The Realization of a Parallel Serverless World: Applying Peer-to-Peer Architecture in Massive Multiplayer Online Games

Cody R. Brown[1]

SUMMARY

In this paper we look at the parallel challenges to implement a fully distributed Massive Multiplayer Online Game or a serverless world. The key issues considered for such a world are implementation, authentication, security storage, communication and load balancing within a peer-to-peer architecture. Because peers cannot rely on a centralized service for communication and storage, everything is distributed through the world. We discuss possible optimizations and interest management approaches to cluster peers and improve on peer-to-peer mutual checking and communications. Basic security and storage techniques are disused to realize an implementation. These tools are implemented on an MPICH-OPeN backbone for the underline message passing communications. The hope is to be able to obtain a very functional Massive Multiplayer Online Game without a reliance on expensive servers.

INTRODUCTION

The gaming industry is an extremely parallel oriented enterprises. A number of parallel component layers can be seen throughout the modern video game. This starts with specialized massively parallel hardware devices such as Graphics Processing Units (GPUs) that are designed to effectively handle and render polygonal based graphics. Further along, one finds message passing architectures implemented into the design of games. These were used extensively since the mid-nineties (26) because of the natural way message passing related to game design. Popular examples of parallel message passing architectures can be seen in simple well defined games such as Sudoku (7) and the Game of Life in Matlab's Parallel Computing Toolbox (4). The pure message passing architecture has been abandoned for more complicated games due to the thorny structure it added to the code. Bringing the problem out of a single computer, we see further parallel components bleed into distributed and network computing pertaining to network play and server related communications between the players or peers. To a larger scale, we can see parallel structure in Massive Multiplayer Online Games (MMOGs) that include a large number of peers playing the same application over network protocols such as TCP/IP. All these commercial MMOGs to date are implemented with a Client/Server (C/S) architecture (25; 15) and typically peers are split into small groups relative to the number of servers in the MMOG world (20).

This quickly leads us to a bottleneck situation. With the current implementation, MMOGs become scalable to the number of servers available, not to the number of peers. With the expensive price for the cost and maintenance of a server (20), a large amount of peers can become economically unrealistic. A server can only accommodate a set number of peers and thus a global maximum number is always present. A solution to cut costs and allow for the problem to be well scalable to the physical number of peers seems to be necessary.

A proposed solution to these problems is a distributed Peer-to-Peer (P2P) architecture for MMOGs. Since the millennium P2P architectures have received considerable attention within the literature especially with respect to simple tasks such as file transfer applications like Napster and Bittorent. They are known to be extremely scalable to the number of peers and organize easily into a coherent network (24). Using this framework would allow a true serverless world to be envisioned and also allow for many other benefits that could be utilize within this framework.

Without a server, peers will be able to communicate directly with each other, allowing for faster communication to direct neighbours. A distributed computational network can be established allowing the worlds physical objects to be computed by the peers inhabiting it. More complicated real-time simulations can easily be incorporated into the MMOGs and distributed over the peers such as true water, wind and sound simulations (9), aspects that have always been dumb-down because of finite computational resources. Without the reliance of a single server, the virtual world will always exist as long as there are peers within it; no necessary reliance to a single server.

There are many benefits to this concept of a serverless world, but applying the P2P framework to more complicated applications such as MMOGs quickly confronts us with some challenges. A few particular challenges involved with implementing this architecture into MMOGs include authentication, security, storage, communication and load balancing. We will look at recent approaches to solving these parallel problems and also look at MPICH-OPeN an implementation for this problem using MPI over an adaptive open P2P network. Analysing these approaches and inspecting the current tools available to us, we can see if the realization of a serverless world concept is plausible, and what the current challenges are to overcome for this realization.

IMPLIMENTATION FOR A SERVERLESS WORLD

The most parallel and important concept of a serverless world would be the middleware involved with implementing the idea. Peers need to be able to communicate and interact with static and dynamic objects within the world. These objects can be thought of like complex structures of data that need to be communicated with nearby peers (17). A message passing architecture seems to be a very logical choice to this problem: the realization of this over a larger network like TCP/IP falls into

[1]Seismic Laboratory for Imaging and Modeling,Department of Earth and Ocean Sciences, University of British Columbia, 6339 Stores Road, Vancouver, V6T 1Z4, BC, Canada

a distributed multi-process environment or a P2P network. As stated before, P2P networks have been well established to be scalable and coherent especially for other simple tasks such as file transfers (24). Message passing throughout the P2P network will allow objects to communicate to nearby peers and then be animated within the world. The parallel needs in establishing the communication within a serverless world match that of a P2P architecture very well. Implementing this architecture to handle more complicated and integrated spatial data is the greatest challenge (25), which could require massive amounts of information to be passed and stored throughout the world, or at least to the nearest neighbours. However, we will see later that specific optimizations can be applied for more efficient communication. Before going into further details about these communication and parallel problems, we will look at a solution to the implementation of such a parallel message passing P2P architecture.

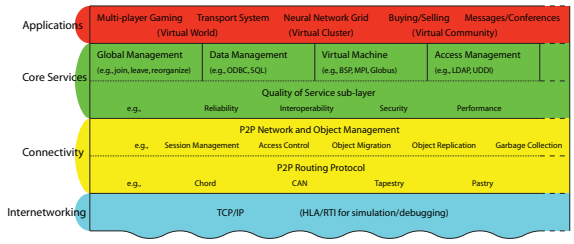

Figure 1: OPeN layered architecture (25).

There are many grid implementations to choose from such as MPICH-G2 (16). One specialized implementation we will consider is MPICH-OPeN (3) created at the University of Melbourne. It was specifically designed for parallel computing over an adaptive P2P network (19), and allocates distributed virtual machines to perform jobs through the dynamic P2P network. The basic design of this implementation is shown in Figures 1 and 2. The work is relatively new but experiments have been shown that this implementation is comparable to other MPICH distributions (19). For our MMOG serverless world concept, the main concerns with the middleware involve a simple, decentralized, scalable, reliable and well performed approach (17) for communication over a P2P network. MPICH-OPeN shows reasonably good performance in each of these aspects for a Message Passing Interface (MPI) over a P2P network (19). A middleware implementation will of course be interchangeable, but showing that these ideas are easily modular and are being actively discussed simplifies many of the underlining implementations of our serverless world. Middleware implementation ideas that focus particularly on an agent or database approach are discussed in (17), but it seems logical that a hybrid between these concepts will be best, and this is the design approach that can be utilized by MPICH-OPeN. CAESAR is a more recent middleware proposal that adds on extra components to the MPICH-OPeN which would be necessary to commercialize a complex service-oriented P2P application (8). This more useful but more complicated approach will not be nessasary for the scope of this report.

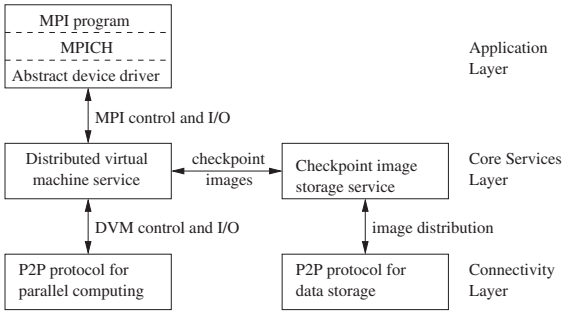

Figure 2: Design of MPICH-OPeN (19).

AUTHENTICATION

Now that we have a scalable and reliable implementation of a necessary P2P backbone, we can look at the parallel counterparts for the components necessary to implement an MMOG within our P2P network. The first main concern would be authentication within the world. Authentication is important in a number of ways: two particular points would include the publisher/subscriber model (11) and administration level of the MMOG (12). A way to grant access rights to individuals can become a complicated task for a serverless world. One possible approach is discussed in (12) that adds an Authentication Component (AC) to the P2P network. This AC can have keys which include the player_id and expiration_date (or equivalent information). This information is published on a Distributed Hash Table (DHT) accessible by all peers. This allows peers to authenticate each other and the AC to update the DHT when new information becomes available. This simple approach implements the idea, but also has a number of flaws that may need to be addressed; the most notable being the security responsibilities of the AC and requiring this AC to be trusted. The AC can be implemented as a permanent server or just as a special peer that logs into the world, which will allow us to keep to our serverless world concept. The addition of a DHT is also another security issue but is a necessary component that will be used for many different aspects of our serverless world. We will see the security responsibilities are not a large concern and that there are many techniques discussed to tackle this issue.

SECURITY

In almost any MMOG architecture a huge concern is the security of the game. Prevent cheating and malicious play provides an equal and fair experience for all peers. These simple tasks in a C/S architecture becomes extremely complicated within a distributed architecture. Particularly in our case where there is not necessarily a centralized server or component. This issue is probably the most causable concern for the commercialized world. Businesses need to be able to control and authenticate their world. Without the tools to do this, a serverless world is implausible for any application in a commercialized sense. One security model is discussed in (14) which involves encrypting the packets in the P2P network, sending it along a random path, with the true destination embedded in the en-

crypted packet. This allows one to protect sensitive identity data such as destination IP from the full world at the cost of a delayed arrival. In (15) four approaches are discussed for addressing further security concerns in a distributed environment such as a P2P network.

Distributed State Dissemination
Distributed state dissemination is the first approach discussed and involves the use of a centralized server to process action requests against the state updates of the peers. This approaches cheating similar to a C/S style and stores the full game state on a centralized peer. Even though this provides a solid attack against cheating, it also strays away from our serverless world concept. With a permanent security server reading random game state information from the world, we now need to maintain and service this server. The global design still incorporates a parallel P2P structure and thus greatly reduces bandwidth and computational costs of such a security component, meaning our approach will still be scalable to the number of peers. Furthermore, in reality the number of physical servers necessary to apply such a security feature would be minimal (15). Later we see this hybrid approach may include more benefits that might make looking at a semi-serverless world concept more appealing.

Mutual Checking
The second approach involves Mutual Checking (MC) of nearby peers to validate game state. In this approach every peer acts like a server or Region Controller (RC) for a small cluster of peers, such as virtual regions within the game world. In the idea from (15) they discuss assigning these RCs with trusted peers or servers. But like the AC we can easily implement this within a servereless setting as well. The basic idea behind this concept is: you may not trust a single client, but you trust the consensus of multiple unaffiliated clients (15). This can easily be realized by automatic checking of small game state information between neighbouring peers. Of course with a semi-serverless world, this idea could become more stable and still allow us to utilize the benefits from a distributed P2P architecture. This clustering of specialized peers will also allow us to utilize optimal P2P region based communications which will be discussed later.

Log Auditing
Similar to the first approach, this requires external servers not a part of the world, which lessens a fully distributed P2P layout. The basic concept involves game states to be stored as logs during particular times. These logs are analysed during non-peak hours to look for discrepancies. The main point here is not to prevent cheating in real time, which would create many load-balancing issues (15) but to store these logs and detect cheaters later. This concept can be applied to a more distributed architecture for the analysis of the logs if needed, but a centralized server would always be necessary to store and maintain these logs. Again the necessary server components for this concept would be minimal and can be combined with other security server components.

Trusted Computing
The final approach discussed involves the concept of Trusted Computing (TC) (15; 14). This is a controversial initiative of the Trusted Computing Group (5), which is not a parallel computation component so will not be discussed in detail. However, the idea involves hardware validation of a computer to ensure the software has not been tampered with. Some less secure software approaches also work here and have already been implemented in C/S MMOGs such as Maple Story (2). Trusting that the client of the peer has not been tampered with would allow more confidence for the peers in the distributed P2P network. With this we could even allow some trivial data to be stored locally, reducing the game state load for the world, which would be extremely useful especially when minimal peers are available.

STORAGE

A great concern with a serverless world would be the storage of data. The simple approach would be to consider a semi-serverless world which would allow a storage server to simply store the game state and distribute this throughout the peers when necessary. A way to implement this Storage Compnent (SC) is discussed in (12). This approach can be applied to a purely distributed sense as well storing the data in a DHT, and distributing over the number of peers. Applying concepts such as MC, and assuming a large number of peers are available at any given time, then this provides a reasonable way to implement the serverless world. With a large number of peers, small game area clustered groups could store localized game data through a DHT and use MC to verify this data. As long as there are sufficient peers, enough to make many small groups, this concept would be easy to implement, and no peer would have to store or transfer a large amount of data. Losing one peer would allow the MC peers to contact the next nearest peer (as pertaining to a neighbour or region based interest management approach discussed further) to store a redundant copy of the data. Likewise the addition of peers would reduce the redundant copy. The trouble comes when minimal peers are available, making cluster groups have a small number of peers but large coverage of game area. The amount of game data to store would become very large and break this formulation. The only way to combat this would be to include an SC or reduce security checking.

COMMUNICATION AND LOAD BALANCING

As stated before, communication is sent through message passing in a P2P environment. The middleware implementation of this is discussed above, but providing specialized interested management clusters will allow us to more efficiently communicate throughout the world (21). Two popular approaches include neighbour based interest management and region based interest management (10). The concepts of a region based approach allows us to incorporate many of the other ideas more easily into the platform. Other interest management concepts are discussed in (18) but the region based approach works well in our setting. Allowing the P2P network to cluster with respect to virtual regions can assist us with the storage issue, and allows use of a simple approach to implement our MC secu-

rity model. An implementation of this region based approach is discussed in (10) and is implemented on an already established P2P algorithm Chord (22). Further implementation and requirement issues are discussed in (10) and show this to be a fast algorithm for communication to relevant peers within the game region of your world. Further analysis using a Chord approach is found in (25) which include specifics on implementation details, distributed hashing, distribution of objects, algorithm and complexity analysis and necessary layers to implement a serverless world. You may see a simple diagram illustrating the method in Figure 3. The indexing in (25) involved octrees and distributed hash tables, but a more recent approach involving quadtrees is discussed in (23; 24).

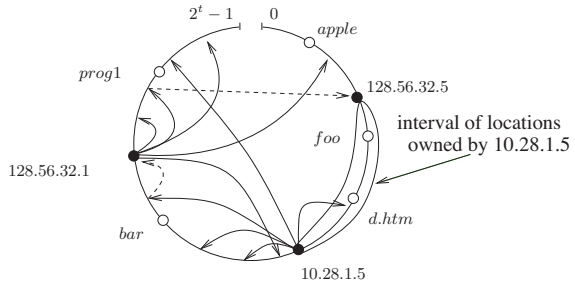

Figure 3: The Chord method (25).

Global event handling within the MMOG can easily be achieved with N-Trees (13). With this event handling scheme, we can divide events within regions by N-equal squares, using a structured ordering within the tree. The global events will be stored in a DHT, and placed within a specialized N-Tree. This is demonstrated with a quadtree in Figure 4. The benefit of this further clusters regions together for quick communication between relative peers for shared regional events. The proposed structure provides very good asymptotic analysis for the most common operations (13). More details about the specific implementation and structure of N-Trees can be found in (13).

The load balancing with the proposed implementation focuses on our region based interest management approach. This approach can easily be enhanced if desired and interesting concepts for tweaking such a management approach that relates to our MMOG P2P environment are discussed in (6).

CONCLUSTIONS AND FUTURE WORK

The necessary tools to implement a pure serverless world seem to be available and regularly discussed in the literature since early in this millennium. The application of a pure serverless world is more suitable to MMOGs that will always have a large set of peers and low security requirements. However in a commercialized setting, the security issue would be an imported aspect and the implementation of a hybrid approach or semi-serverless world would have to be adopted instead. The semi-serverless world still allows most of the benefits of a pure serverless world, making the architecture scalable to the number of peers in the world, not to the number of servers. Furthermore it would greatly reduce the number of servers re-

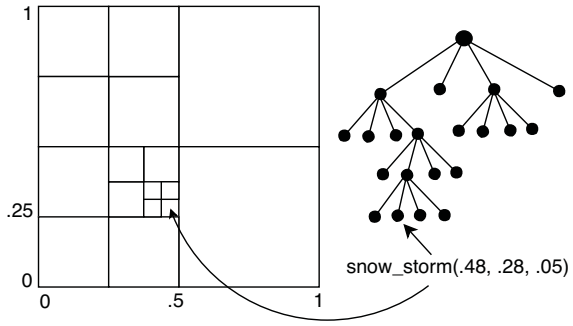

Figure 4: Cartesian applicaion space and quadtree representation. In this example, a snow storm occurs at specified location and radius (13).

quired to operate MMOGs on a global scale, requiring only a few specialized servers or components such as an AC, SC and RC to authenticate, store and distribute regional tasks throughout the network. More specialization with future components is also possible and provides future security and load balancing improvements to the world. The main benefit of a semi-serverless world involve the fact that we remove the bottlenecks and high costs of the standard C/S model which is commonly used throughout the commercialized world currently.

This hybrid idea is not new and a similar proposal can be found in (11). Other hybrid approaches can also be seen in practice, such as the Great Internet Mersenne Prime Search (GIMPS) (1) which uses a distributed multi-process environment with a central authentication component to delegate tasks throughout the network. This seems to work well with computationally expensive tasks such as prime factor searching. Our current model does not include any such high computation costs, but future work can easily be adopted into our MMOG model to provide realistic simulations of the static environmental events on the client level. This would allow computation costs to be distributed over the region so that all peers could benefit from high quality simulations that presently cannot be included in current game implementations because of the limited resources available to a single client. With the implementations discussed above and the exciting extensions that are available in a distributed approach, this will allow a fully distributed semi-serverless world to become realized within the MMOG community and persuade others to adopt this effective distributed framework.

REFERENCES

[1] Great internet mersenne prime search. URL: `http://www.mersenne.org/`.

[2] Maple story. URL: `http://www.maplestory.com/`.

[3] Mpich-open. URL: `http://p2p.cs.mu.oz.au/software/mpi-open/`.

[4] Parallel computing toolbox. URL: `http://www.mathworks.com/products/parallel-computing/`.

[5] Trusted computing group. URL: `http://www.trustedcomputinggroup.org/`.

[6] M. E. Ali, E. Tanin, R. Zhang, and L. Kulik. Load balancing for moving object management in a p2p network. In *Proceedings of Database Systems for Advanced Applications - DASFAA*, pages 251–266, 2008.

[7] H. Bauke, A. Gorlich, S. Korada, C. Measson, T. Mora, and O. Rivoire. Sudoku: An application of message passing, 2006.

[8] L. Chan, S. Karunasekera, A. Harwood, and E. Tanin. Caesar: middleware for complex service-oriented peer-to-peer applications. In *MW4SOC '07: Proceedings of the 2nd workshop on Middleware for service oriented computing*, pages 12–17, New York, NY, USA, 2007. ACM.

[9] K. V. D. Doel. *Sound synthesis for virtual reality and computer games*. PhD thesis, University of British Columbia, 1999. Supervisor-Diresh K. Pai.

[10] S. Douglas, E. Tanin, A. Harwood, and S. Karunasekera. Enabling massively multi-player online gaming applications on a p2p architecture. In *In Proceedings of the IEEE International Conference on Information and Automation*, pages 7–12. IEEE, 2005.

[11] S. Fiedler, M. Wallner, and M. Weber. A communication architecture for massive multiplayer games. In *NetGames '02: Proceedings of the 1st workshop on Network and system support for games*, pages 14–22, New York, NY, USA, 2002. ACM.

[12] C. Gauthierdickey and V. Lo. A fully distributed architecture for massively multiplayer. In *Online Games, in ACM NetGames Workshop*. ACM Press, 2004.

[13] C. GauthierDickey, V. Lo, and D. Zappala. Using n-trees for scalable event ordering in peer-to-peer games. In *NOSSDAV '05: Proceedings of the international workshop on Network and operating systems support for digital audio and video*, pages 87–92, New York, NY, USA, 2005. ACM.

[14] A. Harwood and S. Kulkarni. Delay sensitive identity protection in peer-to-peer online gaming environments. In *International Workshop on Peer-to-Peer Networked Virtual Environments*, page 1, 2007.

[15] P. Kabus, W. W. Terpstra, M. Cilia, and A. P. Buchmann. Addressing cheating in distributed mmogs. In *NetGames '05: Proceedings of 4th ACM SIGCOMM workshop on Network and system support for games*, pages 1–6, New York, NY, USA, 2005. ACM.

[16] N. T. Karonis, B. R. Toonen, and I. T. Foster. Mpich-g2: A grid-enabled implementation of the message passing interface. *J. Parallel Distrib. Comput.*, 63(5):551–563, 2003.

[17] S. Karunasekera, S. Douglas, E. Tanin, and A. Harwood. P2p middleware for massively multi-player online games. In *In Demonstration Proceedings of the ACM Middleware Conference*, 2005.

[18] G. Morgan. Scalable massively multiplayer onlne games. Technical report, University of Newcastle, 2005.

[19] L. Ni and A. Harwood. An implementation of the message passing interface over an adaptive peer-to-peer network. In *In Proceedings of the 15th IEEE International Symposium on High Performance Distributed Computing*, pages 371–372. IEEE, 2006.

[20] D. Pittman and C. GauthierDickey. A measurement study of virtual populations in massively multiplayer online games. In *NetGames '07: Proceedings of the 6th ACM SIGCOMM workshop on Network and system support for games*, pages 25–30, New York, NY, USA, 2007. ACM.

[21] J. Smed, T. Kaukoranta, and H. Hakonen. A review on networking and multiplayer computer games. Technical report, Turku Centre for Computer Science, 2002.

[22] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 149–160, New York, NY, USA, 2001. ACM.

[23] E. Tanin, A. Harwood, and H. Samet. Using a distributed quadtree index in peer-to-peer networks. *The VLDB Journal*, 16(2):165–178, 2007.

[24] E. Tanin, A. Harwood, H. Samet, D. Nayar, and S. Nutanong. Building and querying a p2p virtual world. *Geoinformatica*, 10(1):91–116, 2006.

[25] E. Tanin, A. Harwood, H. Samet, S. Nutanong, and M. T. Truong. A serverless 3d world. In *GIS '04: Proceedings of the 12th annual ACM international workshop on Geographic information systems*, pages 157–165, New York, NY, USA, 2004. ACM.

[26] D. Weinstein. The case for message passing architectures, 2005.