Projekt Sieci Komputerowe 2

Paweł Błoch

1 Opis Projektu

Projekt został zrealizowany samodzielnie. Tematem projektu jest turowa [1] gra logiczna Reversi. Serwer i klient został zaimplementowany w jezyku C++. Klient został zaimplementowany w jezyku C++, ponadto korzysta z biblioteki graficznej SFML. Do komunikacji sieciowej wykorzystywane jest API bsd-socket. Projekt jest kompilowany na systemie operacyjnym linux.

2 Protokój komunikacji

Komunikacja jest prowadzona przy pomocy API BSD Socket. Komunikacja miedzy serwerem i klientami polega na przesyłaniu stanu rozgrywki. Możliwe informacje przesyłane miedzy serwerem a klientem to:

- Kolor zawodnika - 1 bajtowa wiadomość zawierajaca informacje o tym, jakim kolorem gra dany klient.
- tura - 1 bajtowa wiadomości mówiaca o tym, który gracz ma nastepny wykonać ruch. Informacja o turze jest przesyłana po każdym ruchu gracza, ponieważ zasady gry dopuszczaja przypdaki, gdy jeden z graczy wykonuje dwa lub wiecej ruchów pod rzad.
- plansza - 64 bajtowa informacja zawierajaca obecny stan gry. Stan gry jest przechowywany i obliczany na serwerze. Stan gry jest przesyłany po każdym prawidłowym ruchu jednego z graczy.
- zwyciezca - 1 bajtowa informacja przekazujaca zwyciezce rozgrywki
- poprawność ruchu - 1 bajtowa informacja która określa czy ruch wysłany do serwera jest poprawny z zasadami dla aktualnego stanu gry.
- bład połaczenia - 1 bajtowa informacja mówiaca o tym, że przeciwnik utracił połaczenie z serwerem.

Informacje przesyłane miedzy klientem a serwerem to:

- ruch - 2 bajtowa informacja opisujaca ruch wykonany przez gracza.

[1] Zasady gry dopuszczaja sytuacje, kiedy jeden z graczy wykonuje dwa lub wiecej ruchów pod rzad.

3 Opis implementacji

Implementacja po stronie klienta przypomina implementacje zgodna z paradygmatem obiektowym. Głownom klasa jest klasa **Game**. Metoda **Game::run** odpowiada za przebieg i kontrole rozgrywki. Klasa **BSDsocket** odpowiada za metody przeprowadzajace komunikacje sieciowa. Dzieki niej możemy stworzyć połaczenie, wysłać i odebrać dane oraz zamknać połaczenie. Implementacja klienta korzysta z wielowatkowości. W programie tworzone sa 2 watki. Pierwszy odpowiedzialny za przebieg rozgrywki i komunikacje sieciowa. Drugi odpowiada za obsługe zdarzeń. Watki sa tworzone z użyciem **std::thread**.

Implementacja po stronie serwera opiera sie w głównej mierze na funkcji **game_room** która obsługuje rozgrywke pomiedzy dwoma graczami. Klasa **Board** odpowiada za logiczny przebieg rozgrywki. W niej jest przetrzymywany stan gry. Klasa ta również odpowiada za sprawdzanie poprawności ruchu oraz przydzielanie nastepnej tury. Funkcje **my_read** i **send** odpowiadaja za wysyłanie/odbieranie informacji.

4 Kompilacja i uruchomienie projektu

Aby Skompilować projekt należy pobrać projekt z repozytorium. Kompilacji klienta i serwera dokonujemy oddzielnie. Do kompilacji bedziemy potrzebowali biblioteki SFML (link do pobrania) oraz narzedzie Makefile. Aby skompilować klienta/serwer należy przejść do katalogu reversi_client/reversi_server. Uruchomienie klienta/serwera nastepuje przez komende **./bin/client** / **./bin/server**. Ponadto można uruchomić klienta podajac mu jako argument adres ip. Gdy nie jest podany adres jako argument to domyślenie jest wykorzystywany adres 127.0.0.1.