

A Scalable Architecture for Multiplayer Computer Games

Jens Müller and Sergei Gorlatch
Westfälische Wilhelms-Universität Münster
{jmueeller|gorlatch@uni-muenster.de}

Abstract: The concept of Massively Multiplayer Games (MMG) recently has spread into all classical genres of real-time computer games. This paper summarizes our work on a novel proxy server-network topology which provides the required *scalability* to enable massive multiplayer gaming in the genres of First Person Shooter (FPS) and Real-Time Strategy (RTS) games. Besides scalability, i.e., the ability to maintain the game service with an increasing number of participating players, our topology provides a high degree of *responsiveness* which is necessary to support the fast-paced game play of contemporary game designs. This paper compares our proxy server-network to network topologies commonly used in FPS and RTS games. We present the results of an analytical scalability model which allows to forecast maximum player numbers for a given game and discuss experimental results.

1 Introduction

In comparison to traditional computer game designs only suitable for a comparable low number of players, the concept of Massively Multiplayer Games (MMG) provides suitable game designs for hundreds of participating users in a huge game world. There has been a lot of research in the area of scalable network distributions and topologies suitable for the popular genre of Massively Multiplayer Online Role Playing Games (MMORPG), e. g. [CXTL02].

However, such popular game genres as First Person Shooter (FPS) and Real-time Strategy (RTS) have barely been adopted to the MMG concept so far, although they often require the participation of many players. Games of these genres commonly make use of a client-server or peer-to-peer network topology, which do not provide the required scalability for MMG. Therefore, although current commercial team-based FPS games like *Battlefield 1942* [Ar02] provide huge game worlds for a high amount of participating players, the maximum number of players usually is limited to only 64 users.

In [MFGM04], we presented a proxy server topology which provides the required degree of scalability and responsiveness for MMG sessions in FPS and RTS games. We describe this topology in Section 2 of this paper. We compared the scalability of the client-server, peer-to-peer and proxy topology using an analytical model of scalability. In this evaluation, the proxy topology is the only network concept which allows MMG sessions in FPS and RTS games, as we summarize in Section 3. Finally, Section 4 compares our concept to related work and concludes the paper.

2 The Proxy Server-Network

In our proxy topology depicted in Fig. 1, several proxy servers are interconnected and share the same view on the game state. Clients are connected to a single arbitrary proxy in order to participate in the game.

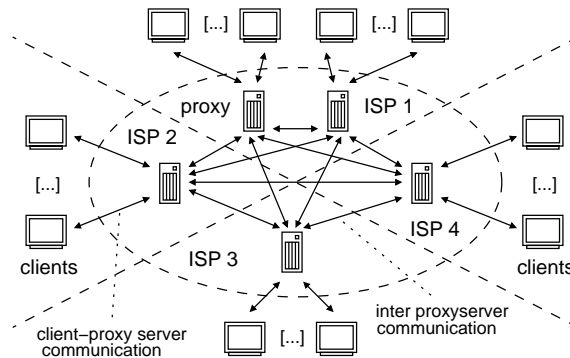


Figure 1: Example session using the proxy server architecture

Clients communicate with their proxy to send user actions and receive state updates. The proxy servers process and forward user actions to other servers in order to synchronize the replicated game state. For the inter-proxy server communication, IP- or application level multicast is used. In order to achieve low communication latency to clients in Internet based games, proxy servers should reside at different Internet Service Providers (ISP). Clients then are able to connect to a proxy at its ISP used for the dial-up Internet connection which results in a comparatively low latency because of a short packet route. This way, comparatively high responsiveness of the game is achieved.

The currently used network topologies for MMORPG partition the game world among several servers which requires clients to change server connections when the user moves into another region of the game world which is not maintained by the current server. In the fast-paced game play of FPS and RTS games, such reconnections during games would be annoying for users. Additionally, the server can only inform clients about the segment of game state it is responsible for and therefore clients do not receive informations about adjacent game regions.

In contrast to this partitioning concept, our topology replicates the game world at all proxies. This allows a proxy to serve arbitrary game clients regardless of their game state or position in the game world and to inform clients about the whole state of the game. However, the replicated data has to be synchronized between all participating proxies.

For the synchronisation of the replicated game state, we use the concept of *eventual consistency* [TvS02]. The proxy server a client is connected to is the only instance which is allowed to alter state informations of that client. After a proxy has changed a client's state, it sends appropriate information to all other servers which update their replicated copies accordingly. This way, consistency of the replicated game state is ensured.

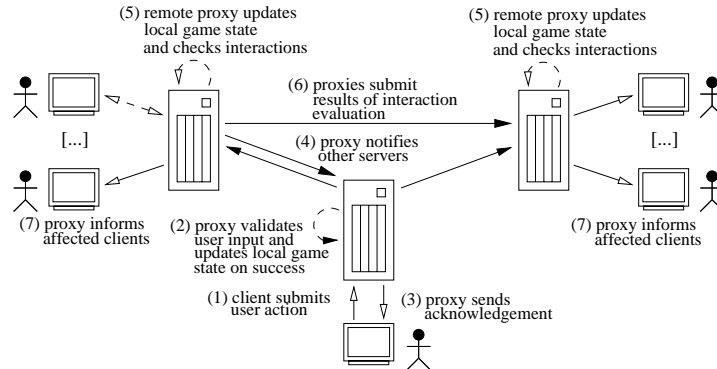


Figure 2: Processing and forwarding a single user action

Figure 2 depicts the steps in the processing and forwarding of a single user action in the proxy topology. After a client has submitted the action to the server (1), the proxy validates the action and updates its local game state (2), and sends an acknowledgement to the client (3). To ensure consistency of the replicated state of that particular client, the server transmits the new state to all other proxies (4) which update their local copies and check whether the client action affects users of their local clients (5). If a user is affected, e.g. by being hit from a shot in a FPS game, the appropriate proxy notifies the other servers in turn (6). Finally, all clients to which the state updates are visible are informed (7).

In [MFGM04], we demonstrated that the game correctness provided by this concept is comparable to the correctness provided by the commonly used client-server topology. In both topologies, there are situations where the order of processed state updates differs from the order of initiation of the corresponding actions by users. However, these situations occur seldom in comparison to the total amount of state updates in a game session. Additionally, the concept of *timewarps* [MVHE04] can be applied to the proxy server topology and further reduce the number of such situations.

3 Scalability Analysis and Experimental Results

In the computer games context, the scalability of a game service architecture is the ability to maintain service when the number of participating players increases. In a running game session, two kinds of resources are utilized by the server processes: (1) *computational resources* in form of computing cycles for the calculations of the game simulation, and (2) *communicational resources* in form of network bandwidth utilized for the transmission of state updates between clients and servers.

We developed an analytical model for scalability in the client-server, peer-to-peer and proxy topology for the comparison of these concepts. The simulation of real-time computer games is usually done at certain points in time which discretizes the actual continuous application. The frequency at which state updates are calculated is commonly referred

to as the *tickrate* t ; the maximum time available for the calculation of a state update therefore is determined as $1/t$ seconds.

In short, the model subdivides the overall utilization of computational and communicational resources in the calculation of the new game state at each tick into several basic tasks. These tasks like the validation and processing of user inputs or the transmission of updated state information to other participating processes occur in all of the considered topologies such that the resource utilization for each task can be directly compared.

In the comparison of the topologies based on the model, the proxy network is the only architecture which provides the required scalability for MMG sessions in the FPS and RTS genre. Furthermore, our analytical model allows to forecast the maximum number of supported clients which is reached when the calculation of the state update at the processes maintaining the game state takes nearly the complete available time of $1/t$ seconds.

We implemented a test game in order to verify our model. In this game, avatars (the virtual representations of players), each controlled by a single client, walk around in a game map and try to catch each other. Although this game is quite simple, its basic design is comparable to sophisticated FPS games. The test game was run at a tickrate of 25 updates per second, which is a common value for FPS games in Internet-based sessions.

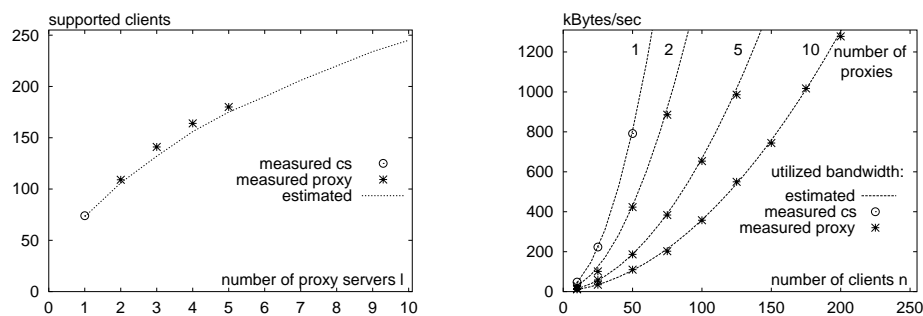


Figure 3: Experimental results for the test game at tickrate 25

Figure 3 compares the forecasting results of our model with the results of measurements: The curve in the left plot shows the estimated number of maximum players throughout different numbers of proxy servers; the actual measured numbers are depicted by the measuring points. The deviation is less than 7%. For the estimation as well as for the measurements, the worst case was assumed, i.e., every client performs an action each tick and all other avatars are visible to each other. The point \circ shows the maximum number of players in the client-server topology which is equal to the proxy topology with a single server. Due to the limited number of available test hosts for servers and clients, we only were able to run the game with a maximum of five proxies at the chosen tickrate. The curves in the right plot of Fig.3 depict the estimated amount of bandwidth utilized at a single proxy per second. The points show the measured values, the deviation is less than 3%. For the setup with ten servers, a lower tickrate was chosen and the obtained values were normalized to a tickrate of 25.

4 Conclusion and Related Work

The forecast using the analytical model as well as the experiments shows that the proxy topology provides the required scalability for massively multi-player sessions in the FPS and RTS genres. We expect that a more efficient and optimized game engine than our test game would allow FPS or RTS game sessions with up to 500 players using an adequate number of proxy servers.

In contrast to scalable architectures for *Distributed and Virtual Environments* (DVE) like DIVE [FS98], our architecture does not introduce additional delay for acknowledgment of actions which only depend on the round trip times to the ISP. This way, it is more suitable for games with direct, hand-eye coordinated movements like FPS, which require latencies below 150 ms [Ar01]. The presented proxy topology does not require reconnects of clients in a running session like the spatial partitioning commonly used in MMORPG [CXTL02], each client can connect to the proxy providing lowest communication latency. RING [Fu95] proposes a network topology similar to ours. However, RING is specially designed for densely occluded virtual environments, while our topology supports arbitrary environments in computer games. [KLXH04] proposes a peer-to-peer overlay in order to achieve the required scalability for MMG, but this approach mainly concentrates on MMORPG and hardly provides the required responsiveness for FPS games.

References

- [Ar01] Armitage, G.: Sensitivity of Quake3 players to network latency, imw2001 poster. 2001.
- [Ar02] Electronic Arts: Battlefield 1942 <<http://www.battlefield1942.ea.com/>>. 2002.
- [CXTL02] Cai, W., Xavier, P., Turner, S. J., and Lee, B.-S.: A scalable architecture for supporting interactive games on the Internet. In: *Proceedings of the 16th Workshop on Parallel and Distributed Simulation*. S. 60–67. Washington, D.C. May 2002. IEEE.
- [FS98] Frecon, E. and Stenius, M.: DIVE: A scalable network architecture for distributed virtual environments. *Distributed Systems Engineering Journal*. 1998.
- [Fu95] Funkhouser, T. A.: RING: A client-server system for multi-user virtual environments. In: *Symposium on Interactive 3D Graphics*. S. 85–92. 1995.
- [KLXH04] Knutsson, B., Lu, H., Xu, W., and Hopkins, B.: Peer-to-peer support for massively multiplayer games. In: *IEEE Infocom*. March 2004.
- [MFGM04] Müller, J., Fischer, S., Gortlach, S., and Mauve, M.: A proxy server-network for real-time computer games. To appear in proceedings of Euro-Par 2004. August 2004.
- [MVHE04] Mauve, M., Vogel, J., Hilt, V., and Effelsberg, W.: Local-lag and timewarp: Providing consistency for replicated continuous applications. *IEEE Transactions on Multimedia*. 6(1):47–57. February 2004.
- [TvS02] Tanenbaum, A. and van Steen, M.: *Distributed Systems: Principles and Paradigms*. Prentice Hall. 2002.