



ZÁPADOČESKÁ UNIVERZITA V PLZNI
FAKULTA APLIKOVANÝCH VĚD
KATEDRA INFORMATIKY A VÝPOČETNÍ TECHNIKY

SEMESTRÁLNÍ PRÁCE KIV/UPS
Síťová hra Kvarteto

Jaroslav Klaus
A13B0347P
jklaus@students.zcu.cz
15. ledna 2017, Plzeň

Obsah

1	Zadání	2
2	Popis hry	2
3	Protokol	2
3.1	Typy zpráv a dat	3
3.2	Posloupnost zpráv v bezproblémové hře	5
4	Implementace	6
4.1	Server	6
4.1.1	Klienti mimo hru	6
4.1.2	Management hry	7
4.1.3	Periodické zasílání „keep-alive“	7
4.1.4	Kontrola dostupnosti klientů	7
4.2	Klient	7
5	Uživatelská příručka	8
5.1	Server	8
5.2	Klient	8
6	Závěr	9

1 Zadání

Vytvořte hru, kterou může hrát více hráčů. Řešení musí obsahovat program serveru, který bude obsluhovat více klientů i her současně. Klient bude po opětovném přihlášení pokračovat tam, kde přestal. Program musí být odolný proti chybným zadáním a musí adekvátně reagovat na výpadky serveru i klientů. Klient bude naprogramován v jazyce *Java* a server v jazyce *C* nebo *C++*. Konkrétně vytvořte hru *Kvarteto* za použití protokolu *TCP*.

Program bude doplněn o zpracování statistických údajů (počet odeslaných a přijatých zpráv a bytů, počet přijatých spojení, počet spojení ukončených z důvodu chybných dat apod.). Dále nesmí chyby v komunikaci znemožnit případně znesnadnit používání grafického rozhraní klienta.

2 Popis hry

Hra Kvarteto je relativně jednoduchá. Jedná se o karetní hru, kterou je možné hrát ve třech a více hráčích (maximálně však 32). Hraje se s kartami označenými písmenem a číslem (1A, 1B, 1C, 1D, 2A, 2B, ..., 8D). Cílem hry je získat čtveřici karet s příslušným číslem, např. 1A, 1B, 1C a 1D. Tah probíhá tak, že hráč některého ze soupeřů požádá o kartu, kterou potřebuje. Pokud ji soupeř má, předá ji hráči, který pokračuje dalším tahem. Pokud kartu soupeř nemá, je na tahu on.

3 Protokol

Protokol TCP je tzv. spolehlivý. Řeší za nás ztrátu zpráv během přenosu, jejich duplicitu, pořadí a další, proto není potřeba tyto stavy řešit. Důležité však je zjistit výpadek klienta, který řádně neukončil spojení. To je zajištěno periodickým posíláním zpráv „*keep-alive*“. Pokud od serveru nebo klient ta určité doby nedorazí tato zpráva, bude prohlášen za nedostupného a spojení s ním ukončeno.

Komunikační protokol bude textový s oddělovačem `;`. Bude se skládat ze tří částí, viz tabulka 1. Data budou jako oddělovač používat `,`. Zpráva bude ukončena znakem `\n` nebo `\r\n` (v závislosti na použitém OS). Vzhledem k tomu platí pro data (přezdívky, id her apod.) omezení, že nesmí obsahovat tyto vyhrazené znaky. Zpráva tedy může vypadat např. takto: `"5;6;nick,3\n"`.

Tabulka 1: Vzhled zprávy

typ zprávy	velikost dat	data
------------	--------------	------

3.1 Typy zpráv a dat

Typ zprávy bude uveden číslem, u kterého bude dále popsán význam a naznačeno, jak budou vypadat data. Jako první v popisu bude uvedeno, zda se jedná o zprávu od klienta na server (C), nebo od severu ke klientovi (S).

0. S, C, **keep-alive**

Data: -

1. C, **žádost o seznam her**

Data: -

2. S, **seznam her**

Data: počet her (int)

seznam her [
 id hry (string)
 počet připojených (int)
 kapacita (int)
]

3. C, **žádost o připojení**

Data: jméno (string)

id hry (string)

4. S, **odpověď na žádost o připojení**

Data: 0 pokud vše proběhlo v pořádku

1 pokud je hra plná

2 pokud je hráč již na serveru

3 pokud již hra neexistuje

5. C, **žádost o vytvoření hry**

Data: jméno (string)

počet soupeřů (int)

6. S, **odpověď na žádost o vytvoření hry**

Data: 0 pokud vše proběhlo v pořádku

1 pokud je hráč již na serveru

2 pokud bylo zadáno moc málo hráčů

7. S, **začátek hry**
 Data: počet soupeřů (int)
 pro každého protihráče [
 jméno (string)
 počet karet (int)
]
 počet mých karet (int)
 seznam mých karet [
 karta (string)
]
8. S, **jsi na tahu**
 Data: -
9. S, **oznámení, kdo je na tahu** (pro ostatní hráče)
 Data: jméno (string)
10. C, **tah**
 Data: protihráč (string)
 požadovaná karta (string)
11. S, **odpověď na tah**
 Data: 0 pokud kartu vlastní
 1 pokud kartu nevlastní
12. S, **oznámení tahu** (pro ostatní hráče)
 Data: výsledek tahu (
 0 pokud kartu vlastnil
 1 pokud kartu nevlastnil
)
 jméno (string)
 karta (string)
 protihráč (string)
13. S, **vyhrál jsi** (= konec hry, má kvarteto)
 Data: -
14. S, **oznámení o někom, kdo vyhrál** (pro ostatní hráče)
 Data: jméno (string)
15. S, **prohrál jsi** (= konec hry, nemá žádnou kartu)
 Data: -

16. S, **oznámení o někom, kdo prohrál**
Data: jméno (string)
17. S, **ukončení hry z důvodu nedostupnosti hráče**
Data: jméno (string)
18. C, **žádost o navázání přerušené hry**
Data: jméno (string)
19. S, **odpověď na žádost o navázání přerušené hry**
Data: výsledek (
0 pokud vše proběhlo v pořádku, následují data
1 pokud hráč nebyl na serveru nalezen
2 pokud byl nalezen, ale je aktivní
pokud vše proběhlo v pořádku, ale hra ještě nezačala
)
počet soupeřů (int)
pro každého protihráče [
jméno (string)
počet karet (int)
]
počet mých karet (int)
seznam mých karet [
karta (string)
]
20. C, **odpojuji se**
Data: -

3.2 Posloupnost zpráv v bezproblémové hře

Každých 2500 milisekund zpráva "0;0;" od klienta i serveru.

Po navázání spojení může vypadat komunikace např. takto:

```
C: 1;0;
S: 2;1;0
C: 5;6;nick,2
S: 6;1;0
S: 7;55;2,enemy1,11,enemy2,10,11,1A,2B,3C,4B,4D,5A,6B,7C,8D,5C,2D
S: 8;0;
C: 10;9;enemy1,2A
S: 11;1;1
S: 9;6;enemy1
```

S: 12;16;0,enemy1,1A,nick
S: 14;6;enemy1

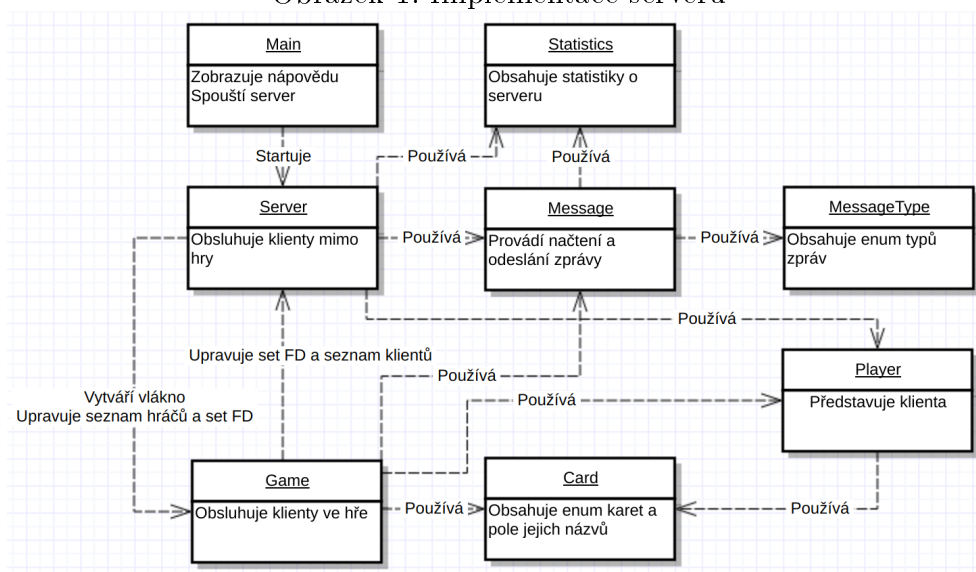
4 Implementace

Zde je popsána dekompozice, rozvrstvení, metoda paralelizace a další implementační záležitosti programů.

4.1 Server

Server je rozdělen do tří hlavních částí. První část je zpracování požadavků klientů, kteří nejsou v žádné hře. Druhou částí je management hry a jejích klientů. Třetí část slouží k periodickému zasílání „keep-alive“ zpráv všem klientům na serveru a ke kontrole, zda jsou klienti dostupní (posílají „keep-alive“ zprávy), a k případnému ukončení spojení s těmi nedostupnými. Každá z těchto částí běží ve vlastním vlákne, přičemž každá hra má vlákno vlastní. UML diagram implementace je vyobrazen na obrázku 1.

Obrázek 1: Implementace serveru



4.1.1 Klienti mimo hru

Tato část volá funkci *select* s krátkým timeoutem nad sockety klientů mimo hru a serveru. Pokud se zjistí, že na některém z klientských socketů jsou

data ke čtení, provede se jejich načtení a zpracování. Pokud se něco událo na socketu serveru, je přijato nové spojení s klientem. Typy zpráv, které budou zpracovány v této části jsou: 0, 1, 3, 5 a 18. Zde probíhá vytváření her, připojení klientů do her apod.

4.1.2 Management hry

Tato část volá funkci *select* s krátkým timeoutem nad sockety klientů ve hře. Pokud se zjistí, že na některém z klientských socketů jsou data ke čtení, provede se jejich načtení a zpracování. Typy zpráv, které budou zpracovány v této části jsou: 0, 10 a 20. Zde probíhá management hry jako je rozdání karet, kontrola tahů apod.

4.1.3 Periodické zasílání „keep-alive“

Tato část prochází seznamy klientů mimo hry i ve hrách a každému pošle zprávu typu „keep-alive“ a poté se na 2500 milisekund vlákno uspí. Toto je jediná funkce této části.

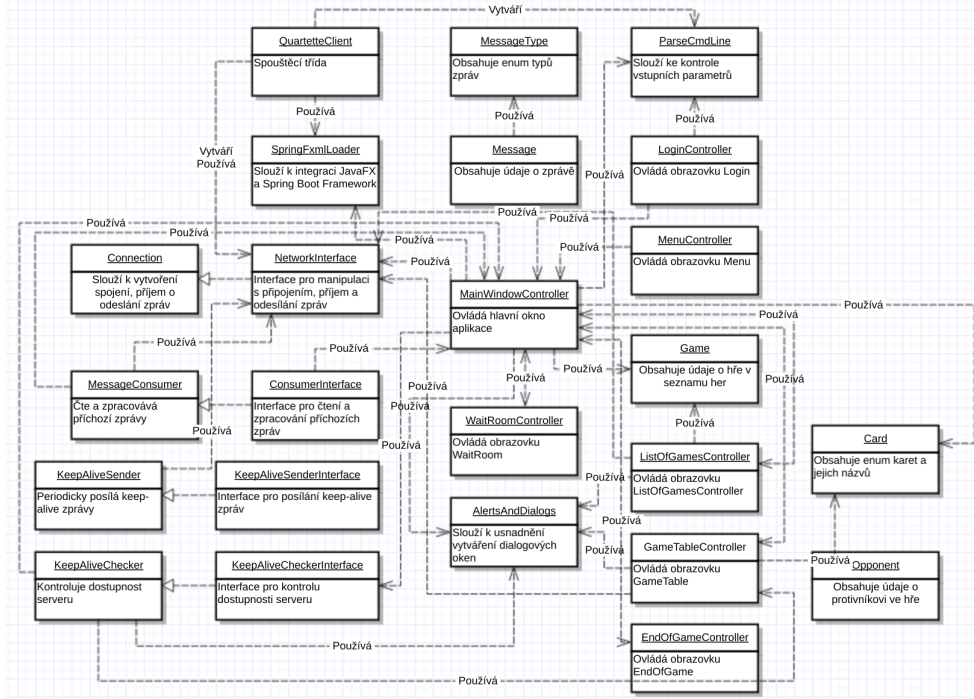
4.1.4 Kontrola dostupnosti klientů

V tomto vlákne se prochází seznamy klientů mimo hry i ve hrách. U každého se kontroluje čas poslední přijaté zprávy „keep-alive“. Pokud je tento čas starší než 5 sekund, nastaví se stav hráče jako neaktivní a je tak možné, aby klient požádal o navázání přerušené hry (pokud v nějaké byl). Pokud je čas starší než 25 sekund, je spojení s klientem ukončeno a případná hra ukončena z důvodu nedostupnosti hráče.

4.2 Klient

Postup práce klienta je téměř identický jako ten na serveru. Liší se např. v tom, že zde existuje pouze jedno vlákno pro příjem a zpracování zpráv. Další vlákno slouží pro odesílání zpráv „keep-alive“, jiné pro kontrolu dostupnosti serveru a poslední k ovládání GUI. UML diagram implementace je na obrázku [2](#).

Obrázek 2: Implementace klienta



5 Uživatelská příručka

5.1 Server

Server je napsán v jazyce *C++* v operačním systému Linux a je k němu vytvořen příslušný *Makefile*, kterým je možné server pomocí nástroje *make* jednoduše přeložit. K přeložení je potřeba použít překladač *g++* s podporou standardu *c++11*, knihovnu *pthread* pro možnost vytvoření a práce s vlákny a knihovnu *uuid* (případně *uuid-dev*) pro vytváření id her. Server se spouští se dvěma parametry, adresou pro naslouchání a portem, např.:

```
./QuartetteServer INADDR_ANY 10000
```

Ukončení serveru je možné provést zasláním signálu *SIGINT* (např. stisknutí Ctrl + C).

5.2 Klient

Klient je napsán v jazyce *Java* za použití *JavaFX*, *JDK 8* a *Spring Boot Framework* a fungovat by měl jak v OS Linux, tak Microsoft Windows. Pro jeho přeložení je potřeba *JDK 8* nebo vyšší a nástroj *maven*, pro spuštění by měl stačit *JRE 8* nebo vyšší. Příložen je soubor *pom.xml* pro automatický

překlad nástrojem *maven*, který se provede např. příkazem:

```
mvn package
```

Vytvoří se soubor *target/QuartetteClient.jar*, který lze spustit příkazem:

```
java -jar QuartetteClient.jar
```

Je možné spustit klienta s parametry, které předvyplní daná pole, např. takto:

```
java -jar QuartetteClient.jar -host localhost
```

```
java -jar QuartetteClient.jar -port 10000
```

```
java -jar QuartetteClient.jar -nick nick, případně jejich kombinací.
```

Po spuštění je požadováno zadání adresy a portu serveru, viz obrázek 3. Po připojení se zobrazí menu jako na obrázku 4. Seznam her se zobrazí po kliknutí na příslušné tlačítko a vypadá jako na obrázku 5. Novou hru vytvoříme kliknutím na příslušné tlačítko a do dialogu 6 zadáme svou přezdívku a počet protihráčů. Hra vypadá jako na obrázku 7. Obsahuje seznam protihráčů, vašich karet a historii. Ovládá se tak, že vyberete z nabídky protihráče, od kterého chcete kartu. Pokud jste na řadě, objeví se dialog, ve kterém vyberete kartu, kterou chcete. Pokud se ze hry odpojíte nekorektně, máte v blízké době šanci se do ní opět vrátit pomocí příslušného tlačítka v menu. To otevře dialog, kam zadáte svou původní přezdívku a budete vráceni do hry.

6 Závěr

Vytvoření zabralo o něco více času, než jsem očekával, cca 80 až 90 hodin. Jedním z důvodů bylo to, že nebylo jednoduché správně dekomponovat problém a uvědomit si všechny možné problémy a situace, které mohou nastat. Další překážkou byl jazyk, ve kterém musel být server implementován.

V práci jsem navrhl komunikační protokol a vzhled zpráv, stavové diagramy serveru, hry a hráče, implementoval a otestoval program klienta i serveru. Zadaní se mi tak, dle mého názoru, podařilo splnit.

Obrázek 3: Připojení k serveru



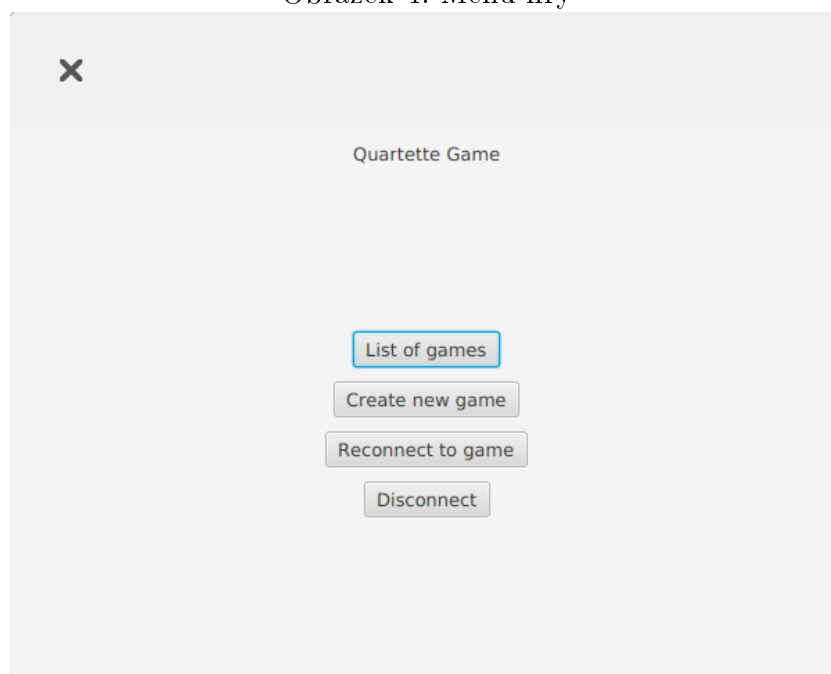
Quartette Game

Enter server address: localhost

Enter server port: 10000

Log In

Obrázek 4: Menu hry



Quartette Game

List of games

Create new game

Reconnect to game

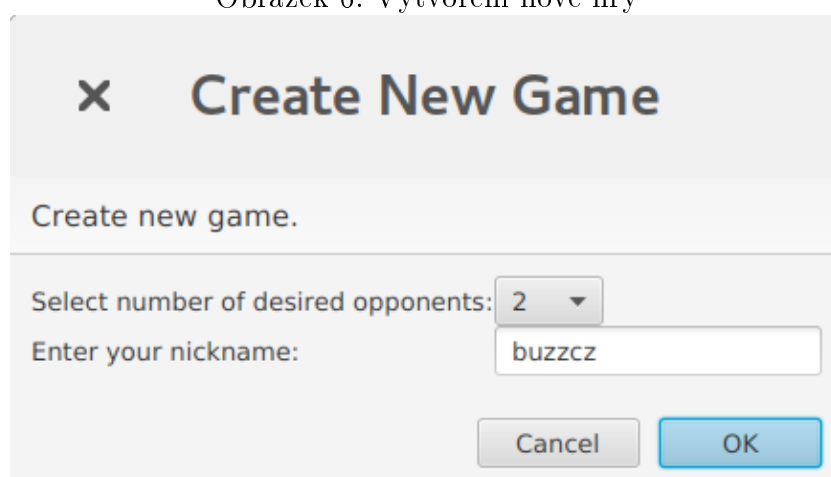
Disconnect

Obrázek 5: Seznam her



A screenshot of a software window titled "Quartette Game". The window has a close button (X) in the top-left corner. Below the title bar, the text "Connected players: 1, capacity: 3." is displayed above a list box. The list box is currently empty, showing only horizontal separator lines. At the bottom of the window is a "Back" button.

Obrázek 6: Vytvoření nové hry



A screenshot of a software window titled "Create New Game". The window has a close button (X) in the top-left corner. Below the title bar, the text "Create new game." is displayed. Below this, there are two input fields: "Select number of desired opponents:" with a dropdown menu showing the value "2", and "Enter your nickname:" with a text input field containing the text "buzzcz". At the bottom of the window are two buttons: "Cancel" and "OK".

Obrázek 7: Hra

