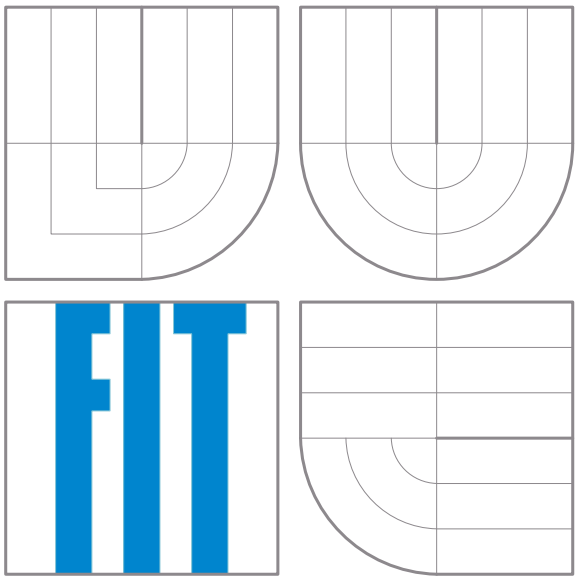


Vysoké učení technické v Brně
Fakulta informačních technológií

SHO Štátna volebná infraštruktúra
Dokumentácia z predmetu IMS

9. prosince 2013

Autori:

Martin Maga xmagam00,
Vojtěch Meca xmecav00

Obsah

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 okrskov a krajských mestskách 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 možňosť konzultácie s pánom doktorom Hrubým(konzultácia ohľadne správnostinášho návrhu Petriho siete).

1.2 Zdroj informácií

Informácie o štátnej volebnej infraštruktúre boli zisťované z dostupnej literatúry. Rovnako sme využili informácii 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 architekrúry: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 nášhu odhadovanému predpokladu. Tak isto sme využili dostupné informácie o spôsobe volieb v Českej republiky 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 co nejvíce reálného 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 dôleţitým 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 časti 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 zaškrtne 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 posielalu na prepočítanie do volebného centra.

2.2 Popis použitých postupov

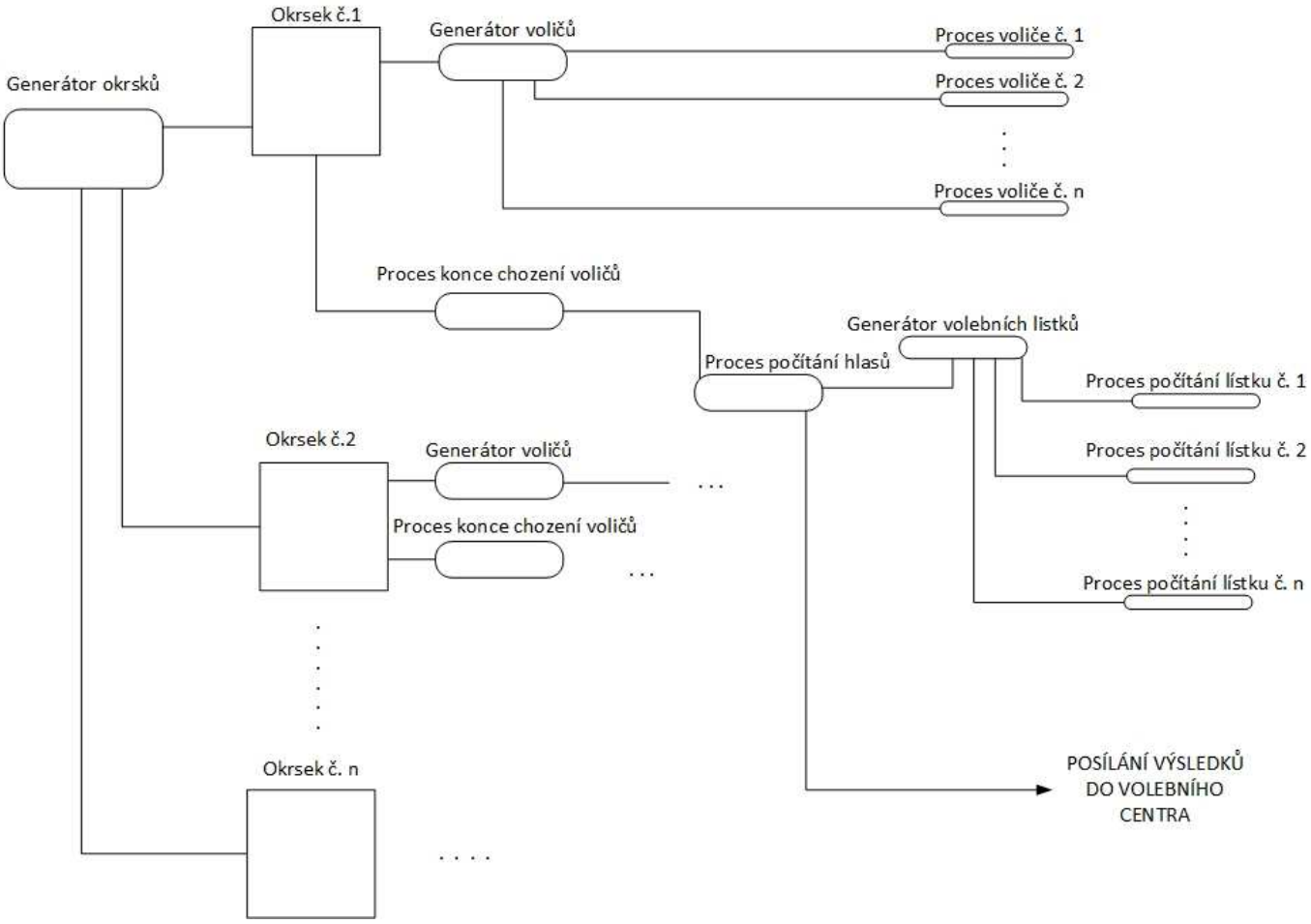
Systém je simulovaný pomocou diskrétnej simulácie (předmet IMS slajd 122) ako systém hromadnej obsluhy (předmet IMS slajd 139). Je to výhodné napr. z hľadiska sledovania dĺžky front (predmet IMS slajd 141) obslužných liniek (predmet IMS slajd 139) jednotlivých častí volebného centra. Použitou technologií k tomuto účelu je jazyk C++ a knižnica Simlib (predmět 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 - modelářska témata

V tejto kapitelo bude načrtnutý koncetuálny model predstavujúci systém volebnej infraštruktúry. Na obrázku č. 1 je uvedeno schéma systému Volební infrastruktury.



Obrázek 1: Konceptuálny model

3.1 Opis konceptuálneho modelu

V obrázku č. 1můžeme vidět několik různých obdelníčků a zaoblených obdelníčků. Popišeme si je jednotlivě:

- a) Zaoblený obdelníček s názvem Generátor okrsků

  Tento zaoblený obdelníček má za úkol generovat obdelníčky Okrsky. Generuje jich přesně kolik jich má vzniknout (určeno například souborem). Po vygenerovaní Okrsků generátor ukončí svou činnost.

- b) Normální obdelníček s názvem Okrsek č. 1...n

  Při vytvoření tohoto objektu se nainicialzují 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 můžou během programu měnit.

- c) Zaoblený obdelníček 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 vygenerovaní všech voličů, zahlásí konec voleb, a tímto její činnost skončí.

- d) Zaoblený obdelníček 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 obdelníček 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 obdelníček s názvem Proces počítání hlasů

  Vytváří Genreátor volibních lístků. Správně spočítáné výsledky posílá do Volebního centra. Poté jeho činnost končí.

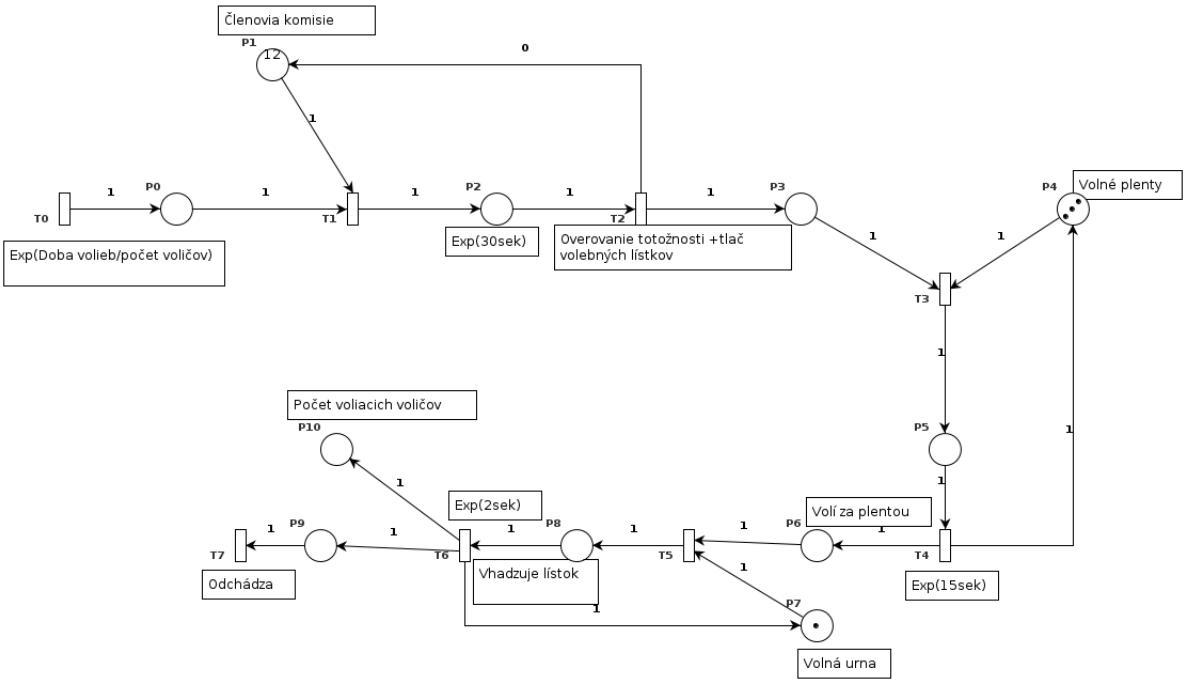
- g) Zaoblený obdelníček s názvem Proces generátor volebních lístků

  Po vytvoření Procesem počítání hlasů začne generovat Procesy počítání listků č.1 .. n. Po vygerenerování všech lístků zkončí svou činnost.

- h) Zaoblený obdelníček s názvem Proces počítání listků č.1 .. n

  Po vytvoření simuluje práci jednoho člena volební komise s tímto lístkem. Po ukončení počítání tohoto lístku jedním členem komise činnost tohoto obdelníčku končí.

3.2 Ukážka Petriho siete pre odovzdávanie hlasu



Obrázek 2: Petriho sieť pre odovzdávanie hlasu

Obrázok č.2 popisuje Petriho sieť, ktorá ukazujem systém volieb. Ukazuje príchod voliča, odvolenie voliča. Na začiatku sa vygenerujú voliči, ktorí prijdu 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 se zařadí do společné fronty a čeká na uvolnění některého ze členů volební komise. Pokiaľ sa občan dostane na radu, tak ho člen volebnej komisie skontroluje. To trvá exponeniciálně 30 sekúnd. Po skončení kontroly občana členem volební komise, občan pokračuje ďalej. Zabere si "Store"s kapacitou 3, ktorý reprezentuje volebnú plentu.
V prípade, že nie je volná plenta, tak sa občan zaradí do jedné společné fronty. Následne užívateľ zajde za plentu a zaškrtá výsledok. To trvá exponenciálně 15 sekúnd. Následne si zabere plentu, ktorá je len 1. Pokiaľ nie je voľná tak sa zaradí do fronty a čaká. Po vhození lístku reprezentující dobu exponenciálně 2 sekundy občan opúšťa systém.

3.3 Implementácia

Program pracuje na základním procesu, který regeneruje v čase nula objekty reprezentující okrsky České republiky. Při vytvoření objektu jednotlivého okrsku se nainicializují data. Objekt reprezentující okresek 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 405 **volatile int mylocal = CANARY1;** na řádek **volatile long mylocal = CANARY1;**. Tím se problém vyřešil a náš model jsme mohli zkompilovat.

4.2 Okrsky navrhnuté k simulácií

Data k okrskům jsme získávali při experimentech ze souboru **okrsky.txt**. Na prvním řádku získáváme informaci o množství okrsků. 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 nejduležitější je údaj o počtu občanů v okrsku.

4.3 Popis architektury

Simulační model má za zálkadní 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 okrsků. 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 oběktu třídy Okrsek a posílá získané data ze souboru do jako parametry konstruktoru. Po vytvoření všech Oksrku, svou činnost končí.

- Třída Okrsek

Tato třída reprezentuje objěkt 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 dekterentují 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 kontruktoru 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 metordu Behavior(). V konstruktoru se předá ukazatel z objěktu Okrsku ze kterého byl generátor vytvořen do svojeho atributu třídy Okrsku. Dále se taky zde definuje čas, který slouži pro znovu aktivovaní této třídy. Zde také měníme parametr účast volíčů pro celou ČR. V mětodě 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 konsturuktoru. 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 metordu 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 místnoti 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 metordu Behavior(). V konstruktoru se předá ukazatel z objěktu Okrsku ze kterého byl timer vytvořen do svojeho atributu třídy Okrsku. V metodě Behavior() se zpustí 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 metordu 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 ulozený v atributu okr třídy a poté se aktivuje. Pomocí metody decrement_poc_lisktu() 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 spoctene_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 metordu 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 objetku 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 metordu 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 repre-

zentující posílání výsledků do volebního centra republiky, se zařízení Centrum_republiky pomocí SIMlib funkce Releise uvolní. Tím končí tento proces.

4.4 Spustenie programu

Pro zkompilování programu je potřeba zadat do příkazové řádky **make** v adresáři s projektem. Po úspěšném zkopilování 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 oksrcích. Nakonec všech výpisů programu získáváme to nejdůležitější a to statistiky modelu.

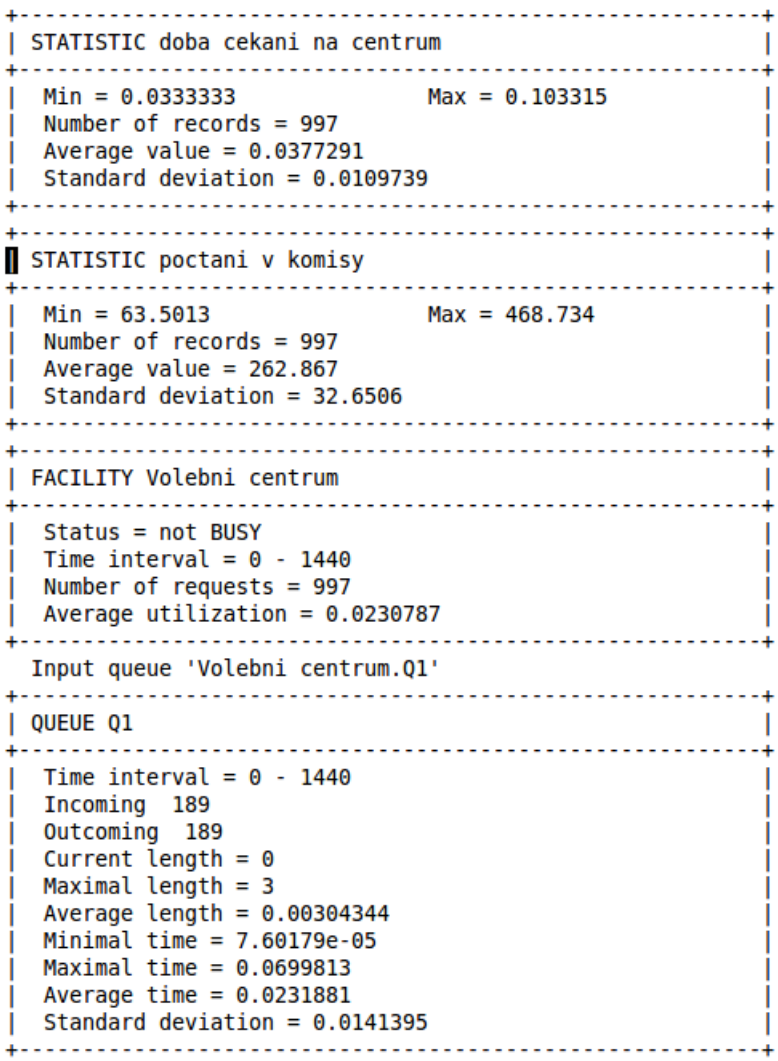
5 Podstata simulačných experimentov

5.1 Postup experimentování

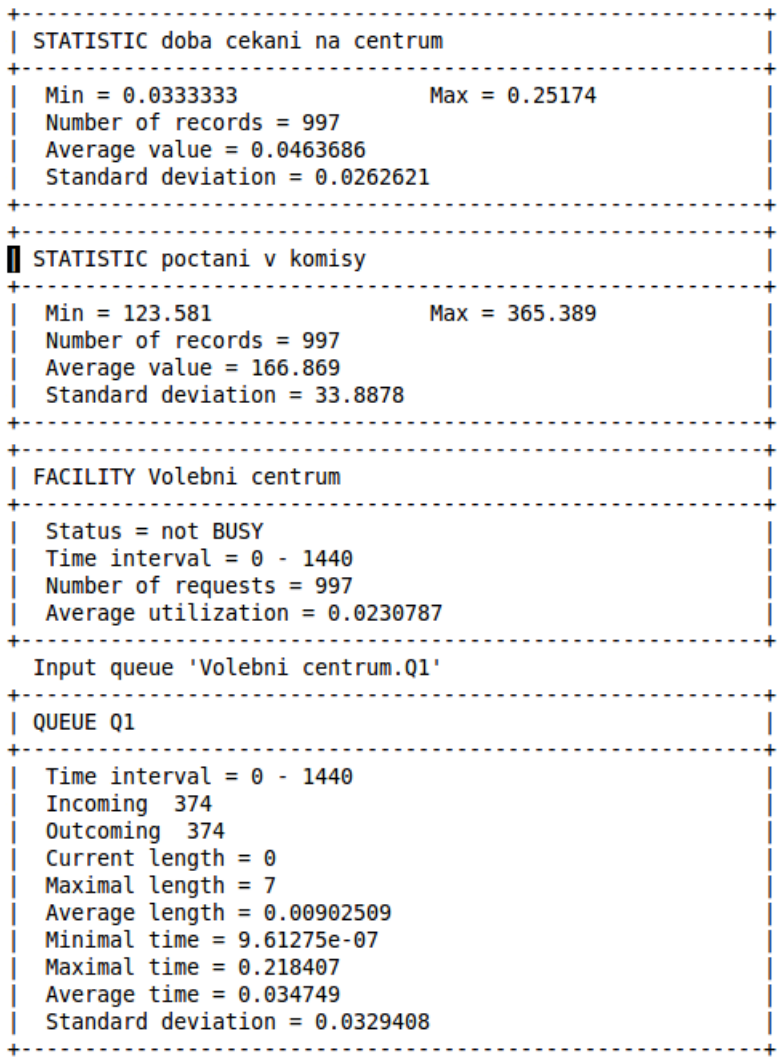
Hlavní cíl, proč byl model vytvořen, bylo zjistit vytíženost Volevního celorepublikového centra a navrhnout změny pro zefektivnění vytíženosti tohoto volebního centra. Dále také jsme získávali informace o průběhu počítání hlasů pro jednotlivé kraje a pak pro celou republiku. K tomu nám pomáhali experimenty. Díky těmto experimentům jsme mohli dosáhnout validitu 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 sledovli, jak se chovají jednotlivé okrsky v délce počítání hlasů a v čekání na volební centrum.

5.2 Popis experimentů

V dalších experimentech jsme se zaměřili pouze na vytíženost volebního centra. Vstupem byla 50% účast voličů a očekávalo se, že výstupem bude doba čekání na volební centrum v průměru okolo 2 sekund. Z obrásku č. 3 získáme výsledné statistiky, ve kterýc jsme zjistili, že průměrně okrsky čekají v minutách 0.0231881, což je přibližně 1.4 sekundy. Další zajímavý udaj byl, že ve frontě na volební centrum bylo maximálně 3 volební okrsky z celé republiky. Obecně se na volební centrum nečekalo. Z obrázku vidíme i další informace o výpisu statistik modelu.



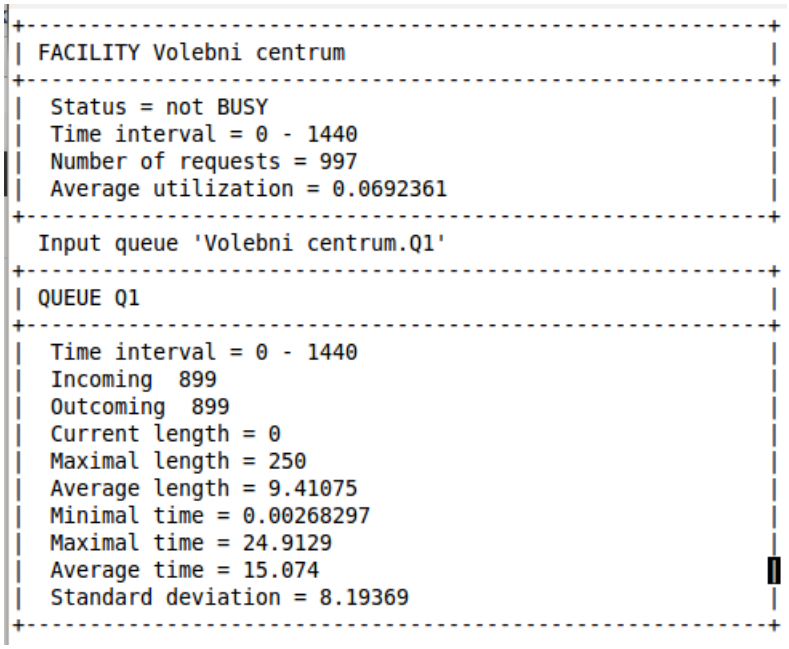
Obrázek 3: 50 % účastníkov



Obrázek 4: 175 % účasť

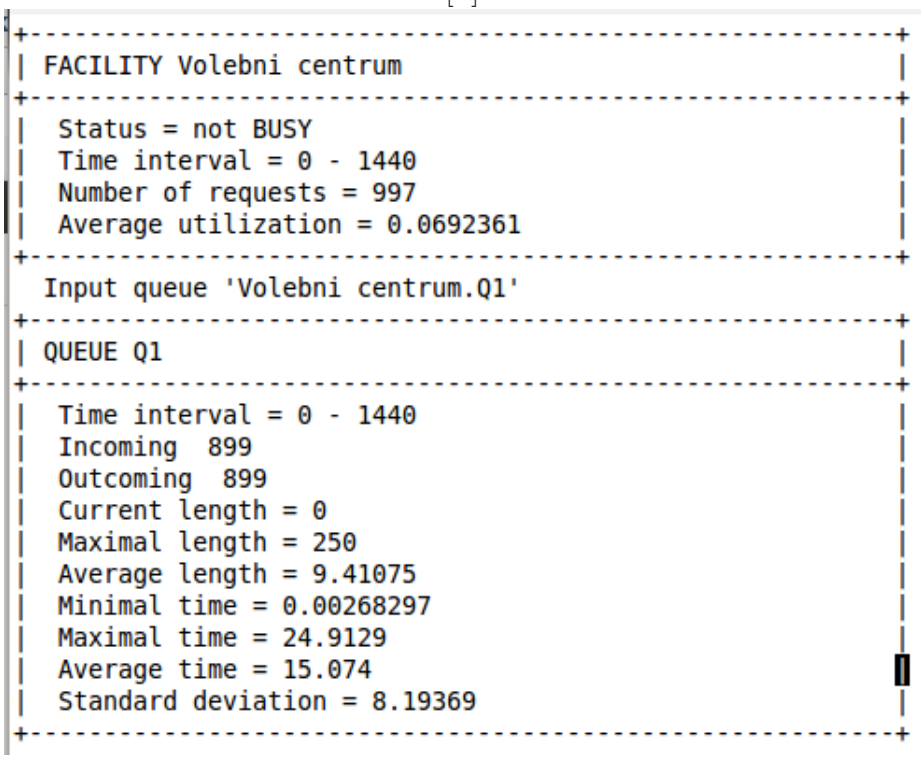
Dalším experimentem jsme chtěli zjistit zda-li se změní čekání okrsků na volební centrum když změníme volevní účast. Tu jsme si změnili na 175%. Očekávali jsme delší fronty orksků na volební centrum. V obrázku č. 4 zišťujeme výsledek, že byla změna v průměru čekání volebních okrsků byla 0.034749 v minutách, což je přibližně 2.1 sekundy. Maximální délka fronty byla 7 okrsků.

Toto nás dovedlo k názoru, že model volebního systému v ČR, který jsme vytvořili není tolik reálný,jak jsme zamýšleli. Změnili jsme čas zpracování výsledků na 6 sekund. A reálnost systému jsme znovu ověřovali experimentem.5
Vstupem byla 100% účast voličů a očekávalo se, že výstupem bude doba čekání na volební centrum v průměru okolo 2-8 minut. Z obrázku 6můžeme vidět, že ve výsledných statistikách zjišťujeme, že průměrně okrsky čekají ve frontě 15 minut. Ve frontě na volební centrum se stav zlepšil na maximálních 250 volební okrsky z celé republiky. Toto byla reálnější situace, než v předchozích případech, což bylo náš cíl experimentů.



Obrázek 5: 100 % účastníkov

[h]



Obrázek 6: 100 % účastníkov

5.3 Závěr experimentů

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

6 Zhrnutie simulačných experimentov a záver

Naším hlavním cílem bylo vytvořit co nejvíce reálný model volebního systému. Proto jsme se striktně nerželi zadání a vyvořili jsme 997 okrsků. 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 zjitili, že model je vrámci školního projektu validní.

7 Referencie

Reference

[1] Chytilek, 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.