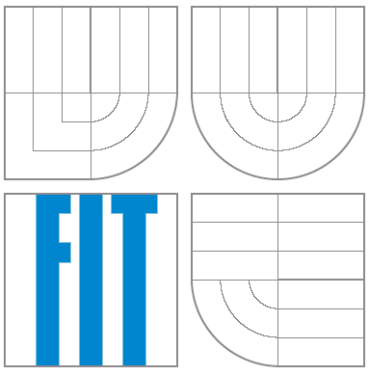

Modelovaní a simulace

2009/2010

Projekt

Diskrétny simulátor založený na riadením udalosťami

Patrik Šebeň, (xseben00@stud.fit.vutbr.cz)
Matúš Kontra, (xkontr00@stud.fit.vutbr.cz)

12. December 2009

Obsah

1. Úvod

Naším zadaním bolo vytvoriť diskrétny simulátor riadený udalostami. Udalosti v našom prípade popisujú zmenu diskrétneho systému. Z hľadiska doby trvania sú to okamžité operácie – zaberajú nulovú dĺžku modelového času. Príkladom udalostí, jako bude neskor podrobnejšie popísane, je napíklad zahajénie obsluhy zákazníka a neskor udalosť dokončenia obsluhy, kde v prvom prípade dojde k zmene stavu obsluznej linky(pokladne) – stane sa obsadenou a neskor dojde k jej uvolneniu udalosťou dokončenia. Z hladiska návrhu nám udalosti umožnujú popísať ľubovolný model avšak, koli ich limitáciam výsledné simulačné modely nie sú vždy dostatočne zrozumiteľné.

Daľsími potrebnými súčasťami simulátora sú nástroje pre spracovanie SHO, ktorých implementácia a funkčnosť budú popísané v nasledujúcej časti.

2.Súčasti simulátora

2.1 Modelovanie udalostí a transakcii

Správanie udalosti je modelované metódou triedy, ktorú može uživatel vo svojej implementácii preťažiť a tým definovať vlastné správanie. Pretože obslužné linky sú obsadzované procesmy, musí ak chce uživatel s týmito nástrojmi pracovať zapúzdriť niekoľko inštancii udalosti do triedy transakcie. Táto trieda NEmodeluje transakciu tak jako sa to deje v procesne orientovanom simulatore – je to len logicke zjednotenie udalosti do celku.

2.2 Prioritné fronty

Často objavujúcov sa datovou štruktúrov při implementácii simulátora boli priority queue – prioritné fronty. Tieto kontajnery poskytujú rovnaké rozhranie jako štandardné fronty – push, top, pop, …, prvky v nich sú však po každom vložení preusporiadané tak aby sa na začiatku nachádzal prvok, ktorý v danom usporiadaní(špecifikovanom operátorom nad uchovávaným datovým typom, príp. Komparačnou funkciou), nachádza na začiatku – v prípade kalendára udaostí záanam s najniším časom a najvyššou prioritou. Kým štandardná knižnica STL jazyka c++ poskytuje takýto kontajner, existovala potreba ho reimplementovať – bola vyžadovaná možnosť odobrať ľubovolný prvok. Na implementáciu sme zvolili neusporiadaný kontajner typu vector, a vlastnosť usporiadania sme udržovali funkciami na vytvorenia a prácu s haldou.

2.3 Algoritmus simulátora – Kalendár udalostí a aktivačné záznamy

Naša implementácia využíva sposob s použitím Kalendára udalostí – je to usporiadaná datová štruktúra postavená nad priority queue, ktorá obsahuje aktivačné záznamy. Aktivačný záznam je štrukturovaný datový typ, nesúci informáciu o aktivačnom čase, priorite a udalosti, ktorá má nastať. Aktivačný záznam je po vložený presunutý na správne miesto vo fronte, takže při volaní pop je navrchu vždycky záznam s udalosťou, ktorá má nastať. Kalendár poskytuje volania umožnujúce vkladať nové záznamy, vybrať udalosť s najmenším aktivačným časom, rušiť už zaradené záznamy atd...

Postup simulácie sa potom riadi nasledujúcim algoritmom:

1. Ak je kalendár prázdny, ukonči simuláciu
2. Vyber prvý záznam, ak je aktivačný čas záznamu vyšší ako čas konca simulácie ukonči simuláciu
3. Nastav simulačný čas na aktivačný čas vybratého záznamu
4. Vykonaj udalosť
5. Choď na krok 1

Algoritmus je implementovaný v triede simulátora a je riadený volaniami v mennom priestore modulu – trieda simulátora není priamo využívaná užívateľom.

2.4 Facility a Store

Sú to typy obslužných liniek . Facility umožnuje obsluhu maximálne jedného požiadavku a Store až n požiadavkov. S každým zariadením je spojená fronta čakajúcich požiadavkov. Využitá je

implementácia prioritnej fronty použitá v kalendáre udalostí, uchováný typ je však iný a komparátor taktiež. Ak udalosť, ktorá je spojená s transakciou požaduje obsluhu, vykoná to volaním Seize (Enter). V tomto momente ak je linka volná zariadenie sa priradí objektu transakcie ktorému patrí udalosť a vráti sa pravdivostná hodnota true – čo užívateľ použije na vykonanie daľších operácii. Ak linka voľná nie je, tak sa použijú parametre volania Seize, ktoré špecifikujú ktorá udalosť transakcie sa má zavolať při získaní linky a vráti sa false. Facility tieto data uloží do čakacej fronty, odkiaľ sú při uvolnení linky v poradí a podľa priority vyberané. Zariadenie sa uvolnuje volaním Release (Leave). V parametroch Seize je možné špecifikovať prioritu obsluhy, požiadavky s vyššou prioritou sú automaticky zaradzované vo fronte na popredné miesta.

2.5 Štatistiky

Každý objekt si sleduje a vyhodnocuje svoje údaje sám. Fronty asociované s oslužnými zariadeniami SHO poskytujú informácie o čakacích dobách, dĺžke fronty a počtoch uložených záznamov. Facility vie vypísať približé využitie a taktiež informuje o počtoch požadavkov a svojom stave. Store poskytuje naviac informáciu o maximálnom obsadení. Simulátor vie vypísať koľko udalosí prešlo simulátorom a jako dlho bežala simulácia(modelový čas).

3. Demonštračný model

Na demonštáciu bol zvolený model čerpacej stanice pre automobily. Na čerpacej stanici sa nachádzajú 4 stojany (modelovane jako Store) a 2 pokladne (modelovane jako Facility). Zákazníci prichádzajú v intervaloch daných exponencialnym rozložením s hodnotou lambda = 0.1 t.j približne každých 10 jednotiek modelového času. Pokúsia sa obsadiť jeden zo stojanov a tankujú čas daný rovnomerným rozložením od 20 do 40 jednotiek. Následne idú zaplatiť. Stavajú sa k voľnej pokladni, alebo sa stavaju do fronty. Čas ktorý tam strávia je 5 jednotiek. Následne opúšťajú systém. (nákres sieťe sa nachádza jednu stranu pod záverom).

4.Záver

I keď sa jedná o dosť zjednodušenú implementáciu, je simulátor použiteľný na experimentovanie s mnohými systémami hromadnej obsluhy. Avšak je nepravdepodobné, že by sa simulátory postavené čisto na udalostiach dalej využívali. Omnoho rozmohnutejšie sú procesne-orientované kniznice, čo je zrejmé už z výsledkou webového vyhľadávania. Dovodom bude pravdepodobne zvýšenie programátorského komfortu tým, že není nutné každú zmenu systému explicitne zachytávať do samostatnej udalosti.

Príloha 1.: Petriho sieť reprezentujúca model čerpacej stanice