Network Architectures and Data Management for Massively Multiplayer Online Games

Minhua Ma, University of Derby, UK
Andreas Oikonomou, University of Derby, UK

ABSTRACT

Current-generation Massively Multiplayer Online Games (MMOG), such as World of Warcraft, Eve Online, and Second Life are mainly built on distributed client-server architectures with server allocation based on sharding, static geographical partitioning, dynamic micro-cell scheme, or optimal server for placing a virtual region according to the geographical dispersion of players. This paper reviews various approaches on data replication and region partitioning. Management of areas of interest (field of vision) is discussed, which reduces processing load dramatically by updating players only with those events that occur within their area of interest. This can be managed either through static geographical partitioning on the basis of the assumption that players in one region do not see/interact with players in other regions, or behavioural modelling based on players’ behaviours. The authors investigate data storage and synchronisation methods for MMOG databases, mainly on relational databases. Several attempts of peer to peer (P2P) architectures and protocols for MMOGs are reviewed, and critical issues such as cheat prevention on P2P MMOGs are highlighted.

Keywords: Data Management, Distributed Systems, Massively Multiplayer Online Games, Multimedia, Networks, Peer-to-Peer, Region Partitioning

INTRODUCTION

MMOGs, also known as MMORPGs, is short for Massively Multiplayer Online (Role-Playing) Games, is a genre of online games which has been quickly and steadily growing since the start of the 21st century. Typically, MMOGs are based on a client-server architecture, where events are computed at the server and sent to clients. Each client controls a single character in the world, known as the player, communicates actions to the game server. Player characters can acquire and improve skills, collect items in their inventory, trade items with other players etc. The server serializes the actions, updates the game world accordingly, and communicates any changes to all affected players. The new game state is then rendered by the client-side software. The condition of the game world is constantly evolving, and in some cases in an unpredictable, unscripted fashion, as players interacts in the game. Some client-server architectures (Quax et al., 2008) have an intermediary layer of proxy servers which can reduce the number of...
connections for the servers and cache non-state data such as mesh or inventory information to improve response time.

Peer to peer (P2P) architectures have recently emerged as a potential alternative design for MMOGs. A few attempts of P2P MMOG have been reported in the research community (Krause, 2008). Compared to peer to peer models, distributed client-server architectures perform well when servers are faster (which is usually the case) or better distributed than clients but one of the drawbacks of client-server based MMOGs is that peer to peer messages must be relayed.

In this paper, we discuss key issues involved in networked computer game development, i.e., MMOG network architectures such as distributed client-server models and various P2P protocols proposed in the literature, database design, data replication, region partitioning, areas of interest, and cheating, with the hope that it would shed some light on future network design of distributed virtual environments and MMOGs.

DATA REPLICATION

Engagement is meant to be for a long period of time in MMOGs with users spending several months or even years playing and improving their character. As these games are designed to run continuously for years it is expected that the hardware infrastructure will at some point inevitably fail, be forced to shut down or otherwise go offline. In such situations it is crucial that the game can be restored to the last correct state it was in. Data replication therefore becomes of paramount importance.

Replication is the process of sharing data to ensure consistency between redundant resources, in order to improve reliability, fault tolerance, and accessibility. It could be data replication if the same data is stored on multiple storage devices or computation replication if the same computing task is executed many times. Current MMOGs are in favour of computation replication due to the issues of bandwidth and network latency. For example, in Russell et al.’s (2008) solution a server send both the initial state of a given object and deterministic code to simulate the object over time to all interested clients, so that clients can independently update the area of interest consistently with updates computed by the server. Computation replication results in efficient use of network bandwidth and the power of multi-core processors of game platforms.

Traditionally if the game state is dynamically stored during game play, it could be restored upon restarting a crashed or failed server. This is usually a relatively simple task where a complete snapshot of the world taken immediately before shutting down the server can be re-issued. Difficulties however arise when the termination is unpredictable, such as in the case of a server crash. Ideally, there should be no difference between the stored states and the actual game states in memory at the time of failure however this may not be realistic due to scalability and efficiency reasons. In this context the major requirements for replication in MMOGs are (Zhang et al., 2008):

- Consistency. This is relatively easy in classic client-server architectures whereas P2P architecture distributes the game state among peers and has no central server, making it very hard to administrate the game state and keep it consistent. Currently, MMOGs cannot solely use P2P architecture and there is no working business model yet.
- Efficiency. MMOGs are real-time systems and players expect fast responses to their actions as high latency could be detrimental to the user’s experience leading to player dissatisfaction which will eventually result in the player unsubscribing from the game. The persistence layer inevitably adds overhead to the server therefore the objective in this case is to minimize the additional cost as experienced by a client when executing an action, e.g. moving, fighting or picking up items in the word.
Related Content

Carrier-Grade Distributed Cloud Computing: Demands, Challenges, Designs, and Future Perspectives
[www.irma-international.org/chapter/carryer-grade-distributed-cloud-computing-82541/](www.irma-international.org/chapter/carryer-grade-distributed-cloud-computing-82541/)

Predicting Room-Level Occupancy Using Smart-Meter Data
[www.irma-international.org/article/predicting-room-level-occupancy-using-smart-meter-data/188856/](www.irma-international.org/article/predicting-room-level-occupancy-using-smart-meter-data/188856/)

Granular Synthesis of Rule-Based Models and Function Approximation Using Rough Sets
[www.irma-international.org/chapter/granular-synthesis-rule-based-models/44713/](www.irma-international.org/chapter/granular-synthesis-rule-based-models/44713/)

SONATA: Social Network Assisted Trustworthiness Assurance in Smart City Crowdsensing
[www.irma-international.org/article/sonata/143903/](www.irma-international.org/article/sonata/143903/)
Utilizing an Augmented Reality System to Address Phantom Limb Syndrome in a Cloud-Based Environment
www.irma-international.org/article/utilizing-an-augmented-reality-system-to-address-phantom-limb-syndrome-in-a-cloud-based-environment/181033/