FAKULTA
APLIKOVANÝCH VĚD
ZÁPADOČESKÉ
UNIVERZITY
V PLZNI

Síťová hra pro více hráčů - Švindl

Úvod do počítačových sítí

23. ledna 2017

Petr Lukašík
A14B0303P
plukasik@students.zcu.cz

Obsah

1. Zadání

- Úlohu naprogramujte v programovacím jazyku C/C++ anebo Java. Pokud se jedná o úlohu server/klient, pak klient bude v Javě a server v C/C++.
- Komunikace bude realizována textovým nešifrovaným protokolem nad TCP protokolem.
- Výstupy serveru budou v alfanumerické podobě, klient může komunikovat i v grafice (není podmínkou).
- Server řešte pod operačním systémem Linux, klient může běžet pod OS Windows XP. Emulátory typu Cygwin nebudou podporovány.
- Realizujte konkurentní (paralelní) servery. Server musí být schopen obsluhovat požadavky více klientů souběžně.
- Zdrojové kódy organizujte tak, aby od sebe byly odděleny části volání komunikačních funkcí, které jste vytvořili na základě zadání, od částí určených k demonstraci funkčnosti vašeho řešení (grafické rozhraní).

2. Hra

2.1 Základní pravidla

Hra se hraje s 28 kartami, které jsou rozděleny do 4 barev, každá po 7 kartách. Hraje se ve 2-7 hráčích. Na začátku hry se všem hráčům rozdá stejný počet karet. V případě přebývajících karet jsou tyto karty položeny do zásobníku karet. Cílem hry je zbavit se všech karet z ruky. Hra končí po zahrání posledního karty a následné rozhodujících akcí, které jsou vysvětleny níže.

2.2 Průběh hry

První karta v balíčku rozhoduje barvu přikládaných karet. V případě prázdného balíčku určuje první hráč, která barva se bude hrát a vyloží první kartu lícem nahoru. Pokračuje následující hráč (po levé ruce) a již přikládá na balíček karty lícem dolu. Může "švindlovat"a vložit do balíčku jinou barvu karty, která není zobrazena jako počáteční. Tato karta je zahraná, pokud ji následující hráč přebije již kartou svou. Kdykoliv ve hře může jakýkoli hráč otestovat zda-li hráč, který vložil poslední kartu na vrch balíčku, podváděl.

Pokud poslední hráč vložil na vrch kartu s jinou barvou než je barva počáteční, podváděl a tedy bere všechny karty v balíčku. První kartu vkládá tedy hráč, který "podvodníka"objevil. V případě souhlasících barev, bere balíček hráč, který nevěřil (otáčel kartu) hráči vykládající poslední kartu. Počáteční kartu tudíž vykládá hráč, který hrál poctivě.

2.3 Konec hry

Hra může skončit třemi způsoby. Hráč vloží poslední kartu a následující hráč ho přebije (nevědomostí) kartou svojí. Nebo bude poslední karta hráče otestována na podvod a vyhraje v případě, že nepodváděl. Případně že hráč s poslední kartou vykládá počáteční kartu.

3. Server

Program lze přeložit pomocí překladače gcc verze 4.8.1+, která již podporuje standart -std=c++11. Ke zdrojovým kódům je přiložen také CMakeList, který usnadňuje kompilaci pomocí nástroje CMake verze 3.0+.

3.1 Spuštění serveru

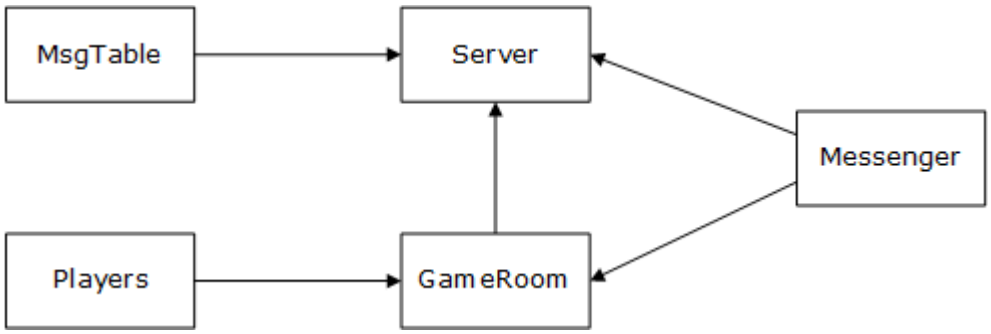
Server lze spustit se základním předdefinovaným nastavením nebo s parametry, určující základní logiku pro server. Jednotlivými parametry se dá přenastavit server na požadované hodnoty. Parametry jsou v pořadí: IP adresa, port, maximální počet herních místností, maximální počet hráčů v jedné místnosti. Při zadání špatný parametrů vyskočí chybová hláška a případná pomoc s jakými parametry lze server spustit.

Příklady:
```
./Server -default
./Server 127.0.0.1 2222 5 6
```

3.2 Implementace

Práce celého serveru se provádí ve třídě *Server*. Jednotlivé sokety jsou zpracovávány funkcí `select()`. Díky této funkci je umožněno spustit server pouze v jednom vlákně a zpracovávat připojení všech uživatelů. Každá herní místnost je však vytvořena v novém vlákně třídy *GameRoom*. Pro udržování informací o uživatelích ve hře je vytvořena pomocná třída *Players* obsluhující data a informace o hráčích v jednotlivých herních místností. Třídy *Server* a *GameRoom* využívají třídu *Messenger* pro posílání zpráv uživatelům pomocí id

soketu.



Obrázek 3.1: Třídy serveru se znázorněnými závislostmi

Jednotlivé místnosti se určují podle id místnosti. Každá si udržuje svůj vektor hráčů, maximální počet a počet momentálně připojených do místnosti. Kromě toho obsahuje informace o stavu hry, který obsahuje: zásobník karet v balíčku, hráč na tahu, předchozí hráč, vítěz a konec hry. Každá herní místnost si také udržuje stav, ve kterém se místnost zrovna ocitá pomocí výčtového typu:

```
enum RoomStatus {
        ROOM_WAIT
        GAME_IN_PROGRESS
        GAME_END
        GAME_WAITING
} roomStatus;
```

Takto lze vždy určit co se v místnosti děje.

3.3 Zprávy

Všechny zprávy a data mezi zprávami jsou oddělovány znakem ':'. Celá zpráva je nakonec ukončena znakem '#'. Přijímání zpráv od uživatele a následnou práci s nimi v serveru je řešeno podle typu zprávy. Typ zprávy je definován ve třídě *MsgTable*. Možné zprávy přijímané od uživatele jsou:

Nekorektní zprávy nejsou přijímány a uživateli je vyslána zpráva o nevalidní zprávě.

Prefix	Parametry	Význam
C_LOGIN	Pepa	Login uživatele
C_USR_READY	-	Uživatel připraven
C_PUT_CARD	K	Vložení karty na vrch balíčku
C_CHECK_CHEAT	-	Kontrola kartu na vrchu balíčku
C_ROOM_INFO	-	Požadavek na informace o uživatelích v místnosti
EOS	-	Chyba v připojení
ERR	-	Chyba v přijímání zprávy
NO_CODE	-	Pouze pro testovací účely
PING	-	Test konektivity

Tabulka 3.1: Vysílané zprávy klienta

3.4 Chyba v připojení

V případě ztráty spojení s uživatelem ve hře je nastaven status herní místnosti na **GAME_WAITING** a je nastaven časový limit pro znovu připojení hráče. V případě, že se hráč nestihne dostavit do daného časového limitu, hra se ukončí bez vítěze. Kontrola správného hráče není nijak unikátní a kontroluje se pouze podle jména. Je tedy možné, že po odpojení hráče se může připojit pod stejným jménem jiný uživatel. Celý systém "reconnectu"běží v novém vlákně, kde se odpočtem kontroluje připojení hráče. Hráč se znovu připojí až tehdy, kdy vyšle zprávu **C_ROOM_INFO#**.

Pokud hráč ztratí spojení pokud je v čekací místnosti, server ho automaticky vyhodí z místnosti a uzavře soket.

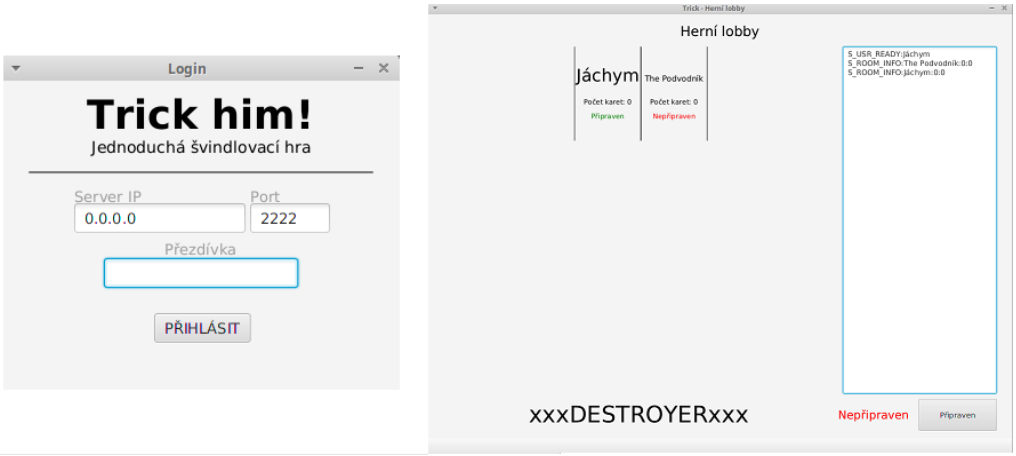
Server a klient si neustále kontrolují spojení zprávami. Klient při navázání spojení se serverem a úspěšném přihlášení začne posílat zprávy **PING#** s intervalem 2sec. Na každou takovou zprávu čeká ihned odezvu od serveru zprávou **PONG#**. Pokud server či klient neobdrží po časovém limitu 5 sec žádnou zprávu mezi sebou obě strany uzavřou soket. A server ukončí spojení v závislosti na hráči a stavu hry.

4. Klient

Překlad klienta je podporován nástrojem Maven verze 3.0+. Klient pro kompilaci využívá verzi SDK 1.8+. Jiné verze nebyli pro kompilaci testovány, není tak jistá jejich funkčnost.

4.1 Hlavní okna

Klientská část je rozdělena na dvě hlavní okna **LoginController** a **GameController** viz Obr. 4.1.



Obrázek 4.1: Hlavní okna klientské části

Přihlašování probíhá v **LoginController**, kde se zadají náležité parametry pro připojení na server. V případě nesprávnosti parametrů vyskočí v

dolní liště chybová hláška, který z parametrů není správný. Kontroluje si i délka jména která musí být v rozmezí 3-15 znaků. Při úspěšném připojení se otevře okno **GameController**.

Ten již představuje herní čekací místnost s informační konzolí po pravé straně. Uživatel může stisknout že je připraven. Tato akce se nedá vzít zpět a hráč již bude připraven do spuštění hry. Ke spuštění hry je třeba, aby všichni hráči byli připraveni. Minimální počet hráčů pro spuštění hry jsou 2.

4.2 Zpracovávané zprávy

Klient odesílá zprávy serveru, který je nazpět potvrzuje svými odpovědmi. Posílané zprávy serveru jsou v tabulce 3.1. Server nazpět posílá zprávy popsané v následující tabulce 4.1.

Prefix	**Parametry**	**Význam**
S_LOGGED	-	Úspěšné přihlášení uživatele
S_NICK_LEN	-	Chyba v délce jména
S_NAME_EXIST	-	Vložení karty na vrch balíčku
S_SERVER_FULL	-	Kontrola kartu na vrchu balíčku
S_JOIN_ERR	-	Chyba připojení do místnosti
S_ROOM_INFO	Pepa:0:0	Informace o hráči v místnosti (jméno:připraven:počet karet)
S_USR_JOINED	Pepa	Nový hráč v místnosti (jméno)
S_USR_LEFT	Pepa	Hráč opustil místnost (jméno)
S_USR_READY	Pepa	Hráč je připraven (jméno)
S_USR_READY_ACK	-	Potvrzení o vašem připravení
S_CARDS_OWNED	K:K:R	Vlastněné barvy karet (barva:barva:barva...)
S_STACK_CARDS	2:K	Informace o stavu balíčku (počet karet:barva ve spodu)
S_ON_TURN	Pepa:Jan:8	Hráč na tahu (momentální:předchozí:počet karet předchozího)
S_CARD_ACK	K	Potvrzení o přijetí karty (barva karty)
S_CHEATED_CARD	G	Informace o kartě na vrchu (barva karty)
S_CARD_NUM_CHANGE	Pepa:12	Změna počtu karet daného hráče (jméno:počet karet)
S_GAME_WINNER	Pepa	Vítěz hry (jméno)
S_GAME_END	-	Konec hry bez vítěze
S_DISCONNECT	Pepa	Informace o ztrátě spojení s hráčem (jméno)
S_RECONNECT	Pepa	Informace o znovuobnovení spojení s hráčem (jméno)
PONG	-	Potvrzení o přijetí PING zprávy

Tabulka 4.1: Vysílané zprávy serveru

5. Závěr

Práce jako taková byla velmi komplikovaná a mohu s naprostou jistotou říct, že by šla ještě vylepšit. Ošetření všech kritických sekcí na serveru i klientu není vůbec jednoduché a je možné že v práci se stále dá najít chybný stav, který mě doteď nenapadl. Spojení serveru a klienta v sobě skrývá mnohá úskalí, které se na první pohled možná nezdají složitá, avšak po bližším prozkoumání mohu říct, že to bylo zatím nejobtížnější programování, které jsem na vysoké škole vyzkoušel.

Nejvíce času zabralo pochopení práce s vlákny a sokety. Herní logika pak byla obtížná pouze v ošetření všech stavů, které mohou nastat v případě zprávy, která do daného stavu nepatřila. Klientská část již tak obtížná nebyla, protože v podstatě pouze reaguje na zprávy od serveru.

V případě snahy pokračovat v tomto projektu bych spíše doporučoval začít úplně od začátku, než se snažit přidat, opravit či vylepšit funkcionalitu.