Scalable Agent-Based Simulation of Players in Massively Multiplayer Online Games

Amund Tveit Øyvind Rein Jørgen V. Iversen
Mihhail Matskin

Department of Computer and Information Science,
Norwegian University of Science and Technology
N-7491 Trondheim, Norway

Abstract. We propose a parallel mobile agent platform - Zereal - for scalable and flexible simulation of Massively Multiplayer Online Games (MMOGs). Players and NPCs in Zereal currently have a sense-reason-act behavior, with reasoning based on Markov chains in addition to hierarchical plans specified in XML. Zereal's primary purpose is to be a MMOG simulation tool that enables testing of usage logging approaches for CRM and data mining purposes, and secondly to allow flexible testing of behavioral AI models for players and non-personal characters. Zereal has shown to be close to linearly scalable in terms of number of players, and has been successfully tested with more than 100 thousand players on 20 CPUs on a Linux-based cluster. Zereal is implemented using Python, MPI and C++.

1 Introduction

Over the last couple of years an emerging commercial online games market has come into existence, fuelled by the increasing availability of cheap Internet access. The main share of this market consists of *Massively Multiplayer Online Games* (MMOGs). The characteristic property of MMOGs is that a large number of players, sometimes even in the six or seven figure realm, take part in a persistent virtual world where they communicate, cooperate, fight, build virtual characters, and undertake game-related missions [11, 24].

The worldwide MMOG market is anticipated to grow annually by 70% to USD 2.7 billion in year 2006 [12]. Traditionally the game industry has been highly product-focused, but with the introduction of MMOGs games have become more like services, where players not necessarily pay for the MMOG game client, but instead pay a monthly subscription fee for having access to the MMOG world [8, 15]. In order to gain and keep subscribing players there is a need to focus on *Customer Relationship Management* (CRM) and *Data Mining* methods [1, 21]. The overall motivation for such methods in a MMOG context is to figure out more about the players and their preferences so the MMOG can be continously improved and keep on generating revenue.

1.1 Research Problem and Approach

A fundamental requirement for data mining and CRM methods is the need for *relevant usage data*. For MMOGs there are currently no open standards for usage data logging, the data is being logged in a proprietary manner by the various MMOG vendors. This makes it hard to develop data mining and CRM approaches that can used for several MMOGs. So the primary research problem is: **How to enable a scalable and flexible simulation environment for testing out approaches for player usage logging in MMOGs?**.

Since the environment needs to simulate *autonomous MMOG participants* (i.e. players and non-personal characters), we choose to use autonomous intelligent *agents* as the primary abstraction. Due to *several types and numbers of agents* we select the Multi-Agent System (MAS) approach from Distributed Artificial Intelligence as the underlying architecture [17].

The rest of this paper is organized as follows. Section 2 describes and discusses the Zereal MMOG simulation platform, section 3 describes related work, and finally the conclusion.

2 Zereal Platform

The overall goal is to create a MMOG simulation platform that provides a (coarse) simulation of active players that can be used to test various approaches for player usage logging.

2.1 Overall Architectural Choices

Creating a realistic activity level with thousands of concurrent players and NPCs is considered to be of greater importance than having a very accurate simulation of each player's and NPC's behavior. However, the architecture should also provide a flexible interface for testing out various intelligence mechanisms.

Requiring support for thousands of concurrent players and NPCs encourages the selection of a *distributed or parallel architecture*, this combined with the prior selection of agents as the main abstraction encourages use of *mobile agents* for the representation of players and NPCs. Due to the relative simplicity of implementation and deployment of message-based parallel systems we choose a platform of type *message-based parallel mobile agent architecture*. A second motivation for parallel message-based systems is that they can run on most current supercomputers and cluster systems. Parallel cluster systems are frequently used to serve MMOGs, see figure 1 for a typical industry configuration [4, 26, 7].

2.2 Simulation Platform Requirements

The six fundamental requirements for our MMOG simulation platform are:

1. Virtual World Model
2. Item Model
3. Player Models
4. Non-Personal Character Models
5. Scalability
6. Logging of Player behavior

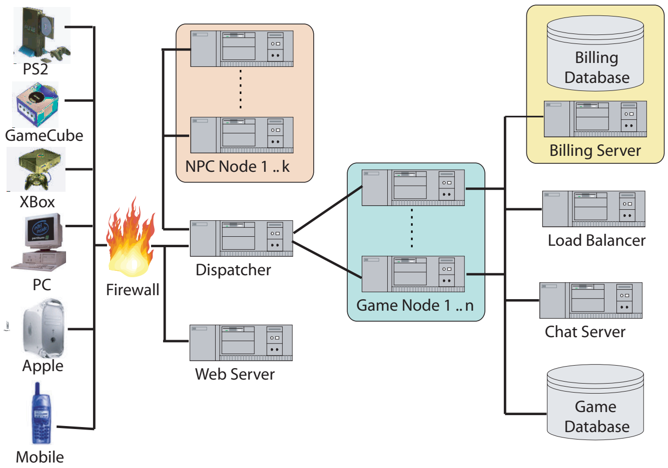

Figure 1: Parallel Cluster Architecture for *an actual* MMOG

2.3 Virtual World Model

Zereal's **MMOG world representation** is a roguelike model, mostly inspired by Nethack (roguelike refers to the game Rogue). Roguelike means that it has simple graphics, typically a 2D environment with a bird's-eye view of the game. Roguelike also usually means that the the game has an enormous variety and amount of features [13], but in our simulation platform we've chosen to provide only some basic features.

The virtual world map is represented as 2D rectangel with borderwalls and doors on the outer limit of the map, and internal walls with doors inside the map to split it up in rooms or smaller outside spaces. Researchers at the Intelligent Computer Entertainment Laboratory at the Ritsumeikan University in Japan has developed a GUI client to Zereal called ZerealViewer, see figure 2 for a screenshot.

Figure 2: Parts of a typical Zereal Virtual World

Each map is a subworld which resides on its own processor in the cluster and the subworlds borders each other after a squared model and send players between them. This world representation is easy to represent in memory, and extension of the world is only limited by memory and power on the clusternodes. Most landscapes can be "made believe" in the maps, and the usually resource demanding task of sensing has less complex data to process. Players and items are represented as ASCII characters in visualization and integer ID's in the internal representation. For scalability reasons it is recommended to map the virtual world into a square (or close to a square) of subworlds, each mapping to one CPU, this ensures a relatively equal load on each CPU (in average), except along the borders (where they have fewer neighbours). A future revision of the platform will enable wrapping between the outmost subworlds (like the game Asteroids).

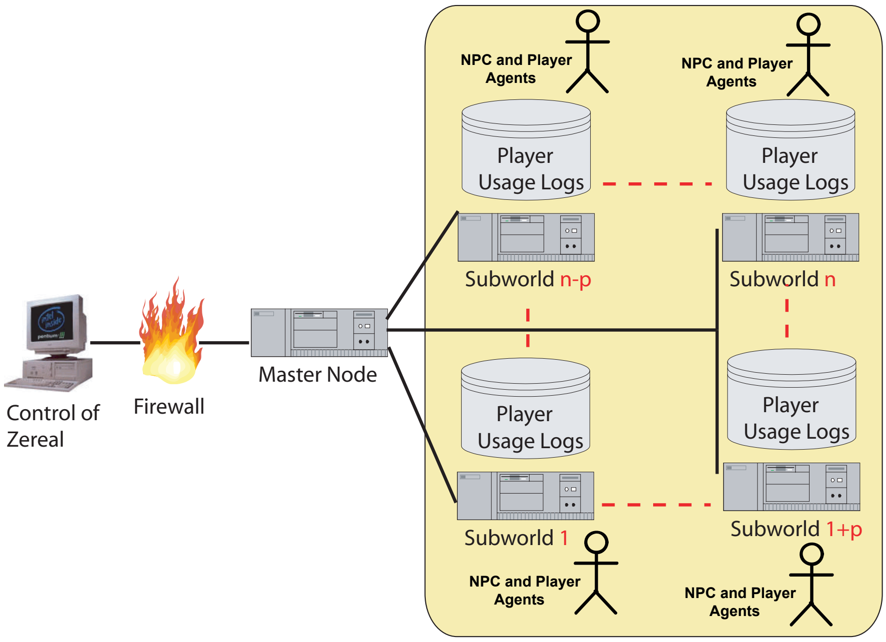

Figure 3: Zereal Macro Architecture

2.4 *Item Models*

Items in Zereal can be of several types: doors, keys, food, potions and weapons. Items occupy one square on the map, and except for doors, all items can be picked up by players. *Doors* transports the agents from room to room or subworld to subworld. If all neighbouring squares on the other side of a door is occupied, agents will not be able to move through them, but in case of a door on the border, agents will be transmitted to the next world but end up in a queue (limbo) untill some of the squares has been freed. Agents may need keys to open locked doors. *Food* and *potions* increases or (if poisonous) decrease the players hitpoints and strength when consumed.

2.5 *Player Models*

Zereal's **player models** are inspired by the typical player categories found in *actual MMOGs*:

1. *Achievers* - people who are entertained by building their avatar character with more skills, hit points, strength etc.

2. *Socializers* - people who wants to build and maintain social relations such as friendship and love, they are typically attracted to people with high intra-game status.

3. *Killers* - people who sees killing as the soul purpose of the game, they typically believe that a player's status depends on how many creatures or other players his/her avatar has killed.

4. *Explorers* - people who roams around the gameworld to discover new places, features, items and non-personal characters.

In general, social status in the MMOG is usually the underlying motivation for all player types in MMOGs [10].

Both **players** and **non-personal characters** are represented as mobile agents with a sense-reason-act approach in Zereal, this is because they need to have autonomous, social, reactive, and pro-active capabilities, hence following the weak properties of agents [25]. The selection of agents as the primary abstraction is also motivated by previous representations of intelligent life-like entities in virtual environments [6, 3] Mobility is selected since the players have to move in a world topology that is spread out on several computational nodes, mobile agents have previously been used in various virtual environment architectures [16].

2.6 Non-Personal Character Models

For our platform we chose the *non-personal characters models* to be equal to player models of the type *killer*, but with minor alterations regarding mobility. The alterations are that NPCs are less mobile than the killers, i.e. they mainly stay within the same subworld at all times.

2.7 Zereal Sense-Reason-Act Approach

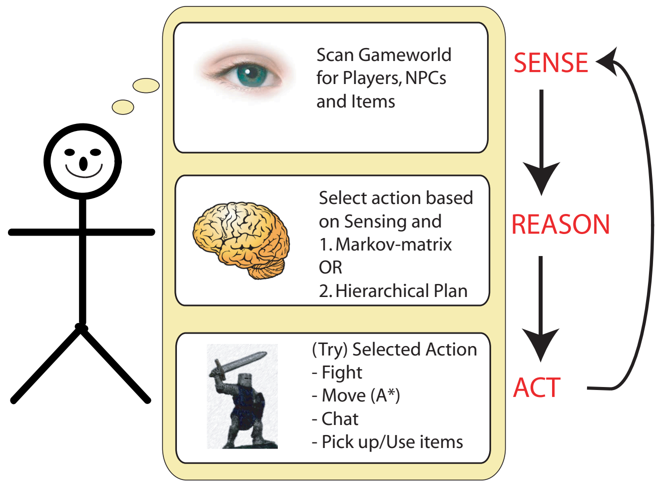

Figure 4: Zereal Agent Architecture

Simulated players and NPCs **senses** using a scan-based vision algorithm that have a configurable vision radius. The vision algorithm can detect walls, items, (other) players and non-personal characters.

There are three types of **reasoning** mechanisms for players and NPCs in Zereal:

1. *Greedy and Agressive* - Both players and NPCs have no social need and will try to pursue the closest player or monster in order to kill it (*Killer* and *Monster* in figure 5).

2. *Action selection using Markov Chain* - The next action is dependent on the previous action and on the possible actions at the current time. E.g. if there are no enemies in sight the 'fight' action in the Markov matrix is pruned and the probabilies for the available actions are normalized. (*MarkovPlayer* in figure 5)

3. *Action selection from hierarchical plan* - The next action is selected from a predefined hierarchical plan (predefined in XML). It also has a long term memory and is able to reason about past experiences, and it is able to use the pathfinding mechanism for determining where to move next. This reasoning mechanism is inspired by the Hap reactive and adaptive architecture used in the Oz project [14]. Plans have so-far been developed for the player types *killer* and *explorer*. (*PlanAgent* in figure 5)

2.8 Scalability of the Zereal Platform

We have used factorial experimental design and identified five (six) factors that have effect on the **scalability** of the simulator. These are: number of players and NPCs, number of CPUs, the vision-radius and size of the world. In addition we have the reasoning factor which is likely to be highly dependent on the reasoning algorithm [22]. By reasoning factor we mean the amount of computational resources spent on AI algorithms for reasoning in the sense-reason-act cycle for each agent.

2.9 Logging of Player Behavior

The main output of Zereal is the log files that are written to disk. The contain player data that can be an important source for data mining in order to find player patterns in MMOGs. Examples of *interesting data mining problems* in MMOGs are:

- How to detect bored or frustrated players?
- How to detect unbalanced features of the game? (E.g. weapons that beat *all* other weapons)
- Which features and data need to be logged to solve the above 2 questions?

The current logging of game events are of 5 types:

1. DEBUG - internal debug messages
2. GAME - game specific events like moves and attacks
3. EVENT - game events regarding receiving and sending agents between subworlds (i.e. CPUs), agent killing and logon/logoff.
4. WARNING - non-critical errors
5. ERROR - severe errors

Logging is done to the local disk of each Subworld's node, and can potentially be joined on the master node for data mining.

Current format of the gamelog file:

```
date||agentID||event||startpos||stoppos||agent type
```

Extracts from a gamelog file:

```
2003-05-22: 12:0:1||6000052||walk||(4,10)||(3,9)||PlanAgent
2003-05-22: 12:0:2||5000018||leaveworld:WEST||(1,2)||(0,3)||PlanAgent
```

Thawonmas et. al used Zereal and its corresponding gamelogs for identification of player types in MMOGs [20].

2.10 Implementation of Zereal

Using Python as the main **implementation** language gives the advantage of high expressiveness and short development time. Regarding performance and memory usage, Python easily competes with Java [18]. Using the stackless implementation of Python gives in adition an extremely fast thread switching, which on most current platforms can reach beyond 1 million switches per second. The heavier tasks like the vision algorithm and A* pathfinding are coded in C/C++ and later glued in to a python module using SWIG. SWIG is a tool for automating the C API coding in order to integrate C/C++ librarys and classes into modules available for languages like Python, Ruby, Perl and Java. PyMPI is used as the interface between the simulator and MPI (Message Passing Interface). The simulator has been tested on parallel clusters with processors based on AMD architecture (at NTNU, Norway) and Intels Xeon architecture (at Ritsumeikan University, Japan).

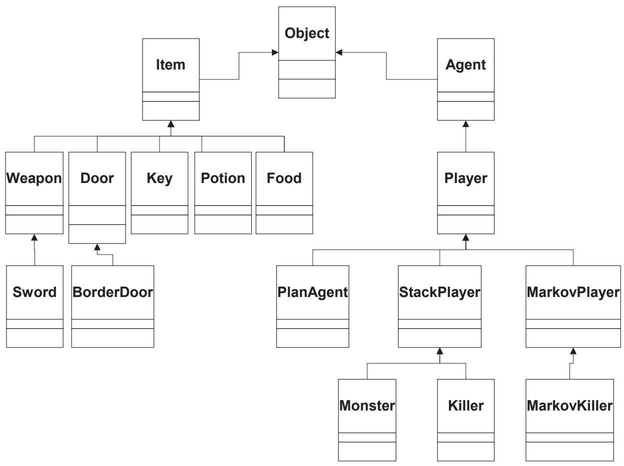

Figure 5: Zereal Class Diagram

3 Related Work

The Creatures software platform simulates agents, represented as virtual pets with humanlike behavior, in a graphical environment [9]. Creatures differs from Zereal by focusing on supporting a few simultaneous agents as opposed to Zereal's support for a very large number of simultaneous player or NPC agents.

Scheutz and Rommer proposed an architecture for interactive believable user agents with personality [19]. Their approach differs from Zereal by focusing on natural language support for human-like agents and not on massive scalability in terms of number of agents.

The SimHuman platform resembles Zereal since it simulates human-like agents with planning capabilities [23], but it differs from Zereal since it focuses on a small number of simultaneous agents and a 3D graphical presentation.

Swarm is a toolkit for large scale agent-based simulation, it differs from Zereal by being generic and not geared toward simulation of MMOGs such as Zereal. Another difference is the language support, Swarm supports Objective C, Scheme and Java; Zereal supports Python and C/C++ but is easily extensible to many other languages using the Simplified Wrapper Interface Generator (SWIG).

Python for Massively Multiplayer Virtual Worlds (PMMVW) resembles Zereal in the sense that it provides support for a MMOG and is Python-based, the main difference is that Zereal is focused on simulation of players and PMMVW supports actual humans who play [2].

In addition can various robotic simulations and software for large-scale multiuser animations be considered related, but not equal to the Zereal approach.

4 Empirical Results

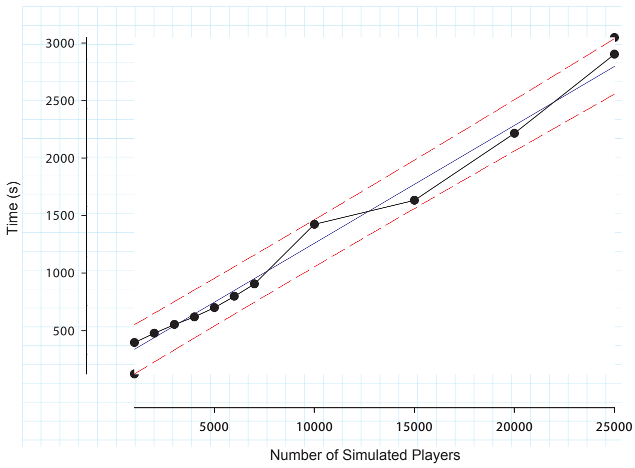

Figure 6: Zereal scalability (with regression line and prediction interval)

In figure 1 the wallclock runtime performance for 5 minutes simulated time in the game (i.e. 300 simulated seconds) as a function of the number of simulated players. The simulation is performed on 5 Athlon 1.6 GHz CPUs on an Linux-cluster. The largest simulation performed so far with Zereal is with 160 thousand agents (simulated players and NPCs) on 20 CPUs. The setup for the experiment was with simulated players and NPCs with Markov chain type reasoning (the largest number of hierarchical planning agents tested so far is 50000). By doubling the number of players and keeping the number of simulated cycles fixed the number of sense-reason-act cycles to be performed doubles, this can explain the close-to-linear scalability.

5 Conclusion and Future Work

We have presented the Zereal Mobile Agent-based Massively Multiplayer Online Game Simulation platform. The implementation has been shown to be close to lineary scalable in terms of number of players simulated. Primary contribution is the platform itself and its scalable implemention that can be used as a testbed for research on MMOGs, in addition to be a MMOG simulation platform Zereal can with minor adaptions provide a generic scalable agent-platform, based on MPI communication it is relatively simple to use on various parallel supercomputers or grid-based environments.

Possible future extensions include: items that can tell the players how they are supposed to be used, player coalition support, guilds, quests, improved intelligence support (e.g. BDI, automatic planning and emotion engine), simulation of natural language based interaction between players and/or NPCs, simulation of e-commerce activities in a MMOG [5] and improved graphical interface support using the VR-environment provided by Silicon Graphics' RAVE.

References

[1] Henrik Andersen. *The CRM Handbook - from group to multi-individual*. PricewaterHouseCoopers, 2000.

[2] Jason Asbahr. Python for Massively Multiplayer Virtual Worlds. O'Reilly Open Source Convention, published Online, July 2001.

[3] Ruth Aylett and Michael Luck. Applying Artificial Intelligence to Virtual Reality: Intelligent Virtual Environments. *Applied Artificial Intelligence*, 14(1):3–32, January 2000.

[4] Butterfly.net: Powering Next-Generation Gaming with On-Demand Computing. Published Online by IBM: `ftp://ftp.software.ibm.com/solutions/pdfs/g325-1938-00.pdf`, January 2003.

[5] Nizami Cummins. Integrating E-Commerce and Games. *Journal of Personal and Ubiquitous Computing*, 6(5-6):362–370, December 2002.

[6] Sorabain Wolfheart de Lioncourt and Michael Luck. Motivating intelligent agents for virtual environments. In *Proceedings of the 2nd International Workshop on Intelligent Virtual Agents (IVA 1999)*, 1999.

[7] Anne C. Elster, Otto Anshus, Amund Tveit, and Cyril Banino. Recent trends in cluster computing. In *Proceedings of the International Conference on Parallel Computing (forthcoming)*. Elsevier Science, September 2003.

[8] Markus Friedl. *Online Game Interactivity Theory*. Charles River Media, 1 edition, October 2002.

[9] Stephen Grand, Dave Cliff, and Anil Malhotra. Creatures: Artificial Life Autonomous Software Agents for Home Entertainment. In *Proceedings of the 1st International Conference on Autonomous Agents*. ACM, ACM Press, 1997.

[10] Neal Hallford and Jana Hallford. *Swords & Circuitry: A Designer's Guide to Computer Role-Playing Games*. Prima Publishing, 1 edition, 2001.

[11] Moon Ihlwan. The champs in online games. *Business Week*, (30):27–28, July 2001.

[12] Zona Inc. and Executive Summary Consulting Inc. State of Massive Multiplayer Online Games 2002: A New World in Electronic Gaming. Technical report, Zona Inc., Redwood City, California, USA, October 2002.

[13] Petri Kuittinen. Introduction to Roguelike Games. Published Online: `http://www.hut.fi/~eye/roguelike/intro.html`, May 2000.

[14] A. Bryan Loyall and Joseph Bates. Hap: A Reactive, Adaptive Architecture for Agents. Technical Report CMU-CS-91-147, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, June 1991.

[15] Ian Maccines, Janusz Moneta, Julio Caraballo, and Dominic Sarni. Business Models for Mobile Content: The Case of M-Games. *The International Journal of Electronic Commerce & Business Media: Special Section on Electronic Commerce in Entertainment and Media*, 12(4):218–227, 2002.

[16] Gonzalo Mndez, Pedro Prez, and Anglica de Antonio. An Overview of hte Use of Mobile Agents in Virtual Environments. In Anglica de Antonio, Ruth Aylett, and Daniell Ballin, editors, *Proceedings of the 3rd International Workshop on Intelligent Virtual Agents (IVA 2001)*, number 2190 in Lecture Notes in Artificial Intelligence, pages 126–136. Springer-Verlag, 2001.

[17] Bernhard Moulin and Brahim Chaib Draa. An Overview of Distributed Artificial Intelligence. In Greg M. P. O'Hare and Nicholas R. Jennings, editors, *Fundamentals of Distributed Artificial Intelligence*, pages 3–56. John Wiley and Sons, 1996.

[18] Lutz Prechelt. An Empirical Comparison of Seven Programming Languages. *IEEE Computer*, 33(10):23–29, October 2000.

[19] Matthias Scheutz and Brigitte Rommer. Autonomous Avatars? From Users to Agents and Back. In Anglica de Antonio, Ruth Aylett, and Daniell Ballin, editors, *Proceedings of the 3rd International Workshop on Intelligent Virtual Agents (IVA 2001)*, number 2190 in Lecture Notes in Artificial Intelligence, pages 61–71. Springer-Verlag, 2001.

[20] Ruck Thawonmas, Ji-Young Ho, and Yoshitaka Matsumoto. Identification of Player Types in Massively Multiplayer Online Games. In *Proceedings the 34th Annual Conference of the International Simulation and Gaming Association (ISAGA)*. ISAGA, To be published by Springer-Verlag, 2003.

[21] Amund Tveit. Game Usage Mining: Information Gathering for Knowledge Discovery in Massively Multiplayer Games. In Hamid R. Arabnia and Youngson Mun, editors, *Proceedings of the International Conference on Internet Computing (IC'2002), session on Web Mining*, volume III, pages 636–642. CSREA Press, June 2002.

[22] Amund Tveit. Scalability Analysis of the Zereal Massively Multiplayer Game Simulator. Technical Report ISSN: 0802-6394, 12/02, IDI, NTNU, December 2002.

[23] Spyros Vosinakis and Themis Panayiotopoulos. SimHuman: A Platform for Real-Time Virtual Agents with Planning Capabilities. In Anglica de Antonio, Ruth Aylett, and Daniell Ballin, editors, *Proceedings of the 3rd International Workshop on Intelligent Virtual Agents (IVA 2001)*, number 2190 in Lecture Notes in Artificial Intelligence, pages 210–224. Springer-Verlag, 2001.

[24] Rusel Demaria & Johnny L. Wilson. *High Score! The Illustrated History of Electronic Games*. McGraw-Hill/Osborne, 1 edition, 2002.

[25] Michael J. Wooldridge and Nicholas R. Jennings. Intelligent Agents: Theory and Practice. *The Knowledge Engineering Review*, 2(10):115–152, 1995.

[26] Terazona White Paper. Available from Zona Inc., May 2003.