Network modeling and simulation of massively multiplayer online games

Selçuk Bozcan and Veysi Isler

SIMULATION 2012 88: 908 originally published online 19 October 2011
DOI: 10.1177/0037549711423283

Simulation

Simulation: Transactions of the Society for Modeling and Simulation International
88(8) 908–920

DOI: 10.1177/0037549711423283
sim.sagepub.com

Network modeling and simulation of massively multiplayer online games

Selçuk Bozcan and Veysi İşler

Abstract

Massively multiplayer online games (MMOGs) have become highly popular in the last decade and now attract millions of users from all over the world to play in an evolving virtual world concurrently over the Internet. The high popularity of MMOGs created a rapidly growing market and this highly dynamic market has forced game developers to step up competitively. However, MMOG development is a challenging and expensive process. In this study, we present a high-level network simulation environment, which can be used to model and simulate typical MMOGs that have client–server architectures. The main objective is to provide a simulation environment to MMOG developers that can be used to test, analyze and verify various aspects of the MMOG network architecture. We have also implemented a graphical user interface that allows one to construct the simulation model visually. We have demonstrated the use of the simulation environment by experimental simulations.

Keywords

massively multiplayer online games, massively multiplayer online role playing games, network simulation

1. Introduction

MMOGs (massively multiplayer online games), sometimes referred as MMORPGs (massively multiplayer online role playing games), are types of multiplayer computer games that are played by a massive amount of concurrent players online over the Internet. With the Internet boom in the mid to late 1990s, MMOGs have rapidly gained in popularity.[1,2] Currently the worldwide online gaming market is dominated by MMOGs and, moreover, in five years time the market revenue of MMOGs is expected to be doubled, according to the DFC Intelligence online gaming forecast report.[3] This rapidly growing and highly dynamic market signifies the importance of MMOGs and continuously attracts game developers from all over the world. However, MMOG development is not an easy task and it is significantly more difficult and costs more than other multiplayer games.[4]

One of the most important and challenging part of the MMOG development process is the design of the network architecture of the game. At any time, thousands of players can be online in a typical MMOG. Designing a network architecture that will utilize several real-time online servers in an efficient and reliable way, creating a smooth game play for a massive number of players, is by no means an easy task.

In this study, we have developed a network simulation tool for MMOGs that can be used to model and simulate MMOG networks. Network simulation is used for various aspects of research and development of networks, including protocol analysis and verification, evaluating the network architectures and testing network scenarios.[5] Use of a network simulation tool can significantly help the developer improve the quality of the network architecture and decrease the development time.

We believe that using a network simulation tool can substantially help the development process of a MMOG. For example, the game network protocol can be analyzed and verified in the early stages of the development process, long before the deployment of the game. Early testing of the network protocol would improve the quality and decrease the cost of deployment.

The network simulation tool developed in this study is based on the general-purpose network simulation environment GTNetS (Georgia Tech Network Simulator).[6] We aimed to develop an easy-to-use network simulation

Department of Computer Engineering, Middle East Technical University, Ankara, Turkey

Corresponding author:
Veysi İşler, Department of Computer Engineering, Middle East Technical University, 06800 Ankara, Turkey.
Email: isler@ceng.metu.edu.tr

environment, which is capable of modeling and simulating typical client–server-based MMOG networks.

2. Massively multiplayer online games

2.1. Background and characteristics

The most important characteristic of MMOGs, which distinguishes them from other multiplayer computer games, is the amount of simultaneous players. Multiplayer computer games, which are played over local area networks (LANs) generally, have between 10 and 40 simultaneous players. In the case of MMOGs, the number of concurrent players online can be as high as several hundreds of thousands. For example, World of Warcraft,[7] which is one of the most popular MMOGs,[1,2] has achieved one million concurrent players in China on 11 April 2008.

The virtual environments presented in the MMOGs are usually huge in size due to the massive amount of concurrent players. Moreover, only a few of the game characters are played by computer. The majority of the characters in the virtual environment are played by the humans. This important characteristic of the MMOG virtual environments leads to high interaction between the players. Such high interaction among the players creates a strong social networking environment, as in Second Life,[8] which is not available in any other kind of computer game.

In addition to the high level of interaction, one of the most important characteristics of MMOG virtual environments is persistency. Unlike other computer games, the virtual environments presented in MMOGs are persistent. In other words, they continue to exist and evolve even if no players are connected to the game. The persistency of the virtual environments presented in MMOGs is so characteristic, that sometimes MMOGs are referred as MMOPWs (massively multiplayer online persistent worlds).

2.2. Architecture of MMOGs

Nearly all of the existing commercial MMOGs have client–server architecture.[9] This is because it is easier to maintain game consistency, persistency, security and administrative control in client–server architectures when compared to other possible architectures.[10] The MMOG client builds up the user interface (UI) of the game and runs on the player's computer. Usually, the MMOG client is responsible for maintaining the communication with the MMOG server and performing audio-visual rendering. On the other hand, the MMOG server is responsible for everything else related with the game, such as receiving client responses, evaluating client actions within the game logic and sending the results back to the clients.

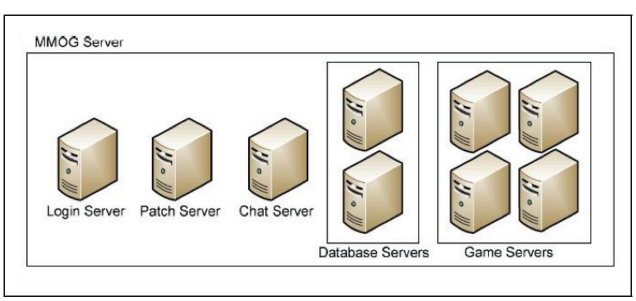

Figure 1. Example MMOG Server Model.

To be able to deal with a massive number of concurrent clients, different approaches and different server configurations have been developed and applied. Usually in MMOGs, the server responsibilities are distributed among a group of machines rather than using a single machine to handle all the work.[9] There are many possibilities for distributing the responsibilities of the servers. One of the possible configurations is shown in Figure 1.

In this sample server configuration, the server responsibilities split into group as follows. The login server is responsible for authentication of the clients. If the client succeeds the authentication process, it can be forwarded to the patch server. The patch server is responsible for ensuring the up-to-dateness of the client software. If the client is not running the latest version, the patch server can send updates to the client. The chat server is responsible for handling the chat messages that the clients send. In this model, the game logic is handled by a group of servers. All game-specific calculations are carried on the game servers. Different approaches are possible for distributing the workload among the game servers. Each server in the game server group can be statically assigned to a geographic location in the virtual environment or the game server can be dynamically configured to share the load according to different metrics. Database servers store the game data, client avatar data and other configuration data related with the game. This server group, referred as the MMOG server farm, acts as a single server in the MMOG network model. MMOGs may have multiple copies of this server farm. For example, World of Warcraft has North America, Europe and Asia servers.[7] These servers are totally independent from each other and each of them runs a copy of the game.

Centralized architectures are claimed to be unscalable by some researchers.[9] However, some of the existing implementations of MMOGs with client–server architecture have shown sufficient scalability. MMOGs, such as World of Warcraft, EverQuest and Lineage, all have a centralized architecture, yet still support hundreds of thousands of users online concurrently. Nevertheless, it is still true that a decentralized peer-to-peer (P2P) architecture can lower the operational costs by making use of the computational power and bandwidth of the clients and improve the performance of the MMOG by reducing the latencies. For this reason, there have been a number of proposals on P2P architectures for MMOGs.[11,12] There are also some hybrid architectures proposed.[13] However, these P2P and hybrid proposals are relatively new and unfortunately are limited to academic studies. To the best of our knowledge, there is no currently available P2P-based commercial MMOG. However, Outback Online,[14] which is a MMOG currently being developed by Yoick, may be the first commercial P2P-based MMOG.

2.3. MMOG development challenges

MMOGs are significantly more difficult and expensive to develop than traditional video games.[4] Development challenges arise from many points. First of all, designing a virtual world at massive scales, and acquiring seamlessness of this world, is itself a great challenge. A virtual world at this size, together with the data produced by thousands of players, brings a massive amount of data to deal with. Managing this data effectively on a very large and heterogeneous network is another challenge. In addition, a MMOG should maintain a smooth game play no matter how many players are online. It should be clear why developing a MMOG is extremely difficult when all these challenges are summed up. In fact, even big companies can fail to create a successful MMOG. For example, Earth & Beyond, which was a science fiction MMOG developed by Electronic Arts & Westwood Studios over five years, was released in September 2002 and shut down in September 2004.[4]

Operational and maintenance costs can be even higher than development costs. The cost of maintaining a huge bandwidth for the operation of thousands of simultaneous player connections makes up the largest share of operational and maintenance costs.[4]

Because of the high development cost of MMOGs, it would be a better approach to measure the game performance during the early steps of the development period. The performance of the important components of a MMOG, such as the network architecture, network protocol and interest management algorithm, should be measured during the early stages of the development period in order to prevent late changes in those components.

3. Network simulation

In MMOGs the proper design of the network architecture design is critical, since the success of a MMOG is heavily dependent on efficient and reliable network operation. However, because of the massiveness of MMOGs, the network architectures can be very complex. Designing a properly operating heterogeneous game network over the Internet is a challenge that MMOG developers face.

Therefore, network simulation itself is an important part of this study, where we have extended a network simulation environment to model and simulate MMOG network architecture. In this section, an introduction to network simulation is given. This section also contains a survey of some of the commonly used Network Simulators. Finally, we present the rationale for choosing the GTNetS as our simulation environment.

3.1. Introduction

As the complexity of the networks increases, their design and analysis gets more challenging. The complexity of MMOG network architecture has made the use of simulation methods necessary for analyzing the game network performance. Various aspects of the game architecture can be analyzed, tested and verified by using the network simulation method. Some of these aspects are listed below.

- Game Network Protocol: the performance and the suitability of the network protocol used by the game can be analyzed and verified by using network simulation methods.
- Distributed Architecture for MMOG servers: network simulation methods can be used to justify the game server architecture. Questions like ‘How is the performance of the system affected when a single server used?’, ‘How is the performance of the system affected when a multiple server configuration instead of a single server is used?’ or ‘If we increase the server number, how will be the latency values in the game network will be affected?’ can be answered by using network simulation tool.
- Performance of the Area of Interest (AOI) algorithm: various AOI (a.k.a. Interest Management Algorithm) algorithms are used by MMOGs to decrease the overall game network traffic.[15] The performance of different AOI algorithms can be measured and improved by using a network simulation tool. The effect of changing parameters of the AOI algorithm on the game network traffic can also be observed. Therefore, it is possible to determine an optimal algorithm with optimal parameters for the MMOG.
- Virtual Map Size: MMOGs (especially MMORPGs) usually divide the virtual environment into maps to distribute the client load among servers. Network simulation methods can help to determine an optimal virtual map size that will cause minimum server latency values.

3.2. Network simulators

Extensive research on network simulators leads us to the GTNetS, Network Simulator 2 (NS-2) and Optimized Network Engineering Tools (OPNet). In this section, we will give a survey of them.

The GTNetS[6] is a full featured network simulation environment that allows researchers interested in computer networks to study the behavior of moderate to large-scale networks, under a variety of conditions.[16] The GTNetS has support for a large number of applications and protocols, including IEEE 802.3, IEEE 802.11 and Bluetooth.

The GTNetS also supports distributed simulations using a federated simulation approach.[16] A single simulation can be distributed either on a network of loosely coupled workstations, a shared memory symmetric multi-processor system or a combination of both. The GTNetS uses the dynamic NIx-Vector routing method to reduce the memory footprint of the simulation. Parallelization ability and the NIx-Vector routing method grants the GTNetS a high scalability. Being a general-purpose network simulator, the GTNetS is also used in sensor network simulations as well as Internet simulations.[17]

The NS-2 is a widely used discrete event simulator.[5,18] It is developed and maintained by the Information Science Institute (ISI) at the University of Southern California and is supported by the Defense Advanced Research Projects Agency (DARPA) and the National Science Foundation (NSF). The NS-2 has considerable support for Transmission Control Protocol (TCP), routing and multicast protocols over both wired and wireless (local and satellite) networks.[18,19] A large number of extensions for the NS-2 can be found on the Internet.

OPNet[20] is a commercial discrete event network simulation environment. It supports analytical and hybrid simulation methods. Moreover, it supports an extensive list of protocols and enables its users to collect detailed statistics regarding the simulation. In addition, it features a convenient UI, which allows users to design, debug and analyze the simulation topology graphically.

The GTNetS seemed to be a reasonable choice for the simulation environment to be used in this study. First of all, the low memory footprint of the GTNetS[6] was important for us, since it is possible to model larger networks with a low memory-consuming network simulator. Furthermore, the extendibility of the network simulation environment is also important for us. We believed that extending the GTNetS would be easy because of the open source license and the object-oriented design. Besides, the GTNetS can be directly compiled and used on both Windows- and UNIX-based operating systems, which would eventually give us more freedom to select the study environment.

4. Related work

The high popularity of MMOGs has attracted many researchers from different disciplines to study various aspects of MMOGs. In this section, we will discuss some of the related work about MMOG network architecture and simulation of MMOGs.

Chen et al.[21] analyzed a 1356-million-packet trace from a MMORPG called ShenZou Online in their study. This research is the first formal analysis of MMORPG server traces. In their study, they have identified important characteristics of MMORPG traffic. They have listed the important properties of MMORPG traffic as tiny packets, strong periodicity, temporal dependence of packet arrivals within the connections and aggregate traffic, irregularity, self-similarity and heavy-tailed session duration. In addition, they have offered explanations for these characteristics of MMORPG traffic. They have stated that the periodicity is due to a common network game design pattern that tends to send out periodic state updates, such as position changes. The temporal locality in game traffic is due to the game's design. The irregularity is due to the diversity and the huge size of the game's virtual environment. The self-similarity of the aggregate traffic is due to the heavy-tailed active/idle activities of individual players. Moreover, they have investigated the suitability of the TCP for MMOGs and found out that the TCP produces a significant overhead because of TCP/Internet Protocol (IP) headers and acknowledgements. They have argued that use of the TCP for MMORPGs can be considered as overkill.

Another similar study about traffic analysis of MMOGs has been done by Svoboda et al.[22] In their study, they have analyzed network traffic of the popular MMOG World of Warcraft.[7] Throughout their analysis, they have extracted parameters that represent the characteristics of the traffic. They have used these parameters to implement a simulation model using the network simulator NS-2.

Jung et al.[23] have proposed a system that provides an automated beta test environment to efficiently test online games. They have called their system VENUS (Virtual Environment Network User Simulator). In the VENUS system, they have implemented virtual clients, which can communicate with the multiplayer game server. The virtual clients, which are controlled from a central station, are implemented in a way that the game server cannot distinguish them from actual clients. The user can control the VENUS system from this central station. The central engineering station (CES) sends control commands to the Virtual Controllers (VCs) and the VCs send game messages to the game server. The game server is monitored by the VENUS system and the statistics regarding its performance are collected. Figure 2 shows the block diagram of the VENUS system.

The VENUS system tries to create an automated beta test environment to test multiplayer games. The virtual clients in the VENUS system connect to the actual game server and try to mimic the behaviors of real clients according to the given commands. In our study, however, we developed a network simulation environment, which can be used to model and simulate the MMOG networks. The VENUS system is developed to be used in the beta testing phase of the development. In contrast, a network simulation tool can be used at the very early stages of the development to evaluate the performance of a specific AOI algorithm or the performance of a specific game network protocol.

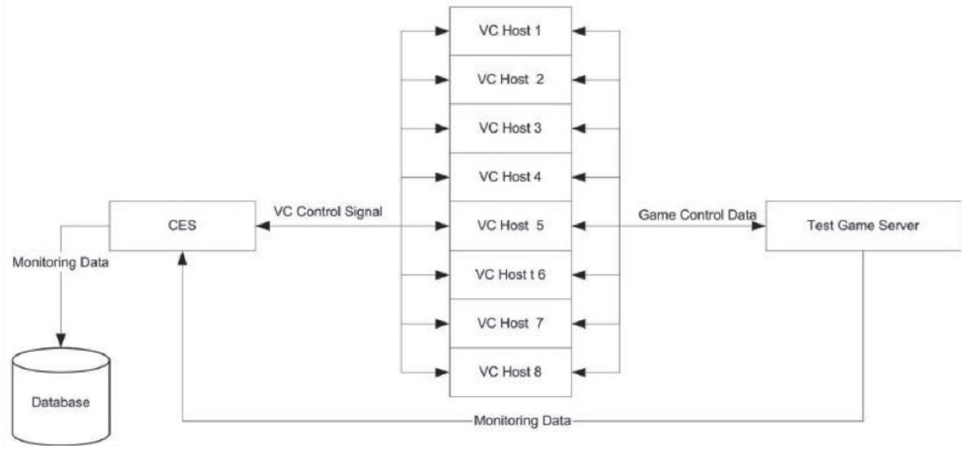

Figure 2. Block diagram of the VENUS system [22].

5. A tool for network simulation of massively multiplayer online games

5.1. Requirements

The aim of this study is to develop a network simulation environment that can be used to model and simulate typical MMOG networks that have client and server architecture.

The network simulation tool for the MMOGs should have good scalability. To be able to simulate a MMOG network, the simulation environment should be capable of simulating networks with at least several thousand nodes.

The network simulation tool should enable its users to easily define custom game messages with variable sizes and frequencies. There is no standard for MMOG network protocol. Every MMOG uses its own custom-defined message set for server and client communication. Therefore, allowing its users to define their own custom messages is a required feature.

A MMOG network simulation tool should provide mechanisms for defining AOI algorithms. AOI algorithms are utilized by the MMOG servers to decrease the network traffic on the game network. Since this algorithm significantly affects the network traffic, it is an important feature of network simulation tool for MMOGs. The tool should support standard TCP. Since nearly all of the commercially available MMOGs use TCP as the transport protocol,[21,22] having TCP support is a required feature of a network simulation tool for MMOGs.

5.2. Architecture of the tool

The MMOG network simulation tool developed in this study is based on the GTNetS[6] network simulation environment. We have extended and added new functionalities to the GTNetS to support simulation of MMOG networks. Furthermore, we have implemented a modeler tool that allows visual modeling. The visual modeling tool that we have developed generates a script file, which represents the simulation model. Then, this script file is taken as input by our simulation environment and the corresponding simulation model is created. For this functionality, we have implemented an extendible parser module for the network simulation environment. After creating the model in the simulation environment and running the simulation, the simulation environment reports the results to the user. The overall architecture of the MMOG simulation tool is given in Figure 3.

We have called the network simulation environment GTNetS-MMOG to emphasize that it is a network simulation environment based on the GTNetS and has specific features to model and simulate MMOG networks.

5.2.1. GTNetS and MMOG extensions. Although the GTNetS has comprehensive support for various network protocols (i.e. TCP, User Datagram Protocol (UDP), Hypertext Transfer Protocol (HTTP)) and application types (i.e. File Transfer Protocol (FTP), Web Browser, Internet Worm), it lacks the required features to model a MMOG network. However, the GTNetS is open source project and provides an extensible architecture. Therefore, we have extended the GTNetS and added new message types, application types and other features to simulate MMOG networks.

5.2.1.1. Contributions to GTNetS code. During the time of our study, we made extensive use of the GTNetS and conducted various simulation experiments as follows.

5.2.1.2. MMOG extensions to the GTNetS. In order to model MMOG networks, we added new applications and data messages to the GTNetS. The extensions that we have

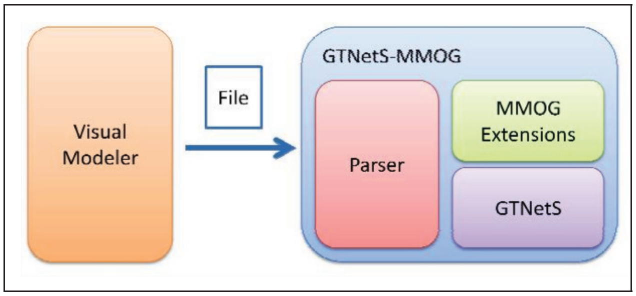

Figure 3. Architecture of MMOG Simulation Tool.

provided can be used to simulate a generic MMOG server and a generic MMOG client with custom game messages. The structure of the extensions provided is described in the Figure 4.

We have implemented two application objects in the GTNetS, which are the MMOG server and the MMOG client. Our MMOG server and MMOG client implementations use TCP for communication. In addition, we have implemented a custom data packet, which allows one to define custom message structures that will be used in client–server communication. GTNetS-MMOG extensions have been implemented using C++ programming language and the MS Visual Studio 2005 Interactive Development Environment (IDE). The detailed descriptions of the MMOG server and MMOG client, as well as MMOG Data Packet, are given below.

- MMOG server: allows incoming connections from the MMOG client application with a configurable port. As the clients connect, MMOG server gives unique IDs to the clients and stores them in a structure for later references.

  MMOG server can also be configured to operate with respect to an AOI algorithm. We implemented a simple Euclidean distance AOI algorithm, yet different types of interest management algorithms can easily be added as needed.

  In Figure 5, the AOI of the red colored player is shown. The small circles in the figure represent the players in the virtual environment. The AOI is a circle around the player with a radius that covers the maximum distance a player can see. If the distance between two players is smaller than the radius of the AOI, then the updates of players are sent to each other.

  Otherwise, the player updates are filtered. AOI algorithms increase the network performance by decreasing the number of messages on the network.

  In our tool, when a MMOG client sends a message that is marked as 'distribute to other clients with respect to AOI', the MMOG server creates copies of this message and sends to the MMOG clients within the AOI of the sender client.

- MMOG client: connects to a MMOG server and sends game messages with respect to defined parameters. For the AOI algorithm, we have also implemented an extendable mobility model for the MMOG client application. The mobility model that we have implemented simulates the movement of a player in the virtual environment by generating random waypoints in a defined virtual area. Other models can be easily implemented and integrated through our interface. As the player moves in the virtual environment, the MMOG client sends messages to the MMOG server reporting its position. The MMOG server in the AOI calculations uses this position information. The position messages of the MMOG client are not mutable. However, the size and the update frequency of the message can be adjusted similar to the custom game messages.

  We have implemented a generic message-sending feature in the MMOG client. The size and sending frequency of the custom messages can be adjusted as needed. Rather than using only constant parameters to define the size and the frequency of the game messages, we have implemented a flexible structure that enables use of random distributions. We have implemented support for following distributions: uniform, exponential, Pareto, Weibull.

- MMOG Data Packet: to enable using of data messages that can include custom message structures, we have implemented a MMOG Data Packet class, which inherits from standard data packet in the GTNetS. The MMOG client and MMOG server communicate with using this message structure. The structure of the MMOG Data Packet is given in Figure 5.

  The header of the message includes a specific signature, which can be used to recognize a MMOG Data Packet. The ID field contains the unique ID of the

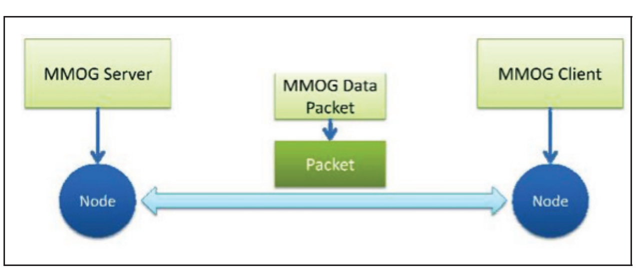

Figure 4. GTNetS MMOG Extensions.

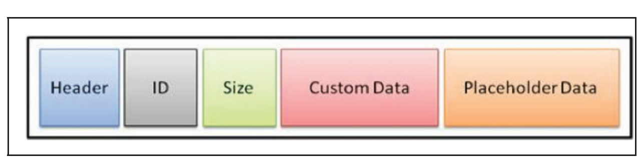

Figure 5. MMOG Data Packet Structure.

message. The Size field contains the size of the actual data in the message. The Custom Data field contains the user-defined custom data structure. The user can define a larger size for the MMOG Data Packet than is actually required for the user-defined data. In that case, Placeholder Data is added to the end of the data packet.

5.2.2. Visual modeler. In the GTNetS, network simulations are written in C++ programming language. In small scale, C++ can be used with no problem at all. However, as the complexity and the size of the simulation model increases, the size of the C++ code required to generate the simulation model also increases. Therefore, we have implemented a Visual Modeler tool to allow the user to easily create and model MMOG networks without writing C++ code. The Visual Modeler tool has been implemented with C# programming language and Microsoft Visual Studio 2005 IDE.

With the Visual Modeler tool, the user can define the network topology by creating network nodes and links. The network nodes are represented with squares in the visual model. A network node can be associated with either the MMOG server or MMOG client applications. Alternatively, a network node can be created with no associated application at all. In that case, the created node will behave as a router in the MMOG network. Nodes with the MMOG server application associated are represented with red squares, whereas nodes with MMOG client application are represented with blue squares. In addition, router nodes are represented with black squares. The user can define the node IP and the application, which will run on the node during the node creation process.

The design of large-scale networks can be time consuming and challenging even with using the Visual Modeler tool. For example, while designing a network with 1000 network nodes, the user should add each node and create links between them manually. Therefore, to ease this task, we have added a feature that allows a user to define a node group with a single operation. Creation of a node group is similar to the creation of a node. While creating a node group, the starting IP of the node group and the number of nodes in the group are defined. The IPs of the nodes in the node group are assigned sequentially starting with the given IP value. In addition, nodes in the node group can be associated with the MMOG client application. The node groups in the visual model are represented with larger squares.

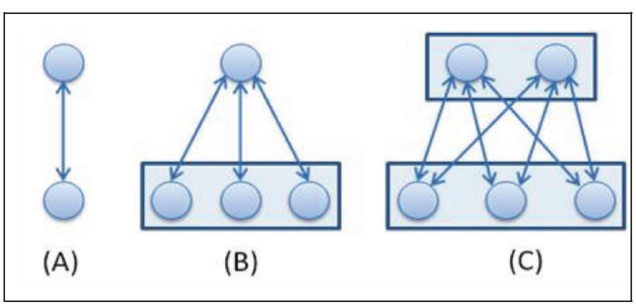

Figure 6. MMOG Data Packet Structure.

In the Visual Modeler tool, the network links between the nodes can be created easily by simply selecting two nodes with the mouse. The links are defined with two parameters: the bandwidth and the link delay. In addition, a color can be assigned with the link, which makes the visual model more understandable. Currently, the Visual Modeler tool only allows creation of full duplex links.

Creation of a link between two nodes is straightforward: Figure 6(a) shows a duplex link created between two nodes. In cases where one end of the link is a node group, a duplex link is created between the node and each node in the node group (see Figure 6(b)). In cases where both ends of the link are a node group, duplex links are created between each node in the first node group and each node in the second group (see Figure 6(c)). There are no links between the nodes in a node group.

The Visual Modeler tool allows one to define custom game messages. A game message can be defined with two parameters, namely message size and message-sending frequency. For both parameters, a random distribution or a constant value can be selected. Selectable random distributions are uniform, exponential, Pareto and Weibull. In addition to that, parameters of the position message of the MMOG client can also be adjusted.

The saving/loading of a simulation model is also supported in the Visual Modeler tool. We have decided to use Extensible Markup Language (XML) formatted files for storing the simulation models instead of using binary files. By using XML files, saved simulation model files can be modified by using simple text editors.

The Visual Modeler tool generates a script file, which includes the definition of the simulation model. This script file can be parsed with the parser module of the GTNetS-MMOG.

5.2.3. The parser. We have implemented a parser module for the GTNetS-MMOG. The parser module parses GTNetS-MMOG scripts and creates the simulation environment.

We have used the Spirit parser framework, which is a part of Boost C++ Libraries. Spirit is an object-oriented recursive descent parser framework implemented using template meta-programming techniques. Spirit tries to mimic the syntax of Extended Backus Normal Form (EBNF) completely in C++ using operator overloading and template programming. Spirit parsers are backtracking and top down, which are capable of parsing rather ambiguous grammars efficiently.

6. Experiments and results

In order to observe the performance and the usability of the GTNetS-MMOG network simulation tool and the Visual Modeler tool, we have conducted network simulation experiments. Our experiments are focused on different aspects of the MMOG network architecture. This section describes the network simulation experiments that we have conducted and comments on the results.

6.1. Simulation environment

The network simulations described in this section were carried on an x86 personal computer (PC) running Intel Pentium at 3.2 GHz with 3 GB of memory. The operating system of the simulation computer was Windows XP Professional with SP2 installed. GTNetS-MMOG and the Visual Modeler tools were compiled for the Windows XP operating system with using Microsoft Visual Studio 2005 development environment. Throughout the network simulations, no tasks other than the critical system tasks were allowed to run.

6.2. AOI algorithm performance evaluation

The effect of the AOI algorithm's performance on the MMOG network traffic is significant. In Experiment 1, we used our GTNetS-MMOG network simulation tool to measure the effect of the interest circle's radius in a Euclidean distance AOI algorithm on the MMOG network traffic.

6.2.1. Simulation model. In Experiment 1, we modeled a MMOG network consisting of 6000 MMOG clients, a single MMOG server and 18 routers. The network topology in Experiment 1 is inspired by the Campus Network topology.[24] We have used the Visual Modeler tool to model the network. Figure 7 shows the network topology in the Visual Modeler tool.

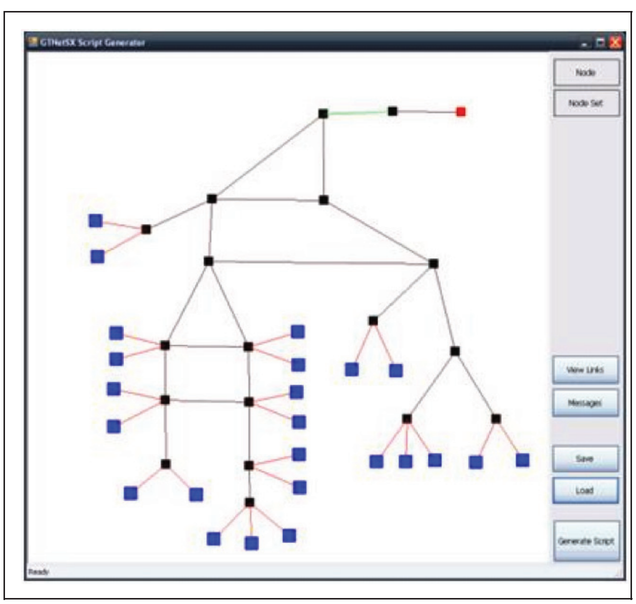

Figure 7. 6000 Client MMOG Network (color online only).

In Figure 7, the blue squares represent the node groups. Each node group consists of 250 MMOG clients. The black squares represent the routers and the red square represents the MMOG server. The MMOG clients are connected to the routers with links that have 1 Mbps of bandwidth and 2 ms of delay (shown with red lines). Except for two routers in front of the MMOG server, all the routers are connected with links that have 16 Mbps of bandwidth and 5 ms of delay (shown with black lines). The two routers in front of the MMOG server are connected with a link that has 16 Mbps of bandwidth and 1 ms of delay (shown with a green line). All the links in the network topology are in point-to-point fashion.

In Experiment 1, we defined a virtual environment of size 300×300. All the MMOG clients are moving randomly within the virtual environment and sending their position updates to the MMOG server. The position messages of the MMOG clients are 50 bytes long. The MMOG clients send their position updates to the MMOG server every 1–3 seconds. We have modeled the sending of the position message with a uniform distribution that has a minimum as 1 second and maximum as 3 seconds. We run the network simulation five times for different values of the interest circle's radius. Each simulation is run for 30 minutes of simulation time.

6.2.2. Results. The network simulations took an average of 4.5 hours to complete on the simulation computer. The simulations in which the radius of the interest circle is smaller took slightly less time than the simulations that have a larger radius of interest circle. During the simulations, we have collected statistics about the MMOG server latency, the packets sent by the server, clients and the packets received by the server. The simulation results are summarized in Table 1.

From the simulation results we have observed that, as the radius of the interest circle increases linearly, the packet send by the MMOG server increases exponentially as expected. This relation between the radius of the interest circle and the packets sent and received by the MMOG server is given in Figure 8.

The MMOG server and client bandwidth requirements can be calculated directly from these results. Table 2 shows the calculated average bandwidth requirements for the MMOG server and MMOG client. The simulation results show that a MMOG client would require an average of 24.92 B/s bandwidth for sending 50 bytes position update messages to the server every 1–3 seconds. On the other hand, a MMOG server would require an 11629130

Table 1. Area of Interest algorithm simulation results for Experiment 1.

Interest circle radius	5	11	17
Best server latency	0.0191	0.0191	0.0191
Worst server latency	0.0401412	2.7099	2.70671
Av. server latency	0.029232	0.0292292	0.0292242
Server RCV (packets)	5,384,258	5,385,530	5,386,353
Server RCV (bytes)	269,212,900	269,286,150	269,323,200
Server SND (packets)	35,703,313	176,558,826	418,639,786
Server SND (bytes)	1,785,165,650	8,828,267,850	20,932,433,900
Client SND (packets)	5,384,323	5,385,782	386,531
Client SND (bytes)	269,216,150	269,289,100	269,326,550

Table 2. Area of Interest algorithm simulation: server and client bandwidth requirements for Experiment 1.

Interest circle radius	5	11	17
Server bandwidth (B/s)	991,758	4,904,593	11,629,130
Client bandwidth (B/s)	24.92	24.93	24.93

B/s bandwidth for only distributing the position updates of the clients when the radius of the interest circle is 17 units.

6.3. MMOG server bandwidth requirement analysis

In typical MMOG networks, the bandwidth requirement of a single client is usually very low. However, MMOG servers require enormous bandwidths due to the massive number of clients. In Experiment 2, we modeled the number of simple MMOG networks with varying client counts and observed the bandwidth requirements of the MMOG server and the client.

6.3.1. Simulation model. The MMOG network model that we used in Experiment 2 is similar to the one in the AOI algorithm performance evaluation experiment (see Figure 7). By changing the node counts in the node groups, we adjusted the MMOG client count on the network. We modeled four MMOG networks with client sizes 960, 2880, 4800 and 6720.

In addition to the position update messages of the MMOG clients, we defined two new messages. The first one is the chat message, which can have a size between 100 and 200 bytes. The MMOG clients send the chat message in a period between 5 and 10 seconds. We used uniform distributions to model this behavior. The second

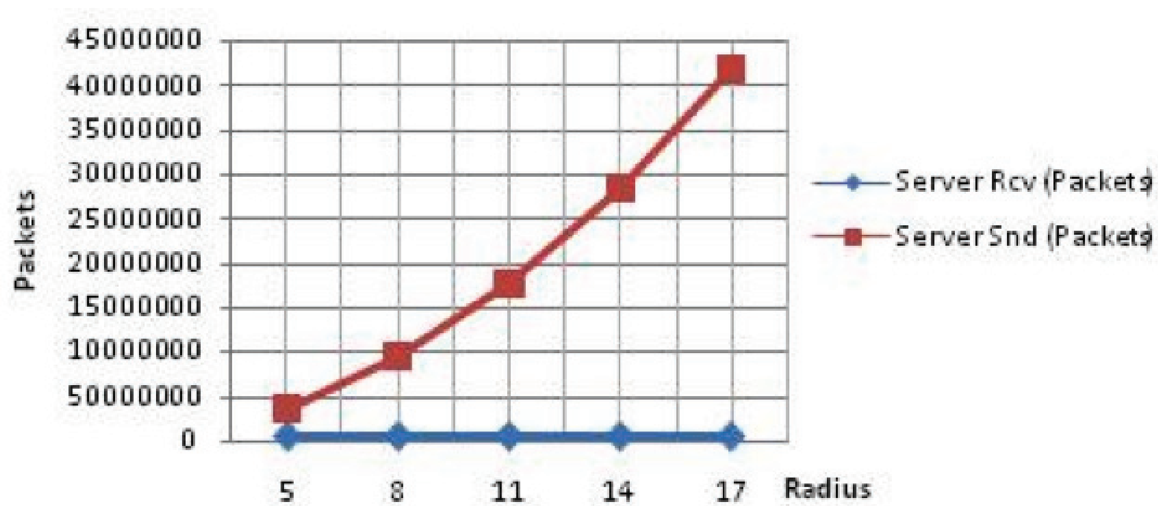

Figure 8. AOI Algorithm Simulation Server Statistics for Experiment 1.

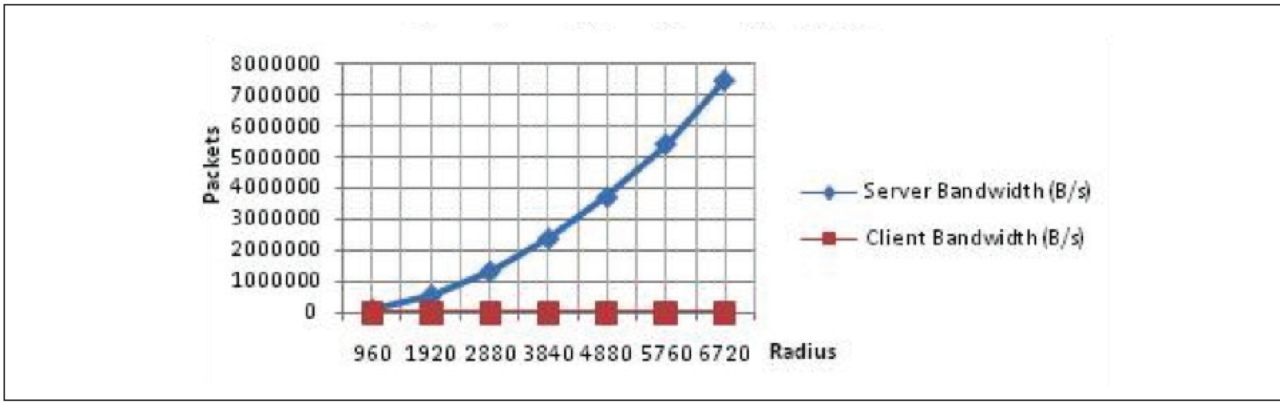

Figure 9. MMOG Server and Client Bandwidth Requirements for Experiment 2.

message that we introduced is the action message, which can have a size between 50 and 200 bytes. The action message is sent by the MMOG client every 5–15 seconds. The position messages are 50 bytes long and are sent to the MMOG server every 1–3 seconds, as in Experiment 1. We defined a virtual environment of size 300×300. In addition, we defined a Euclidean distance AOI algorithm with 8 units of interest circle radius. Each simulation is run for 30 minutes of simulation time.

6.3.2. Results. During the simulations we collected statistics about the messages sent and received by the MMOG server and clients. Table 3 summarizes the collected statistics. Note that the client sent statistics are for average values for a single MMOG client.

Calculated bandwidth requirement values from these results are given in the Figure 9. We have observed that, as the client count increases linearly, the server bandwidth requirement increases exponentially. The exponential increase of the server bandwidth requirement negatively affects the scalability of the overall MMOG network.

6.4. Virtual environment partitioning analysis

In some MMOGs, in order to deal with the massive number of concurrent players, the virtual environment featured by the game is hosted by several servers. In this type of MMOG, usually the virtual environment is geographically divided into regions and each region is associated with a server. As the clients move in the virtual environment, they send their updates to the server that is responsible for their region in the virtual environment.

6.4.1. Simulation model. In order to observe the performance of the overall MMOG network with different virtual environment partitioning, we modeled four network simulation models with 2400 clients and 18 routers. In the first simulation model there is a single server, whereas in the other simulation models there are two, three and four servers, respectively. The link properties and the link connections are the same with the simulation model described in the AOI performance evaluation experiment. Figure 10 shows the MMOG network model with four MMOG servers in the Visual Modeler tool.

The game messages sent by the MMOG clients are similar to the ones that are described in the MMOG server bandwidth requirement analysis experiment. The MMOG clients move randomly in the virtual environment and, as they move, they send their position updates to the MMOG servers with 50-byte long data packets every 1–3 seconds. In addition, we defined two additional game messages similar to the ones in Experiment 2. We defined a chat message, which can be 50–150 bytes long and sent by the MMOG clients every 10–20 seconds. In addition, we defined an action message, which can be 75–100 bytes long and sent by the MMOG clients every 5–15 seconds.

In the simulation models where there is more than one MMOG server, the virtual environment is divided into

Table 3. Massively multiplayer online game server bandwidth requirement analysis results for Experiment 2.

Client count	960	2880	4880	6720
Server RCV (packets)	1,263,027	3,789,908	6,309,364	8,816,587
Server RCV (bytes)	98,832,631	296,452,097	494,130,891	691,673,342
Server SND (packets)	3,489,314	31,521,597	86,209,742	171,784,391
Server SND (bytes)	273,126,945	2,466,443,921	6,753,706,833	13,479,137,198
Client SND (packets)	1316	1316	1294	1316
Client SND (bytes)	102,952	102,936	101,221	102,928

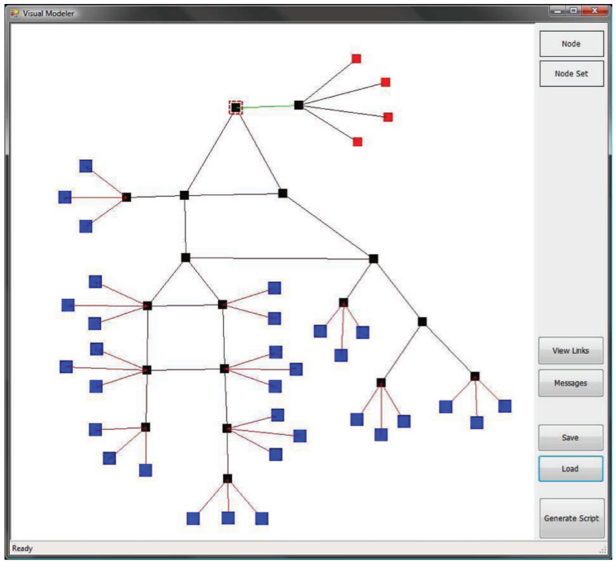

Figure 10. MMOG Network Model with 2400 Client and 4 Servers for Experiment 2 (color online only).

Table 4. Single server configuration results for Experiment 3.

	Server	Clients (total)	Single client
Packets received	2,870,965	NA	NA
Bytes received	173,613,654	NA	NA
Packets sent	11,265,333	2,871,001	1196.250417
Bytes sent	681,322,108	173,616,010	72,340.00417

equal area regions and each region is associated with a server. We defined a 50-byte long connection message to be sent by a client when it enters a new region hosted by a different server. Therefore, as the clients move randomly in the virtual environment, if they enter a new region, the client sends a connection message to the server associated with the new region.

6.4.2. Results. During the simulations, we collected statistics regarding the number of packets sent by the clients and the server, the number of packets received by the servers and the number of connection messages sent by clients. Table 4 summarizes the results obtained from the first simulation model, which has a single MMOG server.

In the case where the whole virtual environment is hosted by a single server, the server receives messages from clients at an average rate of 0.092 MB per second. On the other hand, the server sends messages to the clients at an average rate of 0.361 MB per second. In this case, the average number of bytes sent by a client is 0.039 kB per second.

Table 5 shows the results of the simulation model with two MMOG servers. In the case where the virtual environment is hosted by two servers, the servers receive messages from clients with an average rate of 0.046 MB per second and the average rate of the messages sent by the servers is 0.179 MB per second. On the other hand, a single client sends messages to the servers with an average rate of 0.039 kB per second. The average number of connection messages sent by a single client is 9.2. This means that an average of 0.25 bytes per second overhead is introduced in this configuration.

Table 6 shows the results obtained from the simulation model with three MMOG servers. In this case, the average download and upload rate of the MMOG servers are 0.031 and 0.121 MB/s, respectively. The average upload rate of a single client is 0.040 kB/s. The average overhead caused by the connection messages is 0.44 bytes per second.

Table 7 summarizes the results obtained from the simulation model with four MMOG servers. In this simulation model, the average download and upload rate of the MMOG servers are 0.023 and 0.09 MB/s, respectively. The average upload rate of a single client is 0.040 kB/s. The average overhead introduced by the connection messages is 0.62 bytes per second.

The simulation results show that dividing the virtual environment into regions and assigning a single server for each region can greatly decrease the bandwidth

Table 5. Two server configuration results for Experiment 3.

	Server 1	Server 2	Clients (total)	Single client
Packets rcv.	1,429,318	1,463,246	NA	NA
Bytes rcv.	86,340,754	88,398,087	NA	NA
Packets sent	5,389,191	5,764,235	2,892,623	1205.259583
Bytes sent	326,000,303	348,744,350	174,742,353	72,809.31375
Connection packets	NA	NA	22,115	9.214583333

Table 6. Three server configuration results for Experiment 3.

	Server 1	Server 2	Server 3	Clients (total)	Single client
Pck. rcv.	788,637	1,322,347	797,087	NA	NA
Bytes rcv.	47,599,383	79,801,962	48,107,845	NA	NA
Pck. sent	2,492,743	6,305,001	2,522,738	2,908,912	1212.04
Bytes sent	150,703,283	381,281,843	152,439,571	175,512,334	73,130.13
Con. pck.	NA	NA	NA	38,861	16.19

Table 7. Four server configuration results for Experiment 3.

	Serv. 1	Serv. 2	Serv. 3	Serv. 4	Clients (total)	Single client
Pck. RCV.	513,084	950,132	949,340	509,626	NA	NA
Bytes rcv.	30,930,410	57,311,117	57,268,836	307,83,309	NA	NA
Pck. sent	1,379,515	4,274,244	4,200,727	1,376,650	2,924,346	1218.4
Bytes sent	83,265,754	258,358,737	253,939,367	83,205,816	176,296,300	73,456.7
Con. pck.	NA	NA	NA	NA	53841	22.4

requirement and therefore the workload on the servers. For example, in single server configuration, the bandwidth requirement of the server for upload is about 0.361 MB/s, whereas in the four server configuration this number is 0.09 MB/s. It is clear that dividing the virtual environment will introduce some overhead in the communication because of connection and initialization messages. However, as can be seen from the simulations, the overhead remains insignificant in configurations with a reasonable number of servers.

7. Conclusion

This study presents a network simulation tool for simulating client–server architecture-based MMOG networks. The proposed tool (GTNetS-MMOG) can be used to model and simulate large-scale MMOG networks that consist of several thousands of MMOG clients. For visually modeling the MMOG networks, a network modeler tool (Visual Modeler) is also presented.

The network simulation tool presented in this study is based on the GTNetS. For allowing the simulation of typical MMOG networks, the GTNetS is extended with new application and message types. Furthermore, an infrastructure that allows sending of custom data structures with real data is added to the network simulation environment. The MMOG client application can connect to the MMOG server application using the TCP and can send messages with defined sizes and intervals. The GTNetS-MMOG allows collection of various statistics regarding the MMOG network. To allow users to define their simulation models in script files, a script parser is also added to the GTNetS-MMOG.

A network modeler tool that allows visually modeling of the MMOG networks is also implemented in this study. The Visual Modeler tool provides functionalities that simplify the modeling of large-scale MMOG networks, such as adding a number of nodes to the network topology with a single operation.

Example simulations demonstrate the use of the GTNetS-MMOG network simulation tool as well as the Visual Modeler tool. We have used the GTNetS-MMOG tool to measure the effect of the interest circle radius on the network traffic in a MMOG network. Moreover, the effect of increasing the client count in a MMOG network to the bandwidth of the MMOG server is also observed by a simulation example.

Funding

This research received no specific grant from any funding agency in the public, commercial or not-for-profit sectors.

References

1. Voig, Inc. MMOGData.VOIG.com, 'Online Data about Online Games', http://mmogdata.voig.com/ (2008, accessed 21 August 2011).
2. Woodcock BS. 'An Analysis of MMOG Subscription Growth', http://www.mmogchart.com/ (April 2008, accessed 21 August 2011).
3. DFC Intelligence. 'DFC Online Game Market Forecast 2007', http://www.dfcint.com/index.php (2007, accessed 21 August 2011).
4. Zona, Inc. Executive Summary Consulting, Inc., 'The Challenges of MMOG Development', *Digital Game Developer*, http://www.digitalgamedeveloper.com/2002/11_nov/features/dlmmogd112602.htm (November 2002, accessed 12 June 2008).
5. Obaidat MS. Recent advances in modeling and simulation of network systems. *Simulation* 2006; 82: 365–367.
6. Murray B. 'GTNetS-Home', http://www.ece.gatech.edu/research/labs/MANIACS/GTNetS/ (accessed 17 May 2011).
7. Blizzard Entertainment, Inc. 'World of Warcraft', http://www.worldofwarcraft.com/index.xml (2008, accessed 21 August 2011).
8. Linden Research, Inc. 'Second Life', http://secondlife.com/ (2008, accessed 17 May 2011).
9. El Rhalibi A and Merabti M. Agents-based modeling for a peer-to-peer MMOG architecture. *Comput Entertain* 2005; 3: 3.
10. Cronin E, Filstrup B and Kurc A. *A distributed multiplayer game server system*, EECS589 Course Project Report, University of Michigan, 2001.
11. Knutsson B, Honghui Lu, Wei Xu and Hopkins B. Peer-to-peer support for massively multiplayer games. In: *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2004)*, 11 March 2004, vol. 1, p.107.
12. Hampel T, Bopp T and Hinn R. A peer-to-peer architecture for massive multiplayer online games. In: *Proceedings of 5th ACM SIGCOMM Workshop on Network and System Support for Games (NetGames '06)*, Singapore, 30–31 October 2006, New York: ACM, p.48.
13. Kim K, Yeom I and Lee J. HYMS: a hybrid MMOG server architecture. *IEICE Trans Inf Syst* 2004; E87-D: 2706–2713.
14. Yoick Entertainment. 'Outback Online', http://www.theage.com.au/news/biztech/yoicks-its-another-virtual-world/2007/02/26/1172338546871.html, (accessed 17 July 2008).
15. Boulanger J, Kienzle J and Verbrugge C. Comparing interest management algorithms for massively multiplayer games. In: *Proceedings of 5th ACM SIGCOMM Workshop on Network and System Support For Games (NetGames '06)*, Singapore, 30–31 October 2006, New York: ACM, p.6.
16. Riley GF. The Georgia Tech Network Simulator. In: *Proceedings of the ACM SIGCOMM Workshop on Models, Methods and Tools for Reproducible Network Research (MoMeTools '03)*, Karlsruhe, Germany, 25–27 August 2003, New York: ACM, pp.5–12.
17. Vall EMOA, Riley GF and Heck BS. Large-scale sensor networks simulation with GTSNetS. *Simulation* 2007; 83: 273–290.
18. The Network Simulator - ns-2. http://www.isi.edu/nsnam/ns/ (accessed 21 August 2011).
19. Luc Hogie PB and Guinand F. An overview of MANET's simulation. *Electron Notes Theor Comput Sci* 2006; 150: 81–101.
20. OPNet Technologies, Inc. 'OPNet Modeler', http://www.opnet.com/solutions/network_rd/modeler.html (accessed 21 August 2011).
21. Chen K, Huang P, Huang C and Lei C. Game traffic analysis: an MMORPG perspective. In: *Proceedings of the International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV '05)*, Stevenson, Washington, USA, 13–14 June 2005, New York: ACM, pp.19–24.
22. Svoboda P, Karner W and Rupp M. Traffic analysis and modeling for World of Warcraft. In: *Proceedings of the IEEE International Conference on Communications*, 24–28 June 2007, p.1612–1617.
23. Jung Y, Lim B, Sim K, Lee H, Park I, Chung J, et al. VENUS: The Online Game Simulator Using Massively Virtual Clients. In: *Proceedings of AsiaSim 2004*, Korea, October 2004, pp.589–596.
24. Fujimoto RM, Perumalla KS and Riley GF. Network simulation synthesis. *Lect Commun Networks* 2006; 1: 1–72.

Author biographies

Veysi İşler is a faculty member at the Department of Computer Engineering, Middle East Technical University, Turkey, where he is also the director of Modelling and Simulation Research and Development Center. His research interests include rendering, visualization, game technology, modeling and simulation. Isler has a Ph.D. degree in Computer Engineering in parallel graphics. Email: veysi@metu.edu.tr

Selçuk Bozcan is currently working as senior software engineer in Aselsan Inc., Turkey. He earned his bachelor's and master's degree in computer engineering from Middle East Technical University in Turkey. Email: s.bozcan@gmail.com