Towards decentralized, secure, scalable and consistent MMOGs

Fábio Reis Cecin, Cláudio Fernando Resin Geyer
Universidade Federal do Rio Grande do Sul
Instituto de Informática
Bento Gonçalves 9500, Porto Alegre, Brasil
{fcecin, geyer}@inf.ufrgs.br

Abstract

Real-world, deployed MMOG (massively multiplayer on-line game) services such as World of Warcraft, despite recent efforts from researches, are still implemented as pure centralized systems. Several new MMOG distribution models have already been proposed with the goal of decentralizing current MMOG services through the use of either pure peer-to-peer or hybrid topologies. Most of those proposals fail to address at least one of the three vital problems that arise when MMOG decentralization is attempted: security, scalability or consistency. This paper reviews some of the proposals for the achievement of a decentralized MMOG system that possesses the three key properties. This paper also describes in detail these three properties and attempts to clarify why it is so difficult to solve them at the same time on a fully or even partially decentralized MMOG model. This paper is the starting point of a thesis proposal.

1. Introduction

The massively multiplayer on-line game (MMOG) is a real-time application of distributed simulation with growing popularity. This kind of game supports great amounts of players interacting in real-time, simultaneously, in a persistent-state virtual world. Current examples of MMOGs are World of Warcraft [2] and Guild Wars [1]. As a reference, the World of Warcraft service has been reported to handle hundreds of thousands of simultaneous players [12].

Most, if not all, commercially-deployed MMOG services are designed in a way in which the simulation of the 'virtual world' is distributed over several nodes. Generally, these nodes are dedicated server nodes that must be in secure locations and wired together with dedicated, low-latency networks, with powerful links to the Internet that allow feeding thousands of connected player nodes with frequent game-world state updates. This server-side computing and networking resources over-provisioning allows client-server MMOGs to meet the consistency, security and scalability requirements of those kinds of game services. It is a 'brute force' approach to the MMOG distribution problem that yields significant costs that could be reduced with different approaches to distribution.

The total or partial distribution of MMOG simulation over the machines of the game players is a current research topic, the goal being server-side infrastructure cost elimination or reduction. However, despite all research efforts so far [11], current real-world, deployed MMOG services such as World of Warcraft are still implemented as pure centralized systems, where a service provider runs all of the game simulation in behalf of the clients.

Apparently none of the partially or fully decentralized MMOG models proposed by researchers so far manages to attain the remaining three key properties at the same time: security, scalability and consistency [11]. More specifically, as far as we know, none of the sets of levels of security, scalabilty and consistency attained by each individual proposal seems to be attractive enough to motivate a change, in the commercial sphere, from the centralized MMOG paradigm to a more decentralized model.

This paper reviews some of the recent MMOG distribution models that feature decentralization, while attempting to clarify why it is so difficult to solve the three key problems at the same time on a fully or even partially decentralized MMOG model. The paper then draws some final conclusions and identifies opportunities for future work.

2. Scalability, security and consistency in a client-server MMOG

This section explains what the scalability, security and consistency properties are, and how those are achieved in a client-server, centralized MMOG.

When one mentions 'massively multiplayer on-line games', it is usually a 3D 'virtual world' such as

World of Warcraft's that springs to mind. Although 'massively multiplayer online' implies that the game in question is networked and that it supports a large amount of players simultaneously playing an instance of the game, it doesn't seem to imply a particular interface (3D, 2D or non-graphical) or a particular interactivity paradigm (real-time, turn-based, or other). However, what is known as the 'MMOGs market' today is vastly dominated by 3D virtual worlds with real-time interactivity, and that is what we are referring to in this paper.

2.1. Security

One important aspect of MMOGs that distinguishes them from general-purpose or collaborative virtual environments (CVEs) is that the virtual environment is competitive. This brings forward the main security problem: protection against players that want to cheat in the game to gain unfair advantages over other players [9, 14, 7].

Cheating is rampant in online games played over the Internet. A MMOG model is not adequate for real-world deployment if it doesn't explicitly address the issue of cheating players. This is due to the fact that cheating in MMOGs can be more destructive to the player experience than cheating in regular online games. This is because a key aspect of MMOG virtual worlds is that the state is persisted for months or years. Players invest large amounts of time and money improving their characters (avatars or alter-egos) in virtual worlds, and it is extremely frustrating to see all the rewards for this effort vanish when other players start acquiring 'virtual wealth' by cheating.

This is probably one of the main reasons why most of the current commercial MMOGs are based on centralized client-server architectures. In this model, all of the simulation (computation) executes on servers, which are kept in secure locations. The game clients just send requests (commands) to the servers, which can validate them before applying them to the official, persisted state of the virtual world. This is the main reason why the centralized MMOG is mostly cheat-proof: having no direct access to the machines running the actual simulation, potential cheaters are limited to exploiting vulnerabilities in the game protocol (bugs or design weaknesses).

2.2. Scalability

By centralizing the MMOG simulation on dedicated servers, the problem of scalability arises. With the increase in the supported number of connected clients, it becomes necessary to supply additional processing (CPU) and communication (network) power to the game server-side. To supply this, most commercial MMG implementations today use multiple game servers to serve a single virtual world through employment of some kind of load balancing scheme [13, 5]. These techniques vary from static to dynamic allocation of clients to server clusters or even Grids. Additionally, all those servers must be connected with over-provisioned network links that exchanges something around 20 to 40 packets per second with each client, which is a result of the client-server (or 'interaction terminal' to 'simulator') distribution strategy for interactive virtual worlds.

The large server farms and powerful network links that run centralized MMOG simulations yield significant costs to the game service providers, which usually have to charge monthly fees from the players. Typical MMOGs such as PlanetSide or World of Warcraft supports from a few tens of thousands to a few hundred thousand simultaneous players connected to game servers. Sony Entertainment reportedly had 1,500 servers, distributed across four clusters in different locations, to serve all of its MMOGs, including PlanetSide. So, although the MMOG server-side scalability problem can be dealt with powerful hardware, this approach can be restrictive. For one thing, small or independent game studios and research groups often don't have access to that kind of hardware infrastructure and thus cannot currently employ it.

To emphasize this last point, let's consider an example. Most MMOGs today only support clients with 'broadband' connections. That means assuming an average 5 kB/s transfer rate between a client and the game server-side is reasonable. Then, a game service such as World of Warcraft would be required to deal with 1 GB/s of real-time, timely Internet traffic alone when supporting 200,000 simultaneous clients, which is a figure that was already been surpassed by World of Warcraft and also probably other MMOGs. This is a rough estimate, since games can be optimized to use less than 5 kB/s, but they can also require more. This is clearly a barrier of entry for smaller game publishers and development studios.

2.3. Consistency

Consistency [10], in a distributed interactive simulation, is a qualitative measure based on several factors, including, but not limited to:

- Level of video and audio synchronization attained between all of the interaction terminals;
- Amount of temporal inconsistencies presented to each user (example of temporal inconsistency: showing a dead enemy, receiving a late network event and then rolling-back and re-executing the simulation to show the same enemy alive after a few milliseconds later);
- Responsivity of the simulator to commands issued by human users through input devices.

These have already been studied in depth in previous works on several related research areas, such as distributed interactive games, collaborative virtual environments, military simulation, etc. In this paper we will focus in two additional factors that are strongly related to consistency requirements, and which are more specific to the 'on-line game' problem. The first factor is obvious and is directly related to responsiveness (but also affects scalability and security significantly), but the second factor (seamlessness) is not usually discussed in the context of consistency requirements of MMOGs. It is a key factor that can radically alter the consistency requirements of a MMOG distribution engine. The two factors are:

- Type of game supported (action game, role-playing game, real-time strategy game, turn-based game or other);
- Seamlessness of the 'virtual world' metaphor provided by the game as a whole (distributed simulation engine and the game's actual design).

The first aspect can radically change the consistency requirements of a game. An action game player expects responsiveness to interactive commands in the order of 100 milliseconds, and anything about 200 milliseconds will start to feel inadequate. Role-playing and real-time strategy games tolerate interactions in the range between 500ms and 1000ms, or maybe more, depending on which techniques the particular game uses to hide network latency. Turn-based games can tolerate almost any latency, but, on the other hand, probably won't tolerate temporal inconsistencies as well as the other types of games. Action games are particularly difficult to decentralize in a massively multiplayer context while keeping security and scalability. Fortunately, most MMOGs today are RPGs (role-playing games), which tolerate greater response times and thus making the task of decentralizing the simulator in a secure and scalable way much more palatable.

We will now concentrate on the second aspect which is related to consistency: seamlessness. By default, one assumes that a 'virtual world' is like the real world. That is, it provides a single virtual 3D or 2D space where game characters roam and interact with each other in real-time. However, that is almost always never exactly the case. The early commercial MMOGs like EverQuest estabilished the notion of 'server shards', which are a player metaphor for an instance of the game world. Basically, the 'virtual world', as outlined by the game designers and rendered by game programmers and artists, is copied (instanced) over several servers, or clusters of servers. Players then choose one server or cluster of servers to play on. Each server or cluster of servers manages a separate virtual world, and it works almost exactly as a separate game. The content is the same, but each 'server shard' is a different instance of that content, and players cannot interact between instance boundaries, unless it is a mostly 'off-line' interaction, such as a character's database record being transferred from one 'shard' to another, by an operation carried out through the game's web site interface.

Although this may seem to result in a very seamed game experience, this is not so. This happens because each 'shard' is large enough to hold a healthy dose of players, such as 5,000, which probably satisfies most players as long as a variety of peers is concerned. And, more importantly, all of the 2D or 3D terrain of a single shard is presented to the user as a seamless terrain, even if the terrain is tiled and distributed for processing in different machines on the server-side. This is possible because all server nodes have low-latency, local, dedicated networking to synchronize player interactions that span server machine borders and transfer of players between game tiles or machines. That seamless terrain usually represents a very large area where players can even get lost in and never meet any of the other tens of thousands of players that are also currently there. And, of course, the servers have to support crowded areas with hundreds of people in the line-of-sight of each other just as well. Current client-server MMOGs follow the same approach more or less, varying the distribution technology employed on the server side, which is usually not well published since today they can be considered valuable trade secrets.

Some games present new game designs, approaching the consistency problem in innovative ways. An example is the game Guild Wars. In this game, instead of providing a huge, single 3D virtual space where thousands of avatars roam freely, the game creates collections of smaller spaces which can have two different natures. A player can be at a 'town', chatting and trading with other players, or it can be in an 'instance', having an adventure. In the first kind of space, hundreds of players can gather. In the second kind of space, only up to 16 players can gather, at least for player-versus-player duels (without computer-controlled opponents). This changes the 'consistency game' of MMOGs in a subtle way: if you can partition players in 'instances', which are dynamically-created, non-persisted spaces, then the distribution problem at the server-side can be greatly simplified.

In closing, the amount of consistency offered by a client-server MMOG will increase as more processing and networking resources are allocated on the server-side, provided the number of simultaneous clients supported by the system remains the same. The particular strategy for distributing the simulators on the server-side is also important, and in some way determines the amount of gain or loss with added or removed processing or networking resources at the server-side.

3. The MMOG decentralization problem

Several research papers have been dedicated to peer-to-peer support for massively multiplayer on-line games. As far as we know, most proposals fail to address, at the same time, the security, scalability and consistency issues that arise when partial or total decentralization of the game simulation is attempted. We examine the issues below.

The flow of a client-server interactive simulator can be roughly described by the sequence below:

1. Client reads local input devices, translates them into requests, and sends those to the Server;
2. Server receives player requests, validades, orders and executes them, modifying the game's state;
3. Server sends updates to all clients (e.g. what each client will see or hear on the next slice of simulation time);
4. Client receives updates and renders them as multimedia feedback (video, audio, force-feedback, etc).

The security problem in client-server MMOGs is easier to deal with, because untrusty client machines are only responsible for the bare minimum of tasks (steps 1 and 4), while the servers do all of the critical tasks like validating, ordering and executing events (steps 2 and 3). Decentralization is all about moving steps 2 and 3 partially or totally to untrusty client machines. How this is done determines whether a peer-to-peer MMOG model is exploitable or not. Follows an example that illustrates how the security problem arises in peer-to-peer MMOG proposals.

The common theme in peer-to-peer MMOG proposals is to partition the virtual game space into tiles following some pattern (rectangles [3], hexagons [6], arbitrary shapes [8], or others) and deal with each tile separately. And, one of the several ways in which to deal with the simulation in each tile is to choose a single client machine to act as the simulator of that tile. In essence, one client becomes one of the server machines in a cluster of servers that serves the 'shard' of a MMOG. The clients that are responsible for tiles (the tile 'managers') communicate between themselves to negotiate what happens in situations where events in different tiles affect each other (e.g. players in adjacent tiles seeing each other and shooting projectiles at each other). Clients that are not responsible for simulating tiles act exactly as clients of centralized MMOGs: they simply connect to the manager of the tile that they're currently on, sending requests to them and receiving updates from them.

Those approaches usually fare well in the consistency and scalability problems. Consistency is mostly dependant on choosing good tile managers, and some approaches employ back-up managers that can hot-swap very efficiently if the official manager dies unexpectedly. Tile managers will be analogous to the 'super-peers', the more dependable nodes that are often identified in peer-to-peer file sharing networks. Scalability is also dependant on good tile managers, and also on the size of tiles and the kind of network transport used. Many works assume that the communication that happens within each tile is carried on one or several multicast channels, which reduces the overall bandwidth requirement, particularly for the tile manager.

The security problem with the tile manager approach, however, is severe. If the tile manager is the sole entity responsible for ordering, validating and executing events in each tile, then a single manager can, for instance, produce large amounts of 'virtual wealth' and give it to any player, thus ruining not only the game experience on that tile, but wrecking the whole game economy instantly (note that MMOGs are games with a persisted state). This is completely unacceptable for a real-world MMOG service.

Other approaches try to attain security by turning each and every peer that is playing on a tile into a manager. That usually dictates that every action by one playing peer must be acknowledged by every other peer in the tile before it can be commited. This solves security but, on the other hand, it creates scalability and consistency problems. Scalability is severely impacted, because it creates problems with inter-tile interaction (an N-to-N communication problem) and tile processing (for instance, when all peers, 'super' or not, are required to simulate the whole 3D tile on each machine). Consistency is a problem when inter-tile interaction quality is compromised (for instance, in the FreeMMG [3] model, where it also creates a subtle server-side scalability issue), or when one of the peers suddenly dies or becomes unresponsive, dragging down the interaction responsivity in the whole tile since all peers must confirm all interactions.

3.1. Manipulating consistency requirements

One of the ways to solve both scalability and security in a peer-to-peer MMOGs model is to tackle the consistency problem in creative ways, in a way that options for dealing with scalability and security are not severely limited. This section focuses on two aspects that can be creatively manipulated, and that were mentioned in Section 2.3: type of games supported and seamlessness.

The following discussion intends to serve as examples on how the seamlessness and game-type support can be manipulated to solve the consistency problem while, at the same time, providing a solid foundation where the scalability and security problems can be more definitively solved. But that has a price, and this is also discussed below.

FreeMMG [3] is a proposal for secure and scalable support for real-time strategy games. FreeMMG also partially satisfies the consistency requirements of a massively-multiplayer (MMOG) real-time strategy (RTS) game. Actually, that leads to a rather contrived and academic discus-

sion on the FreeMMG dissertation, which is a reflex of the non-existance of 'true' MMOG RTS games, that is, a deployed MMOG that would have the same consistency requirements of small-scale RTS games such as Age of Empires or Starcraft.

The FreeMMG model is simple: a single server, running on the game provider, acts as a tile manager. The virtual world is divided into tiles that are supposed to be adjacent to each other. Each tile is managed by all peers that are currently playing on that tile (see above for the implications of this). However, since in a RTS game it is expected that no more than 10 or 20 peers will be playing on the same RTS game board (against whose size the 'tile size' of FreeMMG is adjusted), we can guarantee that hot-spots won't be a pressing issue. Security is attained for free, since synchronization is conservative on each tile. All peers must agree on the computation, making it difficult to cheat. The hit on player responsivity is acceptable since it is a RTS game. The main casualty is the seamlessness: interactions between tiles are not fully supported. There is limited support that goes through the server. There is a general assumption in FreeMMG that the player's joy when experiencing a large-scale version of a true real-time RTS would compensate for the lack of interaction support on borders. It is impossible to know if that would be the case unless the model is instanced in a real product, which hasn't happened so far.

P2PSE is an ongoing project in the Group of Parallel and Distributed Processing (GPPD) at UFRGS. The P2PSE model tries to offer a game metaphor very similar to the one employed by Guild Wars (see explanation on Section 2.3). The innovation is to support the 'town' environments (called 'social spaces' in P2PSE) with a client-server, centralized simulator, and the 'instanced' environments (called 'action spaces' in P2PSE) with peer-to-peer, decentralized simulation. P2PSE is thus a hybrid model, which assumes that most of the traffic occurs when players are in action spaces. This is because consistency requirements of social spaces are low: players really don't need to see other players moving in real-time. On the other hand, consistency requirements of action spaces are high: players absolutely need to send updates frequently to each other so that temporal inconsistencies are avoided as much as possible, allowing players to dodge high-speed projectiles and performing other activities that are very time-sensitive.

The P2PSE model deals with the seamlessness aspect in the sense that Guild Wars, a proven commercial MMOG, successfully employs the exact same partitioning of the virtual world. P2PSE is also slated to support action games. The main weakness on the consistency front is the social space, which will attain scalability by drastically reducing the amount of network packets dedicated to updating each connected client. This will cause huge temporal inconsistencies which, fortunately, won't matter since social spaces have a focus on security, since those spaces are where all trading between players occurs.

The main security weakness in P2PSE occurs when an action space game ends, and the results have to be carried over to the social space. At that point in time, the server must collect an 'end-game report' from each player. The main problem is not about when there are disagreements on the reports, but when all the players agree on the result, and the result is a fake. This is known as the 'collusion cheating' problem: all players collude to provide a very positive value for all of them (e.g. provide huge and unjustified awards for each player). A publication related to P2PSE addresses that problem [4], providing some initial results that show promising scalability and consistency properties.

Both FreeMMG and P2PSE are examples of trading off or otherwise manipulating seamlessness and types of games that are supported, with the goal of facilitating the task of supporting the consistency requirements of the game, and in a way that leaves the remaining scalability and security problems in a solvable state. However, they are both unable to support mainstream commercial RPG MMOGs such as World of Warcraft, EverQuest, and others, which comprise the vast majority of MMOGs, commercial or otherwise.

4. Conclusions and future work

This paper presented the current problem of client-server MMOGs: their scalability, consistency and security features come at an economic price. A current research problem is providing peer-to-peer or hybrid models for supporting MMOGs. However, the paper argues that it is difficult to attain scalability, consistency and security at the same time on a partially or fully decentralized MMOG. Although the authors cannot guarantee that such proposal has already been made, the current commercial MMOG landscape, which is comprised exclusively of pure client-server MMOGs, hints at that possibility, and motivates further research. This paper is a first step for a thesis proposal, and, in that scope, future work includes identifying a possibility for addressing consistency, security and scalability in a decentralized MMOG model that supports current MMORPGs (massively multiplayer online role-playing games), with support for seamless virtual worlds, like as provided by all current MMORPGs.

References

[1] ArenaNet. Guild Wars, 2007. http://www.guildwars.com/.

[2] Blizzard. World of Warcraft, 2007. http://www.worldofwarcraft.com/.

[3] F. Cecin, R. de Oliveira Jannone, C. Geyer, M. Martins, and J. Barbosa. FreeMMG: a hybrid peer-to-peer and client-server model for massively multiplayer games. *Proceedings*

of ACM SIGCOMM 2004 workshops on NetGames' 04: Network and system support for games, pages 172–172, 2004.

[4] F. Cecin, C. Geyer, S. Rabello, and J. Barbosa. A peer-to-peer simulation technique for instanced massively multiplayer games. *Proceedings of the Tenth IEEE International Symposium on Distributed Simulation and Real-Time Applications (DS-RT'06)-Volume 00*, pages 43–50, 2006.

[5] T. Duong and S. Zhou. A dynamic load sharing algorithm for massively multiplayer online games. *Networks, 2003. ICON2003. The 11th IEEE International Conference on*, pages 131–136.

[6] S. Fiedler, M. Wallner, and M. Weber. A communication architecture for massive multiplayer games. *Proceedings of the first workshop on Network and system support for games*, pages 14–22, 2002.

[7] C. GauthierDickey, D. Zappala, V. Lo, and J. Marr. Low latency and cheat-proof event ordering for peer-to-peer games. *Proceedings of the 14th international workshop on Network and operating systems support for digital audio and video*, pages 134–139, 2004.

[8] S. Hu and G. Liao. *Scalable peer-to-peer networked virtual environment*. ACM Press New York, NY, USA, 2004.

[9] P. Kabus, W. W. Terpstra, M. Cilia, and A. P. Buchmann. Addressing cheating in distributed mmogs. In *ACM SIGCOMM workshop on Network and system support for games*, pages 1–6, New York, NY, USA, 2005. ACM Press.

[10] D. Roberts and R. Wolff. Controlling Consistency within Collaborative Virtual Environments. *Distributed Simulation and Real-Time Applications, 2004. DS-RT 2004. Eighth IEEE International Symposium on*, pages 46–52, 2004.

[11] G. Schiele, R. Suselbeck, A. Wacker, J. Hahner, C. Becker, and T. Weis. Requirements of Peer-to-Peer-based Massively Multiplayer Online Gaming. *Proceedings of the Seventh IEEE International Symposium on Cluster Computing and the Grid*, pages 773–782, 2007.

[12] T. Surette. World of Warcraft sells more than 600,000 units, 2005. http://www.gamespot.com/pc/rpg/-worldofwarcraft/news.html?sid=6116075.

[13] B. D. Vleeschauwer, B. V. D. Bossche, T. Verdickt, F. D. Turck, B. Dhoedt, and P. Demeester. Dynamic microcell assignment for massively multiplayer online gaming. In *ACM SIGCOMM workshop on Network and system support for games*, pages 1–7, New York, NY, USA, 2005. ACM Press.

[14] J. Yan and B. Randell. A systematic classification of cheating in online games. In *ACM SIGCOMM workshop on Network and system support for games*, pages 1–9, New York, NY, USA, 2005. ACM Press.