Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie
Wydział Elektrotechniki, Automatyki,
Informatyki i Inżynierii Biomedycznej

Programowanie Współbieżne i Rozproszone
Kierunek Informatyka
III rok
2020/2021

Maryna Lukachyk,
Yauheniya Padbiarozskaya
grupa 5a

Pakiet do obsługi urządzeń zewnętrznych USB

20.01.2021

Cel programu:
Stworzenie programu symulującego obsługę urządzeń zewnętrznych przez port USB.
Program został napisany w języku Erlang.

Komunikacja procesów:

Komunikacja procesów odbywa się za pomocą wysyłania :

```
Pid ! message
```

i odbierania wiadomości:

```
receive
    {Pid, message} -> do_something
```

Przykładem takich wiadomości z naszego program mogą być:

```
NewPID ! {self(), adding_device}
```

wysyła wiadomość do procesu, który dodaje nowe urządzenie do listy wszystkich urządzeń.

```
receive
       {Pid, adding_device} ->
              …
       Pid ! {ok, added, NewDeviceMap}
```

Natomiast ten drugi proces odbiera komunikat, wykonuje niezbędne czynności i wysyła wiadomość zwrotną, że ‘ok, udało się dodać nowe urządzenie’.

Opis i schemat struktury programu:

- *usb.erl* – start symulacji, główne menu obsługi portów USB, funkcje realizujące obsługę portów (dodawanie/usuwanie urządzenia, przegląd, obsługa, wyjście z symulacji).

- *filesUtils.erl* – funkcje potrzebne do zarządzania zawartością wybranego USB (dodawanie/usuwanie plików, przegląd).

- *utils.erl* – drugorzędne funkcje do wyświetlania danych wyjściowych.

>>> usb.erl
Program uruchamiany poprzez funkcję *start(),* która tworzy główny proces wyświetlający menu i wywołujący funkcje wybrane przez uzytkownika. Szereg działań dostępnych w symulacji dostępny przez główne menu:

- przegląd podłączonych urządzeń
- dodanie nowego urządzenia USB
- obsługa podłaczonych urządzeń
- odłączenie urządzenia
- i oczywiście, wyjście z programu

show_connected_devices()
sprawdza długość listy urządzeń przekazanej do funkcji: jeśli lista pusta, wypisuje komunikat o braku podłączonych urządzeń, w przeciwnym razie wypisuje nazwy zawierające się w liście.

add_device()
tworzy nowy proces, który wysyła komunikat do funkcji dodającej nowe urządzenia

add_device_function()
ta funkcja z kolei odbiera sygnał, aktualizuje listę przechowującą wszystkie urządzenia, a potem wysyła z powrotem komunikat, który sygnalizuje, ze pomyślnie dodano nowe urządzenie, zwiększ liczbę urządzeń o 1.

manage_devices()
wypisuje listę urządzeń, z której użytkownik wybiera pozycję do obsługi. Program sprawdza listę urządzeń i szuka identyfikatoru, który odpowiada indeksu wprowadzonemu przez użytkownika do konsoli. Jeśli długość listy znalezionych PID'ów jest rózna od zera, wysyła do jej pierwszego elementu sygnał w postaci {DevicePID, menu} – wyświetl menu dostępnych konfiguracji urządzenia.

disconect()
sprawdza długość listy urządzeń, jeśli rózni się od zera szuka PID w liście, który odpwiada indeksu wprowadzonemu przez uzytkownika do konsoli. Do elementu posiadającego PID wysyła sygnał w postaci {kill}, aktualizuje listę urządzeń poprez wywołanie ***updateDevicesMap().***

remove_device()
ta funkcja odbiera sygnał, aktualizuje listę przechowującą wszystkie urządzenia, a potem wysyła z powrotem komunikat, który sygnalizuje, że pomyślnie usunięto urządzenie, zmniejsz liczbę urządzeń o 1.

>>> filesUtils.erl
remove_file() i add_file() w argumentach przyjmują listę plików na urządzeniu.
Add_file() pobiera od uzytkownika nazwę i rozszerzenie pliku, które przekazuje do funkcji *addFiles()*.
remove_file() szuka ID pliku w liście plików, po znalezieniu wywołuje *removeFile()*, menu obsługi urządzenia z aktualną listą plików na urządzeniu.
print_files() łatwo się domyślić, służy do wyświetlania plików zapisanych na urządzeniu.

Instrukcja obsługi:
Przejść do katalogu */src*, aby skompilować:

```
erl
> c(usb).
```

Uruchomienie:

```
> usb:start().
```

Ograniczenia programu:
Ograniczeniem może być przepełnienie pamięci, w rezultacie dodania zbyt dużej liczby nowych urządzeń i plików do list DevicesMap oraz FilesMap.

Dalszy rozwój programu:
Nasz program to podstawowa wersja obsługi urządzeń przez port USB. Tę wersję oczywiście można rozszerzyć o dodatkowe komponenty, takie jak osobny moduł ‘właściwości urządzenia’, przedstawiający informacje o numerze portu, pojemności urządzenia (zajęta/wolna pamięć), jego typie (głośnik, telefon, słuchawki).