vzniká 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 [5].

Workflow systémy sú založené na jednotlivých prípadoch (cases). Ako jednotlivý prípad si môžeme predstaviť konkrétnu požiadavku, ako je napríklad založenie si účtu, objednávka, spísanie závetu 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.

Každý prípad vo WfMS má svoj začiatok a koniec. Medzitým sa proces nachádza v určitom stave, ktorý je definovaný prostredníctvom [5] :

- architektúry daného prípadu - určuje nám ako sa môže daný prípad vyvíjať.
- doposiaľ splnených podmienok - vďaka tomu zistíme 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. Definuje nám, čo treba vykonať aby sa prípad posunul. Ako taká je nedeliteľná a v systéme sa musí vždy vykonať ako celok. V prípade neúspechu úlohy treba úlohu spustiť celú odznova. Je však možné rozlišovať medzi spustením a ukončením úlohy.

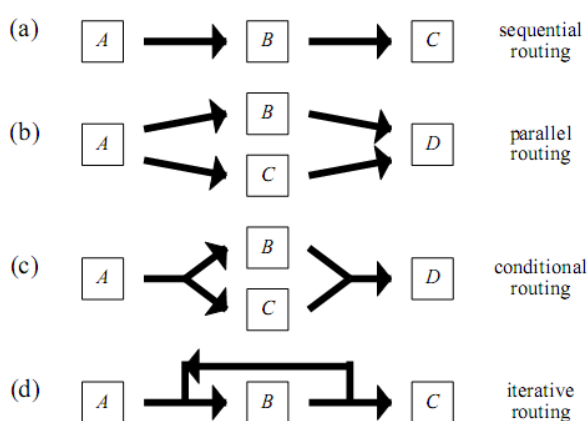
### 1.1.2 Proces

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 vybavovania 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 umožníme vytvoriť pre prípad odlišné životné cykly. Využívame štyri základné konštrukcie: (obrázok 1.1 ) [2]

- 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



Obr. 1.1: Smerovacie konštrukcie vo workflow systémoch podľa [2]



### 1.1.4 Výhody použitia WfMS

Workflow management systém ponúka možnosť oddeliť procesnú časť. Podľa [6] sa tým zabezpečia nasledovné výhody:

- zabezpečí sa jednotná funkcionálna manažmentu a oddelí sa od zvyšku systému. Tradične sa táto funkcionálna rozprestiera v celom systéme. Vďaka tejto separácii je možné rovnakú funkcionálnu použiť na viac ako jeden proces.
- aplikácie sú nezávislé od manažmentu, čo umožňuje upravovať logiku aj po implementovaní systému
- k logickej vrstve je možné pridávať rôzne aplikácie a rozšíriť funkcionálnu systému
- definovanie a návrh procesu je možné diagnostikovať a vopred zabrániť nežiaducim chybám
- z pohľadu manažmentu je možné v konkrétnom prípade identifikovať, v akom stave sa prípad nachádza. Celý proces sa následne ľahšie monitoruje. Dá sa zistiť, kto má v danom prípade a čase čo vykonávať. Celé vykonávanie procesu je pod drobnohľadom, čím sa urýchlí čas vykonania celého procesu. V prípade problému vo vykonávaní prípadu, je možné rýchlo identifikovať zdroj.

### 1.1.5 Riadenie zdrojov vo WfMS

Pod pojmom zdroj rozumieme jednotku, ktorá je určená na vykonávanie úlohy. Môžeme si pod tým predstaviť počítač (server, tlačiareň, fax, atď. ) alebo samotného človeka. 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 rolí. Týmto spôsobom sa oddelí návrh procesu od jednotlivých používateľov. Taktiež sa tým uľahčí administrácia a zabráni sa deadlockom.

## 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 "Communi-

cation with Automata"[16] v roku 1962. Časom sa však Petriho siete menili a dopĺňali. Dnes už podľa [1] poznáme viacero definícií Petriho sietí.

### 1.2.2 Definícia Petriho siete

Petriho sieť je podľa [15] zvláštny typ orientovaného grafu spolu so začiatočným stavom (začiatočným značkováním) -  $PN = (N, M_0)$ . 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.

Rozlišujeme:

- 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 - priraduje 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 [14]

Petriho sieť je orientovaný bipartitný graf reprezentovaný päticou  $(P, T, F, W, m_0)$ , kde:

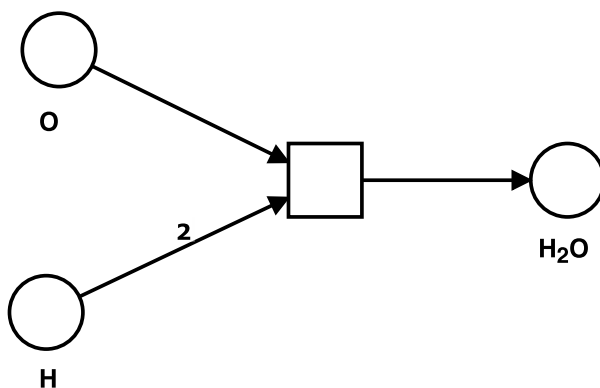
- $P = \{p_1, \dots, p_n\}$  - konečná neprázdna množina miest;
- $T = \{t_1, \dots, t_n\}$  - konečná neprázdna množina prechodov;
- $P \cap T = \emptyset, P \cup T \neq \emptyset$ ;
- $F \subseteq (PxT) \cup (TxP)$  - množina hrán (toková relácia);
- $W : F \rightarrow N \cup \{0\}$  - váhová funkcia ;
- $m_0 : P \rightarrow N \cup \{0\}$  - počiatkové značkovanie;

## 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  $t$ ;

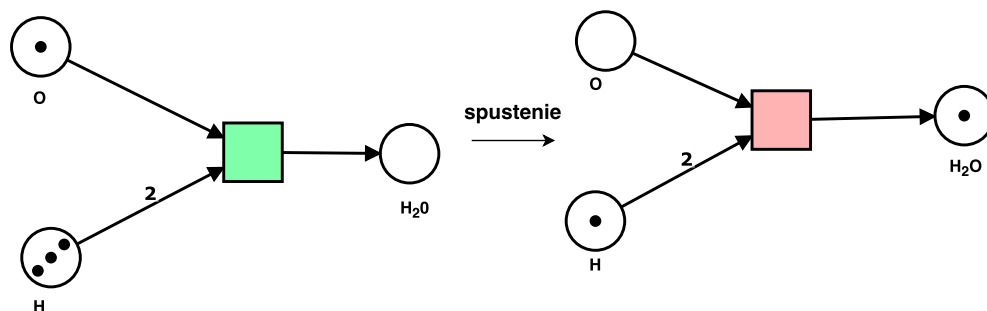
### 1.2.3 Príklad

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  $H_2O$ . 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  $H_2O$ , ktorá nám značí, že po skonzumovaní značiek z miest O a H sa vytvorí práve jedna značka v mieste  $H_2O$  (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 násobnosťou hrany. Naopak vo výstupnom mieste sa značky podľa násobnosti hrany pridajú.



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

### 1.2.4 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 [6] vysvetlil 3 hlavné výhody použitia Petriho sietí na modelovanie workflow management systému:

- Formálna sémantika spolu s grafickou reprezentáciou
- 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

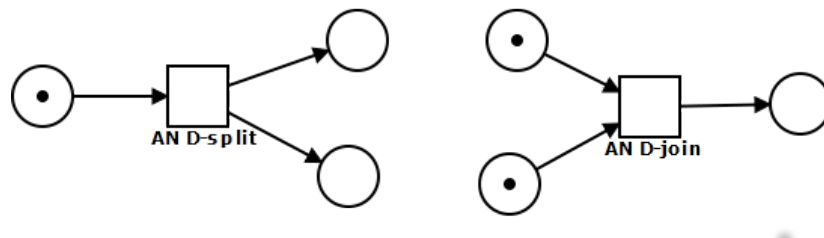
Mapovanie WfMS na Petriho siete je možné ľahko ukázať. Úlohy sú mapované na prechody. Závislosti sa mapujú prostredníctvom miest a hrán. Jednotlivé prípady môžu byť odlišené rôznou farbou tokenov v rozšírených Petriho sieťach. Stav, v akom sa prípad nachádza, definuje rozloženie tokenov v miestach. Petriho sieť, ktorá modeluje WfMS sa nazýva workflow sieť. Aby Petriho sieť bola súčasne WF-sieťou, musí podľa [2] 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  $Y$ .

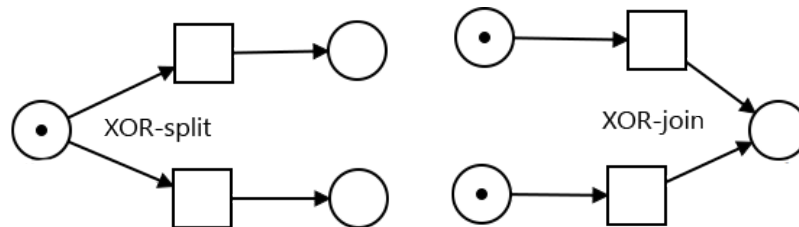
### Smerovacie konštrukcie

Smerovacia konštrukcia AND sa rozvetví prostredníctvom prechodu, ktorý smeruje do viacerých nezávislých miest a na konci sa rovnaký počet miest spojí do koncového prechodu (AND-join). AND konštrukcia umožní dve a viac paralelných vetiev v procese.



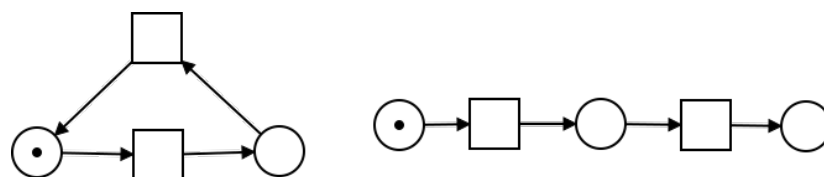
Obr. 1.4: AND konštrukcia v Petriho sieti

Smerovacia konštrukcia XOR funguje opačne. Pre rozvetvenie (XOR-split) sa použije spojenie jedného miesta s viacerými prechodmi , pričom jednotlivé vetvy sa následne spoja prostredníctvom prechodov do miesta. XOR konštrukcia umožňuje výber medzi dvomi vetvami procesu.



Obr. 1.5: XOR konštrukcia v Petriho sieti

Ako sekvenčné smerovanie sa používa jednoduché striedanie miest a prechodov, pokiaľ sa chceme v procese vrátiť späť a použiť iteračné smerovanie, vytvoríme spojenie prostredníctvom jedného prechodu medzi miestom, v ktorom sa nachádzame a miestom, do ktorého sa chceme vrátiť.



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 [4]. 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 sa dá ľahko pomýliť 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 pochopí rozdiel 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, ktorá obsahuje 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 predstavuje 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 [7] 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 operácie.

**Jednoduchosť mechanizmu** Dizajn by mal byť dostatočne zrozumiteľný , aby bolo možné dokázať jeho správne fungovanie. Taktiež by mal byť dostatočne jednoduchý, aby bolo možné prípadné 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. Systém teda priraduje právomoci len subjektom, ktoré majú priamo zadefinované prístupové práva. 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.

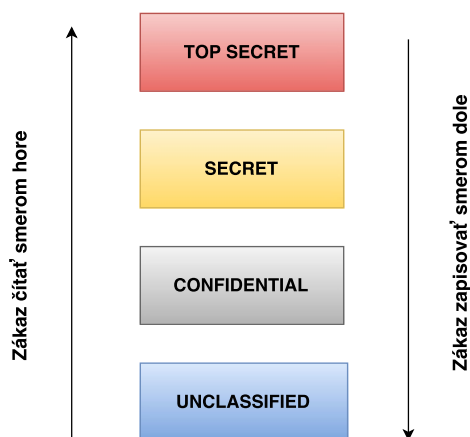
**Mechanizmus najmenej právomoci** V dizajne by sa malo minimalizovať zdieľanie viacerými používateľmi. Implementácia tohto princípu vyžaduje fyzické alebo logické oddelenie systémov

**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 vznikol model Bell-LaPadula [8] (obrázok 1.7). Princíp tohto modelu je jednoduchý a matematicky jasne zadaný. 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 list prezidentovi a zároveň vyžadujeme, aby iba prezident mohol dopis prečítať (pošleme mu ho), ale neoprávňujeme nás to čítať prísne tajné dokumenty.
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 pozíciou jadrových hlavíc. Pridáva sa preto takzvaný *Access Control Matrix*, čiže sa dá vyhľadať v matici podľa používateľa a objektu, aké má daný používateľ nad objektom právomoci.



Obr. 1.7: Model Bell-LaPadula

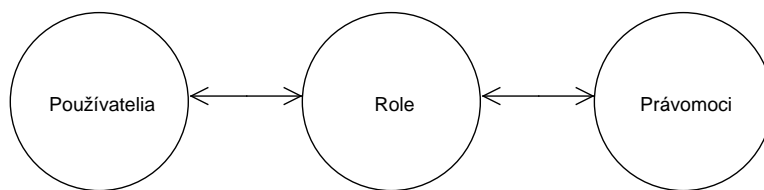


## 1.4 RBAC

### 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.



Obr. 1.8: Vzťahy RBAC podľa [4]

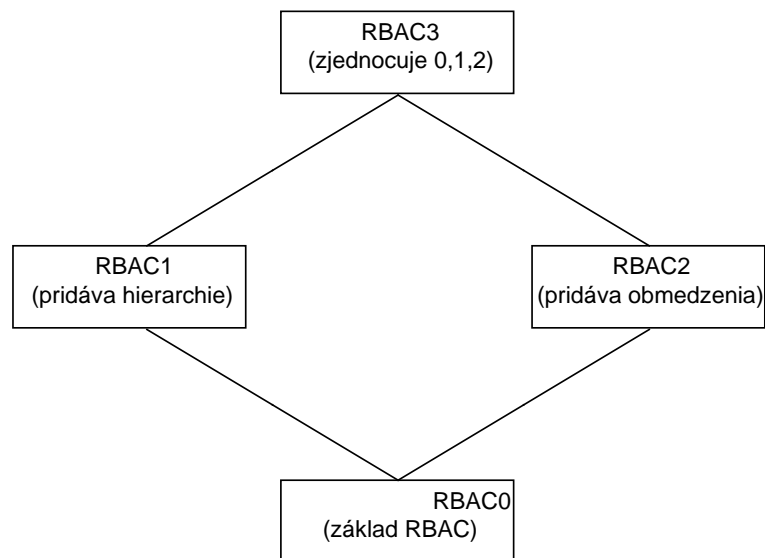
### 1.4.2 Dizajn

V jednej organizácii môže byť viacero rolí, ktoré vykonávajú rôzne funkcie. Práva na vykonávanie určitých operácií 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 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 zabezpečí, že subjekt môže vykonávať iba povolené operácie.

V nasledujúcej publikácii [10] NIST rozobrala RBAC viac do detailov, navrhla prídavné funkcionality a zahrnula špecifické formy vzťahov tak, aby spĺňali požiadavky na oddelenie právomocí. V roku 1996 *Sandhu a spol.* [9] predstavili framework *RBAC96* založený na RBAC a rozložil RBAC do štyroch koncepčných modelov (obrázok 1.9). 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 následne 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).

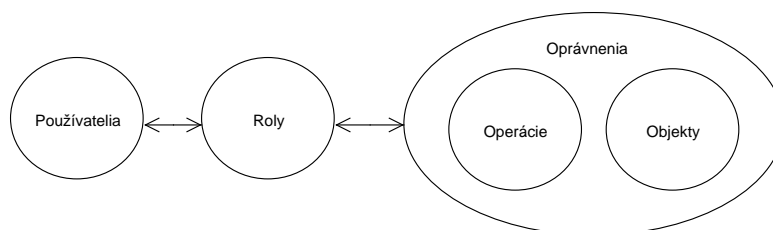


Obr. 1.9: Framework RBAC96 podľa Sandhu a spol. [9]

## 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 operácií aplikovaných na objekty. Základ RBAC pozostáva z definovania oprávnení pre roly, pričom používatelia sú priradení k rolám, a tým nadobúdajú právomoci patriace rolám. Obrázok 1.8 ukazuje vzťah užívateľov, rolí a oprávnení. Dvojitá šípka značí vzťah m:n. Jeden používateľ môže byť

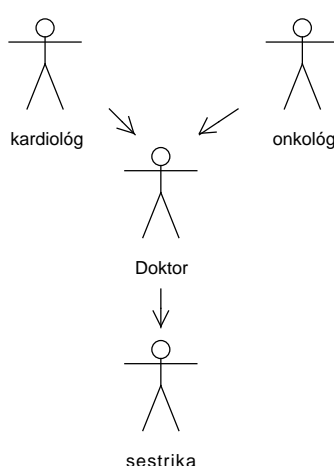
priradený k viacerým roliam a zároveň jedna rola môže pozostávať z mnohých používateľov. Tento princíp predstavuje flexibilné riešenie priradenia práv rolám a používateľov k rolám, čím podporuje princíp minimálnych oprávnení. Každé oprávnenie predstavuje kombináciu objektov a operácií. Celý systém oprávnenia na základe rol môžeme vidieť na obrázku 1.10.



Obr. 1.10: Rozšírené vzťahy RBAC podľa [4]

### 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 splnomocnenie nadradených rol. Môžeme si predstaviť napríklad juniorské a seniorské pozície v organizácii. Seniorská pozícia by mala 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. 1.11: Príklad hierarchie rolí

## Obmedzenia

Do modelu RBAC je možné pridať ďalšie obmedzenia, ktoré bližšie definujú politiku prístupu práv. Medzi základné obmedzenia patrí vylúčenie práv. V niektorých prípadoch chceme zamedziť, aby používateľ mohol pristupovať súčasne k dvom objektom. American National Standard for Telecommunications [11] definuje vylúčenie práv ako rozdelenie zodpovednosti pre citlivé údaje tak, aby žiaden jednotlivец 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) .

#### 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 právomoci, pretože treba prejsť všetky možné objekty v danom systéme.

#### Voliteľné riadenie prístupu DAC

Podľa [12] 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

prístupovať, môže dáta zmeniť alebo rozšíriť práva pre neautorizovaného používateľa .

### Povinné riadenie prístupu MAC

Ako riešenie pre hlavný problém DAC architektúry, povinné riadenie prístupu MAC, definuje hierarchické úrovne bezpečnosti [13]. Vychýľuje sa tak od štandardných prístupov a implementuje MLS (multilevel security). Iba administrátori sú oprávnení spravovať a priradovať prístupové práva. Ku každému objektu, ako aj subjektu, sa priradí určitá bezpečnostná 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.

### Porovnanie RBAC so štandardnými systémami riadenia

#### RBAC

- Rola nemusí obsahovať žiadnych používateľov
- Rola sa ľahko mapuje na štruktúru organizácie
- Prístupové práva sa mapujú na roly
- Používateľom sú priradené role

#### Štandardné systémy riadenia

- 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

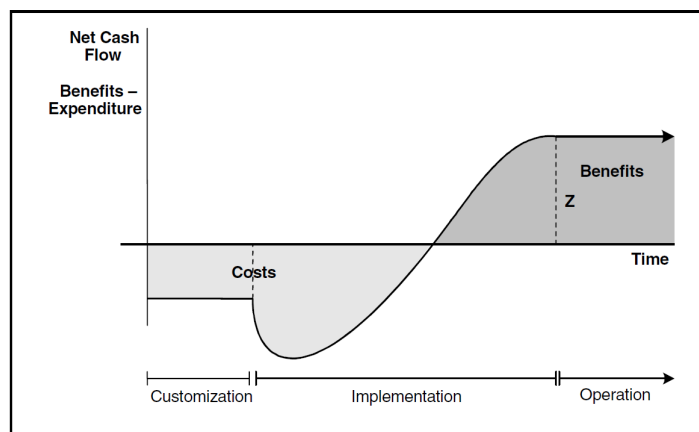
Tabuľka 1.1: Porovnanie RBAC so štandardnými prístupmi riadenia

#### 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 [3] NIST popísala ekono-

mickú návratnosť zavedenia RBAC do systému. Môžeme vidieť (obr. 1.12), 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. 1.12: Ekonomická návratnosť RBAC podľa [3]

# Opis riešenia - Workflow systém

## 2.1 Špecifikácia požiadaviek

### 2.1.1 Základný opis aplikácie

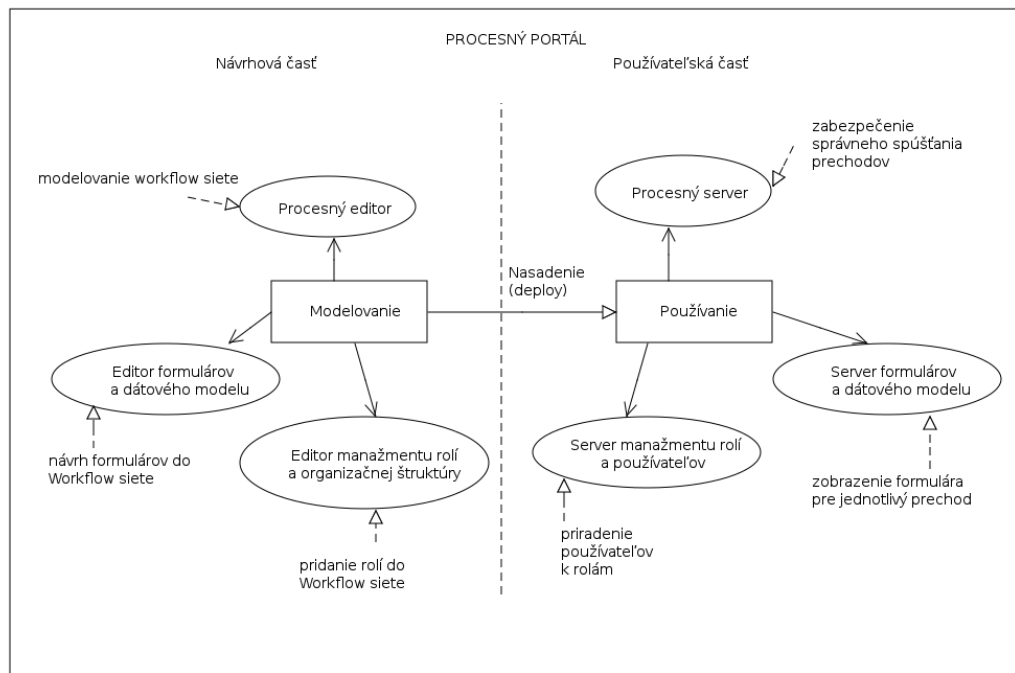
Zistili sme, že workflow management systém je významný prostriedok, ktorý umožní používateľovi oddeliť a jasne znázorniť procesnú časť od aplikácie. Tento systém ponúka mnohé výhody. Jednou z možností ako WfMS implementovať je použitím Petriho sietí. Petriho siete sú jasne formálne definované a matematicky overené. Rozhodli sme sa preto implementovať Petriho siete ako nástroj pre modelovanie procesov.

Celý systém bude fungovať na portáli, cez ktorý bude možné sa registrovať a následne prihlásiť. Aplikáciu môžeme pomyselne rozdeliť na modelovaciu a používateľskú časť (obrázok 2.1). V modelovacej časti je možné navrhovať a modelovať procesy prostredníctvom Petriho sietí. Ku nim sa v *editore manažmentu rolí a organizačnej štruktúry* pripoja role a v *editore formulárov a dátového modelu* sa namapujú formuláre. Takto vytvorená workflow sieť spolu s rolami a formulármi je pripravená na používanie.

Pre používanie potom stačí workflow sieť nasadiť (deploy) do konkrétnej firmy. Vo firme bude potrebné priradiť konkrétnych používateľov ku rolám, ktoré táto workflow sieť používa. Následne tak môžeme považovať workflow systém za funkčný a budeme môcť vytvárať nové prípady. Prechody sa budú dať spúšťať v každom prípade automaticky podľa namodelovanej logiky.

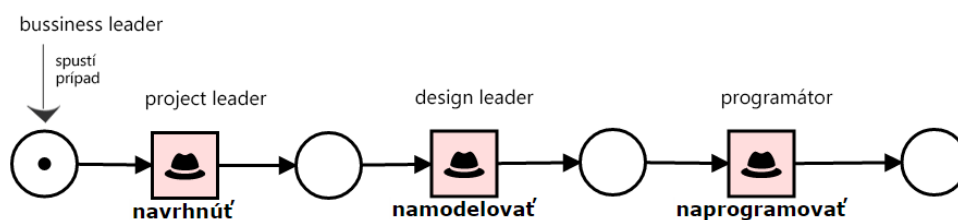
### 2.1.2 Funkcia rolí v aplikácii

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 nové právomoci. Pre efektívny manažment riadenia takéhoto systému nám nestačí využitie klasických modelov, respektíve ich nasadenie by bolo vysoko nákladné. Z týchto dôvodov väčšina workflow manažment



Obr. 2.1: Model Workflow systému

systémov využíva model prístupu na základe rolí RBAC. Pre potreby našej aplikácie preto využijeme základy tohoto modelu (konkrétne implementujeme základný model z frameworku RBAC0 [9]). V každej Workflow sieti priradíme rolu na sieť, 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 role nad prechodom sú určené v dátovej časti. Vo formulári ku prechodu sa dajú políčka nastaviť ako povinné, upraviteľné a viditeľné. To nám zabezpečí ekvivalent k právomociam read, write.



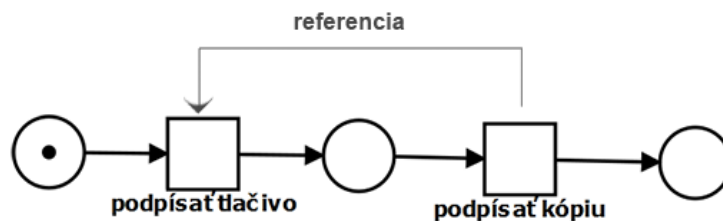
Obr. 2.2: Príklad priradenia rolí k prechodom (vytvorenie novej aplikácie zjednodušene)



### 2.1.3 Referencie

Role v procese zabezpečia prenos prístupových práv z úloh na používateľa. V rámci jednej role si môžeme predstaviť skupinu používateľov, 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 používateľ. Samotné role túto funkcionality nedokážu zabezpečiť. V našej aplikácii je nutné ju explicitne zadať. K tomuto účelu sme použili systém referencií v Petriho sieti. Konkrétne sme implementovali dva typy referencií : *referencia na prechod* a *referencia na prvého používateľa z role*.

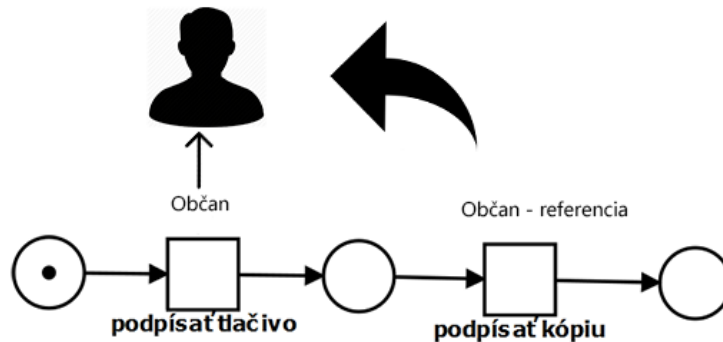
V prvom kroku opíšeme **referenciu na prechod**. Referencia na prechod zabezpečí, aby ten samý používateľ, ktorý spustí prechod, na ktorý odkazuje referencia, bol jediný oprávnený na spustenie prechodu s touto referenciou. Jednoduchým príkladom je, ak rozložíme jeden zložitý prechod, ktorý musí vykonať ten samý používateľ na dve menšie. Predstavme si napríklad podpísanie tlačiva (obrázok 2.3). Ku tlačivu treba podpísať aj jeho kópiu. Naš 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.



Obr. 2.3: Referencia na prechod

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 je, že v procese vopred nevieme určiť poradie vykonávania jednotlivých úloh. Chceme, aby prechod1 aj prechod2 spustil rovnaký používateľ z role. Nevieme však, v akom poradí budú prechody za sebou nasledovať. Z týchto dôvodov sme sa rozhodli pridať *referenciu na prvého používateľa z role*. Táto

referencia rieši obidva problémy zároveň. Prechod, ktorý bude označený touto referenciou, bude môcť spustiť jedine používateľ, ktorý v procese prvýkrát spustil prechod pod takou rolou, akú má prechod 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 používateľ, ktorý je priradený k danej role.



Obr. 2.4: Referencia na prvého z role

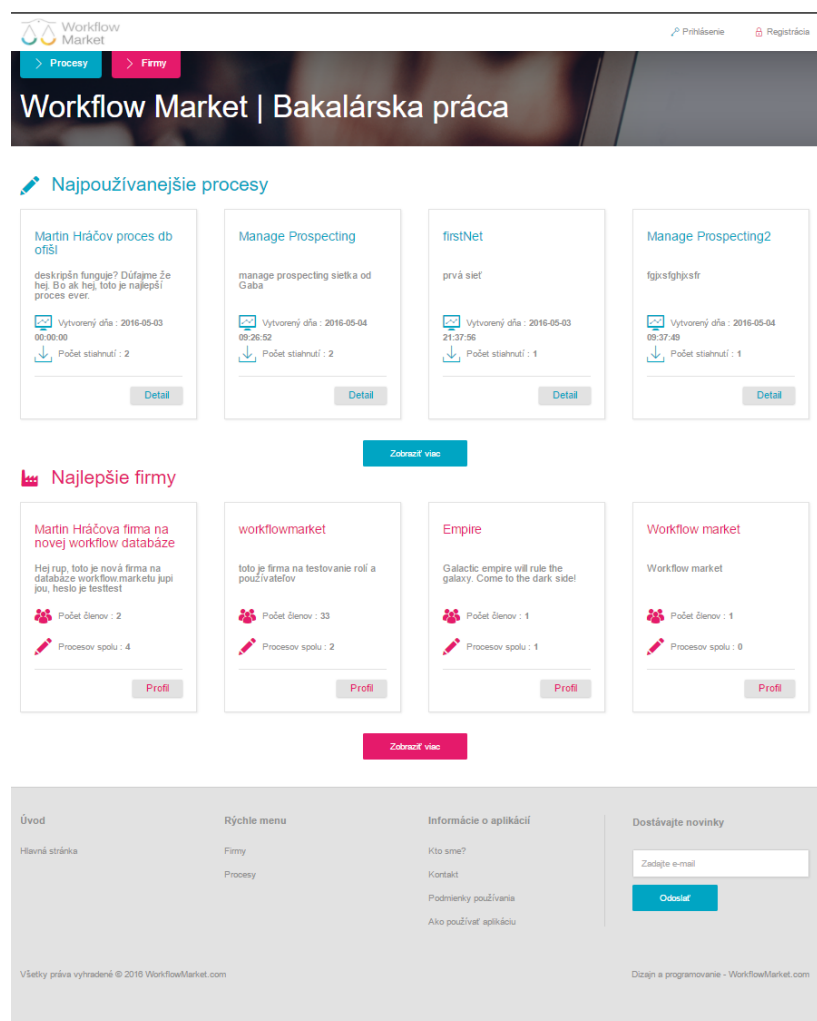
## 2.2 Návrh

V nasledujúcej sekcii bližšie popíšeme jednotlivé časti, ktoré navrhujú jednotliví zúčastnení študenti v rámci spoločného Workflow manažment systému.

### 2.2.1 Procesný portál - Martin Kováčik

Procesný portál slúži na zjednotenie všetkých častí dohromady. Základnou myšlienkou portálu je poskytnúť možnosť vytvárať, využívať a manažovať workflow procesy. Taktiež slúži ako e-shop. Ponúka možnosť vytvorenej workflow siete kopírovať do iných firiem. Všetky firmy a workflow siete sú verejne viditeľné. Vytvára priestor, kde sa dajú procesy vzájomne vymieňať. Vďaka tomu môžu používatelia využívať vo svojich firmách workflow systém bez nutnej znalosti samotného modelovania workflow siete. Portál umožňuje registráciu a prihlásenie používateľov. Prihlásený používateľ môže manažovať svoj profil: prezeráť svoje objednávky, meniť heslo. V prípade straty hesla zabezpečí jeho opätovné zaslanie. Každý používateľ môže vytvoriť a spravovať vlastnú firmu. Vo firme je zahrnutá správa používateľov, ich pridávanie a vyradovanie z firmy. Administrátor určí každému používateľovi ohraničenie prístupu na správu firmy. Nového používateľa je možné pridať do firmy cez zaslanie pozvánky. Zaregistrovaní používatelia majú vstup do firmy voľný, prípadne zabezpečený cez heslo. Portál poskytuje

prístup do modelovania workflow sietí spolu s editorom manažmentu rolí a editorom formulárov, ako aj k správe rolí a ich mapovaniu na používateľov.

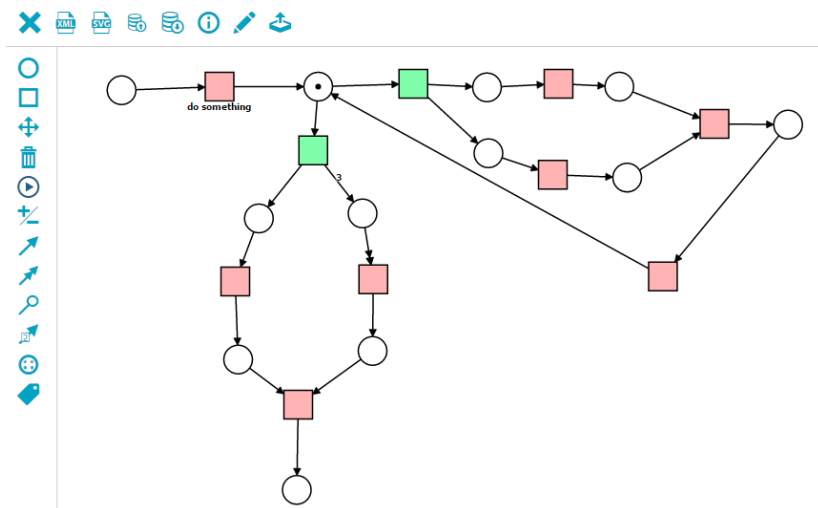


Obr. 2.5: Procesný portál

## 2.2.2 Procesný editor - Tomáš Priboj

Procesný editor ponúka možnosť modelovať Petriho sieť a navrhnúť tak workflow sieť pre vytvorenie firemných procesov. Nájdeme v ňom nástroje na kreslenie rôznych objektov ako miest, prechodov a hrán. Každý takýto objekt je možné presunúť a vymazať. Pri presúvaní je automaticky zabezpečené presúvanie hrán napojených na objekt. Vymazanie objektu analogicky zaručí odstránenie hrán, ktoré by inak nemali začiatkový alebo koncový bod. Každé miesto a prechod je možné označiť menovkou. Udeľovanie značkovania (tokenov) pre miesta je zabezpečené dvoma spôsobmi. Prvý spôsob pridáva značky ľavým tlačidlom myši a pravým ich uberať. Druhý spôsob umožní k miestu ručne napísať počet značiek. Dôležitou funkcionalitou je simulácia Petriho siete. Dovoľuje používateľovi simulovať proces v Petriho sieti, pričom v každom stave zobrazuje

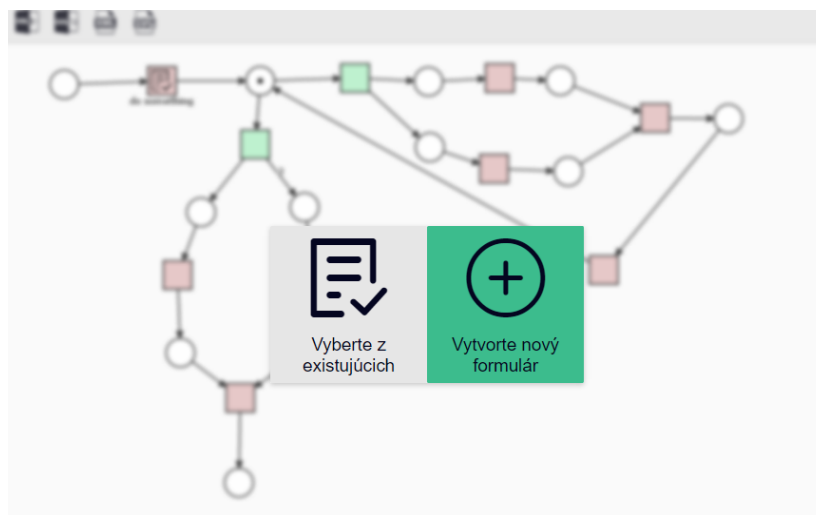
aktívne prechody. Aplikácia umožňuje ukladanie siete buď vo formáte SVG ako obrázok, alebo vo formáte XML, z ktorého je možné neskôr celú sieť importovať na účely modelovania. Pre potreby Workflow systému, editor ukladá popis workflow siete a cez komunikáciu prostredníctvom JSON-u s procesným serverom umožňuje uložiť sieť do databázy. Po uložení do databázy prijme ID, ktoré bolo sieti priradené. Následne môže používateľa presmerovať na editor formulárov a dátového modelu.



Obr. 2.6: Procesný editor

### 2.2.3 Editor formulárov a dátového modelu - Ján Polaček

Po tom, ako sa workflow sieť nakreslí v procesnom editore, je potrebné k prechodom priradiť formuláre, ktoré sa budú neskôr zobrazovať pri spracovaní úloh v prípadoch. Editor formulárov a dátového modelu umožňuje navrhnuť dátový model pre formuláre. Modelovanie formulárov je realizované cez vytváranie, editovanie, prípadne mazanie vstupných polí. Tie môžu byť pridané v podobe *krátkej odpovede* (text input), *dlhej odpovede* (textarea) , *výberu z viacerých* (select boxu) alebo *zaškrtnutých možností* (checkbox) . Ku každému sa dá pridať otázka a popis. Pre jednotlivý vstup definujeme, či je povinný, viditeľný a upraviteľný. Pre potreby znovupoužitia formulárov a ich ukladania do databázy sa vytvorí dataset, ktorý reprezentuje premenné, do ktorých sa ukladajú hodnoty z vstupných polí. Editor dokáže načítať a uložiť Petriho sieť cez formát SVG. Jednotlivé formuláre a dátové typy je možné uložiť vo formáte XML a následne spätne nahráť na načítanú sieť (napr. prostredníctvom SVG). Editor spolupracuje so *serverom formulárov a dátového modelu*. Prostredníctvom JSON komunikácie sú formuláre priebežne ukladané na server.

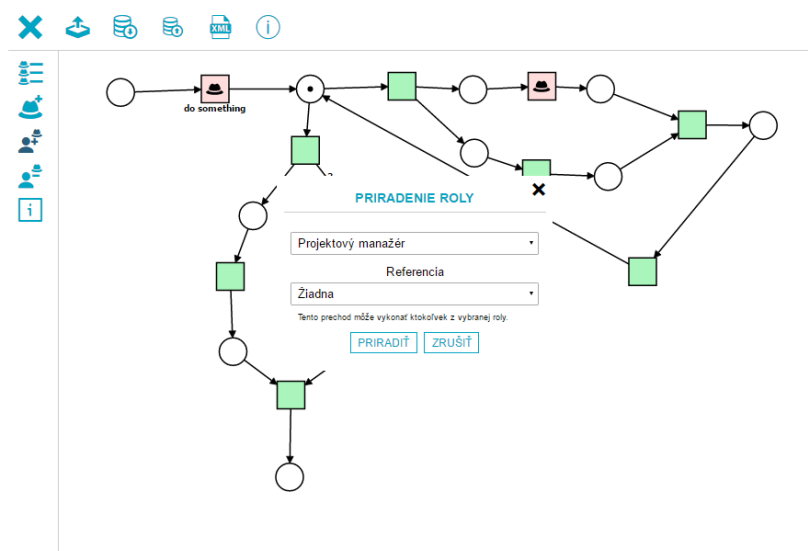


Obr. 2.7: Editor formulárov a dátového modelu

Obr. 2.8: Vytváranie formulára

## 2.2.4 Editor manažmentu rolí a organizačnej štruktúry- Kristián Stroka

Podobne ako pri editore formulárov, *Editor manažmentu rolí a organizačnej štruktúry* slúži na bližšie definovanie prechodov. Konkrétne sa zameriava na otázku, kto môže spustiť prípad a prechody v ňom. Editor umožňuje vytvárať roly a následne každému prechodu vymedziť prostredníctvom rol a referencií práva na spustenie prechodu. Každému prechodu je možné priradiť rolu, prípadne jednu z dvoch referencií : *referenciu na prechod* a *referenciu na prvého používateľa z role*. Toto priradenie je možné následne zrušiť. Editor načítava Petriho sieť prostredníctvom SVG formátu načítaním zo súboru, prípadne z databázy. Mapovanie rolí a referencií je možné uložiť v XML formáte. Pre potreby Workflow systému je ukladanie mapovania do databázy zabezpečené prostredníctvom JSON komunikácie so *serverom manažmentu rolí a používateľov*. Rovnakým spôsobom je možné z databázy načítať roly vo firme, v akej používateľ práve modeluje.



Obr. 2.9: Editor manažmentu rolí a organizačnej štruktúry

### 2.2.5 Procesný server - Anna Demeterová

Procesný server zabezpečuje správne fungovanie Workflow systému po jeho namodelovaní vo firme. Pre konkrétného používateľa vo firme zabezpečí, aby mohol vytvárať nové prípady a preberať úlohy z vytvorených prípadov podľa stavu, v akom sa prípad nachádza a používateľských oprávnení zadaných na základe rolí. Stará sa o to, aby každý prípad nasledoval logiku definovanú modelovanou workflow sieťou. Kontroluje a zabraňuje možným kolíziám referencií a iným deadlockom. Ukladá históriu vykonaných prípadov a úloh. Umožňuje vizualizáciu stavu, v akom sa prípad nachádza. Okrem toho spracováva workflow siete vytvorené v *procesnom editore* a ukladá vytvorenú sieť do databázy. Po prebratí úlohy používateľovi zobrazí možnosť túto úlohu vykonať. To ho presmeruje na *server formulárov a dátového modelu*, kde sa používateľovi zobrazí formulár patriaci danej úlohe. Ak je formulár vyplnený, odošle sa naspäť potvrdenie. Procesný server dokončí úlohu a zabezpečí posunutie prípadu do nového stavu. Procesný server nanovo vypočíta aktívne prechody na spustenie.

Vyskúšajte našu novinku - MARKET +

Kontakt

Obchodné podmienky

Ako nakupovať

+421 910 553 932

Prihlásenie

Registrácia

Vytvor case

Vezmi úlohu

Moje úlohy

História úloh

História case-ov

Aktuálne case-y

Case	Task	Začaté	Formulár	Ukončiť	Opustiť úlohu
manage	Požiadavka na informatívnu kalkuláciu je zadaná	2016-04-24 17:52:13	<div>Zobraz</div>	<div>Dokončiť</div>	<div>Cancel</div>
ManageProspect	Rozdelenie tendrov na ďalšie posúdenie	2016-04-29 13:33:11	<div>Zobraz</div>	<div>Dokončiť</div>	<div>Cancel</div>
Pripad odvolania	Podanie odvolania k rozhodnutiu	2016-04-29 13:33:57	<div>Zobraz</div>	<div>Dokončiť</div>	<div>Cancel</div>

Úvodné informácie

Kto sme?

Kontakt

Úvod

Úvodné informácie

Kto sme?

Kontakt

Úvod

Úvodné informácie

Kto sme?

Kontakt

Úvod

Dostávajúte novinky

Subscribovať

Všetky práva vyhradené © 2016 WorkflowMarket.com

Dizajn a programovanie - WorkflowMarket.com

Obr. 2.10: Procesný server

## 2.2.6 Server formulárov a dátového modelu- Jakub Kováčik

Serverová strana formulárov a dátového modelu rieši zobrazovanie formulárov pre spustené úlohy v konkrétnom prípade. Formulár môže obsahovať krátku odpoveď, dlhú odpoveď, zaškrtnávacie pole a výber z viacerých možností. Po odoslaní formuláru prebieha validácia, či sú vstupné polia správne vyplnené. Taktiež umožňuje priebežne uložiť vyplnené polia, bez toho, aby sa formulár odoslal. To umožní používateľovi odložiť vyplňanie na neskôr. Táto časť je pevne previazaná s editorom formulárov (kapitola 2.2.3). Zatiaľ čo editor rieši modelovanie dátového modelu a návrh formuláru, serverová časť sa stará o ukladanie dát a následné zrekonštruovanie formuláru pre jeho samotné vyplňanie. Komunikácia medzi editorom a serverom prebieha prostredníctvom JSON dát vymieňaných cez AJAX. Pre editor zabezpečuje uloženie a načítanie v rámci databázy.

×

Údaje zákazníka

Meno \*

Ladislav

Priezvisko \*

Pohlavie \*

Muž

Domáce zvieratká \*

Zvoľte Vaše domáce zvieratká

Mačka

Pes

Iné

Ulož formulár

Odošli formulár

Obr. 2.11: Server formulárov a dátového modelu- Jakub Kováčik

# Opis riešenia - Server manažmentu rolí a používateľov

## 3.1 Určenie požiadaviek

Aplikačným výstupom tejto bakalárskej práce je web stránka, ktorej účelom je vytvoriť jednoduché a intuitívne aplikačné rozhranie pre priradenie používateľov k rolám. Hlavné funkcionality zahŕňajú:

- Zobrazenie všetkých používateľov vo firme
- Zobrazenie takých používateľov, ktorí nemajú vo firme priradenú rolu
- Zatriedenie používateľov podľa príslušnosti k rolám vo firme
- Utriedenie používateľov podľa rôznych kritérií - ID, meno, priezvisko, email
- Priradiť používateľov k role vo firme
- Odobrať používateľov z role vo firme
- Vytvoriť vo firme novú rolu
- Pridať do firmy role prostredníctvom XML súboru
- Odstrániť rolu z firmy - pri odstránení roly z firmy, treba dať pozor, aby sme nemohli odstrániť také, ktoré firma práve používa na vykonávanie vlastných procesov

Zároveň má aplikácia poskytovať rozhranie k *editoru rolí a organizačnej štruktúry* pre:

- Načítanie rolí z firmy
- Uloženie mapovania rolí ku Petriho sieti - uloženie rolí a referencií

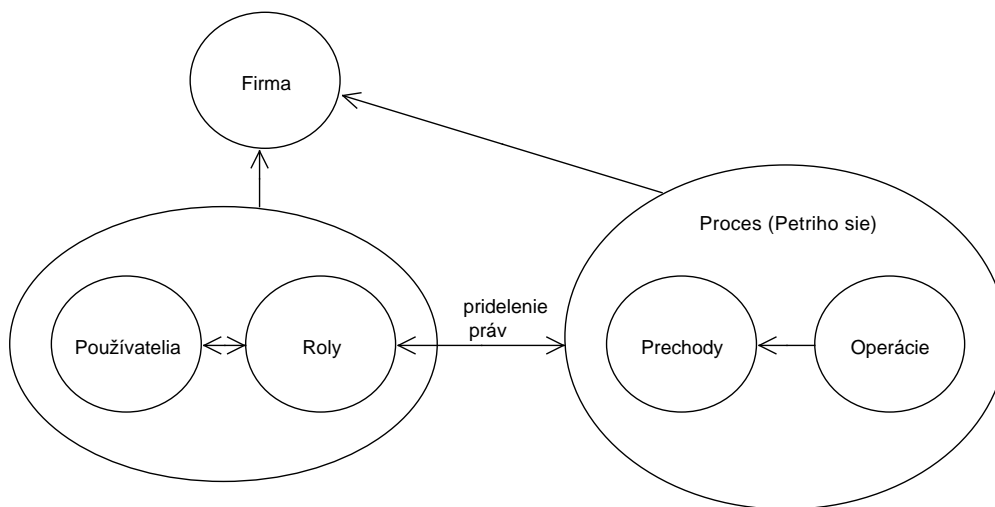


## 3.2 Návrh

### 3.2.1 Popis architektúry

Základnou myšlienkou RBAC architektúry je odstrániť priame priradovanie práv k používateľom. Tento výsledok sa zabezpečí pridaním medzikroku, a teda rolí medzi samotných používateľov a ich právomoci. Samotná implementácia RBAC značne závisí od konkrétneho systému. Naša aplikácia sa zameriava na vytvorenie WfMS za pomoci Petriho sietí.

Ako prvé si definujeme základnú architektúru. Na obrázku 3.1 môžeme zreteľne vidieť dve nezávislé časti fungovania Workflow manažment 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 používateľov. Priradovanie práv je zabezpečené väzbou medzi rolami a procesmi. Pre správnu funkcionality je však potrebné, aby firma mala priradené také roly, ktoré sú použité v jednotlivých procesoch, ktoré firma využíva. Definovanie prístupových práv je zabezpečené nad samotnými prechodmi v sieti.



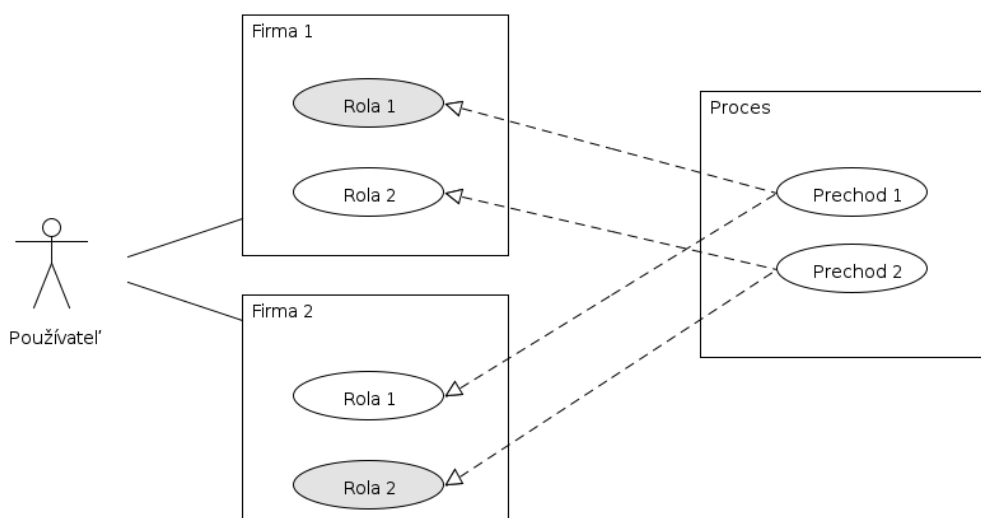
Obr. 3.1: Model RBAC v aplikácii

#### Priradovanie používateľov k rolám

V našej aplikácii nechceme, aby bol používateľ 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 používatelia neviazali len na samotnú rolu. V aplikácii bude väzba používateľa na rolu závislá od konkrétnej firmy. V každej firme bude môcť administrátor - používateľ s právami na riadenie rolí, mať možnosť priradiť používateľa ku konkrétnej role, ktorá je vo firme zaradená. Samotný používateľ môže

byť týmto spôsobom priradený vo viacerých firmách, pričom v každej firme bude mať iné práva.

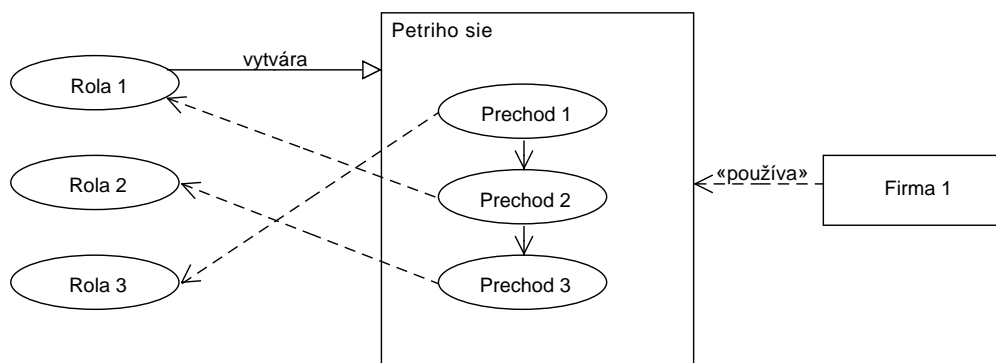
Na obrázku 3.2 môžeme vidieť zjednodušený model mapovania právomocí používateľa prostredníctvom systému rolí. Šedé pozadie v roli znamená, že používateľ je k role priradený. V rovnakom procese vidíme, že používateľ, ktorý môže vo firme 1 spustiť prechod 1, nie je oprávnený vykonať prechod 1 aj vo firme 2, pretože v nej nemá priradenú rolu. Vo firme 2 môže spustiť iba prechod 2.



Obr. 3.2: Mapovanie používateľov na roly vo firme

### Priradenie práv k rolám

Priradenie práv k rolám je definované nepriamo prostredníctvom jednotlivých prechodov. Hlavnou úlohou roly v procese je zdefinovať 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 stavom, v akom sa tokeny momentálne nachádzajú.



Obr. 3.3: Právomoci rolí v sieti

### 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 role, ktorej prináleží daný prechod
3. osoba spĺňa požiadavky referencie

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 používateľom, ktorý spustil referovaný prechod. V prípade, že tieto dáta súhlasia, používateľ je oprávnený spustiť prechod.

Ak má prechod nastavenú *referenciu na prvého používateľa*, overí sa, či sa v procese už daná rola nevyskytla. Ak prechod s danou rolou v danom prípade ešte nebol spustený, používateľ je oprávnený tento prechod spustiť. V opačnom prípade je potrebné overiť ID používateľa s používateľom, ktorý ako prvý spustil prechod s touto rolou. Ak sa zistí zhoda, používateľ 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 (kapitola 2.2.3). Vo formulári je možné nastaviť, aké dáta budú používateľovi prístupné, či ich bude môcť používateľ iba zobrazovať, alebo aj editovať. Takisto sa definuje, ktoré údaje je potrebné vyplniť, aby mohol používateľ tento prechod dokončiť.




Povinné	<input checked="" type="checkbox"/>
Upraviteľné	<input checked="" type="checkbox"/>
Viditeľné	<input checked="" type="checkbox"/>

Obr. 3.4: Právomoci definované v prechodoch

## 3.3 Implementácia


### 3.3.1 Modul na priradzovanie používateľov k rolám













Vo WfMS treba zabezpečiť používateľsky ľahko zrozumiteľnú a jednoduchú administráciu rolí vo firme. Priradzovanie rolí vo firme je realizovaná na dvoch úrovniach: *priradenie jednej roly pre viacero používateľov* a *priradenie viacero rolí pre jedného používateľa*. Pre tieto účely sme vytvorili tabuľku, kde sa zobrazia jednotliví používatelia. Zoznam používateľov sa vytvorí na základe vybranej sekcie. Tieto sekcie delíme podľa filtra na **všeobecné** a **podľa rolí vo firme**. V prvej kategórii sú všetci používatelia a takí, ktorí nemajú priradenú rolu. V druhej zobrazujeme používateľov, ktorí sú priradení k určitej role. Používateľov je možné usporiadať v každej sekcii podľa ich ID, mena, priezviska alebo emailu. Zároveň je možné v každej sekcii na základe týchto kritérií vyhľadávať ľudí.

**workflowmarket**   

General filters: All | Without role

Roles filters: Programátor | Projektový manažér | Vedúci oddelenia

search user... 

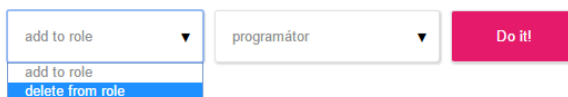
<input type="checkbox"/> Check all	Id	First name	Last name	E-mail	
<input type="checkbox"/>	23	Markéta	Bláznivá	blazniva@gmail.com	 
<input type="checkbox"/>	24	Domofub	Vajcik	vajcik@gmail.com	 
<input type="checkbox"/>	31	Oktávius	Škriniar	skriniaar@yweb.sk	 
<input type="checkbox"/>	34	Zoroslava	Melicharová	melicharova@gmail.com	 
<input type="checkbox"/>	36	Gejza	Železník	zeleznik@stuba.sk	 
<input type="checkbox"/>	37	Eulália	Fisher	fisher@gmail.com	 

add to role ▼ programátor ▼ **Do it!**

Obr. 3.5: Priradzovanie používateľov k rolám

## Priradenie a vyradenie viacerých používateľov z role

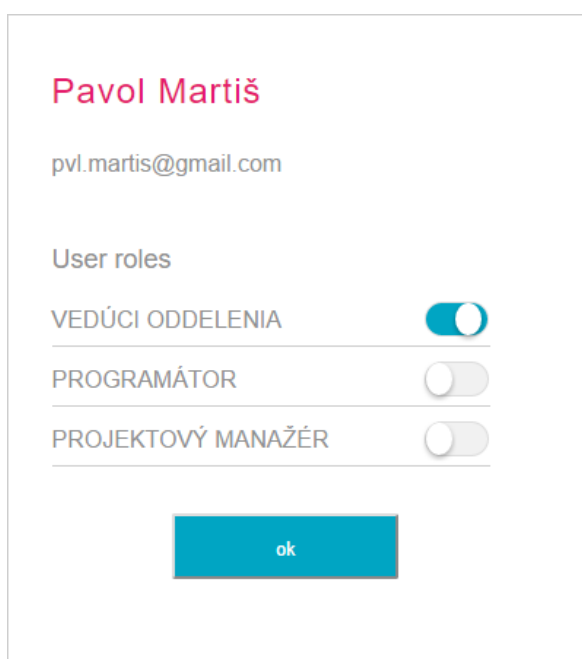
Priradenie a vyradenie viacerých používateľov je založené na dvoch krokoch. Označenie používateľov a následné vybranie žiadanej akcie - add to role (priradenie do role), delete from role (ich odstránenie z role).



Obr. 3.6: Výber akcie pre označených používateľov

## Priradenie a vyradenie jednotlivca

Správa samotného používateľa je realizovaná viacerými spôsobmi. Prvý spôsob je rovnaký ako pri správe viacerých používateľov (označíme ale iba jedného používateľa). Druhý spôsob je detailná správa používateľa. Po kliknutí na podrobnosti konkrétneho používateľa sa zobrazí okno pre konkrétneho používateľa. V ňom sú všetky roly, do ktorých je možné používateľa priradiť. Jednoduchým kliknutím na tlačítko sa používateľ priradí alebo odstráni z role. Takisto je možné odstrániť používateľa z roly priamo z tabuľky kliknutím na ikonu smetného koša.



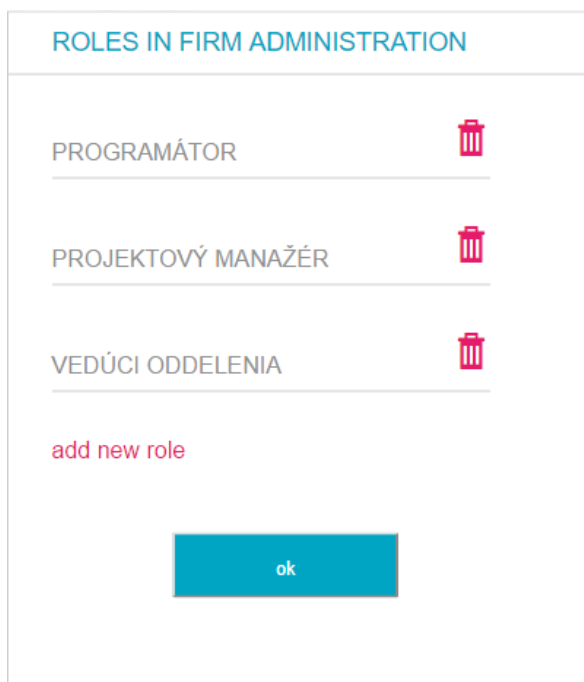
Obr. 3.7: Správa rolí pre konkrétneho používateľa

<input type="checkbox"/>	36	Gejza	Železník	odstrániť používateľa z role
<input type="checkbox"/>	37	Eulália	Fisher	detaily používateľa
				fisher@gmail.com

Obr. 3.8: Možnosti pre konkrétneho používateľa priamo z tabuľky

### Správa rolí vo firme

Firma musí mať vo svojej štruktúre také roly, ktoré používa vo svojich procesoch. Preto sa pri priradení procesu do firmy nahrávajú do firmy roly, ktoré sú v procese využívané. Okrem toho sa však dajú vo firme vytvoriť roly pred tým, ako sa k nemu priradia procesy. Na to slúži administrácia rolí vo firme. Do firmy je tak možné pridať nové roly, prípadne odstrániť staré. Pri odstraňovaní rolí sa najprv skontrolujú v databáze, či firma nepoužíva danú rolu v niektorej z jej procesov.



Obr. 3.9: Administrácia rolí vo firme

Priradenie rolí do firmy je možné aj za pomoci XML súboru v podobnej štruktúre:

```
<?xml version="1.0" encoding="UTF-8"?>
<document>
  <roles>
    <role>
      <name>názov role 1</name>
    </role>
    <role>
      <name>názov role 2</name>
    </role>
  </roles>
</document>
```

```
</roles>
</document>
```

### 3.3.2 Ukladanie rolí a referencií z editoru do databázy

Dôležitou časťou serverovej časti rolí je uložiť výstup z *editora rolí*. Celý proces prebieha v nasledovných fázach :

1. Vytvorenie a priradenie rolí (v editore)
2. Následne po tom, ako používateľ odošle požiadavku na uloženie do databázy, sa odošle na server XML súbor spolu s ID procesom a ID firmy z databázy
3. Na serveri sa najprv vymažú staré dáta (napojenie rolí a referencií na sieť), aby nevznikali problémy pri spätnej úprave
4. Do databázy sa pridávajú roly
5. Do databázy sa pridávajú spojenia medzi rolou a prechodmi. Uloží sa rola, ktorá môže spustiť prípad a pridajú sa referencie ku prechodom.
6. Roly použité v sieti sa priradia do firmy v ktorej sa vykonávalo modelovanie

Príklad XML súboru pre uloženie do databázy:

```
<?xml version="1.0" encoding="UTF-8"?>
<document>
<roles>
  <role> // definovanie role
    <id>0</id>
    <name>programátor</name>
    <start_case>false</start_case> // určuje či môže rola spustiť prechod
    <transitionId>0</transitionId>
    <transitionId>1</transitionId>
    <transitionId>2</transitionId>
  </role>
</roles>
<references>
  <reference> // nulová referencia
    <transitionId>2</transitionId>
    <value>false</value>
  </reference>
  <reference> // referencia na prechod s id 0
    <transitionId>1</transitionId>
    <value>true</value>
  </reference>
  <reference>
```

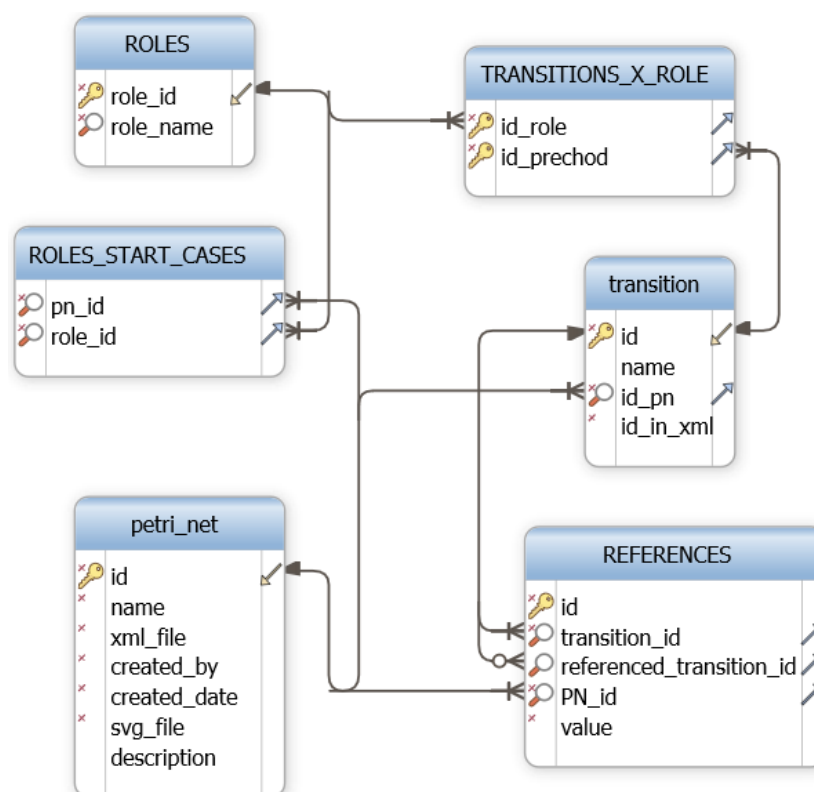
```

    <transitionId>0</transitionId>
    <value>0</value>
  </reference>
</references>
</document>

```

### 3.3.3 Dátový model

Dátová časť našej aplikácie je implementovaná v databáze MySQL. Na začiatku je potrebné definovať návrh tabuliek pre uloženie rolí a referencií ku konkrétnej sieti (obr. 3.10).

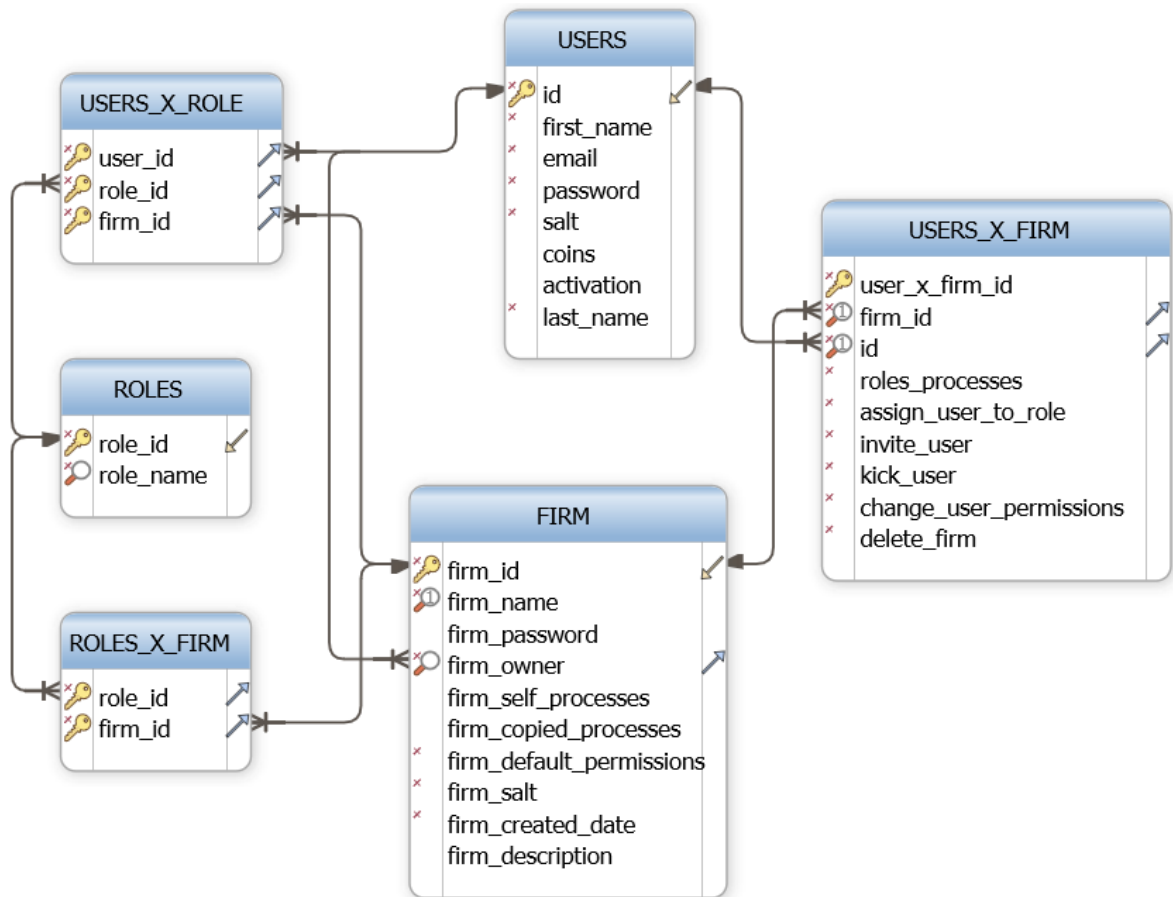


Obr. 3.10: Priradenie právomocí rolám

Do tabuľky ROLES budeme ukladať role a ich názov. Táto tabuľka má okrem ID ako unikátny atribút takisto *názov roly*. Tabuľka *TRANSITIONS\_X\_ROLE* ukladá právomoci rolí na spustenie konkrétneho prechodu v sieti, tabuľka *ROLES\_START\_CASES* ukladá rolu, ktorá môže spustiť nový prípad a tabuľka REFERENCES ukladá jednotlivé referencie ku prechodom. Ak má nastavené *value* na false, prechod nemá referenciu. Ak má hodnotu nastavenú na true, skontroluje sa položka *referenced\_transition\_id*. Pokiaľ má tento atribút číselnú hodnotu, referencia odkazuje na prechod s touto ID. Pokiaľ má nastavenú hodnotu na NULL, referencia je nastavená na prvého užívateľa z role.



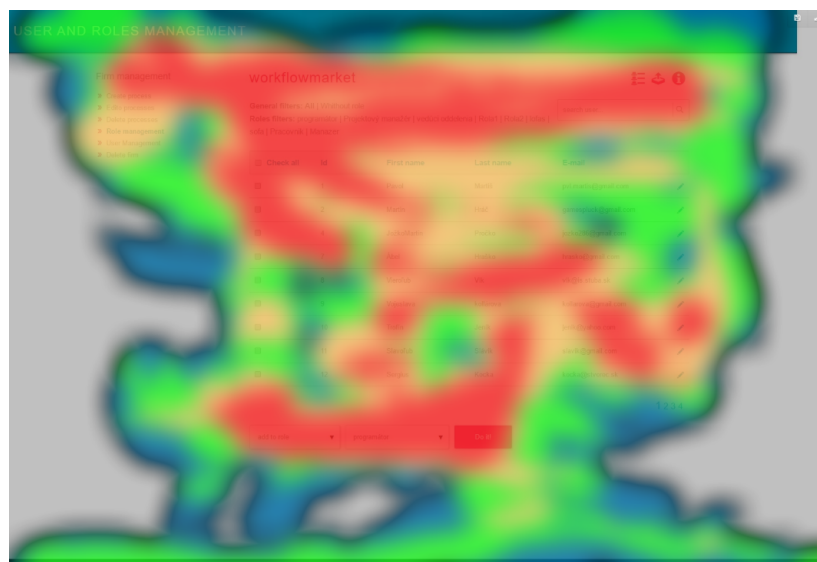
Druhým krokom je návrh databázového modelu, ktorý bude umožňovať konkrétnym používateľom vo firme priradiť rolu (obrázok 3.11). Priradených používateľov vo firme zaznamenávame v tabuľke USERS\_X\_FIRM. Ich priradenie k rolám vo firme je uložené v tabuľke USERS\_X\_ROLES. Priradenie rolí do firmy je zabezpečené takisto priamo prostredníctvom tabuľky ROLES\_X\_FIRM.



Obr. 3.11: Priradenie používateľov do rol vo firmách

Nakoniec na obrázku 3.12 môžeme vidieť spojenú relačnú databázu pre potreby serverovej časti rolí.





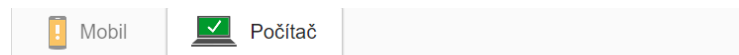
Obr. 3.13: Mapa zobrazujúca hlavné body pozornosti používateľa



Obr. 3.14: Hlavné záujmové body používateľa

### 3.4.2 Google testovanie

Stránka dosiahla na Google PageSpeed Tool [17] dobré výsledky pre rýchlosť na počítači. Hlavné možnosti vylepšenia sa ponúkajú kompresia súborov a ukladanie do vyrovnávacej pamäti. Stránka však má nedostatky pre mobilnú verziu, pretože nespĺňa základné požiadavky na responzívny dizajn.



## 91 / 100 Súhrn návrhov

### ! Zvážte možnosť opravy:

Eliminujte v obsahu nad záhybom stránky jazyk JavaScript a štýly CSS blokujúce vykreslenie

› [Zobrazíť postup na opravu](#)

Využite ukladanie do vyrovnávacej pamäte prehliadača

› [Zobrazíť postup na opravu](#)

Optimalizujte obrázky

› [Zobrazíť postup na opravu](#)

Minifikujte súbor JavaScript

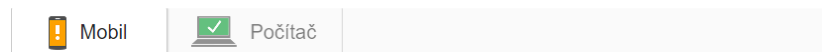
› [Zobrazíť postup na opravu](#)

Minifikujte súbor CSS

› [Zobrazíť postup na opravu](#)

### ✓ Počet úspešne splnených pravidiel: 5

› [Zobrazíť podrobnosti](#)



## 72 / 100 Používateľská skúsenosť

Táto stránka nemusí spĺňať test vhodnosti pre mobilné zariadenia spoločnosti Google. [Prečítajte si ďalšie informácie.](#)

### ! Mali by ste opraviť:

Prispôbte veľkosť obsahu zobrazovanej oblasti

› [Zobrazíť postup na opravu](#)

Používajte čitateľné veľkosti písiem

› [Zobrazíť postup na opravu](#)

Konfigurovanie zobrazenej oblasti

› [Zobrazíť postup na opravu](#)

### ✓ Počet úspešne splnených pravidiel: 3

› [Zobrazíť podrobnosti](#)

## 80 / 100 Rýchlosť

### ! Mali by ste opraviť:

Eliminujte v obsahu nad záhybom stránky jazyk JavaScript a štýly CSS blokujúce vykreslenie

› [Zobrazíť postup na opravu](#)

### ! Zvážte možnosť opravy:

Využite ukladanie do vyrovnávacej pamäte prehliadača

› [Zobrazíť postup na opravu](#)

Optimalizujte obrázky

› [Zobrazíť postup na opravu](#)

Minifikujte súbor JavaScript

› [Zobrazíť postup na opravu](#)

Minifikujte súbor CSS

› [Zobrazíť postup na opravu](#)

### ✓ Počet úspešne splnených pravidiel: 5

Obr. 3.15: Výsledky rýchlosti pre mobil

### 3.4.3 Bezpečnosť

Zistili sme nedostatky v bezpečnosti. Osoba, ktorá môže priraďovať a upravovať role vo firme môže jednoduchou zmenou html kódu alebo javascript súboru prepísať dôležité premenné a zasiahnuť do databázy inej firmy. V budúcnosti preto bude potrebné overovať právomoci administrátora pri jednotlivých prístupoch a zmenách v databáze.

### 3.4.4 Možné rozšírenia

Ako hlavné rozšírenia sa ponúkajú dve dôležité časti. Doimplementovať hierarchiu rolí a pridať obmedzenia na kvalitatívnejšie oddelenie právomocí.

Hierarchia rolí by mala umožňovať jasne definovať štruktúru rolí, tak, aby verne odzrkadľovala štruktúru v organizácii. Určia sa nadradené a podradené roly. Nadradené roly budú automaticky dediť právomoci podradených rolí. Zjednoduší sa tým organizácia štruktúry v podniku. V aktuálnej verzii aplikácie je potrebné používateľa prideliť do každej roly osobitne. Prípadným riešením bude vytvoriť v databáze novú tabuľku, kde budú priradené vzťahy medzi jednotlivými rolami v konkrétnej firme. K tomu bude treba vytvoriť nástroj na definovanie organizačnej štruktúry a zároveň bude potrebné predefinovať procesnú logiku systému.

Ako súčasť tohto rozšírenia bude možnosť zjednotiť vo firme viacero rolí do jednej. V aktuálnej verzii vzniká problém pri priradení procesu do firmy, pretože firma musí automaticky prebrať všetky role v procese. Môže však nastať situácia, keď bude vo firme veľa rolí s rovnakou funkcionalitou, ale odlišným názvom. Ako príklad si môžeme zobrať rôzne synonymá. Rola programátor sa dá nazvať ako "programátor", "vývojár", "developer" a takisto problém môžu spôsobovať viacjazyčné preklady.

Ďalším rozšírením bude pridanie obmedzení právomocí pre jedného používateľa. Pri modelovaní návrhu rolí v editore (kap. 2.2.4) chceme pridať ku prechodu možnosť zápornej referencie. To bude znamenať, že jeden používateľ nebude môcť spustiť dva prechody, ktoré sa vylučujú. Znamená to teda, že pokiaľ používateľ spustí prvý prechod, nebude schopný vykonať aj druhý prechod, ak obsahuje zápornú referenciu na prvý prechod. Za zváženie taktiež stojí obmedzenie štruktúry rolí pomocou statického vylúčenia práv. Znamenalo by to, že budú existovať určité obmedzenia pri priraďovaní používateľov k rolám. Administrátor by teda nebol schopný priradiť jednému používateľovi dve navzájom sa vylučujúce role.

Tretie rozšírenie by mohlo definovať štruktúru zamestnancov vo firme z hľadiska ich organizačnej a geografickej odlišnosti. Väčšie firmy majú viacero oddelení, ktoré môžu znázorňovať isté pracovné alebo geografické odlišnosti. Príkladom môžu byť rôzne oddelenia, ktoré v rámci firmy pracujú na odlišných objednávkach (respektíve prípadoch).

V našom systéme by sa takto mali dať priradiť isté organizačné jednotky pri samotnom vytváraní nového prípadu, prípadne inak ošetriť, aby sa zamedzil prístup v rámci jednotlivých oddelení.

# Záver

Úlohou bakalárskej práce bolo navrhnúť a implementovať server manažmentu rolí vhodný pre komplexný *Workflow manažment systém*, ktorý by fungoval na základe Petriho sietí.

Na začiatku sme analyzovali problematiku workflow manažmentu, Petriho sietí a používané metódy riadenia prístupu. Zistili sme, že Petriho siete poskytujú vhodné riešenie na implementovanie workflow manažment systému. Ich hlavná výhoda spočíva v grafickej reprezentácii, intuitívnom modelovaní a vďaka jej matematickému základu taktiež jasnej analýze procesov. V porovnaní s konkurenčnými WfMS, rozlišuje v prípadoch medzi stavom, do akého proces dostane a jednotlivými udalosťami, ktoré posúvajú prípad do nového stavu. Mapovanie Petriho sietí na workflow systém má určité pravidlá, ktoré sú však dostatočne zrozumiteľné.

Na základe rozboru prístupových práv sme identifikovali systém riadenia rolí RBAC (Role-based Access Control) ako najvhodnejší prístup pre správu zdrojov vo WfMS. Tento model zabezpečuje prepojenie práv používateľov prostredníctvom rolí, a tak umožňuje modelovať procesy nezávisle od používateľskej časti aplikácie. Okrem toho zabezpečuje rýchlu a efektívnu administráciu a vďaka tomu, že je postavený na základoch organizačnej štruktúry, je zrozumiteľný širšej verejnosti. Z dlhodobého hľadiska prevyšuje konkurenčné systémy v administrácii firmy s veľkým počtom zamestnancov a zložitou organizačnou štruktúrou.

Pri návrhu Workflow systému, sme sa vzhľadom k predošlej analýze rozhodli použiť práve Petriho siete ako základ pre logiku WfMS. Každý prechod v Petriho sieti reprezentuje úlohu, ktorú treba vykonať. Ku každej úlohe priradíme formulár a rolu, ktorá je oprávnená ju spustiť. Po namodelovaní procesov ich môžu následne paralelne využívať na portáli viaceré firmy.

Pre tento systém sme vytvorili aplikáciu pre správu rolí vo firmách a ich priradzovanie jednotlivým používateľom. Aplikácia je vo forme webovej služby a takisto poskytuje aplikačné rozhranie pre *editor manažmentu rolí a organizačnej štruktúry* pre ukladanie do databázy. Vytvorená aplikácia umožňuje efektívne spravovať roly vo firme - pridávať a odoberať roly z firmy, mapovať používateľov na roly jednotlivo ale aj skupinovo. Pre tieto účely sa zobrazuje zoznam používateľov, ktorých je možné zotriediť podľa rôznych parametrov, alebo osobitne vyhľadávať na základe textového vstupu. Aplikácia

je kompatibilná s ostatnými časťami Workflow systému - používa rovnakú databázu.

V priebehu testovania sme zistili, že v budúcnosti bude potrebné vylepšiť model referencií, pridať možnosť vytvárať vo firme hierarchie rolí, prípadne obmedzenia ako statické a dynamické vylúčenie práv a zabezpečiť bezpečnosť aplikácie tak, aby používateľ upravením klientskej časti skriptu nebol schopný vykonať neoprávnené operácie.



# 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
- [2] AALST, W.: *The Application of Petri Nets to Workflow Management*. Dostupné na internete: <http://wwwis.win.tue.nl/~wvdaalst/publications/p53.pdf>
- [3] Michael P. Gallaher - Alan C. O'Connor - Brian Kropp, *The Economic Impact of Role-Based Access Control*, National Institute of Standards and Technology, 2002. Dostupné na internete: <http://www.nist.gov/director/planning/upload/report02-1.pdf>
- [4] D. Ferraiolo - D.R. Kuhn - R. Chandramouli: *Role-based Access Control*. Artech House Computing Library. Artech House, 2007, 405 s., ISBN 9781596931138
- [5] AALST, W. - HEE, K.: *Workflow Management Models, methods and systems*. Cambridge, Massachusetts: The MIT Press, 2004, ISBN 0-262-01189-1, p. 267-268
- [6] W. M. Van Der Aalst: *Three good reasons for using a petri-net-based workflow management system*. Medzinárodná vedecká konferencia o informáciach a integrácii procesov v podnikoch (IPIC'96) s. 179–201 , 1996.
- [7] J. H. Saltzer - M. D. Schroeder: *The Protection of Information in Computer Systems*. IEEE, č.63, September 1975, s.1278–1308.
- [8] D. E. Bell - L. J. La Padula: Secure computer systems: Vol. I—mathematical foundations, Vol. II—a mathematical model, Vol. III—a refinement of the mathematical model. Technical Report MTR-2547 (tri zväzky). Bedford: Mitre Corporation, MA, 1973.
- [9] Sandhu, R. a kol.: *Role-Based Access Control Models*. IEEE Computer, zväzok. 29, č. 2, 1996.
- [10] Ferraiolo, D. F. - J. Cugini, - D. R. Kuhn: *Role-Based Access Control (RBAC): Features and Motivations*. New Orleans, LA : Konferencia o bezpečnosti aplikácii , December 11–15, 1995, s. 241–248.

- [11] Telecom Glossary 2000, American National Standard for Telecommunications, American National Standards Institute
- [12] DoD, Trusted Computer System Evaluation Criteria (TCSEC), DoD 5200.28-STD.
- [13] CURPHEY, Mark: *Mandatory Access Control* [online]. 2002 [cit. 2008-10-22] . Dostupné na internete: <<http://www.cgisecurity.com/owasp/html/ch08s02.html>>
- [14] JUHÁS, G.: *Modelovacie formalizmy udalostných systémov*. Bratislava: RT systems s.r.o., 2011, ISBN 978-80-970519-1-4, p. 29-45
- [15] DESEL, J. - REISIG, W.: *Lectures on Petri nets I: Basic models*. Germany: Universit ät Karlsruhe: Springer, 1998, ISBN 3-540-65306-6
- [16] PETRI, C.A. - REISIG, W. 2008 Petri net. [online] [cit. 27.04.2016]. Dostupné na internete: <[http://www.scholarpedia.org/article/Petri\\_net](http://www.scholarpedia.org/article/Petri_net)>
- [17] PageSpeed Insights [online] . Dostupné na internete: <<https://developers.google.com/speed/pagespeed/>>
- [18] Attensee [online] . Dostupné na internete> <<http://www.attensee.com/>>

# Zoznam príloh

A Štruktúra elektronického nosiča .....	II
---	----

## A Štruktúra elektronického nosiča

\

\**application** - obsahuje aplikáciu

\**document** - textová časť bakalárskej práce vo formáte pdf

\database.sql - obsahuje exportovanú databázu

\instructions.pdf - v súbore sú inštrukcie pre spustenie aplikácie