Mayhem in Online Game and Virtual Environment Development

Manuel Oliveira, Jon Crowcroft, Mel Slater

Department of Computer Science, University College London, Gower Street, London WC1E 6BT, UK

{m.oliveira, jon, m.slater}@cs.ucl.ac.uk

Abstract

We present Mayhem, which is a prototype of a Massive Online Role Playing Game (MORPG), where novel research in the field of Virtual Environments (VEs) is combined with the requirements of online games. This results in a novel approach to develop VE systems. The paper analyses the building blocks of the underlying system supporting Mayhem.

The system design is based on component frameworks, with the core infrastructure of the game revolving around the Java Adaptive Dynamic Environment (JADE). We discuss the main extension Modules supporting the system, such as the networking module and the data manager.

1 Introduction

The computer graphics community has consistently provided the computer games industry with inspiration and knowledge to build games designed to catch the attention of the mass market.

The new trend in the game industry is to exploit interconnectivity between users, thus promoting the emergence of online games, such as Ultima Online (UO)[1], EverQuest (EQ)[2] and Asheron's Call (AC)[3]. These systems allow users to share experiences similar to a Virtual Environment (VE). However, unlike with graphics, the games industry has neglected the knowledge or the research value of the VE community. Consequently, the same mistakes are being made again.

2 Online Game vs Virtual Environment

An online game benefits significantly from a large budget investment, most of which is channeled to the production of high quality content.

When we compare online games and VE, it is possible to see that their goals are very similar. Both aim at suspension of disbelief, thus engaging the users in an enjoyable experience. System scalability, in terms of number of users and content size, represents one of the prime objectives.

The fact that online games must be available on a daily basis, week after week, makes persistency and robustness a high requirement on the list. These characteristics are the main strengths to promote cybercommunities.

2.1 Online Game Approach.

The current recipe used for online games development is to adopt the client/server architecture for the infrastructure.

The server dictates the current state of the world, by managing all generated events either by Non Player Characters (NPCs) or end-users. The client collects user feedback and sends all events generated locally to the server. At any particular point in time, a client provides a limited view of the VE to the user. Consequently, it is possible for the server to send each client the relevant events filtered according to their interest.

This approach makes the server a bottleneck in terms of computational and network resources. This means a low threshold is imposed upon the number of simultaneous players supported, since disruption of service due to system overload is not a viable option.

Since every event generated at a client needs to be sent to the server before being distributed to all the remainder clients, latency and jitter become a significant problem. This not only affects user responsiveness, but also the design of the game (i.e.: convoluted streets in cities), which ultimately may cause frustration in terms of playability.

Although the server guarantees total consistency of the VE datamodel, it leads to central point of failure for service provision. Consequently, external activities, such as Denial Of Service (DoS) attacks, represent serious problems in maintenance of online games.

Despite all the associated problems, the client/server architecture has an associated low development complexity and eases deployment of the system. In addition, the possibility of tracking all the users, events and state of the

world, eases the maintenance of the service along with problem resolution.

2.2 Virtual Environment Approach

The research done by the VE community has addressed most of the above problems that plague client/server architectures by exploring alternatives such as distributed and hybrid models. However, a new set of problems arises such as consistency, security, object ownership and deployment. A wide range of different solutions exists.

In VE, the infrastructure architecture model influences the datamodel and its associated mechanisms. The datamodel may be partially or totally replicated across the participating clients. Whenever an event is generated locally, it is disseminated to the remainder of the participating clients. However, the interest of a client is constrained to their immediate surrounding environment. Since there is no server that keeps track of all users, it is not feasible for each client to keep track of all the existing clients within its area of interest. Thus clients are clustered together according to their interest, forming groups that exchange events and state updates of the VE - Area Of Interest Management (AOIM) [4,5,6].

Regarding how the state is distributed, this is delegated to the network by adopting multicast as the communication model. The downside is that deployment of the technology has been slow and problematic[7]. It would be possible to implement distribution via application relays. However, such solutions would experience similar problems to the client/server bottlenecks.

The complexity of a distributed system is much higher than in the case of one dependent on client/server architecture. Traditionally, no software engineering practices are adopted in the development of VE systems, resulting in solutions that are difficult to extend and modify. So the learning curve associated to each solution is quite steep. This leads to the development of a new system each time a new research idea requires implementation.

3 Building Blocks

The dominating development trend in the VE community has resulted in a wide proliferation of systems. The impossibility of code interoperability makes it difficult if not impossible to build customized systems from existing components, building the remaining ones specific to the application. However, modularity begins to shape some of the current VE initiatives, such as Maverick[8], VRJuggler[9], Bamboo[10] and JADE[11].

The Virtual Reality Transport Protocol (vrtp)[12] is an attempt of promoting consensus towards facilitating code interoperability. The initiative provides guidelines for development of VE systems by proposing a common underlying component framework.

The vrtp proposal does not enforce the adoption of all components. In fact, the components described may be aggregated into three layers of flexibility:

- **Infrastructure**. This corresponds to the lowest layer of the framework. The layer consists of a single component, denominated Universal Platform (UP), which provides dynamic runtime extension and management of the remaining components.
- **Middleware**. This layer consists of a set of components with non-application dependent functionality. Some flexibility is exchanged for productivity. These components may be classified as Server, Client and Monitor.
- **Application**. At this layer, it is assumed that the user may use the system as is, without any technical development effort.

3.1 Java Adaptive Dynamic Environment

The Java Adaptive Dynamic Environment (JADE)[11] is the basis for the UP, providing the functionality described in the vrtp proposal.

To provide a flexible and adaptive infrastructure, JADE is based on a component framework where everything is derived from Modules. As depicted in Figure 1, these Modules may be managed by ModuleManagers, which are responsible for their unloading, loading and reloading. In case a Module needs to exist within its own processing thread, then it is considered a RunnableModule. JADE is not dogmatic about how a VE system is designed and implemented into components, rather this responsibility is delegated to the developer.

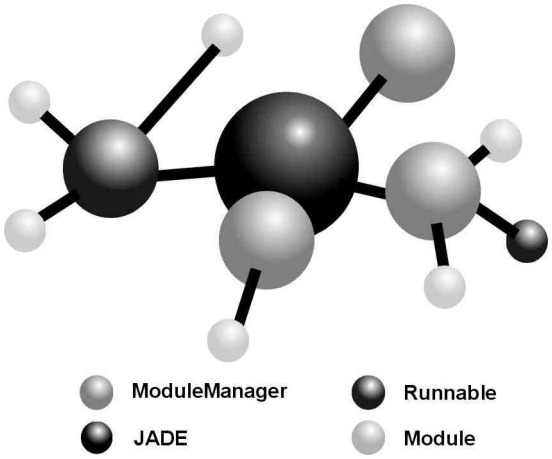

Figure 1: *JADE hierarchical organisation*

JADE provides a consistent namespace where each Module is uniquely identified. This allows for any Module to be retrieved for direct invocation of its methods. Alternatively, JADE provides an event model to decouple Modules from each other.

To support the basic functionality of JADE, several helping components exist, such as the ResourceLocator, which provides the means for a VE to retrieve and handle any resource, independently of its location.

3.2 TreacleWell

Although JADE uses networking to retrieve resources from across the network, it does not provide networking functionality to VE systems. This functionality is provided by an extension Module denominated TreacleWell (TW)[13], which belongs to the middleware layer as described by vrtp.

The design approach of TW is to provide a network component framework that breaks the traditional "black box" paradigm by adopting Application Level Framing (ALF)[14]. This brings the application into the protocol processing of the data.

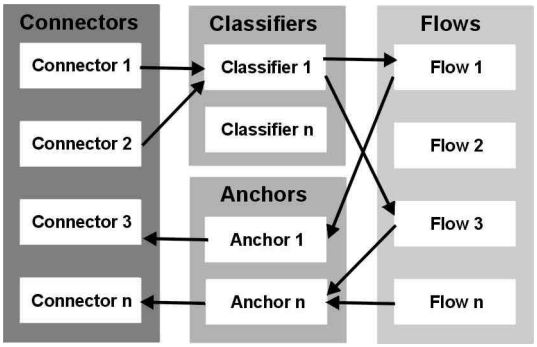

Figure 2: *TreacleWell's 3-tiered framework*

Absent in Figure 2 of the framework, is the Message abstraction. This provides the means by which data may transverse TW, being manipulated by the various elements.

Connectors
The connectors, coupled with the Message abstraction, provide TW isolation from the details of the network access point. In most cases, this will correspond to an IP socket, but it is possible to encapsulate a native device driver with other network protocols besides IP.

Classifiers
This tier affects the incoming messages, classifying them according to the associated criteria. A classifier may be associated to more than a single Flow, placing in the corresponding MessageBuffer for processing.

Anchors
The Anchors share an analogous role to Classifiers, but the message flow is outgoing.

Flows
As illustrated in Figure 3, associated to each Flow is a MessageBuffer where a Classifier deposits Messages. The Flow itself is composed by at least one FlowElement, which process Messages upstream or/and downstream.

The MessageBuffer itself may influence how Messages are inserted. The current TW implementation provides a simple round-based buffer and a time bucket synchronisation buffer. A Flow is available in two flavours depending upon how Messages are processed, either clock or thread based.

A FlowElement may correspond to a single network protocol or to a set of more FlowElement, which is denominated a FlowContainer.

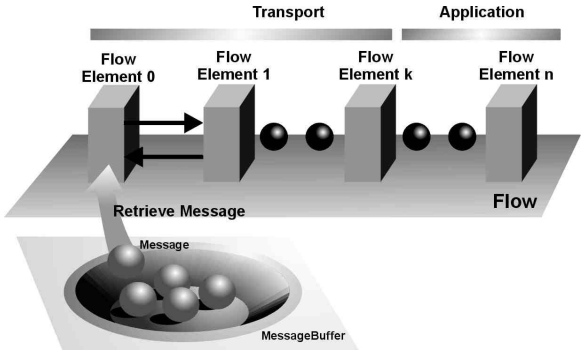

Figure 3: *Internal structure and building blocks of a flow*

The current implementation provides different containers to accommodate different protocol structures:

- **SequenceContainer.** An ordered set of FlowElements that is connected together in linear fashion.
- **ParallelContainer.** A set of FlowElements process in parallel the same Message that is received by the FlowContainer.
- **MuxContainer.** Similar to ParallelContainer, but only one of its FlowElements processes the Message passed to the MuxContainer. The selection of the appropriate FlowElement is permissible by labelling each Message.

The way a protocol designer fits together the different elements of TW determines the combined namespace of all the Application Data Units (ADUs) from the existing Flows.

3.3 Meta Uniform Datamodel

The Meta Uniform Datamodel (MUD) is another JADE extension module that provides a framework to structure datamodels based on the Model-View-Controller pattern, as illustrated in Figure 4.

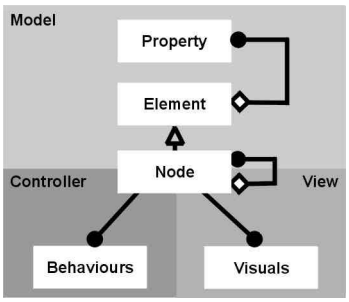

Figure 4: *The Model-View-Control pattern is the core of the Meta Uniform Datamodel*

The framework eases the evolution of existing Modules, since the structure of the datamodel does not undergo radical

changes. In addition, client heterogeneity is permissible by allowing the existence of different Visuals. As an example, one client may support a tile engine while another has a 3D rendererer, but both clients interact together.

4 Mayhem

The Mayhem project consists of the development of a Massive Online Role Playing Game (MORPG), which is an experimental testbed novel VE research ideas.

The game genre is open, meaning that no objectives are set for the users to achieve. Thus, a user may set their own goals and enact according to the character they are playing. So, a user more inclined to socializing may live in the virtual cities and interact with other users (and NPCs) while an achiever may become an adventurer and pursue quests.

4.1 Overview Architecture

The infrastructure supporting Mayhem relies on a hybrid architecture, with three-tiered model for offline services and distributed for real-time interaction.

Offline services consist of character creation, search for other users, etc. The common requirement of all these services is reliability, thus the architecture model is a three-tiered, where the server brokers the information stored in a relational database. The bottlenecks associated with client/server architectures are not significant because at any given instance it is expected that only a reduced set of the user base will be operating transaction services.

Once the user effectively enters the VE, a game session begins where the main requirement is real-time interaction. At any given instance, most of the users who are online will be operating in this mode. Therefore the workload is distributed amongst all the clients.

4.2 Server

The server is the "well-known" portal to the VE, managing the database and offline services.

The server uses JADE, which allows upgrades and system modifications at runtime without disruption of the entirety of the services available.

When a user logs on, their client receives from the server the current configuration file, allowing them to build dynamically their system. Some details depend on user selection, such as selection of the rendering engine (i.e.: tile based or 3D).

The server currently centralises all the content resources of the game, which are retrieved on demand by the clients. However, it is possible to make retrieval be based on http and distribute the content across several web servers.

4.3 Client

The client provides the user a virtual window to the VE while collecting their feedback. Since the client uses JADE, it is possible for the system to be different each time a user enters the VE. Traditionally, the area where hackers have the greatest impact is regarding the game protocols. In Mayhem it is permissible to modify periodically the game protocols making reverse engineering more difficult.

The current AOIM is based on static grid partitioning. However, the Module responsible for AOIM accommodates any interest policy, provided its interfaces are respected. The AOIM Module manipulates all the real-time connectors of TW based on the interest policy in vigor.

5 Conclusions

The development of Mayhem is an ongoing project where the latest VE research is used to support online games.

TW provides the capabilities for a VE system to monitor the state of the network and adapt at runtime by modifying the relevant network components.

With JADE as the core of the system, it is possible for Mayhem to continuously evolve seamlessly in terms of code maintenance and upgrades. The security problem associated a distributed approach is reduced by the authentication mechanisms within JADE and the possibility of periodically modifying the game protocols as deemed necessary.

Mayhem provides the ideal testbed to experiment new ideas regarding online games and VE.

Acknowledgements

This work is sponsored by the Portuguese Foundation for Science and Technology.

References

1 http://www.uo.com

2 http://www.eq.com

3 http://zone.msn.com/asheronscall/default.asp

4 Anderson, D., Barrus, J., Howard, J., Rich, C. and Waters, R., "Building Multi-User Interactive Environments at MERL", IEEE Multimedia, Winter 1995, pp. 77-82

5 Greenhalgh, C. and Benford, S., "MASSIVE: A Collaborative Virtual Environment for Teleconferencing", ACM Transactions on Computer Human Interfaces, Vol. 2, N. 3, pp. 239-261, September 1995

6 Macedonia, M., Zyda, M., Pratt, D., Brutzman, D. and Barham, P., "Exploiting Reality with Multicast Groups: A Network Architecture for Large-Scale Virtual Environments", Proc. IEEE Virtual Reality Annual International Symposium (VRAIS'95), North Carolina, March, 1995

7 Diot C, Levine B, Lyles B, Kassem H and Balensiefen D, "Deployment Issues for the IP Multicast Service and Architecture", IEEE Network Magazine special issue on Multicasting, January/February 2000

8 Hubbold, R., Cook, J., Kaetes, M., Gibson, S., Howard, T., Murta, A., West, A. and Pettifer, S., "GNU/MAVERIK A Micro-Kernel for Large-Scale Virtual Environments", Proc. VRST'99, London, December, 1999

9 Bierbaum, A., Just, C., Hartling, P., Meinert, K., Baker, A. and Cruz-Neira, C., "VR Juggler: A Virtual Platform for Virtual Reality Application Development". Proc. IEEE VR 2001, Yokohama, March 2001.

10 Watsen, K. and Zyda, M., "Bamboo – A Portable System for Dynamically Extensible, Real-Time, Networked, Virtual Environments", Proc. IEEE Virtual Reality Annual International Symposium (VRAIS'98), Atlanta, March 1998

11 Oliveira, M., Crowcroft, and Slater, M., "Component Framework Infrastructure for Virtual Environments", Proc. Collaborative Virtual Environments (CVE), San Francisco, September, 2000

12 Brutzman, D., Zyda M., Watsen, K. and Macedonia, M. "Virtual Reality Transfer Protocol (vrtp) Design Rationale", Workshops on Enabling Technology: Infrastructure for Collaborative Enterprises (WET ICE): Sharing a Distributed Virtual Reality, MIT, Cambridge Massachusetts, June, 1997

13 Oliveira, M., Crowcroft, J. and Slater, M., "TreacleWell: Unraveling the Magic "Black Box" of the Network", submitted for publication

14 Clark, D. and Tennenhouse, "Architectural Considerations for a New Generation of Protocols", Proc. SIGCOMM'90, Philidelphis, September 1990, pp. 200-208