UXP Zadanie Lato 2022, warianty W11- W16

Początek realizacji: 04 . 04 . 2022 Projekt wstępny: 28 . 04 . 2022; Koniec realizacji: 02 . 06. 2022
Punktacja: 0 - 50 p.

Treść: Napisać wieloprocesowy system realizujący komunikację w języku komunikacyjnym Linda. W uproszczeniu Linda realizuje trzy operacje:

```
output(krotka)
input(wzorzec-krotki, timeout)
read(wzorzec-krotki, timeout)
```

Komunikacja między-procesowa w Lindzie realizowana jest poprzez wspólną dla wszystkich procesów przestrzeń krotek. Krotki są arbitralnymi tablicami dowolnej długości składajacymi się z elementów 3 typów podstawowych: **string, integer, float**. Przykłady krotek: (1, ”abc”, 3.1415, ”d”), (10, ”abc”, 3.1415) lub (2,3,1, „Ala ma kota”). Funkcja **output** umieszcza krotkę w przestrzeni. Funkcja **input** pobiera i atomowo usuwa krotkę z przestrzeni, przy czym wybór krotki następuje poprzez dopasowanie wzorca-krotki. Wzorzec jest krotką, w której dowolne składniki mogą być niewyspecyfikowane: „*” (podany jest tylko typ) lub zadane warunkiem logicznym. Przyjąć warunki: ==, <, <=, > , >= . Przykład: **input (integer:1, string:*, float:*, string:”d”)** – pobierze pierwszą krotkę z przykładu wyżej zaś: **input (integer:>0, string:”abc”, float:*)** drugą. Operacja **read** działa tak samo jak **input,** lecz nie usuwa krotki z przestrzeni. Operacje **read** i **input** zawsze zwracają jedną krotkę (choć pasowac może więcej niż jedna). W przypadku gdy wyspecyfikowana krotka nie istnieje operacje **read** i **input** zawieszają się do czasu pojawienia się oczekiwanej danej.

Zrealizować przestrzeń krotek przy pomocy:

- W11 - potoków nienazwanych – z centralnym procesem koordynującym (rodzicem)
- W12 - potoków nazwanych (FIFO) – z centralnym procesem koordynującym;
- W13 - pamięci dzielonej i semaforów (rodzaj semaforów dowolny);
- W14 - kolejek komunikatów (IPC)
- W15 - plików z mechanizmami zajmowania rekordów (np. jeden plik z rekordami zajmowanymi przez określone procesy)
- W16 – zrealizować system nie jako wieloprocesowy lecz jako wielowątkowy, do synchronizacji należy wykorzystać obiekty mutex i cond, synchronziacja powinna być możliwie „drobnoziarnista” (tj. nie jedna prosta sekcja krytyczna na każdej operacji komunikacyjnjej)

Dodatkowe założenia i wymagania

- Nalezy wprowadzić dodatkowe funkcje realizujące podłączenie się do / otwarcie przestrzeni krotek.
- Można przyjąć statycznie określony maksymalny rozmiar krotki
- Dla typu danej float nie ma warunku ==
- Dla danych string warunki: ==, <, <=, > , >= należy rozumieć jako leksykograficzne porównanie stringu.
- System należy zrealizować jako bibliotekę operacji na krotkach, np. linda_output(…), linda_read(…), linda_input(..) plus ew. odpowiedni serwis.
- Konieczne jest zrealizowanie modułu testującego, może to być zestaw programów korzystających z w.w bilioteki i/lub prosty interpreter command-line
- Operacja input powinna być żywotna, tzn. jeżeli krotka, na którą oczekuje zawieszony proces pojawi się odpowiednio wiele razy operacja input powinna ostatecznie zwrócić krotkę (nie powinno dochodzić do zagłodzenia). Najprostrzy sposób realizacji tego wymagania to szeregowanie żądań w kolejce FIFO.
- Uwaga - Linda definiuje też operację eval() - jej implementacja nie jest wymagana

Platforma: Program *musi* być napisany w języku C/C++ wykonującym się w środowisku Unix (Java, Windows i inne rozwiązania nie są dozwolone).

Więcej informacji o systemie Linda:

https://en.bmstu.wiki/Linda_(coordination_language)
https://oneofus.la/have-emacs-will-hack/files/p80-gelernter.pdf

Instrukcje dot. realizacji projektu:

Początek realizacji: 04 . 04 . 2022 Projekt wstępny: 28 . 04 . 2022; Koniec realizacji: 02 . 06. 2022

Kwestie merytoryczne:

Sprawozdanie **wstępne** powinno zawierać:

1. Temat zadania, treść zadania, skład zespołu (wyróżnić lidera), datę przekazania.
2. Interpretację treści zadania (tj. doprecyzowanie treści).
3. Krótki opis funkcjonalny – "black-box", najlepiej w punktach.
4. Opis i analizę poprawności stosowanych: s**truktur danych, metod komunikacja, metod synchronizacji** (wskazane - z rysunkami, np. zależności czasowych przy wymianie komunikatów, oraz postać/formaty komunikatów – np. w postaci tabelek lub rozpisanych w C struktur/obiektów).
5. Planowany podział na moduły i strukturę komunikacji między nimi, w tym koncepcję realizacji **współbieżności.**
6. Zarys koncepcji implementacji (język, biblioteki, narzędzia, etc.).

Nie należy opisywać kwestii znanych i omawianych na wykładzie, np. zasady funkcjonowania API i funkcji systemowych, standardowych narzędzi programistycznych, itp.

Projekt ostateczny powinien zawierać (6-15 stron):

1. To co projekt wstępny, jeśli potrzeba odpowienio zmodyfikowane i rozwinięte.
2. Pełen opis funkcjonalny "black-box".
3. Podział na moduły i strukturę komunikacji między nimi (silnie wskazany rysunek).
4. Opis najważniejszych rozwiązań funkcjonalnych wraz z uzasadnieniem (opis protokołów, struktur danych, kluczowych funkcji, itp.)
5. Szczegółowy opis interfejsu użytkownika.
6. Postać wszystkich plików konfiguracyjnych, logów, itp.
7. Opis wykorzystanych narzędzi, itp.
8. Opis testów i wyników testowania.

Uwagi dodatkowe:

- **Kodowanie**: język C/C++, środowisku Linux (lub inny Unix: MacOs, BSD, ...)
- Testy (pokaz) powinny obejmować wszystkie wymagane funkcje w prostych i bardziej złożonych przypadkach i sekwencjach użycia
- Obowiązuje standardowo przyjęta metodologia testowania programów - b. ważne jest szczegółowe opisanie przeprowadzonych testów – UWAGA: testy nie mają na celu wykazania, że program **działa** poprawnie. Test ma na celu wykazanie, że program **nie działa** poprawnie!
- B. ważne jest precyzyjne opisanie obsługi **sytuacji wyjątkowych i reakcji na błędy.**
- Punktacja: proj. wstępny: 15p; ogólna ocena realizacji projektu: 15 p.; sprowozdanie końcowe: jakość i kompletność: 10 p, jakość kodu z punktu widzenia inżynierii oprogramowania: 10 p.; w sumie: 50 p.

Uwaga: obecnie konsultacje odbywają się w czwartki 11:00-13:00 poprzez sesję Teams (link będzie podany osobno). Wszystkie informacje o udziale w konsultacjach należy obecnie traktować jako dotyczące e-konsultacji.

Kwestie organizacyjne:

- E-mail: grzegorz.blinowski@pw.edu.pl; przypominam też o istnieniu ogólnodostępnej listy e-mail: uxp1a.a@elka.pw.edu.pl
- Zasady korzystania z e-mail przy realizacji projektu: b. proszę zawsze podawać na początku tematu e-maila: "**UXP imię nazwisko ...**" *(dowolnie wybrany ale zawsze ten sam członek zespołu)*
- ~~Konsultacje odbywają się zawsze we **wtorki w godz 10:05-12:00**; pok. 315; w wyjątkowych przypadkach mogą zostać przełożone na inny dzień.~~

- ~~B. proszę w miarę możności stawiać się na konsultacjach w godz 10:05 – 11:00 (bez konieczności uzgodnienia) lub zasygnalizować chęć przybycia na konsultacje w godz 11:00-12:00, co najmniej dzień wcześniej mailem.~~

- Na konsultacje zawsze można zgłaszać się z dowolnymi pytaniami dot. realizacji projektu, zarówno organizacyjnymi jak i technicznymi. Zapraszam także do wysyłania maili w sprawach zarówno technicznych jak i organizacyjnych. **Konsultacje** w trakcie realizacji projektu nie wymagają obecności całego zespołu.

- Projekt wstępny proszę przekazać w wymaganym terminie (lub wcześniej) e-mailem na podany wcześniej adres podając w temacie: "**UXP Imię Nazwisko Projekt Wstępny**". ~~Proszę nie przekazywać wydrukowanych sprawozdań wstępnych (oszczędzajmy środowisko ☺)~~

- Po zebraniu wszystkich (lub większości) projektów wstepnych opublikuję punktację oraz indywidualne uwagi.

- "Zdanie" projektu końcowego wymaga pojawienia się na e-konsultacjach; konieczne jest: (1) przekazanie **e-mailem** dokumentacji końcowej w postaci pliku pdf, (2) przeprowadzenie **pokazu** działania programu; dopiero po wstępnym pozytywnym zaopiniowaniu projektu poproszę indywidualnie o (3) wysłanie e-mailem dokumentacji oraz źródeł do weryfikacj (lub poprawki). **Uwaga**: przez "zdaniem" projektu **nie należy** przekazywać mailem ani w innej postaci źródeł

- **Pokaz** funkcjonowania musi odbywać się w obecności całego zespołu.

- **Przekazanie źródeł:** źródła powinny być przekazane w postaci jednego pliku archiwalnego w formacie .zip, .tgz lub .tar.gz. Archiwum **nie może zawierać plików binarnych** (programów wykonywalnego, plików .o, plików roboczych repozytorium, itp.), dokumentację proszę wysłać jako drugi załącznik (nie powinna być częścią archiwum).

- **Uwaga** – z przyczyn: formalnych, organizacyjnych i technicznych nie akceptuję źródeł w postaci linku do zdalnego repozytorium.