Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Katedra informatiky a výpočetní techniky

Bakalářská práce

Automatizovaný turnajový systém pro platformu Pogamut

Plzeň, 2012 Milan Bárta

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 10. 5. 2012

Milan Bárta

Automated Tournament System for Pogamut

This thesis deals with tournament system for Pogamut. Pogamut is a Java middleware that enables control of virtual agents in multiple environments provided by game engines. It is mainly used in Unreal Tournament 2004 (UT2004). This thesis presents program to create tournaments in UT2004. It allows the user to set the game type, time and score limit as well as a map. Users can load JAR files containing Pogamut bots, set parameters to each bot and create a match list specifying which bots shall compete against each other. When tournament is executed, program automatically launches a UT2004 server with all configured parameters and subsequently loads bots into the game. After all tournament matches are finished, program creates result table and visualizations in which users can analyze progress of each match with details such as which bot was killed by which bot, by whom weapons were picked and how many health and armor bots had at any given point in time. The program developed in the frame of this thesis was successfully used as a support tool in a course of the Department of Computer Science and Engineering of the University of West Bohemia in Pilsen.

Poděkování

Tímto bych chtěl poděkovat vedoucímu bakalářské práce Ing. Ondřeji Rohlíkovi, Ph.D za cenné rady, připomínky a vedení práce.

1 Obsah

1 Úvod

Tato bakalářská práce pojednává o turnajovém systému pro platformu Pogamut ve hře Unreal Tournament 2004. Co je Pogamut? Pogamut je software naprogramovaný v jazyce Java, který umožňuje ovládání virtuálních agentů v různorodých prostředích v herních enginech. Primárně se používá hlavně pro hru Unreal Tournament 2004.

Cílem práce je vytvořit turnajový systém na této platformě. Uživatel si při spuštění programu nastaví, jací boti mají daný turnaj hrát. U každého si nastaví např. jméno týmu, ke kterému bude daný bot patřit, počet botů nebo jejich skill. Dále si nastaví parametry turnaje, tj. o jaký mód se bude jednat, na jaké mapě se bude hrát, nastaví si časový limit a limit na skóre a také si nastaví seznam zápasů, tj. kdo bude hrát proti komu.

Po spuštění turnaje se již uživatel nemusí o nic starat. Program sám před každým zápasem spustí server hry a podle seznamu zápasů připojí jednotlivé boty, které zrovna mají daný zápas hrát. Při běhu zápasu se také vypisuje do konzole jeho průběh (např. kdo zrovna sebral vlajku, kdo koho zabil apod.). Po skončení zápasu se ukončí jak server tak všechny spuštění boti a spustí se další zápas, pokud v seznamu zápasů ještě nějaký je.

Po skončení celého turnaje se vygenerují výsledky a vytvoří se vizualizace jednotlivých zápasů. U každé vizualizace je vidět, kolik životů a štítů měli jednotliví boti, kdo sebral jaké zbraně, kdo koho zabil a změny stavů u vlajky, jestliže se jedná o CTF mód (Capture the Flag). Díky tomu uživatel rychle vidí, jak jednotlivý zápas probíhal.

Využití tohoto programu je hlavně pro předmět KIV/ISW, kde studenti tvoří své vlastní boty a program mohou využít pro testování, jak moc jsou jejich boti dobří a také pro vyučujícího, který na konci školního roku otestuje boty tím, že vytvoří turnaj, kde hraje každý s každým. Tím se zjistí, který student vytvořil nejlepšího bota. Další využití je poskytnout tento program komunitě okolo Pogamutu.

2 Inteligentní agenti

Informace v této kapitole byly čerpány z [1].

2.1 Co je to agent?

Agent jako pojem pro oblast multiagentových systémů je inspirován funkcí zástupce, který je určen pro vykonávání určitých úkolů. Měl by být do co největší míry samostatný – aby operátor nebo správce daného systému prováděl do běhu agenta pouze minimální zásahy. Agent může být softwarové nebo hardwarové zařízení, které operuje pouze ve vyměřeném prostředí.

Hlavní vlastnosti multiagentových systémů:

- autonomie prvků
- decentralizace řízení v systému
- robustnost
- adaptibilita na změny v prostředí

Ve srovnání agenta s objektem, který je nejpropracovanější abstrakcí reality, je agent abstraktnější a má některé důležité vlastnosti navíc:

- autonomii (nezávislost na ostatních agentech)
- existenci v prostředí
- kontinuitu senzorického a aktuálního propojení s prostředím

Srovnání vlastností objektů a agentů shrnuje následující tabulka Tab. 1.

Tab. 1 Porovnání vlastností objektů a agentů

Objekt	Agent
Zapouzdřuje proměnné a metody	Zapouzdřuje chování
Nutná synchronizace vláken	Agenti jsou navzájem nezávislí
Uspořádání objektů prostřednictvím dědičnosti a kompozice	Uspořádání agentů v organizacích
Komunikuje voláním metod (posíláním zpráv)	Komunikuje prostřednictvím vyšších komunikačních jazyků
Prostředí (kromě kompilačního) nehraje žádnou roli	Významnou roli sehrává prostředí

V dalších rovinách lze přidávat agentům dodatečné vrstvy chování, jako je učení, mobilita kódu apod.

2.2 Architektury agentů

Z hlediska složitosti organizace vnitřních komponent agentů lze agenty rozdělit do následujících skupin:

- reaktivní
- deliberativní
- sociální
- hybridní

2.2.1 Reaktivní agent

Tento typ agenta má nejjednodušší vnitřní architekturu. Je tvořen několika paralelními procesy, které jsou spouštěny na základě podnětu z prostředí nebo agentova vnitřního stavu, případně kombinací obojího. Agent nezná prostředí, ve kterém se vyskytuje. Má pouze vyrovnávací paměť, ve které je uložen stav agenta. Nemá možnost plánování.

2.2.2 Deliberativní agent

Deliberativní (uvažující) agent si na rozdíl od reaktivního agenta uchovává symbolickou reprezentaci prostředí a vnitřních stavů, podle nichž si vytváří plány pro dosažení svých cílů. Jednotlivé cíle mají přiřazenou prioritu, která poté slouží pro výběr adekvátního plánu.

2.2.3 Sociální agent

Komunikuje s jinými agenty ve vyšším komunikačním jazyce. Rozšiřují své poznatky o prostředí o modely ostatních agentů. Jedná se především o jména, adresy a specifikace jejich schopností pro případnou kooperaci.

Na rozdíl od multiagentových systémů s centrálním agentem, kterému jsou podřízeni všichni agenti řešící stejný úkol, který je rozdělen na menší podproblémy a každý agent řeší jeho část, musí mít sociální agenti více informací o jiných agentech, se kterými přicházejí do kontaktu. Koalice mezi agenty se vytváří a zanikají podle potřeby. Kromě informací o ostatních agentech si agenti ukládají informace o předchozích koalicích. Každý agent má omezené zdroje, schopnosti či kompetence. Vytváření koalic jim pomáhá tyto problémy překlenout.

2.2.4 Hybridní agent

Architektura hybridního agenta kombinuje některé nebo všechny předchozí architektury do jednoho celku.

2.3 Základní kategorie interakcí mezi agenty

2.3.1 Distribuovaná umělá inteligence

Rozdělení úloh mezi více výpočetních procesů, které řeší podproblémy z dané oblasti podle jejich vhodnosti k danému podproblému a vzájemně kooperují pro vyřešení globálního problému. Využívají se zde centralizované i decentralizované řídící mechanizmy. Díky rozdělení úkolu na menší části je nutné, aby se agenti zúčastnili na řešení problému a podřídili se globálnímu cíli. Této vlastnosti se říká předpoklad benevolence. Koordinaci společenství, ve kterém je splněna uvedená podmínka, se říká kolaborace.

2.3.2 Decentralizovaná umělá inteligence

U decentralizované umělé inteligence agenti slouží k plnění individuálních cílů a jsou spojeny s ostatními agenty mnohem volnějšími vazbami. Nedostatek zdrojů a schopností vede agenty vytvářet kooperativní svazky a koalice. Do popředí zde vstupuje problém koordinace multiagentového systému, protože zde neexistuje žádný centrální řídící mechanismus.

2.3.3 Koordinace a kooperace

Koordinace je proces probíhající v multiagentovém společenství, kterým se dosahuje takové propojení jednotlivých komponent v systému, který umožňuje řešení problému dosažením dosažení decentralizace vykonávaných úkolů a někdy i řídícího procesu. Podle toho lze koordinační procesy rozdělit na ty s centralizovaným a na ty s decentralizovaným řízením. Nejčastějším případem je použití protokolů, které stanovují rámec chování jednotlivých agentů.

Kooperace je řízení forma koordinace s účelovým uspořádáním agentů ve skupině s cílem dosáhnout společného řešení problému. Skupina může být řízena jak centrálně tak decentralizovaně s použitím autonomních prvků. Každý z prvků se řídí svou funkcí užitečnosti, která následně ovlivňuje chování jednotlivých agentů. Každý agent má předem danou roli, kterou musí plnit a má předem dány vztahy s ostatními agenty pro dosažení globálního cíle. Agenti mají tzv. protokol pro komunikaci. Protokol se, jak již z názvu vyplývá, používá pro efektivní komunikaci mezi agenty. Jedná se o přesně stanovené sledy zpráv, které určují, jak lze odpovědět na přijatou zprávu. Agenti mohou komunikovat i bez tohoto protokolu. Jedná se o sofistikovanější, ale také mnohem složitější způsob komunikace.

2.4 Koordinace

Koordinace se podle H. Minzberga rozděluje do 3. Základních kategorií:

Vzájemná dohoda je koordinační proces, kdy máme skupinu o minimálně 2 agentech, kde agenti vzájemnou komunikací vytvářejí dohodu o využití společných globálních zdrojů k dosažení cíle. Všichni agenti jsou si rovni a žádný z nich nemá centralizovanou pozici vůči ostatním.

Přímý dozor je koordinační proces, kde existuje jeden agent, který obstarává centrální řízení procesů ostatních agentů. Na základě zpětné vazby tento agent zasahuje do ostatních procesů za účelem korekce jejich deformovanosti nebo neefektivnosti. Zároveň vykonává kontrolu, a jestliže se nějaký agent „fláká“, tento agent je penalizován. Tento případ lze zobecnit tím, že nebude 1 řídící agent a zbytek si bude roven, ale bude celá hierarchie nadřízenosti a podřízenosti agentů, stejný případ jako je v mnoha firmách a podnicích.

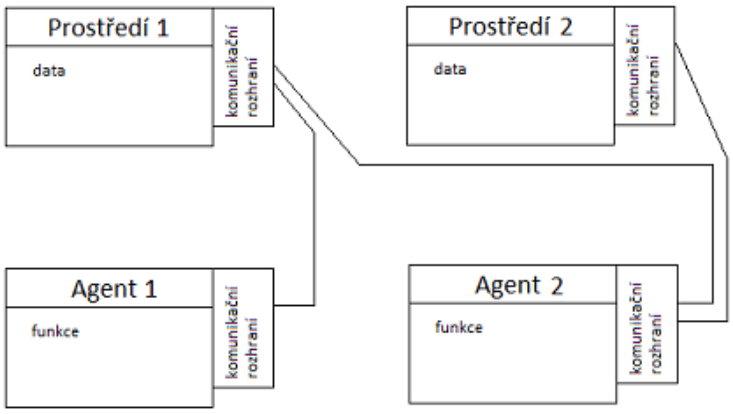
Standardizace je koordinační proces, kde se koordinuje vždy celá skupina agentů na základě pravidel chování. Tyto pravidla jsou předem dány nějakou centrální autoritou. Standardizací chápeme normy stanovené pro uspořádání chování velkého počtu komponent s volnými vztahy, dostanou-li se do vzájemného kontaktu ve specifických situacích.

2.4.1 Reaktivní komunikace

Protokol pro reaktivní komunikaci definuje 3 základní typy:

- značky s časovou platností
- funkci (schopnost) vytváření značek
- funkci (schopnost) interpretace značek

Prostředí si lze představit jako datové struktury, ke kterým mají přístup agenti, kteří jsou paralelně běžícími funkcemi zpracovávajícími data z prostředí (viz Obr. 1).



Obr. 1 Schéma reaktivní komunikace v multiagentovém systému

Komunikace mezi agentem a prostředím je realizována pomocí zpráv. Velkou výhodou je možnost odebrání nebo přidání agentů bez nutnosti změny celého systému. To samé platí i o prostředí.

2.4.2 Aukce

Základní myšlenkou je alokace zdrojů na základě ohodnocení nabídkou a poptávkou. Cílem aukce je nalézt rovnovážnou cenu za dané zdroje právě působením sil nabídky a poptávky. Máme několik typů aukcí:

Anglická aukce

Pravidla aukce umožňují nasadit jakoukoli cenu (pokud není stanovena minimální cena) a všichni agenti znají nabídky všech ostatních agentů. Každou aukci moderuje aukcionář. Aukci vyhrává agent s nejvyšší nabídkou, jestliže již nikdo další nepřihazuje. Strategie je založena na vnitřní hodnotě zdroje (kolik je agent schopný investovat) a také na možnostech ostatních agentů. Strategie spočívá ve zvyšování nabídky o minimální částky až do dosažení hodnoty zdroje. Agent je také ovlivňován historií nabídek v dané aukci.

Holandská aukce

V této aukci aukcionář vyvolá cenu zdroje na nějaké vysoké výši a postupně jí snižuje, dokud nějaký agent nenabídne aktuální cenu danou aukcionářem. Strategie agentů je založena na vnitřní ceně zdroje. Neexistuje žádná dominantní strategie.

Aukce „první nabídka s nejvyšší cenou platí“

U tohoto typu aukce agenti neznají nabídky ostatních agentů a svou nabídkou mohou udělat pouze jednou. Vyhrává agent s nejvyšší cenou nabídky. Strategie je opět založena na vnitřní ceně zdroje s případným odhadem nabídky ostatních agentů. Pro nedostatek informací o ostatních agentech lze špatně identifikovat vítěznou strategii.

Vickreyova aukce

Tento typ je podobně jako předchozí založen na jednom kole nabídek a nabídka s nejvyšší cenou vyhrává. Odlišnost u tohoto typu aukce je však ten, že zdroj je nabídnut vítěznému agentovi za cenu, kterou nabízel agent s druhou nejvyšší nabídkou. Smyslem je přinutit agenty přiznat jejich vnitřní hodnotu alokovaného zdroje. Agent, který chce vyhrát, nesmí cenu zdroje ani přecenit, ani podcenit.

Oboustranná aukce

Tento typ je založen na aktivní účasti jak nabídky (prodávající) tak poptávky (kupující). Obě strany uvádějí cenu zdroje, za kterou chtějí daný zdroj prodat – co nejvyšší cena, nebo koupit – co nejnižší cena. Cílem aukcionářů je najít co největší objem zdrojů, u nichž cena poptávky převyšuje cenu nabídky.

2.4.3 Normy chování agentů

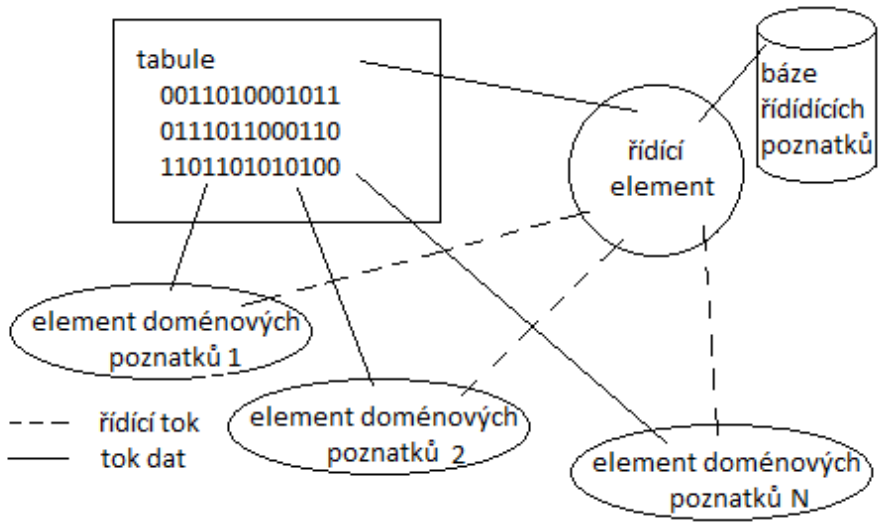
Uspořádání vztahů ve skupině agentů může být důsledkem respektování zásad nebo pravidel chování, která jsou závazná pro každého agenta. Standardní normy pro chování agentů jsou tvořeny ve snaze zabezpečit koordinaci velkého množství agentů. Takové uspořádání by mělo ochraňovat jednotlivce ve skupině, kteří tyto normy dodržují. Nelze však s určitostí říct, jestli se těmito standardizacemi pro koordinaci agentů má smysl zabývat.

2.5 Kooperace

U koordinace je velice obtížné dosáhnout vyššího stupně racionality u celé skupiny, protože zde nejsou specifikovány role, vazby a vzájemné závislosti mezi agenty. Projevuje se to zejména v případě, když nemá jeden agent dostatek zdrojů, schopností nebo kompetencí na vyřešení problému. V takových případech je nutné celý problém rozdělit na menší podproblémy. Tyto podproblémy jsou následně předány k řešení jednotlivým agentům, kteří navzájem kooperují, aby poskládali tyto menší částí do jednoho celku. Jednotlivé podproblémy mohou být provázány různými vazbami. V takových případech je vhodné přiřadit agentům různé role a koordinovat proces řešení problému úplně nebo do jisté míry centralizovaně.

2.5.1 Tabulková architektura

Základní myšlenka spočívá v analogii s řešením problému skupinou odborníků, kde každý se specializuje na jinou oblast. Odborníci sedí v místnosti a dívají se na tabuli, která slouží pro zaznamenávání procesu řešení a ukládají se do ní různé mezivýsledky. Podle obsahu a poznatků odborníků přispívají odborníci k jeho změně. Proces řešení probíhá paralelně, proto je nutné zajistit koordinovaný zápis na tabuli. Tuto roli plní zapisovatel, který rozhoduje o tom, jak bude zrovna obsah pozměněn. Schéma tabulkové architektury je vidět na Obr. 2. [1]



Obr. 2 Základní schéma tabulkové architektury

2.5.2 Kontraktační síť

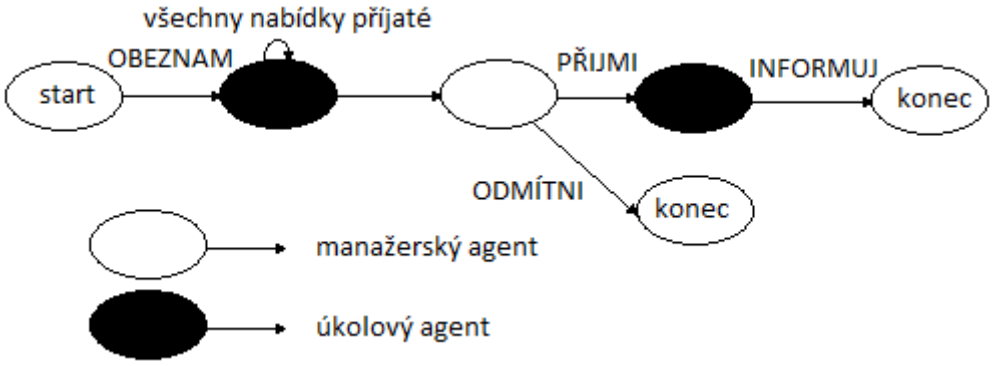
Kontraktační síť je síť agentů s vymezenými rolemi pro distribuované řešení problému. Agenti jsou rozděleni do 2 skupin. Řešitelští agenti se specializují na jistou oblast. Manažerský agent je řídící agent v síti, který zadává úkoly řešitelským agentům a kontroluje, zda jsou úkoly správně plněny. Řešitelský agent se skládá z několika částí:

- Lokální báze dat a poznatků – obsahuje informace poznatky o řešených úkolech a aktuálním stavu
- Řešící modul – zabezpečuje samotné řešení úkolu manažerským agentem
- Kontraktační modul – vedení negociace s manažerskými agenty a vytvoření kontraktu s ním, jestliže je agent schopen vyřešit aktuálně řešený problém

Kontraktační síť lze vyjádřit jako čtveřici (A, R, ρ, P):

- $A = \{a_1, \ldots, a_k\}$ – konečná množina agentů
- R = {řešitel, manažer} – dvouprvková množina rolí
- ρ: A -> R je funkce přiřazující role agentům, např. $\rho(A_1)$ = manažer a $\rho(A_i)$ = řešitel pro $2 \leq i \leq k$
- $P = \{K, \pi_{CNP}\}$je protokol kontrakční sítě, kde
  - K = {ZADEJ, NABÍDNI, PŘIJMI, ODMÍTNI, INFORMUJ} – množina komunikačních terminálů
  - π_{CNP} – protokol komunikačních kanálů

Grafické zobrazení je uvedeno na obrázku Obr. 3.



Obr. 3 Grafické zobrazení protokolu v kontraktační síti

2.5.3 Negociace

Negociace rozšiřují problém kontraktační sítě a slouží k řešení konfliktů mezi agenty, kdy se agenti musí dělit o zdroje nebo ke kooperaci vedoucí k plnění společných cílů. Rámec negociací tvoří 3 základní části:

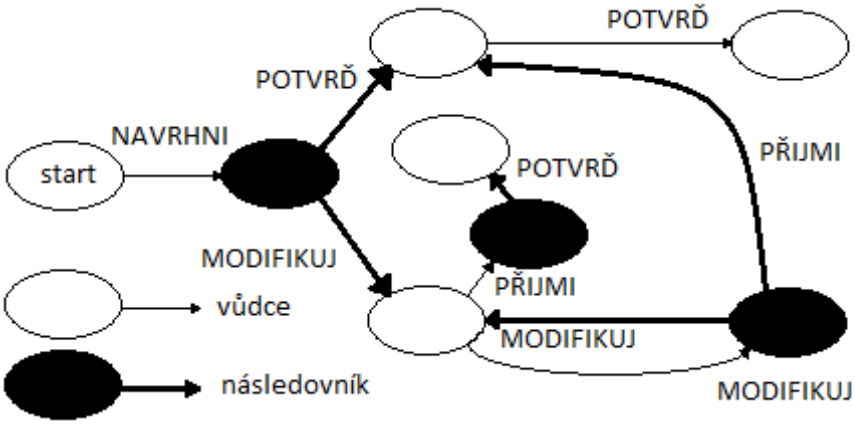
- Prostor možných řešení – množina cílů nebo závěrů, ke kterým se může skupina dopracovat
- Negociační protokol – posloupnost kroků, kterými lze konvergovat k dosažení cíle

- Negociační strategie – rozhodování v bodech, kdy lze pokračovat více než jedním směrem

Negociační množina (A, R, ρ, N, U, P, S) [1]:

- $A = \{a_1, \ldots, a_l\}$ – konečná množina agentů
- $R = \{r_1, \ldots, r_m\}$ – konečná množina rolí
- ρ: A -> R je funkce přiřazující každému agentu roli
- N je konečná negociační množina
- $U = \{U_1, \ldots, U_l\}$ – funkční užitečnost U_i: K -> K jednotlivých agentů, přiřazuje reálné číslo (užitečnost) každému návrhu jiného agenta (komunikačním aktu)
- $K = \{k_1, \ldots, k_n\}$ – konečná množina komunikačních terminálů
- π RxK -> $2^K \mid r \in R$ – funkce mapující prvky množiny K do reakcí vzhledem k rolím agentů
- $P = (K, \pi)$ – negociační protokol
- $S = \{s_1, \ldots, s_o\}$ – konečná množina strategií

Grafické zobrazení je uvedeno na obrázku Obr. 4.



Obr. 4 Zobrazení negociačního protokolu

3 Pogamut

Informace o Pogamutu jsou převzaty z [2].

3.1 Úvod

Pogamut je software naprogramovaný v jazyce Java, který umožňuje vytváření a ovládání virtuálních agentů v prostředí několika herních enginů. Nejrozšířenější je použití ve hře Unreal Tournament 2004. Hlavní účel je zjednodušit fyzické vytváření agenta. Většina akcí v prostředí lze provést jedním či 2 příkazy. Uživatel se díky tomu může soustředit více na vylepšování svého agenta. Aktuální nejnovější verze Pogamut 3.

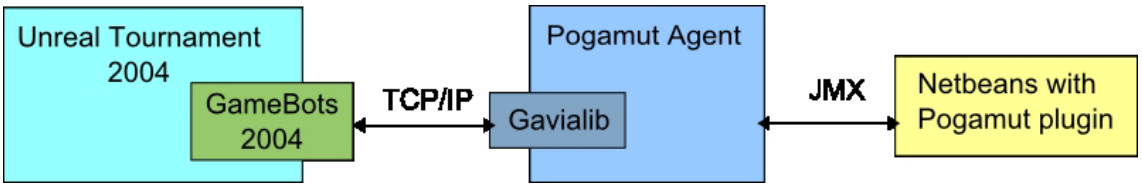
Vlastnosti Pogamutu 3:

- Svázání s virtuálním prostředím hry Unreal Tournament 2004 (UT2004)
- Vývojové prostředí pro ladění chyb
- Knihovna obsahující algoritmy pro pohyb, hledání cesty a podporu pro ovládání zbraní a střílení
- Připojení do tzv. POSH, což je reaktivní plánovač pro ovládání chování agentů a vizuální editor pro POSH plánování
- Podpora pro tvoření a spouštění experimentů

3.2 Přehled

Pogamut 3 integruje 5 základních komponent: UT2004, GameBots2004, GaviaLib library, Pogamut Agent, IDE (Obr. 5). Tyto komponenty implementují funkce zmíněné výše a umožňují uživateli jednoduše vytvářet a ovládat své agenty.

Informace z prostředí UT2004 jsou exportovány přes protokol TCP/IP textovým protokolem GameBots2004. Tyto zprávy jsou zpracovány Java knihovnou GaviaLib. Díky tomu může Pogamut agent pracovat s Java objekty. Pogamut agent může být upraven vzdáleně díky JMX protokolu přes Pogamut plugin do programu NetBeans.



Obr. 5 Architektura Pogamutu

3.2.1 Virtuální prostředí Pogamutu 3

UT2004 je velice známá akční videohra s tzv. částečně otevřeným kódem. Hra obsahuje UnrealScript – skriptovací jazyk vyvinutý autory UT2004, ve kterém je naprogramována velká část hry samotné. Uživatelé tento kód v jazyce UnrealScript mohou modifikovat. Hra také obsahuje velké množství předvytvořených objektů, map a editor map. Toto umožňuje uživatelům vytvořit si nový obsah do hry, který splyne s obsahem dodávaným ke hře.

3.2.2 Rozhraní do virtuálního prostředí

Agenty je možné vytvářet přímo v jazyce UnrealScript. Často je však výhodně vytváření agentů pomocí jiného externího mechanizmu. GameBots2004 propojuje zbytek Pogamutu se hrou UT2004.

GameBots2004 posílají informace z herního prostředí přes TCP/IP protokol a umožňuje uživatelům se připojit do hry, která podporuje klient-server architekturu. To znamená, že uživatel implementuje agenta jako klient a používá GameBots2004 jako server.

3.2.3 GaviaLib Library

Tato knihovna byla vyvinuta v jazyce Java pro připojování agentů do skoro jakéhokoliv virtuálního prostředí. Tato knihovna poskytuje:

- Abstraktní implementaci agenta, která spravuje agentův životní cyklus a umožňuje ovládání agenta pomocí JMX – standardního Java protokolu pro ovládání vzdálených instancí programu
- Agentovo rozhraní ve virtuálním světě, které spravuje objekty a události a na důležité události v prostředí agenta upozorňuje
- XML/XSLT strukturu pro definici objektů a událostí v prostředí

3.2.4 UT2004 Agent

UT2004 agent je soubor tříd a rozhraní napsaných v jazyce Java odvozených od základních tříd z GaviaLib knihovny. Agentovi je přidáno několik vlastností, jako např. možnost pohybu, navigace v prostředí, která využívá way-point systém použitý ve hře a další doplňkové třídy, které nám dávají informace o herních pravidlech.

3.2.5 Vývojové prostředí (IDE)

Vývojové prostředí je vyvíjeno jako plugin do Programu NetBeans. Může komunikovat s běžícími agenty pomocí JMX. Vývojové prostředí nabízí několik funkcí, jako např. šablony prázdných agentů, ukázkové agenty (Navigation bot, Hunter bot), správa UT2004 serverů, seznam běžících agentů, rychlý přístup k proměnným agenta. Není však nezbytně nutné tento plugin používat, lze se obejít i bez něj.

4 Turnajový systém

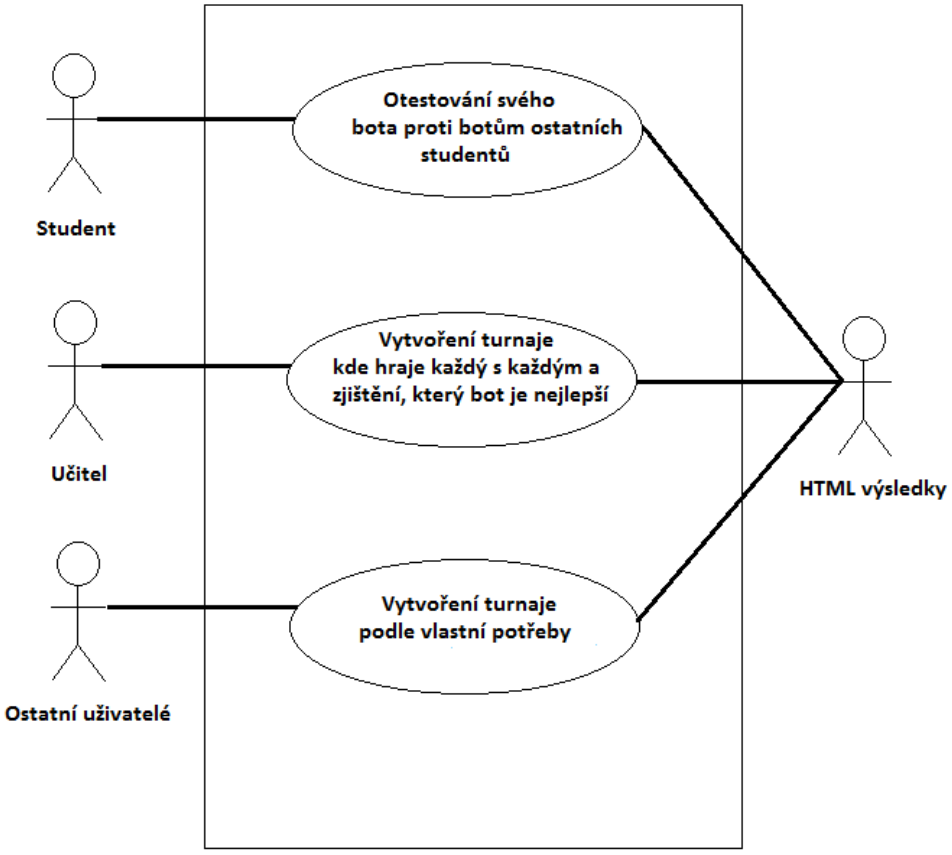
Program byl naprogramován v jazyce Java při využití vývojového prostředí NetBeans. Program je určen pouze pro operační systém Windows. Je to z důvodu práce s adresami souborů a používání dávkových souborů pro spuštění a ukončení serveru a botů.

4.1 Účel programu

Turnajový systém bude sloužit především pro kontrolou semestrálních prací (inteligentních agentů) z předmětu KIV/ISW či podobného předmětu na MFF UK. Zároveň bude sloužit studentům těchto předmětů pro testování svých agentů a možnost jejich doladění.

Druhý velký účel je poskytnout tento program komunitě okolo Pogamutu pro testování jejich vlastních botů nebo pouze pro vyzkoušení, kdo je lepší. Z tohoto důvodu je při konfiguraci botů možnost vybrat si vlastní spouštěč JAR souborů botů, jelikož ne všichni používají stejné parametry, jako jsou stanoveny u předmětu KIV/ISW a aby nemuseli své boty upravovat. Uživatel si naprogramuje jednoduchou JAR třídu, kterou zavolá program a předá jí všechny parametry potřebné pro spuštění bota. Třída si pak již bota spustí sama s vlastními parametry.

Přehledný popis znázorňuje diagram na Obr. 6:



Obr. 6 Use Case diagram

4.2 Specifikace modů

Turnajový systém bude obsahovat celkem 4 módy hry. Popis jednotlivých módů je vidět v tabulce Tab. 2.

Tab. 2 Tabulka módů, které obsahuje turnajový systém

Deathmatch (DM)	Team Deathmatch (TDM)	Capture the Flag (CTF)	Bombing run (BR)
Soutěží proti sobě 2 – n botů, hrají všichni proti všem	Soutěží spolu dva týmy botů, počet botů v jednotlivých týmech může být libovolný (týmy nemusí mít stejný počet botů)	Soutěží spolu dva týmy botů, počet botů v jednotlivých týmech může být libovolný (týmy nemusí mít stejný počet botů)	Soutěží spolu dva týmy botů, počet botů v jednotlivých týmech může být libovolný (týmy nemusí mít stejný počet botů)
Po dosažení limitu na čas nebo na fragy je zápas ukončen, vítězí bot s nejvyšším počtem fragů	Vítězí tým, který má na konci více fragů	Vítězí tým s vyšším počtem donesených vlajek	Vítězí tým s vyšším počtem vstřelených míčů

4.3 XML konfigurace

Turnajový systém je před spuštěním nejdříve nutné nakonfigurovat. K tomuto nám slouží vstupní XML soubor, ve kterém jsou uloženy veškeré parametry pro spuštění turnaje. Soubor můžeme nakonfigurovat 2 způsoby:

- Grafické rozhraní (GUI) – veškeré parametry lze nakonfigurovat pomocí grafického rozhraní, které turnajový systém obsahuje. Grafické rozhraní se spouští automaticky, jestliže je program spuštěn bez parametrů. Podrobnější informace o GUI viz kapitola 4.5
- Z příkazové řádky – XML soubor můžeme nakonfigurovat ručně a poté turnaj spustit rovnou bez použití GUI. Do příkazové řádky poté zadáme:
  `java –jar TournamentSystem.jar` *`jménoSouboru adresaUT2004 čas`*
  - *jménoSouboru* – jméno konfiguračního souboru, program ho bude hledat ve složce `\config`, případně lze zadat absolutní či relativní adresu
  - *adresaUT2004* – absolutní adresa ke hře UT2004
  - *čas* – čas, jak dlouho bude program čekat na spuštění serveru. Tento parametr se používá z toho důvodu, že není možné zjistit, zda server již naběhl či ne. Udává se v sekundách.

Ukázka konfiguračního souboru:

```
<?xml version="1.0" encoding="UTF-8" ?>
<tournament name="Testovaci turnaj"
            xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
            xsi:noNamespaceSchemaLocation="TournamentConfig.xsd">
      <tournamenttype game="capturetheflag" type="individualmatches"
      map="CTF-FaceClassic" translocator="false" />
      <matches>
            <match teamname1="isw1" teamcolor1="red" teamname2="isw2"
            teamcolor2="blue" />
      </matches>
      <settings>
            <timeinminutes limit="5" />
            <score limit="0" />
      </settings>
      <bots>
            <bot teamname="isw1" path="isw-2011 -1.jar" type="team"
            count="3" skill="7" class="" />
            <bot teamname="isw2" path="C:\pogamut\isw-2011-2.jar"
            type="team" count="3" skill="7" class="" />
      </bots>
      <weapons assaultrifle="1" shockrifle="1" linkgun="0"
      biorifle="0" minigun="1" flakcannon="1" rocketlauncher="1"
      sniperrifle="1" lightninggun="0" redeemer="1" ionpainter="1" />
      <parameters parameter="" />
</tournament>
```

Ukázkové konfigurační soubory pro deathmatch a teamdeathmatch jsou dodávány spolu s programem a jsou ve složce `\config`. Nyní následuje popis jednotlivých elementů.

4.3.1 Hlavní element

Hlavní element `tournament` značí začátek a konec konfiguračního souboru. Má jeden parametr:

- parametr `name` – značí jméno turnaje

Dále je zde odkaz na XML schéma, které kontroluje jeho správnost.

4.3.2 Nastavení turnaje

Nastavení turnaje se provádí v elementu `tournamenttype`. Nastavuje se zde mód hry, typ turnaje, mapa, a zda je povolen translokátor či nikoliv. Parametry:

1) `game` – mód hry, povolené tagy:
   - deathmatch (Deathmatch)
   - teamdeathmatch (Team Deathmatch)
   - capturetheflag (Capture the Flag)
   - bombingrun (Bombing Run)
2) `type` – typ turnaje, povolené tagy pro deathmatch
   - allteams – turnaj budou hrát všechny týmy uvedené v seznamu botů
   - individualteams – turnaj budou hrát pouze týmy vyjmenované v elementu matches
   - Povolené tagy pro týmové hry:
   - allagainstall – každý tým bude hrát s každým
   - firstagainstall – první tým uvedený v seznamu týmů bude hrát s ostatními
   - individualmatches – pouze jednotlivé zápasy, které uživatel nastaví v elementu matches
3) `map` – mapa, na jaké se bude turnaj odehrávat
4) `translocator` – zda je povoleno používat transfokátor ve hře, tagy:
   - true – translokátor je povolen
   - false – translokátor není povolen

4.3.3 Nastavení jednotlivých zápasů

Jestliže je zvolen typ turnaje individual matches nebo individual teams, musí uživatel v elementu `matches` vyjmenovat všechny zápasy, které se mají odehrát. Každý zápas je reprezentován elementem `match`. Parametry pro deathmatch:

1) `teamname` – jméno týmu

Povinné parametry pro týmové hry:

1) `teamname1` – jméno prvního týmu
2) `teamcolor1` – barva prvního týmu, povolené tagy:

- red nebo blue – pevné nastavení barvy danému týmu
- auto – barvy se nastavují automaticky

3) `teamname2` – jméno druhého týmu
4) `teamcolor2` – barva prvního týmu, povolené tagy:
   - red nebo blue – pevné nastavení barvy danému týmu
   - auto – barvy se nastavují automaticky

4.3.4 Element nastavení limitů

V elementu `settings` se nastavují limity na čas a na skóre. Obsahuje 2 elementy:

1) Element `timeinminutes` – časový limit v minutách, povinný parametr:
   - `limit` – nezáporné celé číslo, 0 značí nekonečno
2) Element `score` – limit na score nebo fragy (podle modu hry), povinný parametr:
   - `limit` – nezáporné celé číslo, 0 značí nekonečno

4.3.5 Nastavení botů a týmů

V elementu `bots` se nastavují informace o všech JAR souborech reprezentující jednotlivé boty. Pro každý soubor se vytvoří element `bot`, který obsahuje následující parametry:

1) `teamname` – jméno týmu, ke kterému bude JAR soubor botů náležet, více souborů může míst stejný název týmu, potom všichni tyto boti hrají spolu
2) `path` – cesta k souboru, může se jednat o relativní nebo absolutní cestu, případně lze pouze uvést jméno souboru. Potom je tento soubor hledán ve složce `\bots`
3) `type` – team nebo single
   - team – jeden JAR soubor reprezentuje celý tým – lze vytvořit více botů podle parametru count
   - single – jeden JAR soubor reprezentuje vždy 1 bota, nezáleží na parametru count
4) `count` – počet, kolik botů se má vytvořit, pouze pro typ team
5) `skill` – nastavení skillu daného týmu od 0 do 7
6) `class` – odkaz na uživatelem vytvořenou třídu, která spustí bota s vlastními parametry nebo v jiném pořadí, než je spouští turnajový systém. Při uvedení default se soubor spouští s defaultními parametry:
   - 1. parametr: 2011 nebo 2012 – aktuální rok
   - 2. parametr: 0 nebo 1, kde 0 je červený tým a 1 je modrý
   - 3. parametr: 0 až 7, skill botů
   - 4. parametr: počet botů v týmu
   - 5. parametr: localhost nebo xx.xx.xx.xx – IP adresa serveru

4.3.6 Nastavení zbraní

V elementu `weapons` se nastavuje, jaké zbraně budou zobrazeny ve vizualizace (kdo tuto zbraň sebral). Parametry reprezentují jednotlivé zbraně. Povolené tagy:

- 0 – nezobrazovat tuto zbraň
- 1 – zobrazovat

4.3.7 Vlastní parametry

V elementu `parameters` může uživatel uvést nějaké vlastní parametry, se kterými se má spouštět UT2004 server. Povinný parametr:

1) parameter – vlastní parametry
2)

4.4 XML výstup výsledků turnaje

Ukázka výstupního XML souboru:

```
<?xml version="1.0" encoding="UTF-8" ?>
<tournament name="Tournament System">
      <tournamenttype game="capturetheflag" type="individualmatches"
      map="CTF-FaceClassic" translocator="false" />
      <bots>
            <bot teamname="isw1" path=" isw-2011 -1.jar " type="team"
            count="3" skill="7" class="default" />
            <bot teamname="isw2" path="C:\pogamut\isw-2011- 2.jar"
            type="team" count="3" skill="7" class="default" />
      </bots>
      <results>
            <match name="1" timeinminuts="2">
                  <team1 teamname="isw1" teamcolor="red" score="4"
                  winner="true" />
                  <team1 teamname="isw2" teamcolor="blue" score="1" />
            </match>
      </results>
</tournament>
```

Nyní následuje popis elementů. Popis elementů `tournament`, `tournamenttype` a `bots` viz kapitola 4.3.

4.4.1 Výsledky turnaje

V elementu `results` budou uvedeny výsledky všech odehraných zápasů. Každý zápas je ohraničen elementem `match`, který má následující parametry:

1) `name` – číslo zápasu, číslování podle pořadí, v jakém by zápas odehrán
2) `timeinminutes` - časový limit v minutách
3) Element `team`, parametry pro deathmatch:
   - `teamname` – jméno týmu, ke kterému bot patří
   - `playername` – jméno bota
   - `playerscore` – počet fragů daného bota
   - `playerdeaths` – počet smrtí daného bota

   Parametry pro týmové hry:

   - `teamname` – jméno týmu

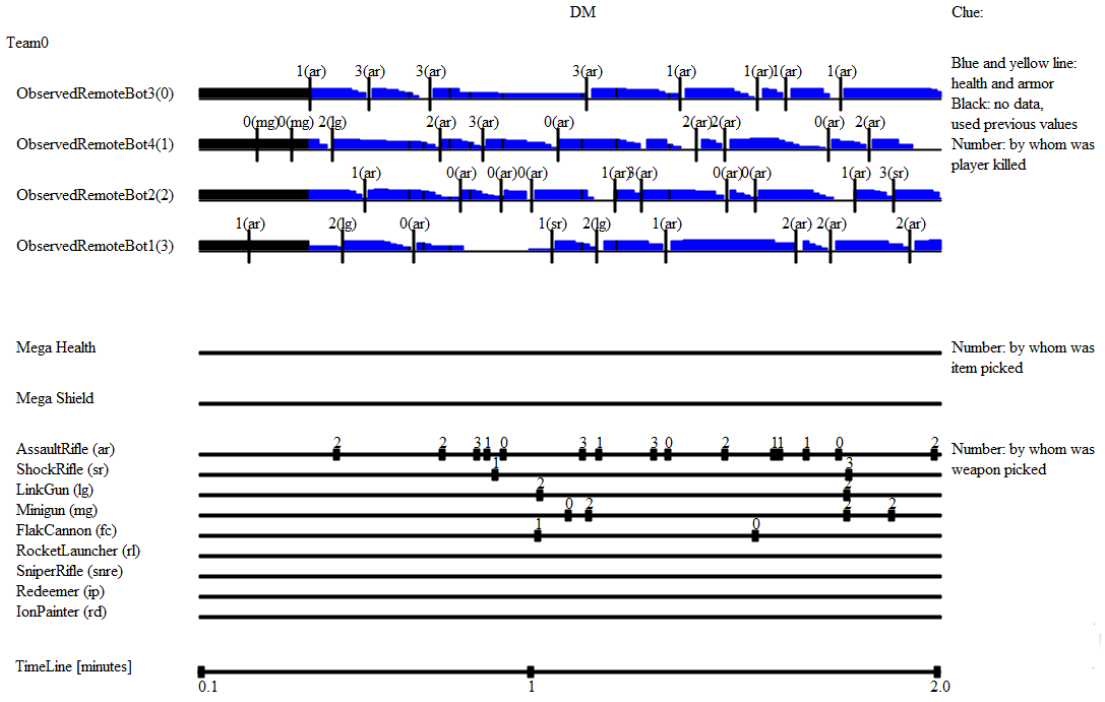
- `teamcolor` – barva týmu
- `score` – skóre týmu

4.5 Vizualizace

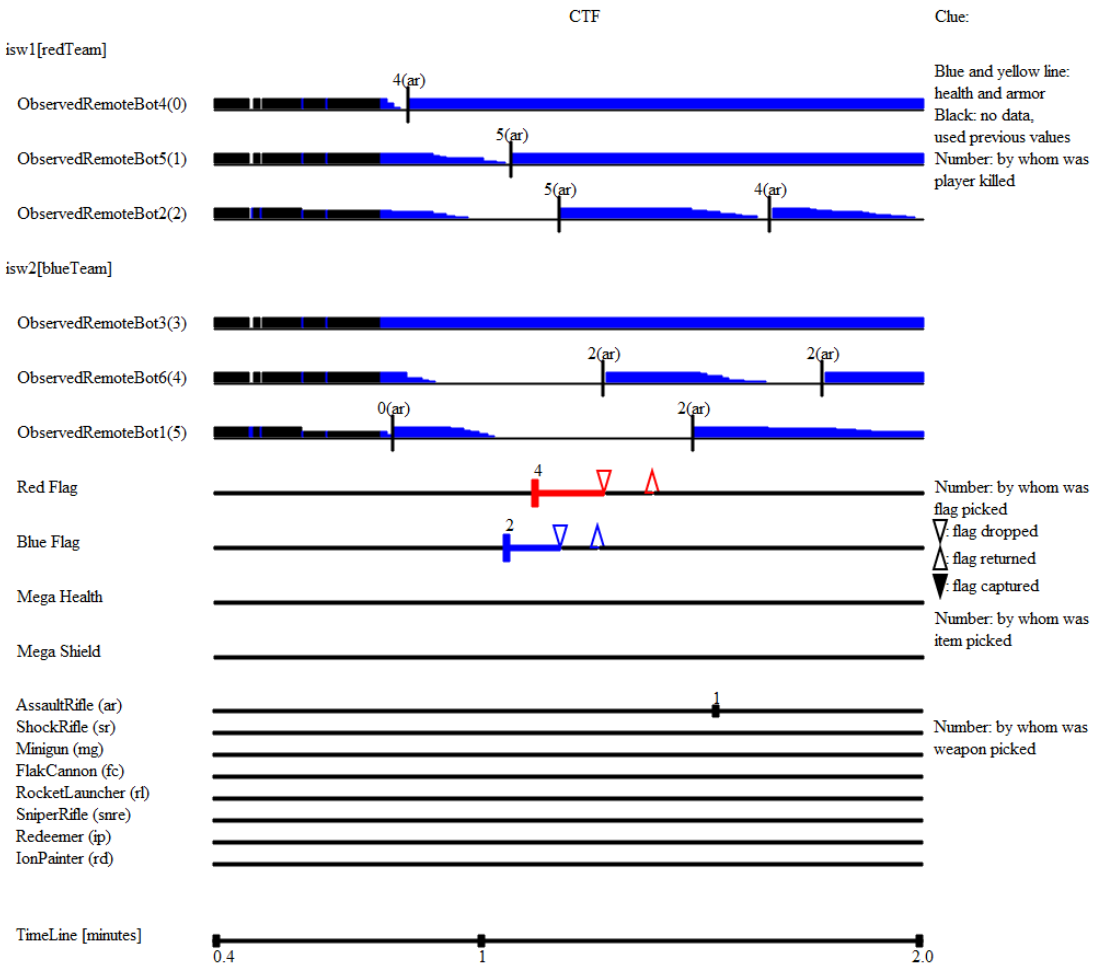
Po každém zápase se vygeneruje SVG obrázek s časovou linií a hlavními událostmi daného zápasu. Obrázek bude obsahovat:

- Seznam botů u červeného a modrého týmu. U jednotlivých botů bude znázorněno, kolik měli v jednotlivých časových úsecích života a štítů a jestliže byl bot zabit, bude znázorněno, kdy byl zabit, kdo ho zabil a s jakou zbraní.
- Kdy byl sebrán a kým MegaHealth a MegaArmor.
- U CTF (Capture the Flag) bude znázorněno kdy a kým byla vlajka sebrána, kdy byla upuštěna na zem, kdy byla vrácena a kdy byla donesena.
- Podle nastavení v konfiguračním XML souboru (viz kapitola 4.3) budou zobrazeny jednotlivé zbraně a u každé bude vidět kdy a kým byla zbraň sebrána.
- Ve spodní části obrázku bude vidět časová osa rozdělená po minutách
- Vpravo bude zobrazena legenda se všemi informacemi.

Ukázky vizualizace pro deathmatch (Obr. 7) a capture the flag (Obr. 8).



Obr. 7 Ukázka vizualizace pro Deathmatch



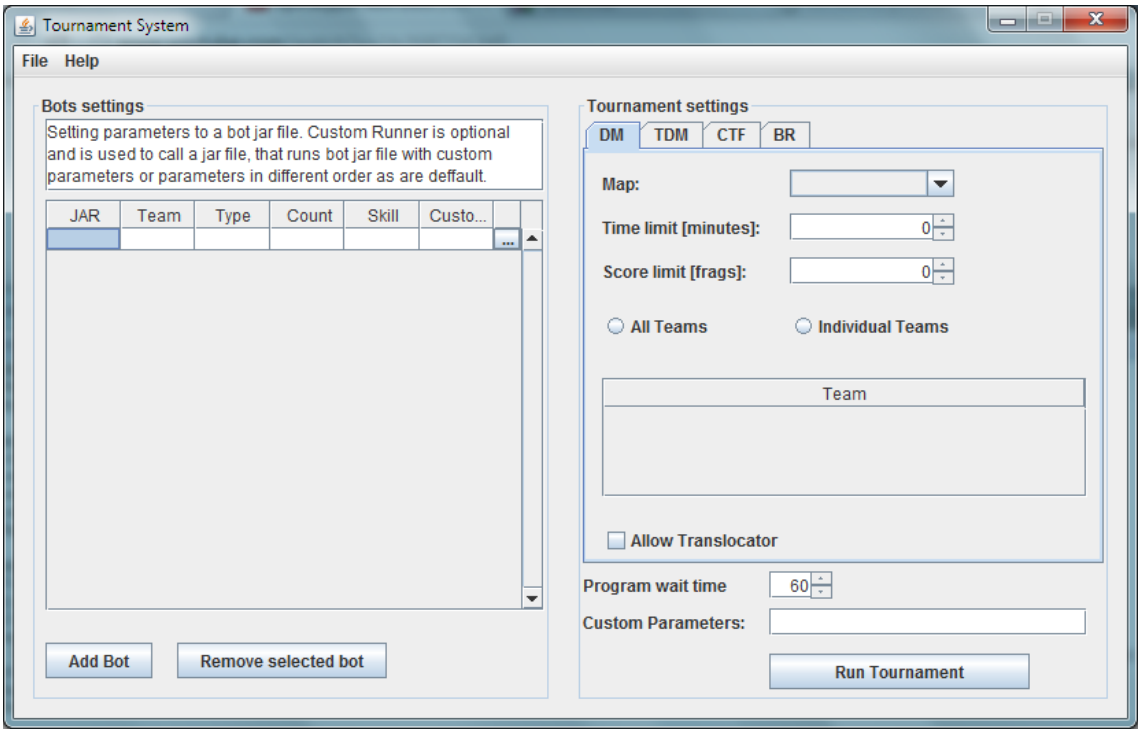
Obr. 8 Ukázka vizualizace pro Capture the Flag

4.6 Grafické uživatelské rozhraní

Turnajový systém obsahuje grafické uživatelské rozhraní (GUI) pro snadnou konfiguraci turnaje. GUI se spouští automaticky, jestliže je program puštěn bez parametrů. GUI je vytvořeno za použití knihovny Java Swing ve verzi 1.7.

GUI po spuštění programu je vidět na Obr. 9. Je rozděleno na 2 části. V levé části se provádí nastavení jednotlivých botů a týmů a jejich parametrů, v pravé části se provádí nastavení samotného turnaje – mód hry, mapa, limity, seznam zápasů aj.

GUI bylo vytvořeno za pomocí editoru pro vytváření uživatelského rozhraní v prostředí NetBeans. Více o tomto prostředí viz [3], Kapitola 5.



Obr. 9 Grafické uživatelské rozhraní

4.6.1 Nastavení botů

V nastavení botů se, jak z názvu vyplývá, nastavují jednotliví boti resp. JAR soubory jednotlivých botů. U každého souboru se musí nastavit několik parametrů:

1) JAR – jméno JAR souboru, po přidání bota se vyplní samo
2) Team – jméno týmu, ke kterému bude tento soubor patřit. Více souborů může míst stejný název týmu. Ty potom hrají spolu
3) Type – team nebo single
   - team – jeden JAR soubor reprezentuje celý tým – lze vytvořit více botů podle parametru count
   - single – jeden JAR soubor reprezentuje vždy 1 bota, nezáleží na parametru count
4) Count – počet, kolik botů se má vytvořit, pouze pro typ team
5) Skill – nastavení skillu daného týmu od 0 do 7
6) Custom Runner – odkaz na uživatelem vytvořenou třídu, která spustí bota s vlastními parametry nebo v jiném pořadí, než je spouští turnajový systém. Při uvedení default se soubor spouští s defaultními parametry:
   - 1. parametr: 2011 nebo 2012 – aktuální rok
   - 2. parametr: 0 nebo 1, kde 0 je červený tým a 1 je modrý
   - 3. parametr: 0 až 7, skill botů
   - 4. parametr: počet botů v týmu
   - 5. parametr: localhost nebo xx.xx.xx.xx – IP adresa serveru

Tabulka nastavení botů byla vytvořena za pomocí tutorialu [4].

Dále jsou zde 2 tlačítka pro změnu dat v tabulce:

- Add bot – otevře okno pro otevření souboru. Uživatel si vybere soubor, který chce otevřít a ten se následně přidá do tabulky s defaultními hodnotami. Při vybírání souboru jsou zobrazovány pouze soubory s příponou `*.jar`.
- Remove selected bot – odebere vybraný řádek z tabulky.

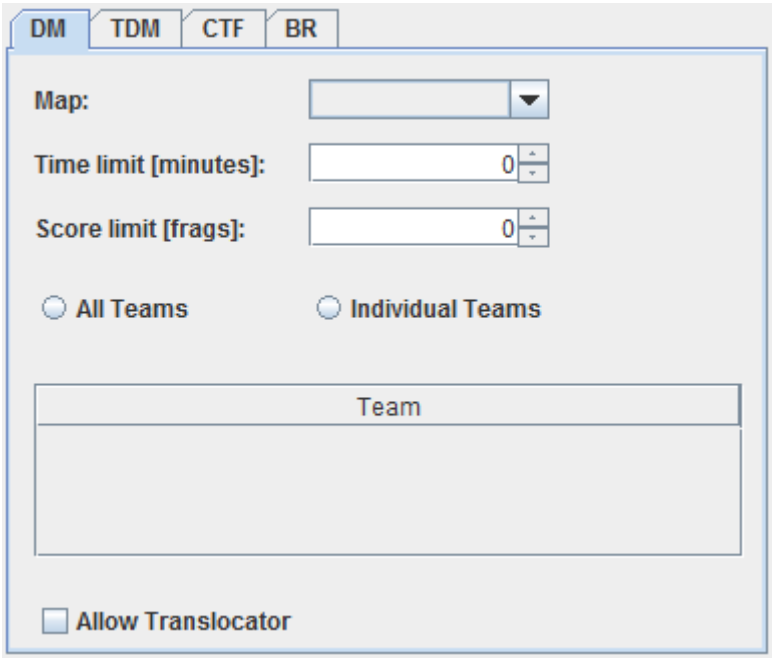
4.6.2 Nastavení turnaje

Zde se nastavují parametry turnaje, se kterými se následně spouští UT2004 server. Parametry:

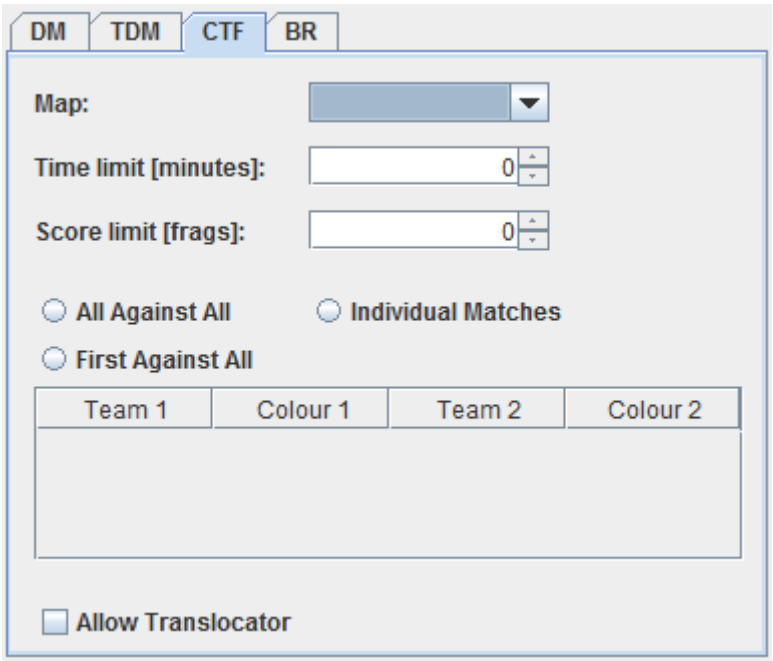
1) Záložka - záložka určuje mód hry. DM - deathmatch, TDM - team deathmatch, CTF - capture the flag, BR - bombing run. Záložka pro DM (Obr. 10) je lehce odlišná než pro ostatní hry (Obr. 11)
2) Map – mapa, na které se bude turnaj hrát
3) Time limit – časový limit v minutách. 0 = nekonečno
4) Score limit – limit na fragy (DM, TDM) nebo na skóre (CTF, BR). 0 = nekončeno
5) Typ turnaje – určuje, kdo bude hrát proti komu. Typy pro DM:
   - All teams - turnaj budou hrát všechny týmy uvedené v seznamu botů
   - Individual teams – turnaj budou hrát pouze týmy, které budou v tabulce jednotlivých zápasů vybrány
     i) Typy pro týmové hry:
   - All Against All – každý tým botů bude hrát s každým
   - First Against All – první tým ze seznamu botů bude hrát s ostatními
   - Individual Matches – uživatel si sám nastaví, kdo bude hrát proti komu v tabulce jednotlivých zápasů
6) Tabulka jednotlivých zápasů – při vybrání typu individual teams nebo individual matches se zobrazí tabulka pro nastavení jednotlivých zápasů. Její podoba opět závisí na vybraném módu hry. Pro DM:
   - Team – vybere se tým ze seznamu týmů

   Pro ostatní módy:

   - Team 1 – první tým
   - Color 1 – barva prvního týmu
   - Team 2 – druhý tým
   - Color 2 – barva druhého týmu
7) Allow Translocator – zaškrtnout pro povolení používání translokátoru ve hře
8) Remove row – odstranění řádku z tabulky jednotlivých zápasů



Obr. 10 Záložka pro DM



Obr. 11 Záložka pro týmové hry

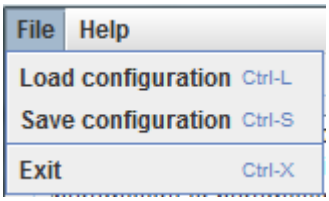
4.6.3 Ostatní nastavení

- Program wait time – jak dlouho bude program čekat na spuštění serveru po spuštění turnaje v sekundách. Čím pomalejší počítač, tím tento čas musí být větší. Tento parametr byl použit jako nouzové řešení, jelikož není možné zjistit, zda je již server spuštěn nebo ještě ne.
- Custom Parameters – vlastní parametry, s jakými se má spouštět server UT2004

- Run Tournament – spuštění vlastního turnaje s danými parametry. Jestliže chybí nějaké parametry, program vypíše informační hlášku.

Menu File (Obr. 12) :

- Load configuration – načtení kofigurace botů a turnaje z konfiguračního XML souboru. Více o konfiguračním souboru viz kapitola 4.3
- Save configuration – uložení aktuální konfigurace do XML souboru
- Exit – ukončení programu



Obr. 12 Menu File

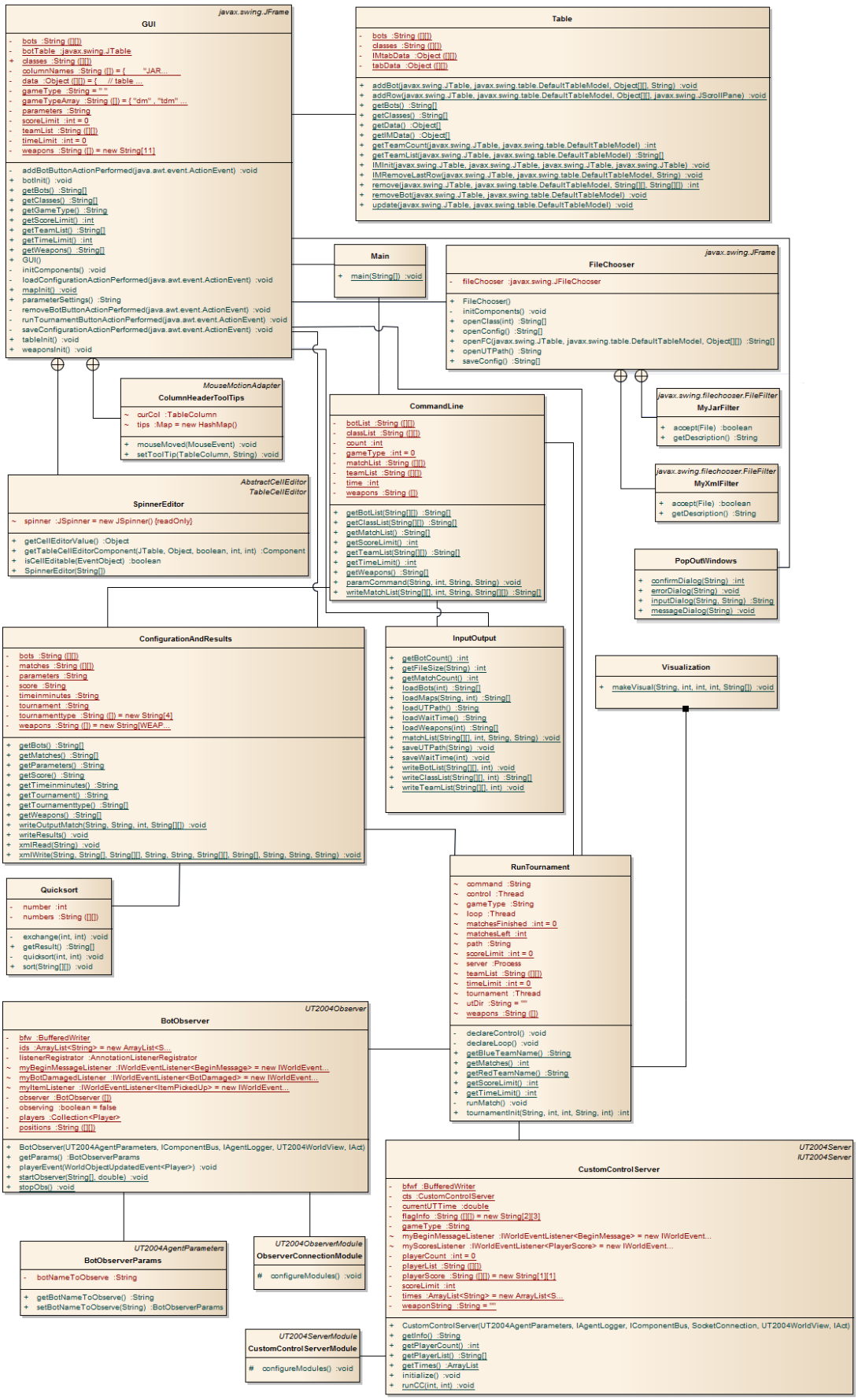
4.7 Průběh turnaje

Po nastavení všech parametrů a puštění turnaje (ať již z GUI nebo z příkazové řádky) se začne spouštět server hry UT2004 a program bude čekat zadaný počet vteřin na jeho spuštění. Po uplynutí této doby program spustí všechny JAR soubory botů, které mají daný zápas hrát. Také se spustí Control Connection (více o něm v kapitole 4.8.9), pomocí kterého se získává aktuální čas a informace o serveru a připojí se Bot Observeři (kapitola 4.8.10), kteří získávají informace o jednotlivých botech. V průběhu zápasu jsou do konzole vypisovány všechny důležité události v daném zápase (někdo někoho zabil, někdo sebral vlajku, apod.).

Po skončení zápasu se ukončí server i všichni běžící boti, vypíší se výsledky zápasu, vytvoří se jeho vizualizace a do výstupního XML souboru se přidají informace o tomto zápasu. Jestliže se nejedná o poslední zápas, opět se spustí server hry a program bude čekat na připojení botů. Jestliže to již byl poslední zápas, vytvoří se HTML stránka s tabulkou všech týmů a jejich pořadím a celkovým skóre. Zároveň zde budou odkazy na jednotlivé stránky jednotlivých zápasů. Ty obsahují tabulku se skóre botů (DM) nebo týmů včetně SVG grafu s tímto skóre (CTF) a také je zde vizualizace daného zápasu (více o vizualizaci viz kapitola 4.5).

4.8 Popis tříd

Program byl napsán v jazyce Java. Program je rozdělen do tří hlavních bloků. První se stará o nastavení všech parametrů turnaje, včetně ukládání konfigurací do souboru. Druhý obstarává samotný běh turnaje a stará se o spouštění a ukončování serveru a botů a získání potřebných dat. Poslední se stará o vytvoření vizualizací k jednotlivým zápasům, zápis celkových výsledků turnaje a vytvoření výstupní HTML stránky s odkazy na jednotlivé zápasy a tabulkou s celkovým pořadím. Na Obr. 13 je vidět UML diagram s přehledem všech tříd včetně vazeb mezi nimi.



Obr. 13 UML diagram

4.8.1 Třída Main

Třída Main se spouští jako první při spuštění programu. Zjišťuje se, zda byl program spuštěn bez parametrů či s parametry. Jestli by spuštěn bez parametrů, spustí GUI. Jestli byl spuštěn s parametry, zjistí, zda byla zadání relativní nebo absolutní cesta u konfiguračního souboru a převede ji na absolutní. Poté volá třídu pro načtení konfiguračního XML souboru.

Také se zjišťuje, zda byl předchozí turnaj dokončen. Ověří se, zda soubor, ve kterém jsou uloženy jednotlivé zápasy, je prázdný či nikoliv. Jestliže prázdný není, znamená to, že turnaj byl přerušen a nabídne se uživateli možnost pokračovat v předchozím turnaji.

4.8.2 Třída GUI

Tato třída reprezentuje grafické uživatelské rozhraní. Vytvoří se nové okno (popis viz kapitola 4.6). Při spuštění se provedou následující akce:

1) Volání metody pro načtení seznamů map pro jednotlivé módy ze souborů, které poté budou na výběr v příslušných rozbalovacích seznamech
2) Nastavení tooltipů
3) Inicializace tabulek
4) Volání metody pro načtení všech JAR souborů, které se při spuštění programu nacházejí ve složce `\bots` a jejich přidání do tabulky botů
5) Volání metody pro načtení, jaké zbraně se mají zobrazovat, ze souboru `weapons.cfg`

Změny v nastavení se ihned provedou. Vždy se do souboru zapisuje aktuální seznam botů, aktuální seznam týmů spolu s tím, jací boti patří k jakému týmu, a aktuální seznam zápasů. Při load/save configuration se provede načtení nebo uložení konfigurace z/do XML souboru.

Při spuštění turnaje se vytvoří příkaz pro spuštění serveru na základně aktuálních parametrů turnaje a poté se volá se třída pro spouštění a ukončení serveru a botů.

4.8.3 Třída InpoutOutput

Tato třída slouží pro čtení a zápis informací do souboru. Je zde také metoda pro vytvoření a zapsání seznamu zápasů. Zjistí se, o jaký typ turnaje jde (všichni proti všem či jednotlivé zápasy) a podle toho se seznam vytvoří. Ten se následně zapíše do souboru.

4.8.4 Třída FileChooser

Tato třída obsahuje dialog pro otevření JAR souborů botů. Jedná se o kontejnerový komponent FileDialog. Vybraný bot je poté přidán do tabulky. Zároveň zde jsou dialogy pro otevření a uložení XML souborů, kde má uživatel možnost si daný soubor vybrat. Také je zde dialog pro vybrání složky ke hře UT2004.

Při použití pro JAR soubory nebo pro konfigurační XML soubory je zde použit filtr, aby byly zobrazeny soubory pouze s danou příponou. Při vybírání složky ke hře UT2004 jsou zase zobrazeny pouze složky.

4.8.5 CommandLine

Třída CommandLine se volá v případě, kdy je program spuštěn s parametry. Na začátku se načtou informace ze zadaného konfiguračního souboru. Na základně těchto informací se vytvoří, stejně jako v třídě GUI, příkaz pro spuštění serveru.

Poté se vytvoří seznam týmů. Na každém řádku v souboru je uloženo jméno týmu a u každého jména týmu jsou jména JAR souborů botů, které mají dané jméno. Seznam se poté uloží do souboru stejně tak jako seznam botů s adresami jejich JAR souborů. Nakonec se zapíše seznam zápasů.

4.8.6 Třída ConfigurationAndResults

Tato třída slouží pro načtení a uložení informací z/do konfiguračního XML souboru. Soubor se načte jako XML soubor a čte se tak, že se vždy vyhledá potřebný element a vezmou se jeho parametry. Ukládání probíhá jako při zápisu do normálního textového souboru. Metodě jsou předány všechny potřebné parametry, které se mají zapsat.

Dále je zde metoda pro zápis do výstupního XML souboru, nejdříve se zapíše úvod. Po skončení zápasu se tento zápas zapíše spolu se jmény týmů a skóre. Po posledním zápasu se zapíše ukončení souboru.

Poslední metoda slouží pro zápis výsledků a vytvoření HTML souboru s celkovým skóre týmů a odkazy na jednotlivé zápasy a vytvoření HTML souborů jednotlivých zápasů se skóre daného zápasu, SVG grafem znázorňující dané skóre a vizualizací daného zápasu.

4.8.7 Třída Table

Tato třída obsahuje metody pro práci s tabulkou botů a tabulkou individuálních zápasů. Jsou zde metody pro přidání a odstranění botů a řádků. Dále je zde metoda udpate, která se volá při změně tabulky a nastavuje defaultní hodnoty, jestliže je nějaká buňka prázdná.

Metoda updateIM aktualizuje tabulku jednotlivých zápasů. Aktualizuje rozbalovací seznamy pro výběr týmu vždy podle aktuálního seznamu týmů. Metoda také zjišťuje, zda nebyly náhodou nastavené stejné barvy týmů u jednoho zápasu. Jestliže k tomu došlo, uživatel je informován a barva druhého týmu bude změněna. Také automaticky přidá nový řádek do tabulky, jestliže všechny hodnoty u předchozího zápasu jsou vyplněny.

Poslední metoda vytvoří seznam týmů (viz CommandLine).

4.8.8 Třída RunTournament

Tato třída zajišťuje spouštění jednotlivých zápasů – spouštění serveru a JAR souborů a jejich ukončování a zjišťování, zda po skončení zápasu ještě nějaké další zbývají.

Hlavní metodě, která se volá při spuštění turnaje, se předají parametry pro spuštění. Uživatel je dotázán na zadání cesty he hře UT2004 (pouze při puštění z GUI) a spustí se nové vlákno declareLoop, které opakuje cyklus, dokud zbývají nějaké zápasy a čeká se na jeho dokončení. V tomto cyklu se spouští nové vlákno tournament, které obsluhuje samotné spouštění a ukončování serveru a botů. V cyklu je dále smyčka, kde vlákno čeká, dokud není překročen limit na skóre nebo na čas. Jestliže se tak stane, ukončí se všechny instance třídy BotObserver (viz kapitola 4.8.10), ukončí se CustomControlServer (viz kapitola 4.8.9) a spustí se dávkový soubor `terminate.bat`, který zajisti ukončení serveru a všech puštěných botů. Poté se volá metoda pro vytvoření vizualizace (viz kapitola 4.8.11). Jestliže zbývají nějaké zápasy, volá se opět vlákno tournament, jinak vlákno končí.

Vlákno tournament vytvoří dávkový soubor `start.bat` pro spuštění serveru a tento soubor spustí. Dále si zjistí všechny potřebné informace o botech a týmech a načte si ze souboru zápasů následující zápas. Program si poté zjistí, jaké JAR soubory jsou u obou týmů a pro ty následně vytvoří dávkové soubory na jejich spuštění a ty poté spustí. Po spuštění všech JAR souborů se vytvoří nové vlákno declareControl, které spouští CustomControlServer.

4.8.9 Třída CustomControlServer

Tato třída vychází z Pogamut třídy CustomControlServer, která získává informace z prostředí UT2004. Při spuštění se vytvoří nový agent, který se do hry připojí na portu pro CustomControlServer. Tím může získat veškeré informace o hře (o agentech, předmětech apod.). Dále se vytvoří soubory, do kterých se zapisují jednotlivé informace o hře (o smrtích jednotlivých botů, o vlajkách, atd.).

Třída obsahuje listener pro získávání aktuálního uplynulého času ve hře. Vedle toho se zjišťuje několik informací:

- informace o stavu vlajek a o případné změně stavu je uživatel informován do konzole (pro CTF).
- informace o změně stavu bomby (pro BR).
- informace o skóre hračů (pro DM) nebo skóre týmů (pro ostatní módy).

Dále se při jakékoliv změně počtu hráčů ukončí a spouští znova instance třídy BotObserver, kteří slouží pro získání informací o jednotlivých agentech ve hře.

Další listener, který třída obsahuje, slouží k získání informací o skóre hráčů a z toho se následně odvozuje, kdo koho zabil (jestliže v jeden okamžik je změna jak u počtu bodů, tak u počtu smrtí). V tomto případě ale mohou nastat chyby a to když jsou dva boti zabití v 1 čas. Nedá se poté určit, kdo zabil koho.

4.8.10 Třída BotObserver

Tato třída vychází z Pogamut třídy BotObserver a slouží pro získání informací o jednotlivých agentech ve hře. Při spuštění se vytvoří nový agent, který se do hry připojí na portu pro BotObserver. Dále se vytvoří soubory, do kterých se zapisují jednotlivé informace o hráčích (jména a tým, o jejich životech a štítech, o sebraných předmětech).

Tato třída je spuštěna tolikrát, kolik je ve hře agentů. Každý agent ve hře má svého pozorovatele (observer), který o něm zjišťuje všechny informace.

Třída obsahuje stejný listener pro získávání aktuálního uplynulého času jako u CustomControlServer. V každém časovém intervalu se zjistí, kolik měli všichni agenti života a štítů a tyto informace se zapíší do souboru s informací aktuálního času.

Další listener zjišťuje, zda byl sebrán nějaký předmět. Jestliže se jedná o zbraň, anebo velkou lékárničku popř. velké brnění, tuto informací zapíše do souboru spolu s aktuálním časem. K této informaci také připojí to, kdo tento předmět sebral. To se zjistí tak, že se projdou pozice všech agentů v daný čas a vybere se ten, který se nacházel nejblíže sebranému předmětu. V tomto případě také může dojít k odchylkám, jestliže více botů stojí od předmětu v podobné vzdálenosti.

4.8.11 Třída Visualization

Tato třída slouží pro vytvoření vizualizace k aktuálně odehranému zápasu. Otevřou se všechny souboru s uloženými daty. Poté se v cyklu prochází všechny časové okamžiky uložené v poli a zjišťuje se, jaká akce se v daný okamžik stala (někdo byl zabit, změna stavu vlajky nebo sebrání předmětu či zbraně) a podle toho se aktuální akce nakreslí do vizualizace.

Velikost pole závisí na délce turnaje. Při delším turnaji by mohlo docházet k problémům s pamětí, ale muselo by se ale jednat o opravdu dlouhý zápas. Pro 10 minut dlouhý zápas obsahuje pole zhruba 2500 údajů. Pro 1 hodinu je to údajů asi 16 000. Pro 24 hodin by to bylo údajů zhruba 400 tisíc. Každá položka je typu double a má velikost 8 bajtů. V dnešní době je takováto náročnost vzhledem k délce zápasu minimální. V každý časový okamžik se také u každého bota nakreslí, kolik měl daný bot životů a štítů. Vzhled vizualizace je vidět na Obr. 6 a 7.

Vizualizace je tvořena jako SVG obrázek. Jedná se o vektorový obrázek, jehož hlavní výhoda spočívá v tom, že když obrázek přiblížíme, nezhoršuje se nám kvalita.

4.8.12 Třída PopOutWindows

Tato třída slouží k zobrazování informačních, chybových a vstupních dialogů. Jestliže např. uživatel nezadal všechny požadované informace při nastavování turnaje a botů, vyskočí informační okno s textem, co je ještě potřeba vyplnit (např. chybí jméno týmu u nějakého bota nebo není nastavena mapa apod.).

Je zde také vytvořen vstupní dialog pro vložení adresy složky hry UT2004. Uživatel má možnost si složku hry vybrat pomocí okna FileDialog, kde jsou filtrovány pouze adresáře.

4.9 Možnosti rozšíření

Program je snadno rozšiřitelný. Lze ho rozšířit např. o jiné módy, než jaké je možné aktuálně nastavit. Uživatel si vytvoří novou záložku v GUI, kde použije stejné informace jako v předchozích. Stáhne si mapy pro tento mód a uloží je do souboru s příslušným názvem. Poté projde kód a všude, kde je podmínka na aktuální mód turnaje, přidá podmínku pro test svého módu.

Lze také vytvořit nový způsob sestavování zápasů. Např. udělat si playoff, kde jsou boti rozděleni do dvojic. Vítězové hrají spolu, až nakonec zbude pouze jeden vítěz. Jde o tzv. „pavouka“. Uživatel si přidá do metody pro vytváření seznamu zápasů podmínku pro svůj typ turnaje a doprogramuje, jak se mají zápasy sestavit. Poté již pouze přidá tlačítko pro výběr tohoto typu turnaje do GUI.

5 Ověření funkčnosti aplikace

Aplikace byla otestována na několika botech s různými parametry turnaje – různé módy, rozdílná mapa, malý a velký časový limit, přerušení turnaje.

V akademickém roce 2011/2012 byla nasazena v ostrém provozu pro hodnocení semestrálních prací studentů na předmětu KIV/ISW. Bylo otestováno 11 botů a bylo odehráno celkem 110 zápasů, každý o délce 5 minut. Turnaj se odehrával na 2 mapách a hrál vždy každý s každým. Na konci bylo vidět, kdo měl kolik bodů a podle toho se určil vítěz.

Vyučující byl s prací spokojen. Měl několik doplňujících připomínek k fungování programu. Všechny připomínky vyučujícího byly zapracovány do nové verze programu.

6 Závěr

Tato práce pojednávala o turnajovém systému pro platformu Pogamut ve hře Unreal Tournament 2004. Práce vycházela ze specifikace požadavků vytvořeného před samotným programem. Byly splněny všechny požadavky v něm uvedené vyjma malých vizualizací, kde je vidět, jak po sobě jednotliví boti střílí, jelikož tuto informaci nelze spolehlivě získat, a grafické logovací okno.

Program bude sloužit na předmětu KIV/ISW pro ověřování prací, kde se vždy na konci roku uspořádá turnaj mezi prácemi studentů o to, kdo vytvořil nejlepšího bota. Zároveň bude sloužit studentům pro testování svých prací. Další využití programu je poskytnout tento program komunitě okolo Pogamutu pro tvoření svých vlastních turnajů.

V akademickém roce 2011/2012 byl program nasazen v ostrém provozu hodnocení semestrálních prací studentů. Do nové verze byly zapracovány všechny doplňující připomínky vyučujícího.

Program je snadno rozšiřitelný. Lze ho rozšířit např. o jiné módy, než jaké je možné aktuálně nastavit nebo si uživatel vytvoří nový způsob sestavování zápasů. Např. udělat playoff. Boti jsou rozděleni do dvojic. Vítězové hrají spolu až nakonec zbyde pouze jeden vítěz.

Použité zkratky a termíny

Bot	software umělé inteligence, který za každou vytvořenou instanci programu ovládá jednoho hráče v počítačových hrách
Dávkový soubor	jedná se o textový soubor obsahující sérii příkazů, které jsou zpracovány interpretem příkazového řádku
Frag	bod za zabitého protivníka, vychází ze slova fragging – slovo označující zabít nadřízeného důstojníka výbušným granátem
Herní engine	jádro počítačové hry či jiného programu
HTML	značkovací jazyk pro hypertext; používaný jako jeden z jazyků pro vytváření internetových stránek
Java	objektově orientovaný programovací jazyk
JAR	formát souboru pro programovací jazyk Java založen na ZIP kompresi
Listener	metoda volaná při změně stavu nějakého objektu
SVG	značkovací jazyk, který popisuje dvourozměrnou vektorovou grafiku pomocí XML
TCP/IP	obsahuje sadu protokolů pro komunikaci v počítačové síti
Way-point	bod ve fyzickém prostoru využívaný pro potřeby navigace
XML	značkovací jazyk, popisuje strukturu z hlediska věcného obsahu jednotlivých částí u dokumentů

Literatura

[1] KUBÍK, Aleš. Inteligentní Agenty. 1. vydání. Brno: Computer Press, 2004. ISBN 80-251-0323-4

[2] Gemrot, J., Kadlec, R., Bida, M., Burkert, O., Pibil, R., Havlicek, J., Zemcak, L., Simlovic, J., Vansa, R., Stolba, M., Plch, T., Brom C.: Pogamut 3 Can Assist Developers in Building AI (Not Only) for Their Videogame Agents. In: Agents for Games and Simulations, LNCS 5920, Springer, 2009, pp. 1-15, [online] [cit. 2012-05-02] <http://diana.ms.mff.cuni.cz/main/papers/pogamut3_AGS_final.pdf>.

[3] BŐCK, Heiko. Platforma NetBeans, Podrobný průvodce programátora. Přeložili Jaroslav Bien, Miloš Šilhánek, Michal Petřík. 1. vydání. Brno: Computer Press, 2010. ISBN 978-80-251-3116-9

[4] Oracle Corporation. Using Swing Components, How To Use Tables. [online] [cit. 2012-05-02] <http://docs.oracle.com/javase/tutorial/uiswing/components/table.html>

Příloha 1 – uživatelská příručka

1) Spuštění programu

Program se spouští příkazem `java -jar TournamentSystem.jar`. Program lze spustit 2 způsoby:

Bez parametrů

Spuštěním programu bez parametrů se otevře grafické okno s nastavením vlastního turnaje.

S parametry

program lze spustit se 3 povinnými parametry:

a) adresa konfiguračního XML souboru - lze použít absolutní či relativní adresu začínající „\“, nebo lze napsat pouze jméno souboru. Tento soubor pak bude hledán ve složce `\config`.

b) složka se hrou Unreal Tournament 2004. Adresa se píše s uvozovkami či bez.

c) Jak dlouho bude program čekat na spuštění serveru po spuštění turnaje. Podrobnější popis viz níže.

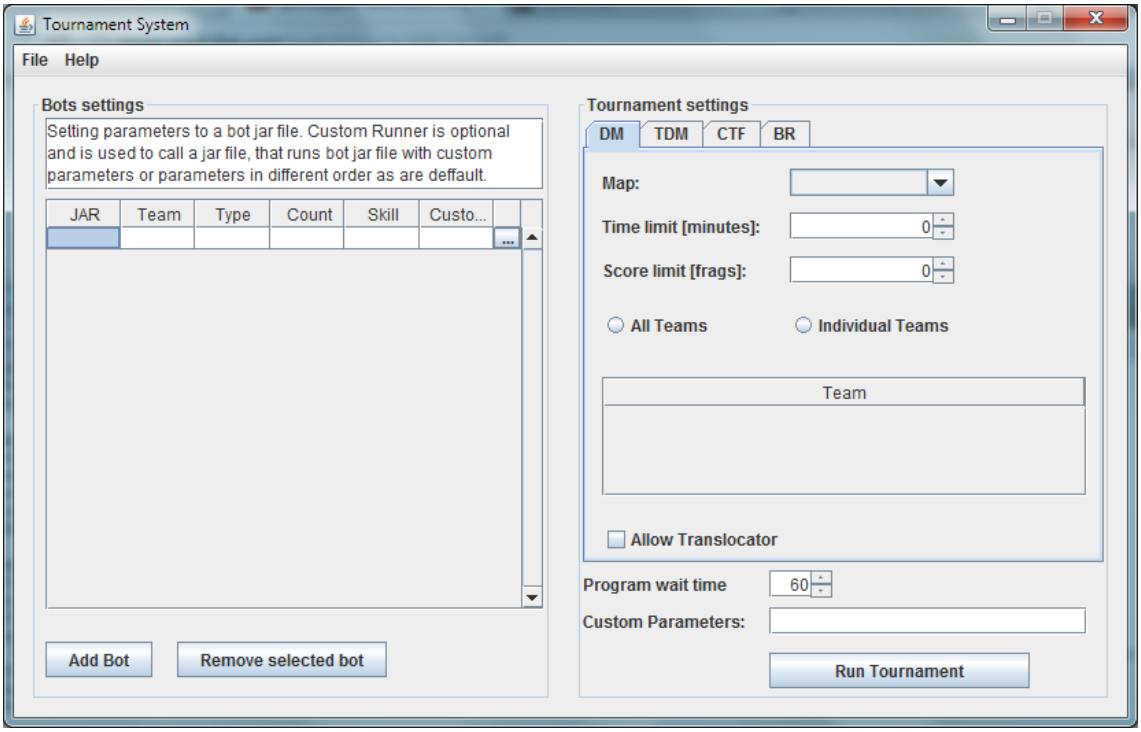
Příklad spuštění:

```
java -jar TournamentSystem.jar config.xml C:\UT2004\ 40
```

Program musí být nainstalován do adresáře, kde má uživatel právo zapisovat do souborů. Bez tohoto nelze program spustit. Také je nutné, aby program mohl zapisovat do adresáře hry UT2004, kvůli vytvoření dávkového souboru pro spuštění serveru, a také do adresářů, kde se nachází jednotliví boti.

2) Grafické uživatelské rozhraní

Po spuštění programu bez parametrů se objeví okno GUI:

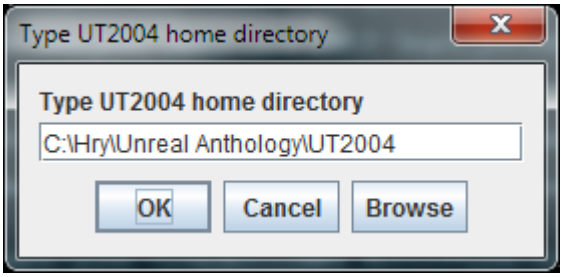


Podrobnější nastavení je uvedeno v kapitole 4.6.

3) Spuštění a běh turnaje

Nastavení cesty ke hře UT2004

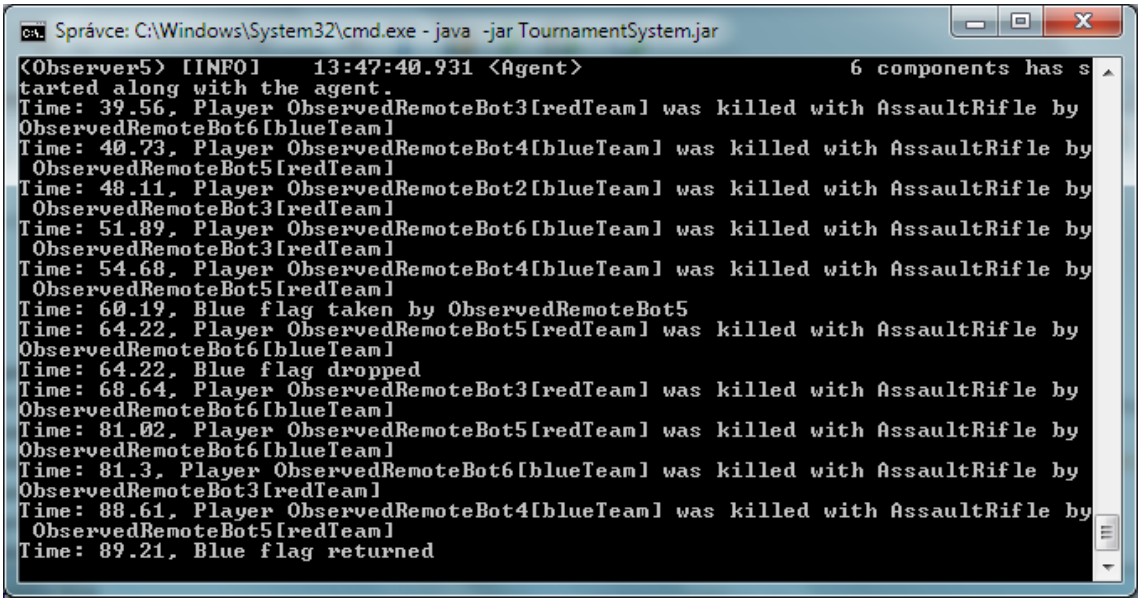
Po kliknutí na tlačítko Run Tournament bude uživatel dotázán na zadání cesty ke hře Unreal Tournament 2004. Bude vždy zobrazena poslední použitá cesta, která se načte ze souboru:



Průběh turnaje

Po zadání adresy se začne spouštět server hry a program bude čekat zadaný počet sekund na start serveru. Po uplynutí této doby začne připojovat jar soubory podle toho jaký tým (týmy) mají daný zápas hrát a zároveň se spouští ControlConnection pro získání informací o serveru a instance třídy BotObserver pro získání informacích o jednotlivých botech (připojování observerů trvá nějakou dobu a je závislé na počtu botů

ve hře). Uživatel je o všech akcích, které se zrovna ve hře dějí, informován do konzole (někdo sebral vlajku, někdo někoho zabil apod.). Část výpisu vypadá např takto:



Na konci každého zápasu jsou vypsány jeho výsledky (skóre týmů u týmových her nebo skóre hráčů u deathmatch)

4) Vizualizace

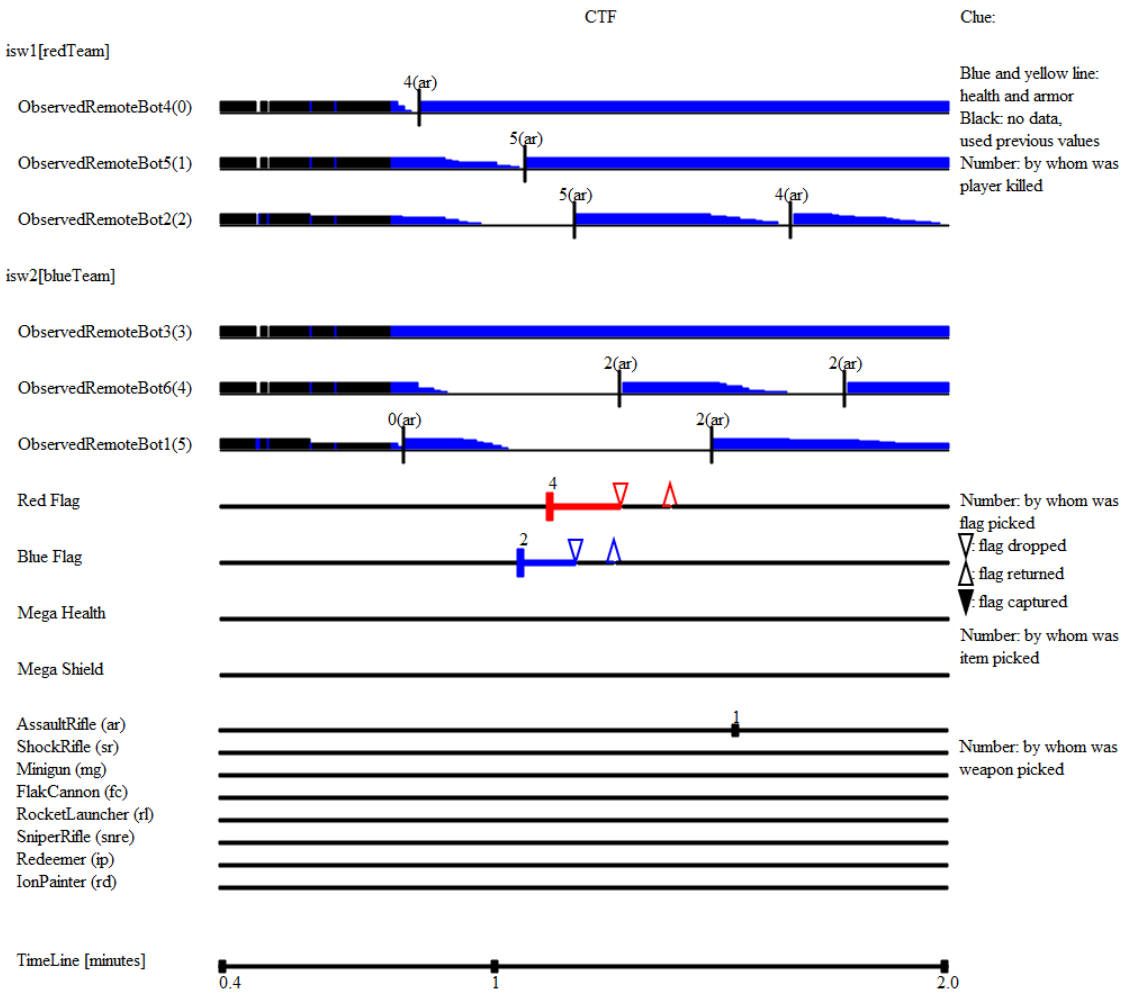
Po skončení celého turnaje budou ve složce `\output` výsledky turnaje.

Soubor output.xml

Bude se zde nalézat soubor output.xml, ve kterém jsou vidět parametry turnaje, jednotliví boti a také výsledky všech zápasů.

Soubor index.html

HTML stránka obsahující opět parametry turnaje a v případě týmových her i tabulku s celkovým pořadím, skóre a počtem bodů jednotlivých týmů. Dále jsou zde odkazy na jednotlivé zápasy. U každého zápasu je uvedena tabulka se skóre týmů nebo hráčů, v případě týmových her graf znázorňující skóre obou týmů a dále je zde vizualizace celého zápasu s vysvětlivkami. Ukázka vizualizace pro CTF:



5) Konfigurace

Soubor config.xml

Soubor `config.xml` slouží pro rychlou konfiguraci turnaje bez nutnosti vše vyplňovat manuálně v GUI. Popis viz kapitola 4.3.

Soubor weapons.cfg

Soubor weapons.cfg slouží pro nastavení, jaké zbraně se mají zobrazovat ve vizualizaci, 1 - zbraň bude zobrazena, 0 - zbraň se zobrazovat nebude. Soubor je umístěn ve složce `\config`.

Rychlé načtení botů

JAR soubory lze načíst ještě jedním způsobem: Jakékoliv JAR soubory nahrané do složky `\bots` před spuštěním GUI budou při spuštění GUI automaticky načteny do tabulky. Urychluje to organizování turnajů. Uživatel nemusí všechny boty po jednom přidávat.

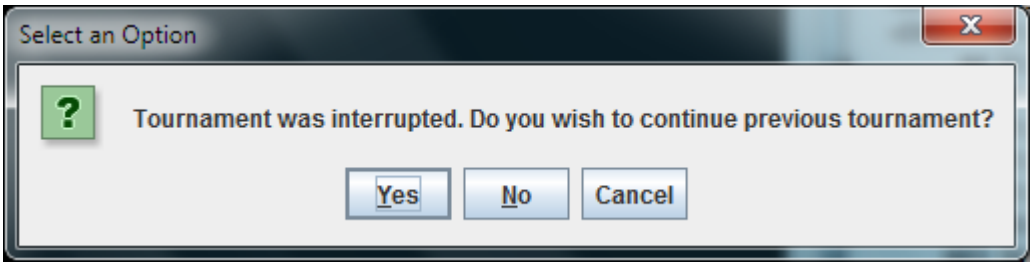
6) Speciální případy

Chyba při běhu serveru

Jestliže nastane chyba při běhu serveru, uživatel bude informován chybovou hláškou a v případě závažné chyby bude program ukončen.

Pokračování v přerušeném turnaji

Jestliže nastala chyba při běhu programu nebo uživatel program ukončil sám před dokončením, soubor se seznamem zápasů nebude prázdný. Jestliže uživatel spustí program znovu, dostane možnost pokračovat od místa, kde byl turnaje přerušen. Při spuštění se objeví hláška:



Při vybrání možnosti ano bude turnaj pokračovat v místě, kde byl přerušen. Při vybrání možnosti ne se pustí klasicky nové GUI.

Přidání bota do již dokončeného turnaje

Tento případ může nastat, když např. při kontrole semestrálních prací někdo odevzdá pozdě a do turnaje se nedostane. Uživatel může udělat to, že edituje soubor `matchList.txt` a přidá tam zápasy, které chce odehrát. Po spuštění si bude program myslet, že turnaj byl přerušen a nabídne uživateli možnost pokračovat s tím, že jestliže nový tým nebude v tabulce výsledků, do tabulky se přidá.