Programowanie Współbieżne (zadanie)

Wojskowa Akademia Techniczna im. Jarosława Dąbrowskiego
prowadzący: mgr inż. Wacław Olech

Artur M. Wolff (grupa nr H7X2S1)

18 maja 2019

1 Treść zadania

W sklepie jest n regałów, na każdym regale k rodzajów produktów w określonej cenie, każdego rodzaju produktu jest m sztuk. Klienci robiąc zakupy zmniejszają liczbę produktów na półkach regałów. W momencie, gdy zabraknie jakiegoś produktu na regale, zatrudniony magazynier uzupełnia jego ilosć blokując przy tym regał dla kupujących. W programie są:

- klienci o ograniczonej cierpliwości (czekają tylko określony czas na magazyniera i rezygnują);
- klienci zdeterminowani, którzy czekają aż magazynier uzupełni regał produktami.

2 Krótki opis problemu

Trudnością w implementacji jest z pewnością synchronizacja dostępu do zasobów współdzielonych (regałów) przez procesy sekwencyjne (klienci). Każdy klient powinien (musi) robić zakupy samodzielnie i ewentualnie czekać na magazyniera. Niepożądany dostęp współbieżny, w tym wypadku zakup tego samego produktu przez klientów w tym samym czasie mógłby mieć nieprzewidziane skutki. Zależnie od implementacji, taki błąd mógły spowodować awarię programu i nawet jego nieoczekiwany koniec (pomijając niespójności w danych w pamięci).

2.1 Przyjęte założenia

2.1.1 Klienci

Aby lepiej oddać rzeczywistość, każdy klient ma ekskluzywny (blokowany) dostęp do regału. Każdy klient losowo wybiera rodzaj produktu, który zamierza zakupić.

2.1.2 Klienci o nieograniczonej cierpliwości

Każdy klient o nieograniczonej cierpliwości losowo wybiera regał, przy którym dokona zakupu.

2.1.3 Klienci o ograniczonej cierpliwości

Każdy "niecierpliwy" klient czeka 1000 milisekund przy zablokowanym przez magazyniera regale. Jeżeli magazynier nie uzupełni przez ten czas towaru, niecierpliwy klient przechodzi do kolejnego regału, aż zakupi towar lub wyjdzie ze sklepu z pustymi rękoma.

2.1.4 Magazynier

Magazynier dostarcza produkty na półkę z szybkością produkt / 1000 ms.

2.1.5 Launcher

Kolejne wątki klientów uruchamiane są co losowany od 0 włącznie do 1000 wyłącznie milisekundowy interwał.

2.1.6 Interfejs

Interfejs programu jest minimalistyczny. Informacje potrzebne do określenia, co w danym momencie dzieje się w programie wypisywane są na standardowe wyjście. Można o nich myśleć jak o logu z urządzenia rejestrującego wyjście ze sklepu klienta, który kupił lub nie zakupił żadnego produktu.

```
Client no. 46 bought product of kind no. 1
Client no. 50 left the store
Client no. 51 left the store
```

3 Wykaz zasobów dzielonych

Zasoby dzielone to część danych zawartych w obiekcie każdego ze sklepowych regałów. Mówiąc obrazowo: są to półki z produktami (zaimplementowane jako lista list / dwuwymiarowa macierz produktów – atrybut `productGroups` klasy `ProductRack`).

4 Wykaz wyróżnionych sekcji krytycznych

- Metoda `acquireOne` klasy `ProductRack`;
- Metoda `buy` klasy `Store` (preambuła sekcji krytycznej).

5 Wykaz obiektów synchronizacji

- Lista semaforów binarnych FIFO (atrybut `rackMtxs` klasy `Store`).

6 Wykaz procesów sekwencyjnych

Procesy sekwencyjne (klienci) uruchamiane są w metodzie `run` klasy `Launcher`. Ich liczba może zostać dowolnie zmieniona (tak jak parametry n, k, m) podczas tworzenia nowej instancji klasy `Launcher`.

7 Wnioski

Problem związany z zadaniem został rozwiązany i zaimplementowany w języku Java. Odmiennie od języków programowania, takich jak C – Java posiada dobre wbudowane wsparcie dla współbieżności.