Jak rychle prototypovat chování umělých bytostí: Pogamut 2

Ondřej Burkert, Rudolf Kadlec, Jakub Gemrot, Michal Bída, Jan Havlíček a Cyril Brom

Univerzita Karlova v Praze, Matematicko-fyzikální fakulta
Katedra software a výuky informatiky, Praha, Česká Republika
ondra@atrey.karlin.mff.cuni.cz, brom@ksvi.mff.cuni.cz
URL: http://artemis.ms.mff.cuni.cz

Abstrakt *Zájem o nasazování umělých bytostí v různých aplikacích od her přes výukové a terapeutické nástroje po film neustále roste. V souvislosti s tím se objevují nové výzkumné problémy (např. problematika umělých emocí či sociálních interakcí). Výzkumníci však často tráví neúnosně mnoho času vytvářením vlastního vývojového prostředí pro bytosti a integrací bytostí do virtuálního světa. Tento článek představuje platformu pro rychlý vývoj chování umělých bytostí vtělených do trojrozměrného světa počítačové hry. Platforma nabízí sadu knihoven pro snadnou definici chování umělé bytosti a integrované vývojové prostředí, které umožňuje rychlé ladění kódu, parametrizaci a vytváření experimentů. Platforma je především určena pro výzku. Díky souboru tutoriálů je ovšem přístupná i širší odborné veřejnosti a lze ji využít také jako výukový nástroj.*

1 Úvod

V dnešní době stále stoupá zájem o umělé bytosti. Umělými bytosmi rozumíme speciální softwarové agenty dle Wooldridge [1] vtělené ve virtuálním světě. Vývoj chování umělých bytostí však není snadný. Důvody jsou zřejmé: agenti jednají v dynamickém, nepředvídatelném, interaktivním světě, jejich úkoly zahrnují, mimo pohybu po světě, různorodé netriviální úkoly, agenti spolu mohou také sociálně interagovat, projevovat emoce apod.

Příklady takových umělých bytostí můžeme nalézt například v moderních počítačových hrách [2], výukových hrách [3], terapeutických nástrojích [4] a virtual storytellingu [5]. Mnoho odlišných aplikací s sebou bohužel nese mnoho různých vývojových prostředí. Kvalitní vývojové prostředí je nezbytným předpokladem pro rychlý vývoj agentů. Studenti a výzkumnící, kteří chtějí vyvíjet vlastní agenty, však nemají k dispozici kompletní stáhnutelnou aplikaci. Vývoji samotného agenta tak musí předcházet náročný vývoj vlastního vývojového prostředí i virtuálního světa, či připojení vývojového prostředí ke světu počítačové hry (např. Unreal Tournament, Quake). Dostupná vývojová prostředí jsou (a) komerční (např. AI-Implant [7], Xaitment [9]), (b) proprietární řešení jednotlivých výzkumných skupin (např. [6]), (c) volně šiřitelné aplikace pro usnadnění vývoje umělých bytostí.

Komerční nástroje jsou pro nováčky v oboru nevhodné ze tří důvodů. Zaprvé předpokládají znalost mnoha pokročilých algorimů umělé inteligence, které implementují. Zadruhé vyžadují integraci do konkrétního virtuálního světa. Připojení do něj však obecně není triviální záležitost. Zatřetí jsou poměrně drahé.

Proprietární řešení jsou obecně nepoužitelná, neboť jsou určena pro konkrétní virtuální svět a zkoumaný problém. Navíc bývají často nedostatečně dokumentovaná a podporovaná.

Mezi volně šiřitelné nástroje určené pro tvorbu vlastních agentů patří například projekt Gamebots [10], který představuje rozhraní pro připojení agentů do hry Unreal Tournament 1999 [15], nebo F.E.A.R. [11], který k integraci agentů do hry Quake přidává framework pro návrh a vývoj agentů. Bohužel tyto aplikace neobsahují moduly, jako je integrované vývojové prostředí, podpora experimentů, knihovna modulů pro agentovy základní funkce, manažer žurnálů apod. Tyto moduly jsou nezbytné pro rychlý vývoj chování agentů. Naším cílem je tuto mezeru zaplnit a nabídnout nekomerční, stáhnutelnou platformu pro rychlý vývoj umělých bytostí.

Představovaná platforma se jmenuje Pogamut 2 a navazuje na naši předchozí práci (Pogamut 1 [12]). Nabízí navíc následující sadu nástrojů: (1) komunikaci agenta s virtuálním světem, (2) pomocné knihovny – senzory, správu paměti agenta, inventář, reprezentaci světa, A* algoritmus (algotimus pro hledání cest [13]) atd., (3) integrované vývojové prostředí (IDE) s širokou podporou vývoje, ladění a experimentů, (4) POSH [14] – rozhodovací systém (action selection mechanism - ASM).

Pogamut 2 je určen: a) pro výzkumné projekty zaměřené na chování umělých bytostí (mezi jinými výzkum spolupráce agentů a výzkum emocí), b) pro výuku problematiky umělých bytostí na akademické půdě. Jako virtuální 3D svět byl použit svět hry Unreal Tournament 2004 [15] (UT04). UT04 nabízí mimo editoru lokací mnoho předdefinovaných předmětů, lokací a typů her.

Obrázek 1. Záběr z hry Unreal Tournament 2004^{TM}.

Beta verzi Pogamutu 2 lze nalézt na našich stránkách [1]. Obsahuje veškerou funkcionalitu prezentovanou v tomto článku. Momentálně pracujeme na dokončení pokročilých nástrojů, přesněji na video tutoriálech a zpětným přehráváním zaznamenaných simulací a experimentů včetně inspekce zaznamenaných dat (např. stav agenta, stav ASM).

Článek má následující strukturu. V druhé kapitole probereme cíle projektu. Třetí kapitola popisuje architekturu Pogamutu 2. Čtvrtá kapitola nastiňuje styl práce s aplikací. Článek uzavírá diskuze řešení, popis probíhajících prácí a shrnutí.

2 Cíle projektu

Pogamut 2 si klade následující cíle:

1. Rozšiřitelnost a modularita – licence LGPL [2] umožňuje zásahy do kódu, architektura počítá s připojením libovolných ASM (např. SOAR[16]) a dalším rozšiřování IDE a knihoven.
2. Uživatelsky přívětivé prostředí – IDE podporující intuitivní vývoj a ladění.
3. Paralelizace – díky architektuře klient/server jsou odděleny UT04 a ASM, lze tedy dělit zátěž na více strojů při zapojení mnoha agentů najednou.
4. Rychlý začátek práce – dokumentace, příklady a tutoriály snižují startovní čas
5. Snadná validace modelu – podpora experimentů usnadňuje validaci

Komerční nástroje [7], [8], [9] splňují všechny požadavky mimo experimentů. Dostupným volně šířitelným nástrojům (F.E.A.R., Gamebots, JavaBots [17], Tielt [18], Pogamut 1 apod.) nejčastěji chybí komplexní IDE a podpora experimentů.

3 Architektura Pogamutu 2

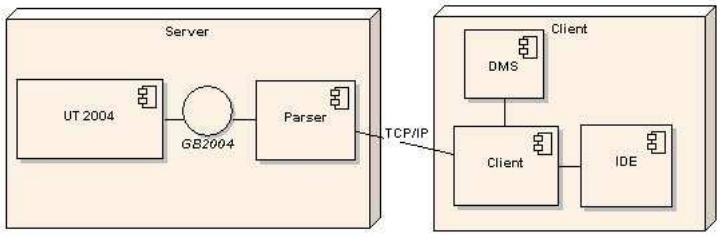
Pogamut 2 se skládá z šesti základních modulů: (1) UT04, (2) Gamebots2004 (GB04), (3) Parseru, (4) Klienta, (5) IDE a (6) ASM. Většinu modulů jsme naprogramovali v jazyce Java. Výjimku tvoří UT04, což je komerční software, a GB04, který vznikl úpravou a rozšířením starší verze (GB [10]). Nyní představíme podrobněji jednotlivé moduly.

Unreal Tournament 2004 je počítačová hra, kterou používáme jako virtuální svět pro agenty. Umožňuje připojení až 30 agentů do jedné lokace, což je tedy horní hranice i pro Pogamut 2. UT04 lze modifikovat dvěma způsoby. Zaprvé pomocí editoru, který umožňuje například vytváření nových lokací a úpravu stávajících. Zadruhé lze upravovat přímo skripty definující chování hry, předmětů a postav ve hře. Skripty jsou zapisovány v interním skriptovacím jazyku UnrealScript. Kombinací těchto způsobů lze vytvořit v podstatě libovolný virtuální svět.

Gamebots 2004 je server zabudovaný do UT04 starající se o komunikaci mezi avatarem ve hře a jejím externím ASM. GB04 je adaptací původních GB určených pro UT99 [15]. Mezi rozšíření oproti GB patří posílání všech předmětů a navigačních bodů v lokaci před zahájením simulace, automatický raytraycing, příkazy pro ovládání serveru a nahrávání záznamů simulace.

[1] http://artemis.ms.mff.cuni.cz

[2] Lesser General Public License



Obrázek 2. Architektura platformy je následující: UT04, GB04 a Parser na jednom stroji zajišťují běh simulace, propagaci podnětů k agentovi a provedení vybraných akcí. ASM, Klient a IDE mohou běžet na jiném stroji a starají o rozhodování a zobrazují práci agenta uživateli.

Parser převádí řetězce z GB04 na Java objekty, které posílá dál ke Klientovi. Hlavním účelem tohoto modulu je optimalizovat komunikaci s Klientem a umožnit tak připojení většího počtu agentů.

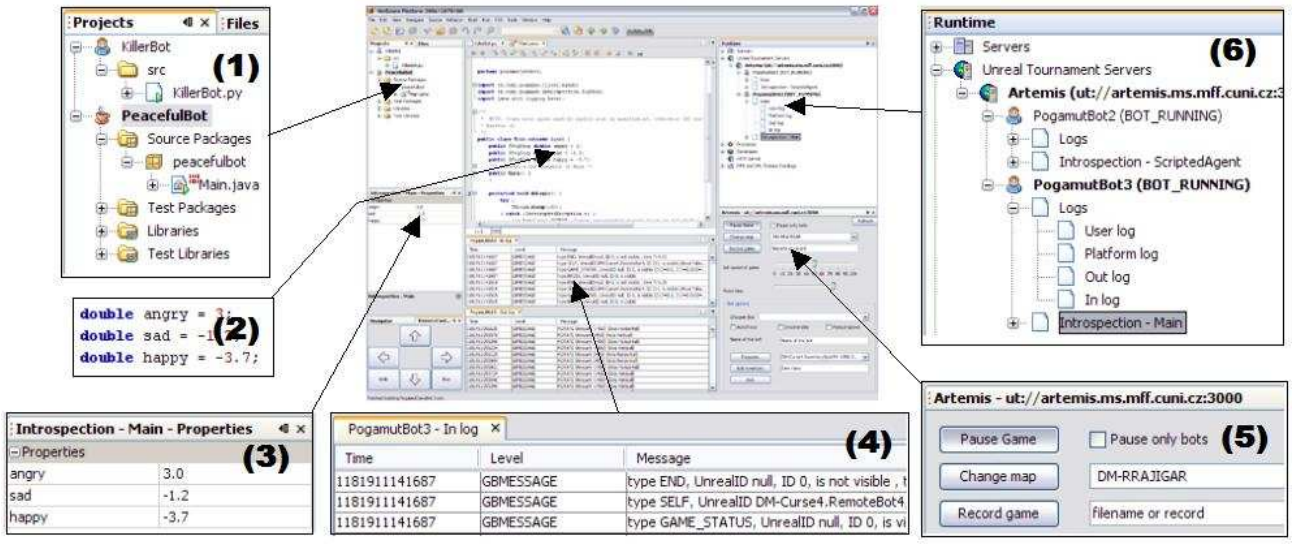
Klient je balík Java tříd. Nabízí: (a) krátkodobou paměť na externí vjemy, (b) příkazy - primitiva pro kontrolu těla agenta, (c) inventář předmětů, které agent posbíral, (d) metody pro pohyb a navigaci v lokaci (zahrnují i interní A*).

IDE představuje plug-in pro NetBeans^{TM} [19]. IDE pomáhá ve všech podstatných částech vývojového cyklu: implementaci, ladění a experimentech. Obsahuje:

- podporu skriptování chování agentů (Java, Python, reaktivní plánovač POSH [14]).
- správu knihovny dostupných agentů.
- ladící nástroje – inspektor interních proměnných agenta, pohledy na agentovu paměť, přehled žurnálů z ASM a komunikace, atd.
- sadu metod pro definici experimentů – uživatel může definovat počáteční konfiguraci, počet opakování, ukončovací podmínku apod.

4 Práce s Pogamutem 2

Obecně lze proces vytváření umělé bytosti rozdělit do následujících kroků: (1) definice modelu, (2) implementace modelu, (3) ladění implementace, (4) ladění parametrů modelu a (5) experimentů. IDE Pogamutu 2 bylo navrhnuto tak, aby podporovalo poslední čtyři kroky. V této sekci nastíníme, jak je toho dosaženo.



Obrázek 3. Ukázka aktuální verze IDE v ladícím módu. Popis jednotlivých oken se naléza v následujícím textu.

Implementace modelu. Platforma aktuálně podporuje vytváření ASM v Javě, Pythonu nebo v POSHi. Balík tříd Klienta zajišťuje komunikaci s GB04 a nabízí vysokoúrovňové rozhraní pro ovládání agenta. Stará se také o paměť agenta a inventář. IDE obsahuje správu projektů (Obr. 3, okno 1) a editor se zvýrazňováním syntaxe (3.2).

Ladění. Ladění v IDE je složeno z osmi částí, které jsou vhodné pro odstraňování chyb i parametrizaci. Seznam běžících serverů a agentů (3.6) pomáhá se správou více agentů. Introspekce a vlastnosti platformy (3.3) nabízejí rychlý přehled o interních proměnných agentů i platformy. Žurnály (3.4) zobrazují zprávy zaznamenávané v komunikaci mezi Parserem a GB04, interní zprávy platformy a zprávy ASM. Žurnály lze filtrovat podle typu zprávy, navíc jsou podle typů barevně odlišeny. Ruční ovládání agenta umožňuje umístění agenta na konkrétní pozici. IDE nabízí možnost zastavit simulaci a prohlédnout si aktuální situaci ve virtuálním světě společně s žurnály. Zastavení simulace je obsaženo v panelu ovládání serveru (3.5), který dále umožňuje nastavovat rychlost hry, lokaci, měnit pozice jednotlivých agentů či vytvářet nahrávky z aktuální simulace.

Ladění parametrů. Modely jsou typicky velmi citlivé na nastavení mnoha parametrů. IDE umožňuje nastavování parametrů za běhu. Nástroje zmíněné v předchozím odstavci také slouží k rychlejšímu nalezení vhodného nastavení parametrů.

Experimenty. Celková evaluace modelu je možná díky specifikaci experimentů v IDE (lze specifikovat kontrétní lokaci, počet agentů, jejich startovní pozice, vybavení, ukončující podmínku, počet opakování experimentu). Do simulace lze rovněž přidávat breakpointy. Představené vlastnosti umožňují snadnou práci s experimenty a jejich opakování.

5 Diskuze a probíhající práce

Hlavní přínos Pogamutu 2 spočívá v integrování vývojového prostředí, bohaté knihovně předdefinovaných metod pro design agenta a možnosti vytváření agenta v Pythonu za použití reaktivního plánovače POSH. Nicméně systém má jistá omezení. K serveru lze připojit maximálně 30 agentů, není tedy vhodný pro masové simulace. Dalším aspektem je tok času; rychlost hry lze měnit jen do jisté míry, což činí platformu nevhodnou například pro nasazení evolučních algoritmů.

Nyní pracujeme na implementaci pokročilých nástrojů. Patří mezi ně podpora více skriptovacích jazyků a dalších ASM (např. SOAR [16]), videotutoriály a timeline. Timeline umožní zaznamenat simulaci společně s žurnály. Následně bude možné přehrát simulaci simultánně se zobrazením výpisu ze žurnálů, a tak sledovat přehledně dění v simulaci i změny v ASM agenta. Tento nástroj tak bude užitečný při odstraňování chyb a parametrizaci.

6 Závěr

Představili jsme platformu pro vývoj umělých bytostí v komplexním virtuálním světě počítačové hry Unreal Tournament 2004. Platforma obsahuje knihovny pro tvorbu agenta. Knihovny obsahují moduly pro vnímání, motorická primitiva, paměť, inventář a reprezentací světa. Klíčovou komponentou systému je IDE, které nabízí inspektor proměnných, ovládání serveru, pohledy na žurnály, agenta a paměť, editor skriptů atd. Dohromady tyto moduly tvoří platformu pro rychlé prototypování chování umělých bytostí ve světě UT04.

Platforma je zamýšlena pro pětifázový vývoj agenta. Fáze jsou: definice modelu (specifikace), implementace modelu, odstraňování chyb v implementaci, parametrizace, experimenty. Poslední čtyři jsou podporovány platformou. Vývojář je tak ušetřen zbytečné práce s komunikací, pamětí, reprezentací objektů světa atd. a může se zaměřit přímo na řešený problém.

Platforma je primárně určena pro výzkum. Konkrétně je to výzkum (1) emocí, (2) navigace v prostoru a (3) spolupráce agentů. Dále je možno využít platformu pro výuku. Používání platformy je intuitivní a přímočaré. Obsáhlý soubor tutoriálů a instruktážních videí ještě víc zkracuje startovní čas. Naše skupina zamýšlí použití platformy na MFF UK pro demonstraci praktických problémů umělých bytostí studentům se zájmem o tuto oblast. Snadná práce s platformou může být lákavá pro komunity hráčů, kteří již dnes experimentují se svými oblíbenými hrami [20], [21].

Poděkování. Tato práce byla podpořena granty GA UK 1053/2007/A-INF/MFF, GA UK 351/2006/A-INF/MFF a programem ”Information Society” pod projektem 1ET100300517.

Literatura

[1] Wooldridge, M., Jennings, N. R.: Intelligent Agents - Theories, Architectures and Languages. In: Volume 890 of Lecture Notes in Artificial Intelligence. Springer-Verlag (1995)
[2] Microsoft: Halo 2. URL: http://www.bungie.net [24. 4. 2007]
[3] Tactical Iraqi. URL: http://www.tacticallanguage.com.[24. 4.2007]
[4] Hodges, L.F., Anderson, P., Burdea, G.C., Hoffman, H.G., Rothbaum, B. O.: Treating Psychological and Physical Disorders with VR. IEEE Computer Graphics and Applications (2001) 25-33
[5] Mateas, M., Stern, A.: Façade: An Experiment in Building a Fully-Realized. Interactive Drama. Game Developers Conference (2003)
[6] Cavazza, M., Charles, F., Mead, S.J.: Developing Re-usable Interactive Storytelling Technologies. IFIP World Computer Congress 2004, Toulouse, France (2004)
[7] Engenuity Technologies Inc.: AI-Implant. URL: http://www.ai-implant.com [24. 4. 2007]
[8] Softimage: XSI. URL: http://www.softimage.com [24. 4. 2007]
[9] X-AItment GmbH: X-AItment, URL: http://www.x-aitment.net [24. 4. 2007]
[10] Adobbati, R., Marshall, A. N., Scholer, A., and Tejada, S.: Gamebots: A 3d virtual world test-bed for multi-agent research. In: Proceedings of the 2nd Int. Workshop on Infrastructure for Agents, MAS, and Scalable MAS, Montreal, Canada (2001). URL: www.planetunreal.com/gamebots [24. 4. 2007]
[11] Champandard, A.J.: AI Game Development: Synthetic Creatures with Learning and Reactive Behaviors. New Riders (2003). URL: http://fear.sourceforge.net [24. 4. 2007]
[12] Brom, C., Gemrot, J., Bida M., Burkert O., Partington, S. J., Bryson, J. J.: POSH Tools for Game Agent Development by Students and Non-Programmers. Proc. of CGAMES 2006, Dublin, Ireland (2006) 126-133. URL: http://ail.jinak.cz [24. 4. 2007]
[13] Higgins, D.: Generic A* Pathfinding. In: AI Game Programming Gems (Mark DeLoura, ed.). Charles River Media (2000)
[14] Bryson, J.J.: Inteligence by design: Principles of Modularity and Coordination for Engineering Complex Adaptive Agent. PhD Thesis, MIT, Department of EECS, Cambridge, MA (2001)
[15] Epic Games: UnrealTournament 2004. URL: http://www.unrealtournament.com [24. 4. 2007]
[16] University of Michigan: SOAR. URL: http://sitemaker.umich.edu/soar/home [24. 4. 2007]
[17] JavaBots. URL: http://utbot.sourceforge.net [24. 4. 2007]
[18] Molineaux, M., Aha, D.W.: TIELT: A Testbed for Gaming Environments. Proceedings of the Twentieth National Conference on Artificial Intelligence (Intelligent Systems Demonstrations), Pittsburgh, PA: AAAI Press (2005)
[19] Sun Microsystems, Inc: Netbeans. URL: http://www.netbeans.org [24. 4. 2007]
[20] Counter Strike bots. URL: http://www.cstrike.ro/bots.php [24. 4. 2007]
[21] Bots United. URL: http://www.bots-united.com [24. 4. 2007]