Semestrální práce KIV/UPS
Síťová hra Piškvorky (TCP)

Tomáš Klepač

12. ledna 2026

Obsah

1 Úvod

Tato semestrální práce implementuje síťovou dvouhráčovou hru Piškvorky nad transportním protokolem TCP.

Serverová část aplikace běží na operačním systému Linux a je implementována v jazyce C (standard C17).
Klientská část je multiplatformní grafická aplikace (testováno na Windows i Linux) implementovaná v jazyce C# na platformě .NET 8.0 s využitím UI frameworku Avalonia.

Cílem bylo vytvořit:

- plně funkční server obsluhující více místností současně,
- moderního grafického klienta,
- definovaný textový protokol umožňující implementaci libovolného alternativního klienta či serveru,
- robustní zpracování výpadků připojení, reconnect a ošetření chyb.

Projekt splňuje požadavky KIV/UPS pro implementaci síťové hry včetně konfigurace serveru, validace vstupů a dokumentace protokolu.

2 Popis hry

Implementovaná varianta hry je klasická mřížka 3×3. Hráči se střídají v tazích. Cílem je umístit tři své symboly do řady.

Vítězství nastává při vytvoření řady tří symbolů ve směru:

- horizontálním,
- vertikálním,
- diagonálním.

Pokud se hrací pole zaplní bez vítěze, hra končí remízou. Po skončení hry mohou hráči hlasovat pro `REPLAY` (odvetu).

3 Architektura aplikace

3.1 Serverová architektura

Server je implementován v jazyce C a využívá BSD sockety. Architektura je založena na modelu **"Thread-per-Client"**, kde každý připojený klient je obsluhován vlastním vláknem, zatímco hlavní vlákno přijímá nová spojení.

Klíčové moduly:

- **main.c** – inicializace socketů, hlavní accept cyklus, spouštění vláken.
- **config.c/h** – načítání konfigurace ze souboru `server.config`.
- **client.c/h** – správa klientských struktur, odesílání zpráv.

- **room.c/h** – správa herních místností, párování hráčů, herní smyčka.
- **game.c/h** – logika hry, validace tahů, detekce výhry.
- **utils.c/h** – pomocné funkce (logging, bezpečné čtení/zápis).

Server dále využívá samostatné vlákno pro **heartbeat** mechanismus, které periodicky rozesílá `PING` zprávy a kontroluje dostupnost klientů.

3.2 Klientská architektura

Klient je napsán v C# na frameworku .NET 8.0 a využívá knihovnu **Avalonia UI** pro grafické rozhraní. Díky tomu je aplikace spustitelná na Windows, Linuxu i macOS.

Struktura projektu (MVVM pattern):

- **Views** – definice vzhledu oken (MainWindow, GameView, LobbyView) v XAML.
- **ViewModels** – propojení logiky a zobrazení, zpracování stavu hry.
- **Models** – datové modely a síťová vrstva (TCP komunikace).

Klient komunikuje asynchronně a ukládá session ID do souboru `session.txt` pro umožnění opětovného připojení (reconnect).

4 Konfigurace serveru

Server načítá konfiguraci ze souboru `server.config`. Pokud soubor neexistuje, použijí se výchozí hodnoty.

Příklad konfigurace:

```
PORT=10000
MAX_ROOMS=16
MAX_CLIENTS=128
BIND_ADDRESS=0.0.0.0
DISCONNECT_GRACE=60
```

Parametry:

- **PORT** – TCP port, na kterém server naslouchá. Lze přepsat argumentem příkazové řádky.
- **MAX_ROOMS** – maximální počet aktivních herních místností.
- **MAX_CLIENTS** – maximální počet připojených klientů.
- **BIND_ADDRESS** – IP adresa pro bind (0.0.0.0 pro všechny rozhraní).
- **DISCONNECT_GRACE** – čas v sekundách, po který je místo v místnosti rezervováno pro odpojeného hráče (pro reconnect).

5 Textový protokol

Protokol je textový, řádkový, kódovaný v UTF-8 (ASCII kompatibilní). Zprávy jsou odděleny znakem nového řádku `\n`. Jednotlivá pole ve zprávě jsou oddělena znakem `'|'`. Každá zpráva začíná prefixem `##`.

5.1 Zprávy od klienta (Client -> Server)

- ##**JOIN**|***name*** – Požadavek na registraci hráče do lobby.
- ##**LIST**| – Požadavek na aktuální seznam místností.
- ##**CREATE**|***room_ name*** – Vytvoření nové místnosti.
- ##**JOINROOM**|***id*** – Vstup do existující místnosti dle ID.
- ##**MOVE**|***x***|***y*** – Provedení tahu na souřadnice x, y (0-2).
- ##**REPLAY**|**YES/NO** – Odpověď na nabídku nové hry.
- ##**EXIT**| – Opuštění místnosti (návrat do lobby).
- ##**QUIT**| – Úplné odpojení od serveru.
- ##**RECONNECT**|***name***|***session*** – Pokus o obnovu spojení po výpadku.
- ##**PING**| – Heartbeat zpráva od klienta (volitelné).

5.2 Zprávy od serveru (Server -> Client)

- ##**HELLO**| – Uvítací zpráva po připojení.
- ##**SESSION**|***id*** – Přidělení unikátního session ID pro budoucí reconnect.
- ##**JOINED**|***name*** – Potvrzení úspěšného přihlášení do lobby.
- ##**INFO**|***text*** – Informativní zpráva pro uživatele.
- ##**ERROR**|***text*** – Chybové hlášení (např. plná místnost, špatný tah).
- ##**PING**| – Heartbeat výzva, klient by měl odpovědět jakoukoliv aktivitou (ideálně PONG, pokud je implementován, nebo jen udržuje spojení).
- ##**ROOMS**|***count***|***id***|***name***|***state***|***players...*** – Seznam místností.
- ##**CREATED**|***id***|***name*** – Potvrzení vytvoření místnosti.
- ##**JOINROOM**|***id***|***name*** – Potvrzení vstupu do místnosti.
- ##**START**|**Opponent:*name*** – Hra začíná.
- ##**SYMBOL**|**X/O** – Přidělení herního symbolu.

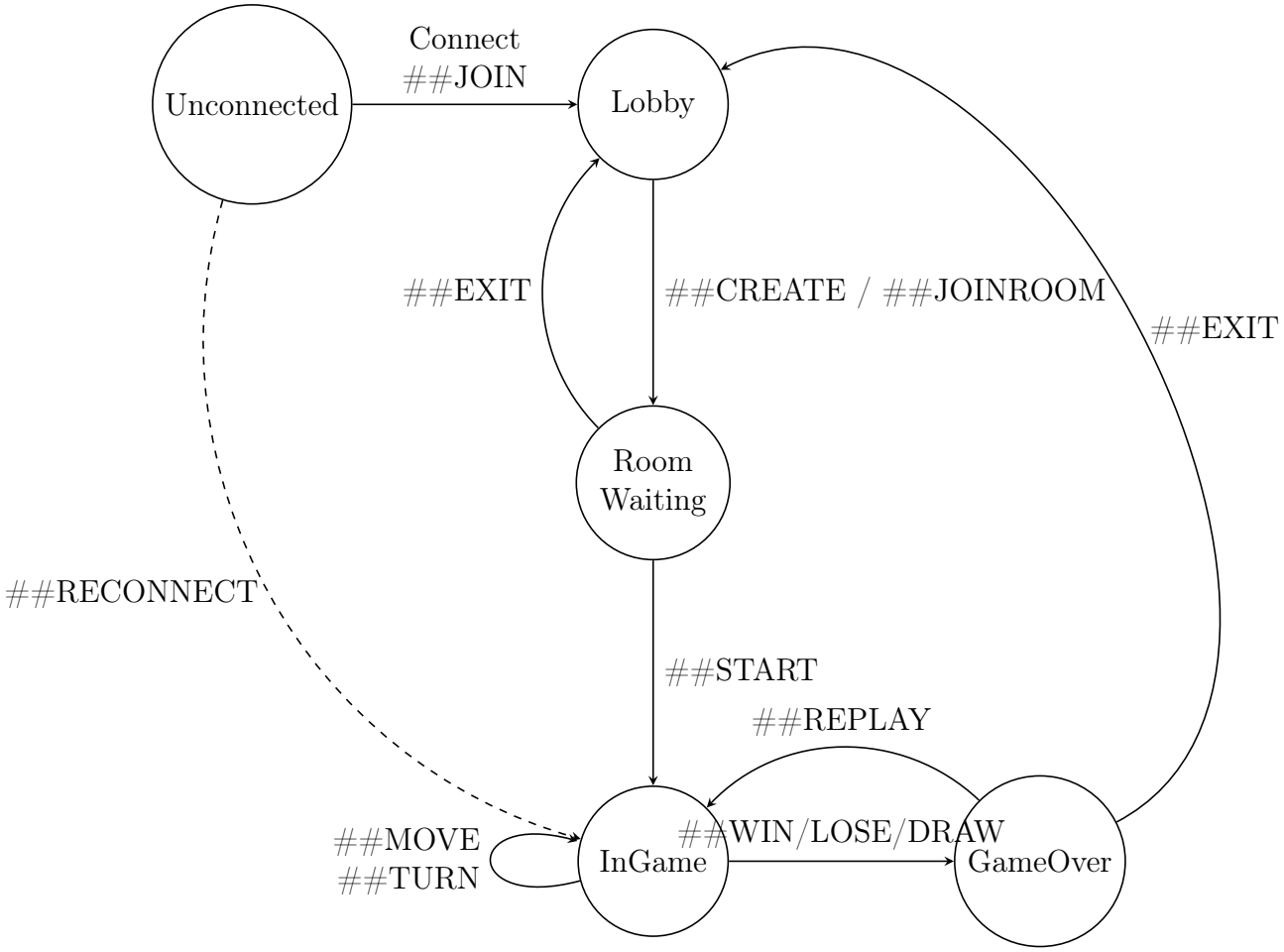
- ##**TURN**|***text*** – Výzva k provedení tahu.
- ##**MOVE**|***player***|***x***|***y*** – Informace o provedeném tahu (vlastním i soupeřově).
- ##**WIN**|***who*** – Konec hry, výhra.
- ##**LOSE**|***who*** – Konec hry, prohra.
- ##**DRAW**| – Konec hry, remíza.
- ##**RECONNECTED**| – Úspěšné obnovení stavu po reconnectu.

5.3 Validace a omezení

- Jména a názvy nesmí obsahovat znak `'|'`.
- Souřadnice tahu musí být v rozsahu 0–2.
- Session ID je generovaný řetězec.
- Server ignoruje prázdné řádky a zprávy bez prefixu `##`.

5.4 Stavový diagram komunikace

Následující diagram znázorňuje základní stavy klienta a přechody mezi nimi na základě zpráv protokolu.



Obrázek 1: Stavový diagram klientské aplikace

6 Chybové stavy a Reconnect

6.1 Ošetření chyb

Server reaguje na nevalidní stavy odesláním zprávy ##**ERROR**. Příklady:

- `##ERROR|Invalid session` – při neplatném pokusu o reconnect.
- `##ERROR|Room full` – při pokusu vstoupit do plné místnosti.
- `##ERROR|Not your turn` – při pokusu o tah mimo pořadí.

6.2 Reconnect (Obnovení spojení)

Aplikace podporuje zotavení po ztrátě spojení (např. pád wi-fi, restart klienta).

1. Po prvním přihlášení server pošle klientovi `##SESSION|<id>`.
2. Klient si toto ID uloží lokálně (do souboru).
3. Při opětovném spuštění klient automaticky zkusí poslat `##RECONNECT|name|session`.
4. Pokud server stále drží stav hráče (v paměti po dobu `DISCONNECT_GRACE`), obnoví spojení a vrátí hráče do rozehrané hry.

7 Návod k překladu a spuštění

7.1 Server (Linux)

Vyžaduje GCC a GNU Make.

1. Přechod do složky serveru:

```
cd server
```

2. Překlad:

```
make
```

Tím vznikne spustitelný soubor `build/server`.

3. Spuštění:

```
make run
```

Nebo manuálně se specifikací portu:

```
./build/server 10000
```

7.2 Klient (Windows/Linux/macOS)

Vyžaduje .NET 8.0 SDK.

1. Přechod do složky klienta:

```
cd client
```

2. Spuštění:

```
dotnet run
```

Pokud chcete specifikovat parametry nebo sestavit release verzi:

```
dotnet build -c Release
./bin/Release/net8.0/PiskvorkyClientAvalonia
```

8 Závěr

Vytvořená aplikace splňuje zadání semestrální práce. Byla ověřena funkčnost ve vícevláknovém prostředí serveru i na moderním klientu Avalonia. Protokol je robustní a umožňuje korektní synchronizaci stavu hry i po výpadku sítě.