FAKULTA
APLIKOVANÝCH VĚD
ZÁPADOČESKÉ
UNIVERZITY
V PLZNI

KIV/UPS - Úvod do Počítačových sítí

Semestrální práce

Three Tunnelers

Jméno: Štěpán Ševčík

Osobní číslo: A13B0443P

E-mail: kiwi@students.zcu.cz

Datum: 27. ledna 2017

Obsah

1 Zadání

Počítačová hra Three Tunnelers založená na původní DOSové hře Tunnelers sestávající z:

- Herní klient implementovaný v jazyce JAVA
- Herní server implementovaný v jazyce C

Zásady vypracování semestrální práce viz. CW

- 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.
- Součástí programu bude trasování komunikace, dovolující zachytit proces komunikace na úrovni aplikačního protokolu a zápis trasování do souboru.
- Každý program bude doplněn o zpracování statistických údajů (přenesený počet bytů, přenesený počet zpráv, počet navázaných spojení, počet přenosů zrušených pro chybu, doba běhu apod.).
- 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 Úvod

Three Tunnelers je síťová adaptace rozšiřující možnosti originální hry Tunnelers pro DOS. V této hře hráč přebírá kontrolu nad hrabacím tankem v podzemní oblasti a jeho cílem je zabrat tuto oblast pro sebe. V oblasti se nachází další hráči, jejichž cílem je to samé. Zabrání oblasti je možné docílit tím, že bude hráč poslední žijící nejvíce-krát z daného počtu kol.

Hlavní vlastností hry Tunnelers je, že si hráči musejí postupně získat přehled o oblasti z pohledů na půdorys. K dispozici totiž nemají pohled na celou oblast ale jen na malé části kolem tanků. Každý hráč vidí půdorysy nad všemi tanky.

3 Programátorská dokumentace

3.1 Datové struktury

Pro zjednodušení popisu funkcí systémů jsou předpokládány následující struktury u klienta i u serveru:

Mapa

Mapa představuje oblast, o kterou se bojuje. Oblast se skládá z (W * H) sekcí, takzvaných chunků, stejné velikosti N * N bloků. V oblasti je několik sekcí, které představují základny hráčů.

Chunk

Chunk je malý útvar skládající se z bloků, který reprezentuje terén oblasti. Některé chunky obsahují seskupení bloků představující hráčskou základnu, u těchto chunků je nastaven identifikátor odpovídajícího hráče.

Blok

Blok na mapě je základní jednotku mapy. Jedná se pouze o číselnou hodnotu identifikující typ bloku, možné hodnoty, kterých blok může nabývat jsou: Prázdný blok = 0, Zemina = 1, Kámen = 2, Zeď základny = 3.

Klient

Klient je struktura reprezentující klientskou aplikaci, pomocí které hráči vstupují a ovládají hru. Sdílené informace v této struktuře jsou klientské jméno a odkazy na hráče v místnosti.

Hráč

Hráč je skupina informací o účastníkovi boje o oblast. Tato struktura je oddělena od Klienta pro jednodušší realizaci více hráčů v jedné spuštěné aplikaci. Struktura hráče obsahuje odkaz na klienta, kterému náleží, ovládání, což je struktura uchovávající stav stisknutých vstupů, a číslo barvy tohoto hráče.

Směr

Směr je výčet identifikující jeden z 8 světových směrů reprezentovaný celočíselnou hodnotou. Možné hodnoty jsou: 0 = Žádný směr, 1 = Sever, 2 = Severovýchod, 3 = Východ, 4 = Jihovýchod, 5 = Jih, 6 = Jihozápad, 7 = Západ, 8 = Severozápad.

Tank

Tank, rozlehlý až 7*7 bloků, je struktura definující hráčem ovládaný prvek v oblasti. Tank je definován hráčem, kterému přísluší, svojí polohou na mapě a směrem, kam je natočen. Dále stav tanku definují jeho aktuální počet bodů zásahů, které zbývají do jeho zničení, zbývající energií a časem zbývajícím do možností vystřelení dalšího projektilu.

Projektil

Projektil zabírající oblast pouze 3 bloky je stejně jako Tank definován hráčem, kterému náleží, svojí polohou na mapě a směrem kam je natočen a kam letí.

Herní místnost

Herní místnost je kontejner nesoucí informace aktuálním stavu místnosti, informace klientech, kteří se účastní dění v místnosti a identifikátor klienta, který má místnost "na starost". Dále místnost nese informace o hráčích účastnících se hry a v určitém stavu i o bitevní zóně. Stavy místností jsou reprezentovány celočíselnými hodnotami 0 = Čeká, 1 = Lobby, 2 = Bitva začíná, 3 = Bitva, 4 = Souhrny.

Bitevní zóna

Bitevní zóna je střed dění a zde se uchovávají informace o mapě definující oblast, o tancích v oblasti a o projektilech létajících oblastí a případně další doplňující informace o průběhu hry.

3.2 Herní server

Server je sestaven z co nejmenších funkčních celků, které se dají seskupit do následujících kategorií:

1. **Jádro** Zde se vyskytuje převážná většina výkonného kódu a páteřní struktury, které vše takzvaně drží pohromadě
2. **Hra, model a mapa** Do této kategorie patří struktury zachycující stavy různých prvků hry a funkce usnadňující práci s nimi
3. **Síť** V této části je funkcionalita obstarávající síťovou komunikaci s klienty.

3.2.1 Spouštění serveru

Po spuštění programu proběhne načtení parametrů a inicializace globálních proměnných, jako jsou směrové proměnné nebo tvarové definice. Dále se zpracují a ověří zadané běhové parametry portu pro naslouchání a počtu místností. Podle požadovaného počtu místností se alokuje paměť pro připojení, klienty a herní místnosti. Dalším krokem v pořadí je běh programu, ve kterém jsou inicializovány páteřní struktury jádra a síťového adaptéru a spuštěna vlákna pro aktualizaci jádra, pro síťový adaptér a pro rozhraní příkazové řádky. Po dokončení běhu všech vláken jsou prostředky uvolněny a běhové statistiky vypsány.

3.2.2 Běh síťového rozhraní

Vlákno síťového rozhraní hlídá serverový socket a sockety připojených klientů pomocí funkce select(). Pokud nastane aktivita na serverovém socketu, vlákno přijme spojení a přiřadí jej struktuře spojení. Pokud aktivita nastane na socketu, ověří že se nejedná o chybu spojení. Pokud nastane chyba, spojení uzavře a označí jej jako neplatné. V opačném případě uloží příchozí znaky do vyrovnávací paměti, přeformátuje je do struktury představující příkaz a ten předá jádru do fronty příkazů pro zpracování.

3.2.3 Běh jádra

Běh jádra se skládá z pravidelných "tiků", což jsou obnovovací cykly. V každém cyklu je prováděno několik kroků, které jsou obaleny časomírou. Díky tomu je možné ovlivňovat dobu, jakou bude vlákno spát než spustí další obnovovací cyklus.

V prvním kroku jsou kontrolováni připojení klienti - pokud je některý z klientů označen jako odpojený, je spuštěna uklízecí procedura. V případě, že je klient v herní místnosti, vyšle

ostatním klientům místnosti aktualizaci stavu. Pokud v místnosti nezůstal žádný další klient připojený, místnost je uklizena a připravena pro další použití. Pokud klient nebyl v místnosti, informace o jeho připojení jsou odstraněny, čímž je vytvořeno místo pro následující klienty.

V dalším kroku jsou zpracovány příkazy klientů. První příkaz klienta vždy musí být autorizace, dále viz protokol. Po úspěšné autorizaci jsou všechny příkazy zpracovávány stejným způsobem - jejich obsah je předán skenovací struktuře s ukazatelem na do zpracovávaného textu, podle čísla typu příkazu je vybrán odkaz na funkci zpracování a tato funkce je spuštěna. Její výsledek představuje úspěch vykonání funkce. Pokud jádro zaznamená příliš mnoho neúspěšných příkazů, klienta odpojí.

V posledním kroku obnovovací smyčky je prováděna aktualizace jednotlivých místností. Aktualizace místností jsou spouštěny pouze pro stavy "Příprava bitvy", kdy jsou kontrolováni klienti - spuštění totiž nemůže proběhnout dokud nejsou všichni připraveni a pokud by některý z klientů v tomto stavu nečekaně ukončil svoji činnost, ostatní klienti by zůstali čekat na ostatní. Další aktualizovaný stav hry je "Bitva", kde jsou zpracovány uživatelské vstupy a výsledky jsou synchronizovány s klienty.

3.3 Herní klient

Klient je obdobně typově rozdělen do částí, nebo do takzvaných vrstev. Nejvyšší vrstvou je pohled, což je uživatelské rozhraní. O vrstvu níže se nachází jádro, které se dále dělí na model a funkční logiku. Pohled s jádrem a jádro s pohledem komunikují prostřednictvím rozhraní. Pod jádrem se nachází rozhraní pro komunikaci se sítí, která je přímo využívána jádrem. Vrstva síťového rozhraní je obdobně implementována jako na serveru - vláknem, které čeká na zprávy od serveru, formátuje je do příkazů a předává je pro zpracování jádru.

Pohledová vrstva je postavená na platformě JavaFX, která nabízí bohatou množinu komponent. V této aplikaci pohledová vrstva také slouží jako spouštěč jádra.

Přepínání pohledů má na starost třída `TunnelersStage`, která přímo implementuje pohledové rozhraní jádra. Jednotlivé pohledy jsou potom realizovány jako samostatné třídy rozšiřující třídu `ATunnelersScene`. Scény jsou složeny z kořene `AnchorPane` komponenty, která umožňuje "ukotvit"pod-komponenty na libovolné místo. V této komponentě jsou potom umístěny komponenty `Canvas`, do které je vykreslován obsah, `FlashArea`, pomocí které se zobrazují upozornění a ve většině případů i samotný obsah, jež je složený z dalších pod-komponent.

4 Uživatelská dokumentace

4.1 Server

Pro vytvoření spustitelného souboru serveru je třeba zdrojové soubory[1] přeložit spuštěním příkazu `make` z adresáře se zdrojovými soubory. Zkompilovaný soubor se spouští s parametry: `TT_Server <port> <počet místností>` Za běhu serveru lze do textového rozhraní zadávat následující příkazy:

- **clients** - výpis klientů
- **connections** - výpis existujících spojení
- **rooms** - výpis místností
- **status** - aktuální statistiky
- **exit** - ukončení serveru

4.2 Klient

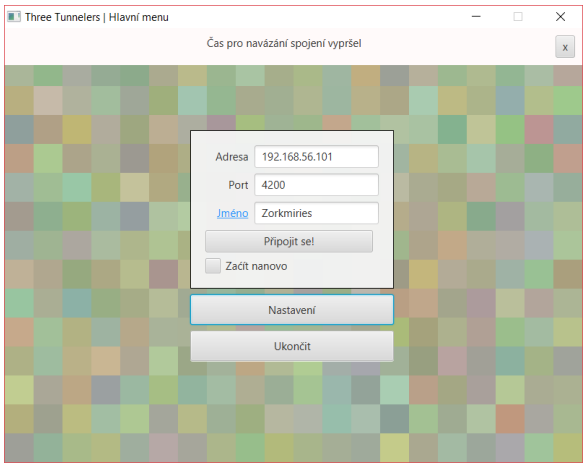
K zkompilování zdrojových souborů[2] klienta je použit nástroj Ant. Získání spustitelného souboru lze docílit těmito příkazy: `ant compile` a `ant jar`. Klienta lze spustit "otevřením"souboru .jar nebo spuštěním z konzole příkazem `java -jar ThreeTunnelers.jar`. Pokud je klient spuštěn z konzole, lze vidět několik vývojářských výpisů.

[1] https://github.com/Thoronir42/ThreeTunnelers_Server/
[2] https://github.com/Thoronir42/ThreeTunnelers/

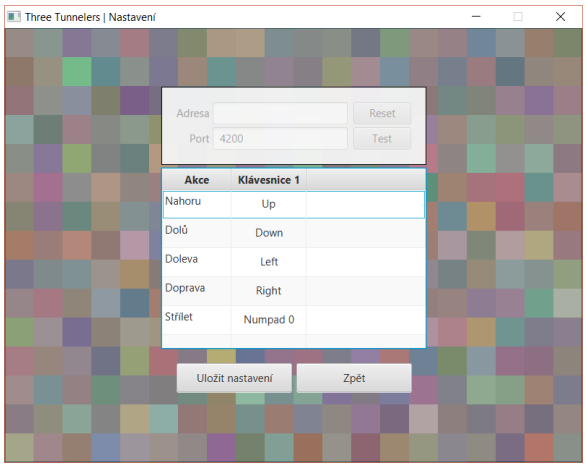
4.3 Průvodce uživatelským rozhraním

Pohledy nevyžadující spojení se serverem

Po spuštění aplikace se uživatel ocitá v hlavním menu viz Obrázek 1. Uživatel má možnost možnost z hlavního menu přejít do pohledu nastavení viz Obrázek 1. V nastavení může uživatel změnit konfiguraci kláves, kterými bude ovládat svůj tank. Později bude možno ukládat i výchozí konfiguraci serveru.



Obrázek 1: Hlavní menu

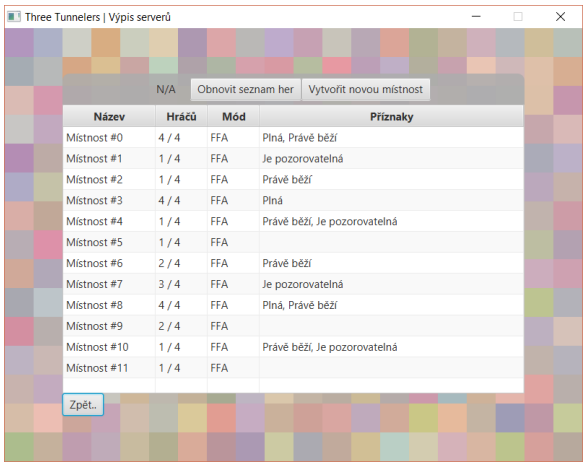


Obrázek 2: Nastavení

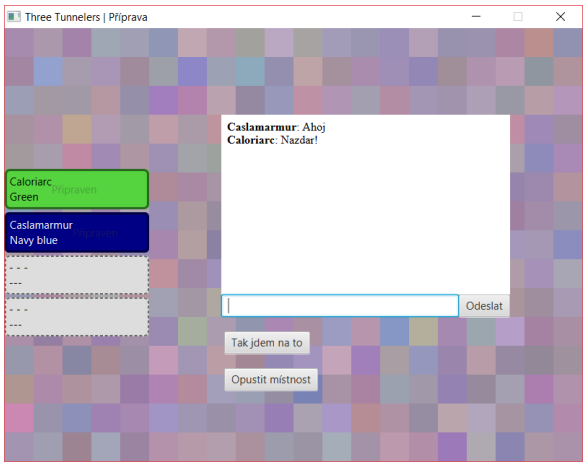
Z hlavního menu se může také klient připojit na server zadáním adresy a portu cíleného serveru. Před stiskem tlačítka má klient možnost zvolit si vlastní jméno, které je jinak pseudonáhodně generováno. Pokud při navazování spojení se serverem nastane chyba, uživatel je o tom informován v liště na vrcholu okna viz Obrázek 1. Po úspěšném připojení na server se uživateli zobrazí výpis místností viz Obrázek 3, odkud může dvojklikem na řádek s požadovanou místností vstoupit do dané místnosti nebo příslušným tlačítkem vytvořit novou místnosti.

Pohledy vyžadující spojení se serverem

Při úspěšném vytvoření místnosti nebo připojení se do místnosti je přepnut pohled na lobby viz Obrázek 4. Zde je pro členy místnosti k dispozici chat a tlačítko, kterým signalizují, že jsou připraveni na hru.

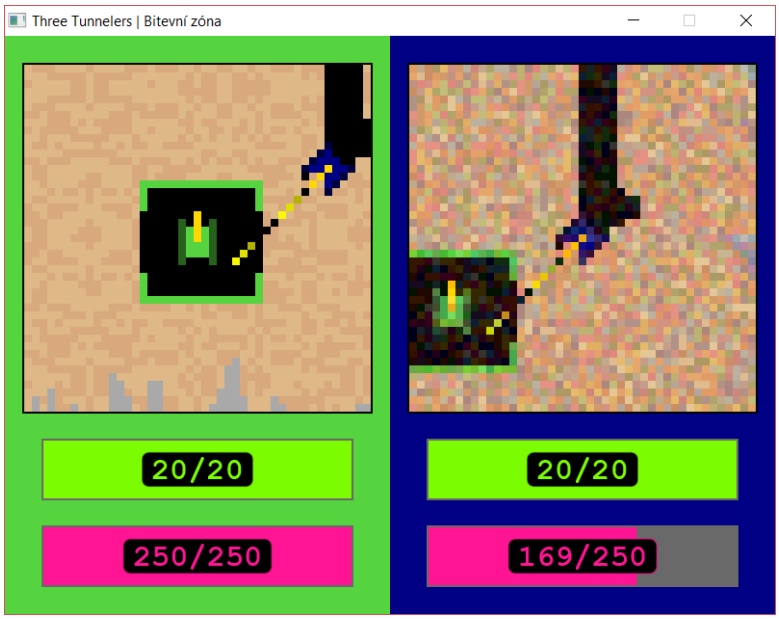


Obrázek 3: Výpis místností



Obrázek 4: Lobby

V momentě, kdy jsou všichni účastníci místnosti připravení hrát a server připraví bitevní zónu, členům místnosti se přepne pohled na bitevní zónu, viz Obrázek 5. Zde každý hráč ovládá svůj tank pomocí nastavených kláves. Každý hráč má na obrazovce vyhrazenou část, kde má "kukátko"do bitevní zóny a indikátory zbývajících bodů zásahů a zbývající energie. Po zničení jednoho z tanků je bitva ukončena a členové místnosti jsou zpět v Lobby.



Obrázek 5: Bitevní zóna

5 Závěr

Vypracované aplikace splňují zadání ačkoliv neobsahují některé neklíčové funkce z důvodu tísnivého času.

Díky této semestrální práci jsem si vyzkoušel, co obnáší proces návrhu a implementace síťové aplikace skládající se ze serveru z klienta a aplikačního protokolu, pomocí kterého tyto aplikační části komunikují. Procvičil jsem si vytváření uživatelského rozhraní a připomenul si důležitost očividné reakce na uživatelské vstupy.

Práci mi značně zjednodušily jednotkové testy, které jsem implementoval na straně klienta pomocí testovacího frameworku JUnit a na straně serveru, které jsou realizovány debugovacími výpisy. Obě aplikace nejsou ani zdaleka otestovány celé ale klíčové prvky lze testy ověřit.

Hra má veliký potenciál pro rozšířitelnost díky návrhu, který klade důraz na všeobecnost. Mezi jednodušší rozšíření patří navýšení maximálního počtu hráčů v místnosti, podpory více hráčů na jednom počítači nebo i přidání možnosti různých herních módů.

6 Přílohy

6.1 Aplikační protokol TTP

Protokol TTP (Three Tunnelers Protocol) definuje rozhraní mezi dvěma účastníky komunikace po síti. Komunikace je asynchronní, což znamená že účastníci nemají stejné role, zde mají mezi sebou vztah Klient-Server. Označení Klient označuje roli účastníka komunikace, klient potom označuje konkrétní reprezentaci připojeného uživatele. Aplikační protokol TTP je realizován nad transportním protokolem TCP. Zprávy nemusí mít jednotnou délku ale vždy musí odpovídat následujícímu formátu:

`<hlavička><tělo><LF>`

- **hlavička** zprávy se skládá ze 4 hexadecimálních znaků, které představují číselný typ zprávy
- **tělo** zprávy se odvíjí podle typu zprávy, čísla jsou přenášena v hexadecimální podobě s nevýznamovými nulami na začátku pro zachování jednotné délky bloků
- **LF** představuje znak "Line Feed"a je vyžadován na konci každé zprávy

Protokol je bezestavový, což znamená, že žádná ze zpráv nevyžaduje okamžitou odpověď. Většina druhů zpráv však významově předpokládá návaznost jiných zpráv. Komunikace tedy probíhá stylem Požadavek (P) - Odpověď (O) s možností vyvolání delších komunikačních řetězů některými zprávami a s možností signalizací změny stavu (S).

Definice zpráv protokolu podle hlaviček určují formát těla. V definici je možné se setkat s následujícími symboly:

X = Znak představující hexadecimální hodnotu

text = Neformátovaný text proměnlivé délky. Obvykle se tento blok vyskytuje na konci těla a jeho délku lze určit jako počet znaků od posledního bloku do konce zprávy.

Následuje výčet definic skládající se z čísla a názvu typu zprávy na prvním řádku a popisu těla zprávy v kontextu požadavku (P), odpovědi (O) nebo signálu (S).

Organizační zprávy mezi klientem a serverem

0x01 Představení klienta

P: (XX)(text) [indikátor znovupřipojení, tajný kód], pokud indikátor znovupřipojení nabývá nulové hodnoty, zpráva nemusí obsahovat blok tajný kód

O: (XX)(text) [indikátor znovupřipojení, tajný kód], Server vždy posílá oba bloky.

Na požadavek Představení klienta Server odpovídá potvrzením Představení (0x01) nebo Odpojením klienta (0x02).

0x02 Odpojení

S: (text)[důvod odpojení], po přijetí zprávy Klientem nebo Serverem lze očekávat ukončení spojení.

0x03 Nastavení jména

P: (text) [Jméno klienta], Klient posílá jméno, kterým chce být reprezentovaný

O: (text) [Jméno klienta], Server posílá ošetřené jméno, které klient skutečně získal

Na požadavek lze očekávat odpověď Nastavení jména (0x03).

0x04 Marco

P: (XX)(XXXXXXXXXXXXXXXX) [počet předcházejících zpráv - nulová hodnota, časový otisk odesílající strany]

O: (XX)(XXXXXXXXXXXXXXXX)(XXXXXXXXXXXXXXXX) [počet předcházejících zpráv - nenulová hodnota, časový otisk odesílající strany, časový otisk přijímající strany]

Na požadavek lze očekávat odpověď typu Marco (0x04) nebo odpověď typu Polo (0x05)

0x05 Polo

O: (XXXXXXXXXXXXXXXX) [Časový otisk přijímající strany]

Zprávy spravující místnosti

0x10 Výpis místností

P: -

O: (XX)(M)*n [počet místností ve zprávě (N), N popisů místnosti]

Popis místnosti M je tohoto formátu: (XXXX)(XX)(XX)(XX)(XX) [id místnosti, maximální počet hráčů, aktuální počet hráčů, stav místnosti, herní mód]

Požadavek je vždy prázdný a lze na něj očekávat odpověď typu Výpis místností (0x10)

0x12 Vytvořit místnost

P: -

Požadavek je vždy prázdný a lze na něj očekávat odpověď typu Vstoupit do místnosti (0x13) nebo Opuštění místnosti (0x14)

0x13 Vstoupit do místnosti

P: (XX) [Číslo místnosti]

O: (XX)(XX)(XX) [Číslo místnosti, číslo lokálního klienta, číslo klienta leadera místnosti] Na požadavek lze očekávat odpověď typu Vstoupit do místnosti (0x13) nebo Opustit místnost (0x14).

0x14 Opustit místnost

P: -

S: -

Mimo-herní zprávy v rámci místnost

0x20 Zpráva - prostý text

P: (text) [tělo zprávy]

S: (XX)(text) [číslo klienta místnosti, tělo zprávy] Na požadavek lze očekávat signál typu Zpráva - prostý text (0x20).

0x21 Systémová zpráva

S: (text) [tělo zprávy]

0x22 Zpráva - vzdálený příkaz

P: (text) [Příkaz ovládání serveru jako z příkazové řádky]

Na požadavek lze očekávat signál typu Systémová zpráva (0x21)

0x40 Synchronizace fáze

S: (XX) [stav místnosti]

0x41 Stav připravení

P: (XX) [signalizace připravenosti (nulová / nenulová hodnota)]

S: (XX)(XX) [signalizace připravenosti, číslo klienta v místnosti]

0x42 Informace o klientovi

S: (XX)(text)[číslo klienta v místnosti, jméno]

0x43 Stav klienta

S: (XX)(XX)(XXXX) [číslo klienta v místnosti, stav klienta, doba odezvy v milisekundách]

Klient NESMÍ poslat tuto zprávu.

0x44 Odebrání klienta

S: (XX)(text) [identifikátor klienta v místnosti, důvod odebrání klienta]

0x45 Předat vedení klientovi

P: (XX) [číslo klienta v místnosti

S: (XX) [číslo klienta v místnosti]

Na požadavek lze očekávat signál typu Předat vedení klientovi.

Herní zprávy místnosti

0x50 Přidání hráče

S: (XX)(XX)(XX)(XX) [slot v místnosti, barva, číslo klienta v místnosti, číslo hráče klienta]

0x51 Odebrání hráče

S: (XX) [slot v místnosti]

0x52 Přesunutí hráče mezi sloty

S: (XX)(XX) [slot odkud, slot kam]

0x53 Nastavení barvy hráče

S: (XX)(XX) [slot hráče, barva]

0x60 Specifikace mapy

S: (XX)(XX)(XX) [velikost chunku, šířka v chuncích, výška v chuncích]

0x61 Specifikace základnových bloků na mapě

S: (XX)((XX)(XX)(XX)) * N [N - počet specifikací základen, N specifikací: (X chunku, Y chunku, číslo hráče v mínstnosti)]

0x62 Chunková data

S: (XX)(XX)(XX)(text) [X chunku, Y chunku, checksum chunku, <domluvená velikost chunku> * X]

0x63 Seznam změn bloků

S: (XX)(změna) * N [Počet změn bloků N, N změn bloků]

Změna bloku je následujícího formátu: (XXXX)(XXXX)(X) [X souřadnice bloku, Y souřadnice bloku, nový blok]

0x70 Nastavení ovládání

P: (XX)(XX) [číslo hráče v místnosti, bitová mapa stisklých kláves]

S: (XX)(XX) [číslo hráče v místnosti, bitová mapa stisklých kláves]

Na požadavek lze očekávat signál typu Nastavení ovládání 0x70

0x74 Informace o tanku

S: (XX)(XX)(XXXX)(XXXX)(XX)(XX)(XX) [číslo hráče v místnosti, stav tanku, pozice X, pozice Y, směr, body zásahu, energie tanku]

0x7A Vytvoření projektilu

S: (XX)(XX)(XXXX)(XXXX)(XX) [identifikátor projektilu, číslo hráče, X na mapě, Y na mapě, počáteční směr]

0x7B Odebrání projektilu

S: (XX) [identifikátor projektilu]