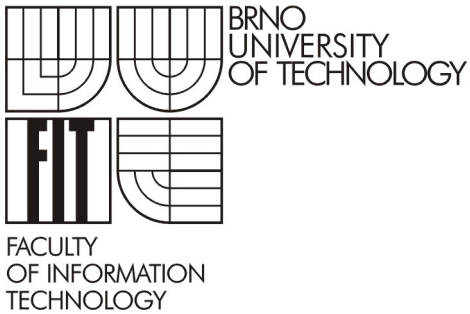


IMS
modelování a simulace

Implementace diskr. simulátoru založeného na řízení UDÁLOSTMI

(opak procesně orientovaného přístupu)

dokumentace

Jaroslav Sendler, xsendl00, mailto:xsendl00@stud.fit.vutbr.cz
Dušan Kovačič, xkovac21, mailto:xkovac21@stud.fit.vutbr.cz

Obsah

1.Úvod....3
2.Popis simulátoru....4
simulace....4
použití simulace:....4
fronta (queue)....4
použití fronty:....4
zařízení (fascility)....4
použití zařízení:....4
sklad (storage)....4
použití skladu:....4
statistiky (stats)....4
použití statistiky:....4
generátor pseudonáhodných čísel....4
Použití generátorů:....5
kalendář událostí....5
příklad užití třídy sCalendar:....5
3.Příloha....6
zadání demonstračního modelu....6
Petriho síť....7

1. Úvod

Knihovna pro diskrétní simulátor s kalendářem událostí založeného na řízení událostmi je implementována v jazyce C++. Obsahuje všechny potřebné třídy pro zpracování SHO, jako je fronta (queue), zařízení (fascility), sklad (storage) a statistiky (stats). Funkčnost knihovny byla ověřena na příkladu modelu SHO s názvem Knihovna, viz příloha.

2. Popis simulátoru

Simulátor splňuje základní požadavky pro simulování v diskrétním čase.

simulace

použití simulace:

fronta (queue)

použití fronty:

zařízení (fascility)

použití zařízení:

sklad (storage)

použití skladu:

statistiky (stats)

Knihovna umožňuje vytvářet statistiky ze simulací. Vytváří se pomocí třídy *sStats()*.

použití statistiky:

generátor pseudonáhodných čísel

Generátory použité v knihovně jsou implementovány třídou *sGen()*. Konkrétně to je generátor normálního (Gaussova) rozložení `normalGen(double mu, double lambda)`,jež je implementován pomocí Box-Muller transformace založené na generovaní bodu umístěného v jednotkové kružnici a na následné transformaci. Generátor exponenciálního rozložení `expGen(double lambda)`, `uniformGen(double a, double b)`, `randomGen()`,

`randomGen(int range)`, `randomGen(int rangeMin, int rangeMax)`, `poissonGen(double lambda)` (inspirace Simlib). Dále se využívá generátor rand() z knihovny cmath.

Použití generátorů:

```
sGen gen; // vytvoreni generatoru
double normal = gen.normalGen(double mu, double lambda);
double exp = gen.expGen(double lambda);
```

kalendář událostí

Kalendář je tvořen lineárním obousměrným zřetězeným cyklickým seznamem s hlavou, třída *sCalendar()*. Tato datová struktura byla vybrána pro následnou jednodušší práci s kalendářem. Metoda `dbInsertEvent(sEvent* event)` implementuje vkládání události do kalendáře, kde event je událost, jež se bude v daný čas provádět. Každá vkládaná událost se zařadí na správné místo podle času a při stejném čase se řadí podle priority. Z toho vyplývá že datová struktura *sCalender* obsahuje jen seřazené prvky. F-ce `dbIsEmpty()` zjistí zda je kalendář prázdný, vrací `true` pokud neobsahuje žádnou událost jinak `false`. Pro zjištění události jež se má provádět a pro její výběr z kalendář použijeme metodu dbGetNextEvent(). Mazání určité události z kalendáře se provádí metodou dbDelete(sEvent* event).

příklad užití třídy *sCalendar:*

```
sCalendr cal;  // vytvoreni kalendare
sEvent event;  // vytvoreni udalosti
cal.dbInsertEvent(new sEvent(event));   // vlozeni udalosti
cal.dbShow();  // zobrazeni obsahu kalendare
event = cal.dbGetNextEvent(); // ziskani první udalosti
sEvent* event1 = event;
cal.dbDelete(event1);    // smazani udalosti
```

3.Příloha

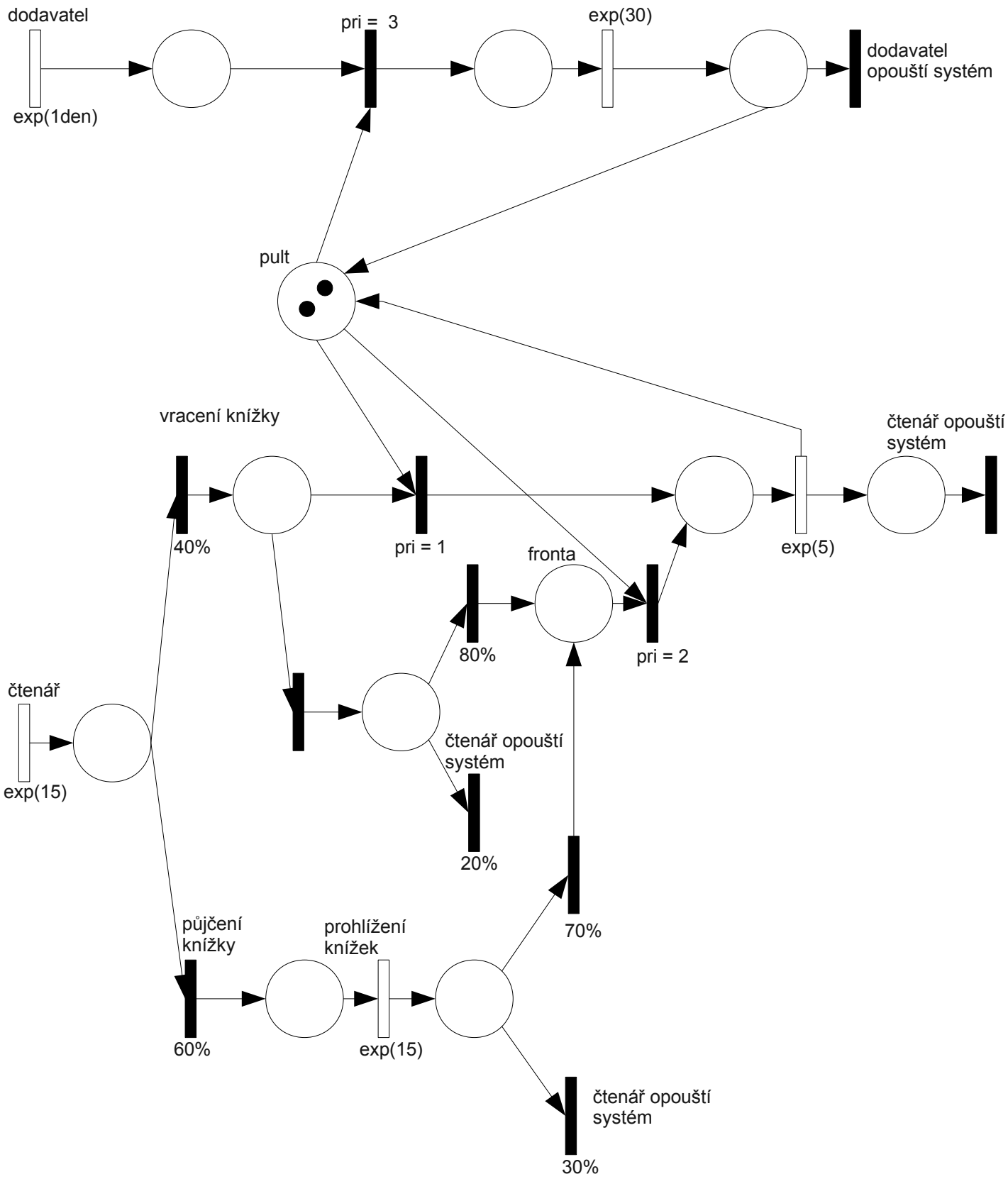
zadání demonstračního modelu

Pro demonstraci funkčnosti knihovny (diskrétního simulátoru s kalednářem událostí) jsme zvolili SHO model Knihovna.

Zadání:

- knihovna obsahuje dva pulty u kterých lze vracet a půjčovat knížky
- studenti do knihovny přichází v intervalech daných exponenciálním rozložením se středem 15 min *exp(15)*
- *40%* studentů jde knížku jen vrátit, *60%* studentů si jde knížku vypůjčit
- pokud je pult volný, student jež vrací knihu jej obsadí a vrací knížku *exp(5),* po tomto čase opouští sytém a uvolňuje pult, pokud je ale obsazený tak *80%* se jich postaví do řady a *20%* z knihovny odchází
- studenti jež si šli půjčovat knihu *exp(15)* hledají nebo prohlíží knihy, následně jich *30%* odejde aniž by si vybrali a *70%* jde k pultu
- do systému *exp(1den)* vstupuje dodavatel, ten má přednost před všemi, pokud přijde a pult je obsazen počká, ale neřadí se do řady ke studentům, po obsazení pultu *exp(30)* dodává knihy, následně opouští systém a uvolňuje pult

Petriho síť


dodavatel
exp(1den)
pri = 3
exp(30)
dodavatel
opouští systém
pult
vracení knížky
40%
pri = 1
fronta
80%
pri = 2
exp(5)
čtenář opouští
systém
čtenář
exp(15)
čtenář opouští
systém
20%
půjčení
knížky
60%
prohlížení
knížek
exp(15)
70%
čtenář opouští
systém
30%