

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

SHO Štátna volebná infraštruktúra
Dokumentácia z predmetu IMS
9. prosince 2013

Autori:

Martin Maga xmagam00,
Vojtěch Meca xmecav00

Obsah

1	Úvod	2
1.1	Autori	2
1.2	Zdroj informácií	2
1.3	Testovacie prostredie	2
1.4	Validita	2
1.5	Ciele projektu	3
2	Rozber témy a použitých technológií	3
2.1	Téma práce	3
2.2	Popis použitých postupov	3
2.3	Pôvod použitých technológií	3
3	Koncepcie - modelárska témata	4
3.1	Opis konceptuálneho modelu	5
3.2	Ukážka Petriho siete pre odovzdávanie hlasu	6
3.3	Implementácia	7
4	Architektura simulačného modelu	7
4.1	Zmena implementácie v knižnici SimLib	7
4.2	Popis štruktúry programu	7
4.3	Okrsky navrhnuté k simulácií	7
4.4	Popis architektúry	7
4.5	Spustenie programu	11
5	Podstata simulačných experimentov	12
5.1	Postup experimentov	12
5.2	Popis experimentů	12
5.3	Závěr experimentů	15
6	Zhrnutie simulačných experimentov a záver	15
7	Referencie	16

1 Úvod

Táto dokumentácia sa zaoberá vývojom, implementáciou a testovaním systému hromadnej obsluhy "Štátna volebná infraštruktúra", ktorá simuluje[2] volebný systém v Českej republike zahrňujúci voľby v okrskoch a krajských mestských a následne odoslanie a počítanie hlasov v informačnom centre.

Na základe modelu a simulácie bude ukázané chovanie systému so zreteľom na ukázanie slabých miest pri voľbách. Tento projekt môže byť použitý na optimalizáciu systému voliev v Českej republike vzhľadom na zrýchlenie celého systému počítania hlasov a rozdelenia do okrskov.

1.1 Autori

Na projekte SHO "Štátna volebná infraštruktúra" sa podieľali nasledujúci autori:

- Martin Maga(xmagam00)
- Vojtěch Meca(xmecav00)

Okrem vyššie spomenutých ľudí sme využili možnosť konzultácie s pánom doktorom Hrubým(konzultácia ohľadne správnosti nášho návrhu Petriho siete).

1.2 Zdroj informácií

Informácie o štátnej volebnej infraštruktúre boli zisťované z dostupnej literatúry. Rovnako sme využili informácie dostupné na internete, ohľadne počtu okrskov a ich voličov. Rovnako sme využili štatistiky, ktoré sme generovali pri skončení programu.

1.3 Testovacie prostredie

Pre testovacie účely boli použité architektúry: Linux 3.2.0-56-generic 86_64 GNU/Linux pre menšie vzorky dát a pre rozsiahlejšie testovanie na väčšej vzorke dát: FreeBSD eva.fit.vutbr.cz 9.2-STABLE FreeBSD amd64.

1.4 Validita

Experimentovaním sme overovali validitu modelu, ktoré vo forme štatistik a jeho následnej analýze odpovedali našu odhadovanému predpokladu. Tak isto sme využili dostupné informácie o spôsobe volieb v Českej republike a štatistických informáciách, ktoré sú verejne prístupné na internete.

1.5 Ciele projektu

Ciele projektu zahŕňajú:

- Analýza aktuálneho volebného systému v Českej republike
- Analýza slabých miest volebného systému
- Návrh efektívnejšieho prístupu, ktoré by dokázalo zvýšiť rýchlosť počítania hlasov
- Návrh čo najviac reálneho systému voleb v České republice

2 Rozber témy a použitých technológií

2.1 Téma práce

Témou práce bola implementácia systému hromadnej obsluhy štátnej volebnej infraštruktúry. Volebná infraštruktúra v Českej republike funguje na nasledovnom princípe: Česká republika sa zaraďuje medzi dvojkomorové parlamentné systémy. Tvorí ju Poslanecká snemovňa a Senát. Do PS ČR sa volí na základe pomerného voličského systému. Ďalším dôležitým zákonom je zákon č. 247/1995 Sb., o voľbách. Tieto právne pramene sú základnými právnymi prameňmi v oblasti volieb v ČR.[1]

Voľby prebiehajú štandardne každé 4 roky, alebo po predčasných voľbách. Každému občanovi republiky prislúcha 1 volebný hlas, ktorý musí odovzdať. Každý občas patrí pod istý volebný kraj, ktorý sa delí na menšiu časť zvané okrsky. Voľby prebiehajú štandardne v sobotu alebo v nedeľu dokopy 14 hodín. Každý občan, ktorý spĺňa podmienky príde v mieste svojho bydliska do volebnej miestnosti, kde sa preukáže platnosti dokladom totožnosti volebnej komisii, ktorá skontroluje údaje a povolí občanovi voliť. Každý občan ide jednotlivo za urnu, kde zaškrtnie možnosť a následne vhodí svoj hlas do urny a zvyšné zahadzuje.

Na konci volieb sú hlasí v jednotlivých okrskoch sčítané a odnesené na obecný úrad, odkiaľ sa posielajú na prepočítanie do volebného centra.

2.2 Popis použitých postupov

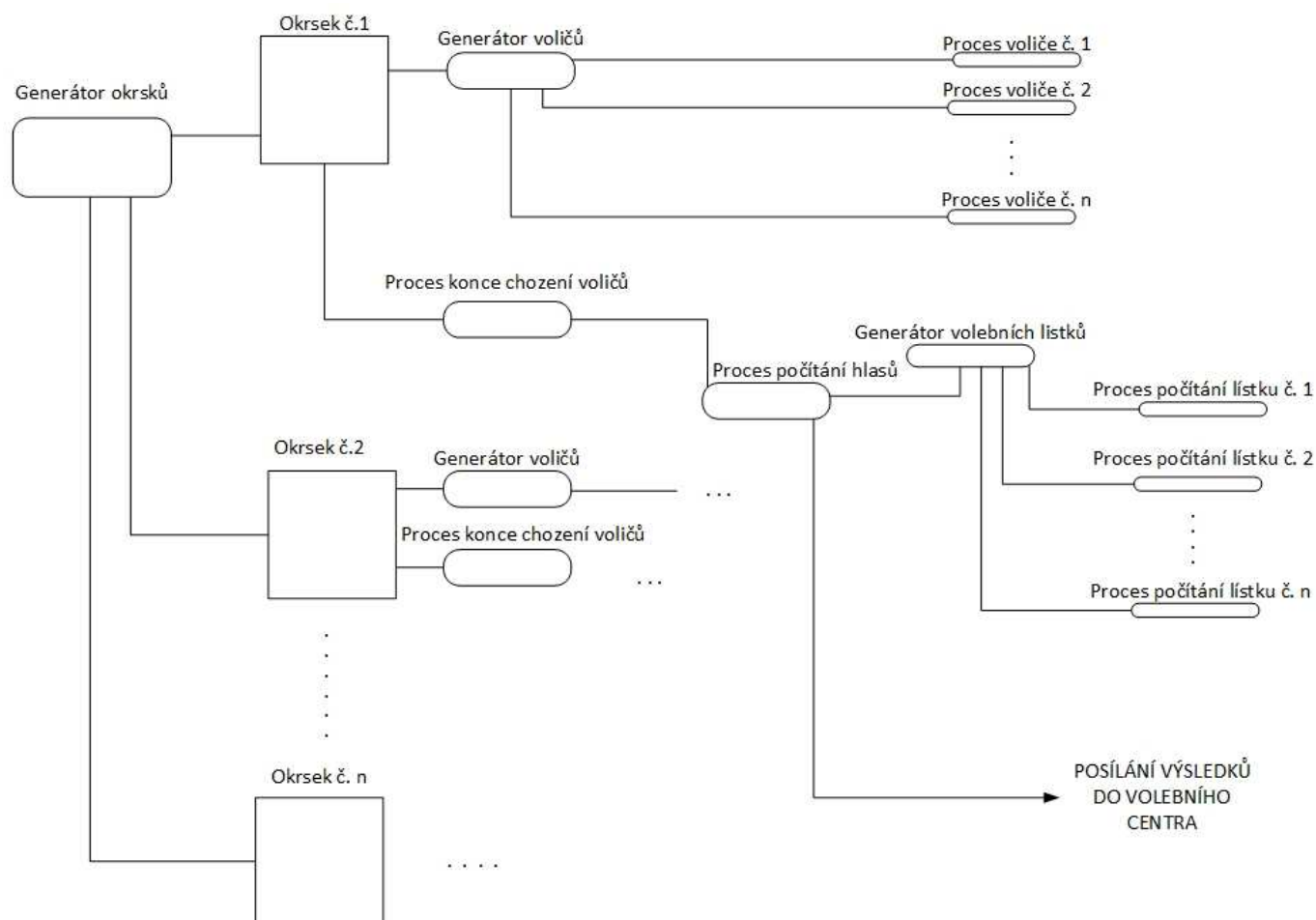
Systém je simulovaný pomocou diskkrétnej simulácie (predmet IMS slajd 122) ako systém hromadnej obsluhy (predmet IMS slajd 139). Je to výhodné napr. z hľadiska sledovania dĺžky front (predmet IMS slajd 141) obslužných liniek (predmet IMS slajd 139) jednotlivých častí volebného centra. Použitou technológiou k tomuto účelu je jazyk C++ a knižnica Simlib (predmet IMS (1) slajd 72). Výhodou spomínaných technológií je rýchlosť simulácie.

2.3 Pôvod použitých technológií

- Simlib - <http://www.fit.vutbr.cz/peringer/SIMLIB/> (GNU LGPL)
- C++ - <http://en.wikipedia.org/wiki/C++>
- Ubuntu - <http://www.ubuntu.com/>
- Petriho siete - http://en.wikipedia.org/wiki/Petri_net

3 Koncepce - modelářská témata

V této kapitole bude načrtnutý konceptuální model představující systém volební infrastruktury. Na obrázku č. 1 je uvedeno schéma systému Volební infrastruktury.



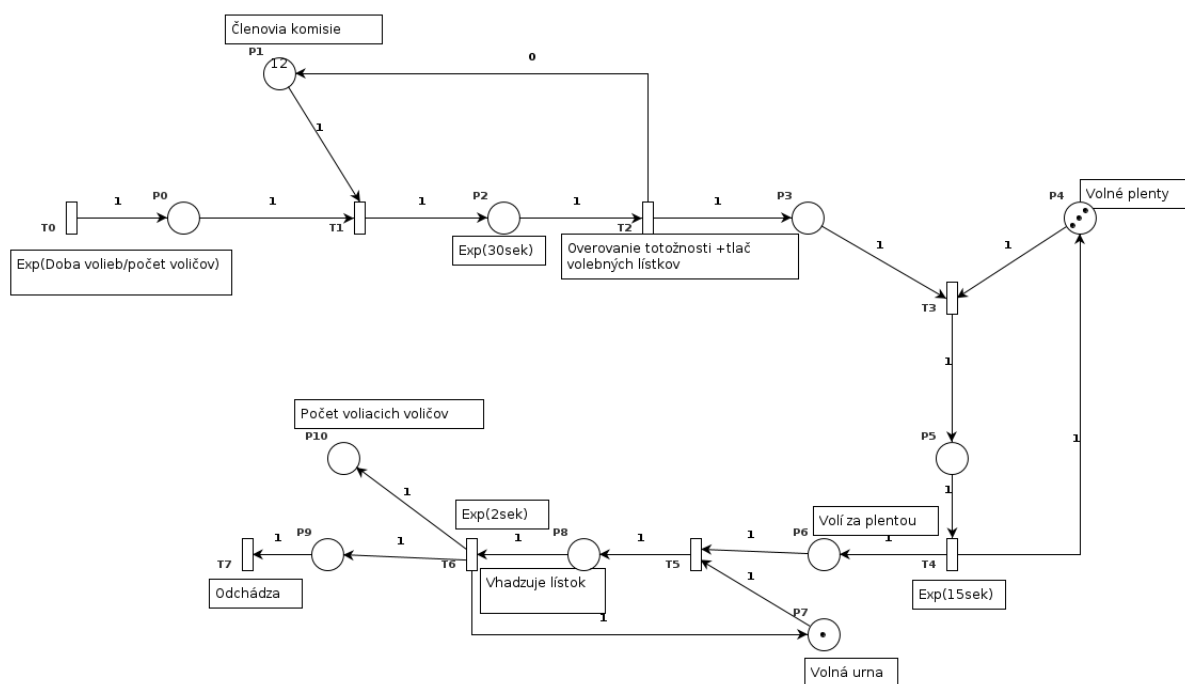
Obrázek 1: Konceptuální model

3.1 Opis konceptuálního modelu

V obrázku č. 1 můžeme vidět několik různých obdelníků a zaoblených obdelníků. Popíšeme si je jednotlivě:

- a) Zaoblený obdelník s názvem Generátor okrsků
Tento zaoblený obdelník má za úkol generovat obdelníčky Okrsky. Generuje jich přesně kolik jich má vzniknout (určeno například souborem). Po vygenerování Okrsků generátor ukončí svou činnost.
- b) Normální obdelník s názvem Okrsek č. 1...n
Při vytvoření tohoto objektu se nainicializují hodnoty, které jsou získány z Generátoru okrsků a dále vytvoří si Generátor voličů a taktéž si vytvoří Proces konce chození voličů. Poté objekt slouží pouze pro ukládání hodnot, které se mohou během programu měnit.
- c) Zaoblený obdelník s názvem Generátor voličů
Po vytvoření Okrsekem začne generovat voliče, kteří jdou volit. Vytvoří jich podle parametru maximálního počtu voličů a dále taky podle parametru volební účast voličů. Po vygenerování všech voličů, zahlásí konec voleb, a tímto její činnost skončí.
- d) Zaoblený obdelník s názvem Proces voliče č. 1 .. n
Je vytvořen Generátorem voličů. Po vytvoření simuluje všechny zásadní pokyny po správné odvolení voliče. Po samotném odvolení voliče proces končí.
- e) Zaoblený obdelník s názvem Proces konce chození voličů
Je vytvořen Okrskem. Zapne v sobě časovač reprezentující dobu voleb. Po uplynutí časovače nebo po příchodu všech voličů v okrsku se vyvolá Proces počítání hlasů. Poté jeho činnost končí.
- f) Zaoblený obdelník s názvem Proces počítání hlasů
Vytváří Genreátor volibních lístků. Správně spočítané výsledky posílá do Volebního centra. Poté jeho činnost končí.
- g) Zaoblený obdelník s názvem Proces generátor volebních lístků
Po vytvoření Procesem počítání hlasů začne generovat Procesy počítání lístků č.1 .. n. Po vygenerování všech lístků zkončí svou činnost.
- h) Zaoblený obdelník s názvem Proces počítání lístků č.1 .. n
Po vytvoření simuluje práci jednoho člena volební komise s tímto lístkem. Po ukončení počítání tohoto lístku jedním členem komise činnost tohoto obdelníku končí.

3.2 Ukážka Petriho siete pre odovzdávanie hlasu



Obrázek 2: Petriho sieť pre odovzdávanie hlasu

Obrázok č.2 popisuje Petriho sieť, ktorá ukazuje systém volieb. Ukazuje príchod voliča, odvolenie voliča. Na začiatku sa vygenerujú voliči, ktorí príjdu do volebnej miestnosti. Pri vstupe do miestnosti si občan zabere člena volebnej komisie, ktorý je reprezentovaný štruktúrou "Store" s kapacitou 12 v Simlibe.

Pokiaľ nie je voľný nejaký člen volebnej komisie, tak sa zařadí do spoločnej fronty a čaká na uvoľnení niektorého z členov volebnej komisie. Pokiaľ sa občan dostane na radu, tak ho člen volebnej komisie skontroluje. To trvá exponenciálne 30 sekúnd. Po skončení kontroly občana členom volebnej komisie, občan pokračuje ďalej. Zabere si "Store" s kapacitou 3, ktorý reprezentuje volebnú plentu.

V prípade, že nie je voľná plenta, tak sa občan zařadí do jednej spoločnej fronty. Následne užívateľ zajde za plentu a zaškrtná výsledok. To trvá exponenciálne 15 sekúnd. Následne si zabere plentu, ktorá je len 1. Pokiaľ nie je voľná tak sa zařadí do fronty a čaká. Po vhození lístku reprezentujúci dobu exponenciálne 2 sekundy občan opúšťa systém.

3.3 Implementácia

Program pracuje na základním procese, který regeneruje v čase nula objekty reprezentující okrsky České republiky. Při vytvoření objektu jednotlivého okrsku se nainicializují data. Objekt reprezentující okrsek po inicializaci spustí po sobě jdoucí 2 procesy. První z nich je proces, který po svém vzniku začne generovat procesy příchodů voličů do volební místnosti. Druhý z procesů vytvoří časovač, který udává legální délku voleb. Po uplynutí určeného času na příchod voličů tento proces vytvoří nový proces. Nově vytvořený proces se stará o počítání jednotlivých volebních lístků ve volební místnosti, dále simuluje kontrolu výsledků a taky posílání správně vypočtených výsledků do Volebního centra České Republiky. Problém počítání jednotlivých volebních lístků řeší pomocí procesu na generování procesu počítání volebního lístku. V případě, že po spočítání výsledků se zjistí, že nastala chyba v počtech, proces zopakuje počítání volebních lístků znovu.

4 Architektura simulačního modelu

4.1 Zmena implementácie v knižnici SimLib

Při kompilaci modelu jsme narazili na problém v knihovně SimLib konkrétně s proměnou `CANARY1`. Po konzultaci s Dr. Petrem Peringerem jsme změnili v knihovně SimLib v souboru `process.cc` řádek `volatile int mylocal = CANARY1`; na řádek `volatile long mylocal = CANARY1`; . Tím se problém vyřešil a náš model jsme mohli zkompileovat.

4.2 Popis struktury programu

Zdrojový soubor programu se nazývá `projekt_v4.cc`.

4.3 Okrsky navrhnuté k simulácií

Data k okrkům jsme získávali při experimentech ze souboru `okrsky.txt`. Na prvním řádku získáváme informaci o množství okrků. Další řádky jsou ve formátu 4 posobě jdoucích informací. První z nich je číslo kraje. Druhý z nich je název kraje, další údaj v sobě nese název krajského města. Pokud kraj nemá krajské město, tak zde nacházíme znak '-'. Poslední a nejdůležitější je údaj o počtu občanů v okrsku.

4.4 Popis architektury

Simulační model má za základní jednotku minutu.

V programu je několik tříd, které zajišťují jeho spávný průběh.

- Třída `GenOkrsku`

Tato třída reprezentuje proces Generování okrsku z konceptuálního modelu na Obr. č. 1. Tato třída se dědí od třídy `Event` z knihovny `SIMlib` obsahuje pouze jednu metodu `Behavior()`. Metoda `Behavior()` ze začátku načítá první řádek ze souboru. V souboru na prvním řádku je informace o celkovém počtu okrků. Dále pokračuje jedním cyklem, který při každém průchodu načítá data jednoho řádku ze souboru, což reprezentuje data určené právě pro jeden okrsek. Poté cyklus pokračuje vytvořením nového objektu třídy `Okrsek`

a posílá získané data ze souboru do jako parametry konstrukturu. Po vytvoření všech Oksrku, svou činnost končí.

- Třída Okrsek

Tato třída reprezentuje objekt Okrsek z konceptuálního modelu na Obr. č. 1. Tato třída má konstruktor a mnoho dalších metod na získávání, nastavování atributů třídy. Také tato třída obsahuje metody, které inkrementují nebo dekrementují některé atributy třídy. Třída je určena jako ukládání a uchovávání dat se kterými se během programu pracuje. V konstruktoru nainicializuje potřebné atributy. A po té vytvoří proces GenVolicu a předá mu parametr, jako ukazatele na sebe (pomocí ukazatele this). Poté vytváří podobně jako před tím další proces TimerVoleb. Předává mu do konstruktoru také ukazatel na sebe. Tím končí konstruktor této třídy.

- Třída GenVolicu

Tato třída reprezentuje proces Generování voličů z konceptuálního modelu na Obr. č. 1. Tato třída se dědí od třídy Event z knihovny SIMlib a obsahuje konstruktor a metodu Behavior(). V konstruktoru se předá ukazatel z objektu Okrsku ze kterého byl generátor vytvořen do svého atributu třídy Okrsku. Dále se taky zde definuje čas, který slouží pro znovu aktivování této třídy. Zde také měníme parametr účast voličů pro celou ČR. V metodě Behavior() se testuje jestli nenastal konec voleb. Zda-li nenastal, tak vytvoří objekt procesu voliče a předá mu parametr objektu Okrsek, ze kterého byla tato třída vytvořena. Inkrementuje počet generovaných voličů v okrsku a testuje jestli počet vygenerovaných voličů se rovná maximálního počtu voličů v daném okrsku, což je uloženo v atributu Okrsku s názvem pocet_volicu. Pokud bylo vytvořeno méně voličů, tak se znovu tato třída aktivuje podle součtu času Time a času určeného v konstruktoru. Ale pokud je vytvořeno stejně voličů jako maximální počet, tak se tato třída znovu neaktivuje, ale nastaví parametr end_voleb třídy Okrsku na 1 a to pomocí metody konec_voleb() v objektu Okrsek a tím ukončí dobu konce voleb daného okrsku.

- Třída Volic

Tato třída reprezentuje Proces voliče z konceptuálního modelu na Obr. č. 1. Tato třída se dědí od třídy Process z knihovny SIMlib a obsahuje konstruktor a metodu Behavior(). V konstruktoru se předá ukazatel z atributu okrsek třídy GenVoličů, ve kterém je uložen ukazatel na konkrétní okrsek. Metoda Behavior() určuje chování voliče ve volební komisi. Při vstupu voliče se nejprve zvýší číslo aktuálních voličů v místnosti o 1 pomocí metody v objektu Okrsku increment_lidi_v_mistnosti(). Poté proces voliče pokračuje zabráním si jednoho člena komise, reprezentujícího Store s názvem komise pomocí metody get_komise() z objektu Okrsek. Po uplynutí dané doby pomocí SIMlib funkce Wait() uvolní zabraného člena komise. Podobně si zabere i jednu z plent. Planta je taky typu Store. Nakonec podobně jako předchozí zařízení zabere i urnu, která je reprezentována zařízením Facility. Proces skončí po odečtení aktuálního čísla voličů pomocí metody decrement_lidi_v_mistnosti().

- Třída TimerVoleb

Tato třída reprezentuje Proces konce chození voličů z konceptuálního modelu na Obr. č. 1. Třída je určena pro ukončení voleb po uplynutí celkového času voleb. Tato třída se dědí od třídy Process z knihovny SIMlib a obsahuje konstruktor a metodu Behavior(). V konstruktoru se předá ukazatel z objektu Okrsku ze kterého byl timer vytvořen do svého atributu třídy Okrsku. V metodě Behavior() se spustí cyklus, který reprezentuje dobu voleb příchodu voličů způsobem, že v každém cyklu čeká minutu pomocí

Wait() a testuje jestli nenastal konec voleb pomocí metody třídy Okrsek get_konec_voleb(). Poté spustí cyklus, který čeká na uvolnění místnosti od všech voličů. Testuje se atribut lide_v_mistnosti třídy Okrsek. Před ukončením svého konání proces TimerVoleb vytvoří nový proces s názvem PocitaniHlasu a předá mu ukazatel na objekt Okrsku, ze kterého byl vytvořen a aktivuje ho.

- Třída GenHlasu

Tato třída reprezentuje Generátor volebních lístků z konceptuálního modelu na Obr. č. 1. Tato třída se dědí od třídy Event z knihovny SIMlib a obsahuje konstruktor a metodu Behavior(). V konstruktoru se předá ukazatel z atributu okr třídy TimerVoleb, ve kterém je uložen ukazatel na konkrétní okrsek. V metodě Behavior() se vytvoří proces PocitaniKonkretnihoHlasu s ukazatelem na Okrsek uložený v atributu okr třídy a poté se aktivuje. Pomocí metody decrement_poc_listku() se sníží počet nespočtených volebních lístků. Dále se testuje jestli se vygenerovali všechny lístky pro daný okrsek pomocí metody get_poctu_zvolenych_listku(). Pokud metoda vrátí 0 nastaví se atribut spocetene_hlasy objektu Okrsku. Pokud metoda vrátí jiné číslo, znovu aktivuje se tato třída v čase Time.

- Třída PocitaniKonkretnihoHlasu

Tato třída reprezentuje Proces počítání lístků z konceptuálního modelu na Obr. č. 1. Tato třída se dědí od třídy Process z knihovny SIMlib a obsahuje konstruktor a metodu Behavior(). V konstruktoru se předá ukazatel z atributu okr třídy GenHlasu, ve kterém je uložen ukazatel na konkrétní okrsek. Metoda Behavior() určuje chování počítání lístku členem volební komise. Při začátku počítání lístku se nejprve zvýší číslo aktuálních právě počítaných lístků o 1 pomocí metody v objektu Okrsku increment_prave_pocitanych_listku(). Poté proces počítání lístku pokračuje zabráním si jednoho člena komise, reprezentujícího Store s názvem komise pomocí metody get_komise() z objektu Okrsek. Po uplynutí dané doby pomocí SIMlib funkce Wait() uvolní zabraného člena komise. Poté vyhodnotí zda-li je lístek platný nebo neplatný. Pokud platný pomocí metody increment_plat_listku() v objektu Okrsek se zvýší číslo platných lístků okrsku. Zda-li je lístek neplatný použije se po zvýšení atributu poc_nep_listku metoda increment_nep_listku() v objektu Okrsek. Proces skončí po odečtení aktuálního čísla voličů pomocí metody decrement_prave_pocitanych_listku().

- Třída PocitaniHlasu

Tato třída reprezentuje Proces počítání hlasů z konceptuálního modelu na Obr. č. 1. Tato třída se dědí od třídy Process z knihovny SIMlib a obsahuje konstruktor a metodu Behavior(). V konstruktoru se předá ukazatel z atributu okr třídy TimerVoleb, ve kterém je uložen ukazatel na konkrétní okrsek. V metodě Behavior() se spustí cyklus, který testuje, zda-li má proběhnout počítání hlasů. Pokud ano, tak vytvoří proces GenHlasu a předá mu ukazatel na objekt Okrsek. Poté spustí cyklus, který končí v případě, že jsou vytvořeny všechny procesy třídy PocitaniKonkretnihoHlasu. Poté se spustí další cyklus, který kontroluje končí v případě, že jsou všechny lístky spočtené. Po nějaké době (pomocí Wait()) vyhodnotí, jestli výsledky jsou správné a nebo ne. Pokud výsledky správné nejsou, tak zavolá metodu get_nul_prepoctenym_listkum(), která nastaví atributy poc_nep_listku a poc_plat_listku na 0 a atribut pocet_zvolenych_listku na původní hodnotu v objektu Okrsek. A cyklus pro vyhodnocení zda-li se mají výsledky počítat se spustí znovu. Pokud jsou výsledky správné, tak si proces třídy PocitaniHlasu

zabere zařízení Centrum_republiky pomocí SIMlib funkce Seize. Po uplynutí času reprezentující posílání výsledků do volebního centra republiky, se zařízení Centrum_republiky pomocí SIMlib funkce Release uvolní. Tím končí tento proces.

4.5 Spustenie programu

Pro zkompileování programu je potřeba zadat do příkazové řádky **make** v adresáři s projektem. Po úspěšném zkompileování projektu zadáme příkaz **make run**, když budeme chtít program spustit. V průběhu výpočtu programu se na obrazovku vypisují příchody a odchody jednotlivých voličů a taky začátek zpracování volebních lístků a jejich konec zpracování v jednotlivých okrscích. Nakonec všech výpisů programu získáváme to nejdůležitější a to statistiky modelu.

5 Podstata simulačních experimentů

5.1 Postup experimentování

Hlavní cíl, proč byl model vytvořen, bylo zjistit vytíženost Volebního celorepublikového centra a navrhnout změny pro zefektivnění vytíženosti tohoto volebního centra. Dále jsme získávali informace o průběhu počítání hlasů pro jednotlivé kraje a pak pro celou republiku. K tomu nám pomáhali experimenty. Díky těmto experimentům jsme mohli dosáhnout validity samotného modelu voleb. V experimentech jsme měnili parametr volební účasti, kdy jsme sledovali vytíženost volebního centra v závislosti na tomto parametru. Dále jsme takéž podle tohoto parametru sledovali, jak se chovají jednotlivé okrsky v délce počítání hlasů a v čekání na volební centrum.

Pro validitu našeho modelu jsme si stanovili výstup experimentů, že při 100% účasti voličů našeho modelu by mělo nastat: 2-3 hodiny počítání hlasů, 13-17 minut čekání okrsku na zpracování volebního centra. Tyto údaje jsme určili z praxe, protože člen Vojtěch Meca se zúčastnil jako člen několika voleb v ČR a v průměru odpovídají tyto hodnoty výsledkům z Moravskoslezského kraje.

5.2 Popis experimentů

Zde si popíšeme jedny prvních experimentů, které jsme prováděli. Vstupem byla 50% účast voličů a očekávalo se, že výstupem bude doba čekání na volební centrum v průměru 13-17 minut. Dále jsme předpokládali průměrné počítání hlasu za 2-3 hodiny. Z obrázku č. 3 u kolonky a vidíme výsledné statistiky, ve kterých jsme zjistili, že průměrné počítání hlasů v okrscích je 263 minut, což je 4.4 hodiny. Dále můžeme z obrázku č. 3 u kolonky b vyčíst, že průměrně okrsky čekají na volební centrum v minutách 0.0231881, což je přibližně 1.4 sekundy. Další zajímavý údaj byl, že ve frontě na volební centrum bylo maximálně 3 volební okrsky z celé republiky.

STATISTIC	poctani v komisy
Min = 63.5013	Max = 468.734
Number of records = 997	a
Average value = 262.867	
Standard deviation = 32.6506	
FACILITY	Volebni centrum
Status = not BUSY	
Time interval = 0 - 1440	
Number of requests = 997	
Average utilization = 0.0230787	
Input queue	'Volebni centrum.Q1'
QUEUE	Q1
Time interval = 0 - 1440	
Incoming	189
Outcoming	189
Current length = 0	
Maximal length = 3	
Average length = 0.00304344	
Minimal time = 7.60179e-05	
Maximal time = 0.0699813	b
Average time = 0.0231881	
Standard deviation = 0.0141395	

Obrázek 3: 50 % účast

Dalším experimentem jsme chtěli zjistit zda-li se změni čekání okrsků na volební centrum když změníme volební účast a také počítání hlasů v okrscích. Volební účast jsme si změnili na 175%. Očekávali jsme delší fronty okrsků na volební centrum a kratšího počítání hlasů v okrscích. V obrázku č. 4 kolonky a zjišťujeme výsledek, že se průměrné počítání hlasů snížilo na náš požadovaný předpoklad 100% účastí voličů na necelých 167 minut. Což odpovídá našemu původnímu předpokladu. Dále z kolonky b jsme vyčetli, že byla změna v průměru čekání volebních okrsků 0.034749 v minutách, což je přibližně 2.1 sekundy. Maximální délka fronty byla 7 okrsků.

Z těchto prvotních experimentů můžeme říci, že model bylo potřeba změnit, aby dosáhl naším původním předpokladům. V dalších experimentech jsme zjišťovali, jak se model mění při změně čekání zpracování údajů od volebního centra. Zvyšovali jsme postupně tento údaj, až jsme dostali tohoto experimentu.

```

+-----+
| STATISTIC poctani v komisy |
+-----+
| Min = 123.581                Max = 365.389 |
| Number of records = 997      |
| Average value = 166.869      |
| Standard deviation = 33.8878 |
+-----+
| FACILITY Volebni centrum |
+-----+
| Status = not BUSY |
| Time interval = 0 - 1440 |
| Number of requests = 997 |
| Average utilization = 0.0230787 |
+-----+
| Input queue 'Volebni centrum.Q1' |
+-----+
| QUEUE Q1 |
+-----+
| Time interval = 0 - 1440 |
| Incoming 374 |
| Outcoming 374 |
| Current length = 0 |
| Maximal length = 7 |
| Average length = 0.00902509 |
| Minimal time = 9.61275e-07 |
| Maximal time = 0.218407 |
| Average time = 0.034749 |
| Standard deviation = 0.0329408 |
+-----+

```

Obrázek 4: 175 % účast

Vstupem byla 100% účast voličů a očekávalo se, že výstupem bude doba čekání na volební centrum v průměru okolo 13-17 minut a předpokládali jsme průměrnou dobu počítání jako v předchozích experimentech 2-3 hodiny. Z obrázku č. 5 můžeme vidět ve výsledných statistikách, že průměrně okrsky čekají ve frontě 15 minut. Můžeme si taktéž všimnout, že na volební centrum čekalo více okrsků a to maximálně 250. Dále si z obrázku 6 můžeme všimnout, že doba počítání hlasů je přibližně 152 minuty. Díky výsledkům tohoto experimentu jsme došli názoru, že model funguje podle našich původních představ.

FACILITY Volebni centrum
Status = not BUSY
Time interval = 0 - 1440
Number of requests = 997
Average utilization = 0.0692361
Input queue 'Volebni centrum.Q1'
QUEUE Q1
Time interval = 0 - 1440
Incoming 899
Outcoming 899
Current length = 0
Maximal length = 250
Average length = 9.41075
Minimal time = 0.00268297
Maximal time = 24.9129
Average time = 15.074
Standard deviation = 8.19369

Obrázek 5: 100 % účast

celorepublikovy cas pocitani hlasu
STATISTIC
Min = 94.7627
Max = 347.349
Number of records = 997
Average value = 152.558
Standard deviation = 32.4842

Obrázek 6: 100 % účast

5.3 Závěr experimentů

Bylo provedeno 8 experimentů na vytíženost centra. V průběhů experimentů byla odstraněna nereálná situace čekání orsků na volební centrum.

6 Zhrnutie simulačných experimentov a záver

Naším hlavním cílem bylo vytvořit co nejvíce reálný model volebního systému. Proto jsme se striktně nedrželi zadání a vytvořili jsme 997 orsků. Také jsme zvažovali zda vytváření procesů volebních lístků je zbytečné. Ačkoli tato operace není příliš efektivní pro výpočet, pro reálnost systému je zásadní. Přístup k jednotlivým volebním lístkům je v praxi normální a proto i tento způsob v modelu jsme zachovali.

Díky sadě experimentů jsme zjistili, že model je v rámci školního projektu validní.

7 Referencie

Reference

- [1] Chytilík, R.: *Volební systémy*. Praha:Portal, 2009, ISBN 978-80-7367-548-6.
- [2] Kutiš, V.: *Základy modelovania a simulácie*. Bratislava:STU, 2010, ISBN 978-80-227-3345-8.