[DMLSt04b] *Vasilios Darlagiannis, Andreas Mauthe, Nicolas Liebau, Ralf Steinmetz;* **Distributed Maintenance of Mutable Information for Virtual Environments;** IEEE International Workshop on Haptic Audio Visual Environments and their Applications - HAVE'04, Ottawa, Oktober 2004. S.

Distributed Maintenance of Mutable Information for Virtual Environments

Vasilios Darlagiannis Andreas Mauthe Nicolas Liebau Ralf Steinmetz
Darmstadt University of Technology
Multimedia Communications (KOM)
Merckstr. 25, 64283 Darmstadt, Germany
{*Vasilios.Darlagiannis, Andreas.Mauthe, Nicolas.Liebau, Ralf.Steinmetz*}*@KOM.tu-darmstadt.de*

Abstract

Collaborative Virtual Environments are distributed systems that offer collaboration environments containing multiple types of media sources. Taking into account additional constraints such as bounded latency scalability becomes a challenging problem in such systems. This paper explores the design space of developing scalable, heterogeneous and dynamic Virtual Environments based on Distributed Hash Tables. Such structures have been exploited for large scale Peer-to-Peer systems to provide fast lookup operations in a well distributed way.

In this paper it is argued that large scale Collaborative Virtual Environments can operate more effectively using Peer-to-Peer communication paradigms. Such approaches reduce the high communication cost and avoid potential performance bottlenecks of centralized approaches. The load-balancing issues are addressed by distributing the maintenance responsibility for storing and providing dynamic, persistent information that describes the state of the virtual environment. Heterogeneity in participants' capabilities is also being considered.

1 Introduction

Collaborative Virtual Environments (CVEs) have developed into an interesting research topic during the last decade. International bodies have specified standards to model their content such as VRML [9], MPEG-4 Systems [10] or their architecture such as High Level Architecture [1].

CVEs are very demanding systems since they integrate multiple media types with various and most importantly demanding network requirements. A number of architectures have been proposed in the past to deal with the communication overhead and the management of the collaboration sessions. Two major directions have been followed: (i) the traditional client-server paradigm and (ii) the all-to-all serverless approach. However, both of them have limitations in terms of scalability. Client-server approaches require additional server resource capabilities as the number of clients increases. All-to-all approaches have high bandwidth requirements in order to enable communication among the participants and even harder to solve problems, such as global information consistency. Further, additional issues have to be addressed, e.g. fault-tolerance or heterogeneity. Network-layer multicasting could (if available) improve the efficiency of the all-to-all approach.

However, in different application domains large scale systems are becoming a reality. For instance, *Peer-to-Peer* (P2P) file sharing applications reach populations of millions of users. Advanced *Distributed Hash Table* (DHT) based mechanisms promise even larger size systems. In this paper, a DHT-based approach is adapted and applied to CVE system design. Though the addressed issues are limited to a certain type of information (i.e. the state of mutable objects) it can be extended to efficiently address additional communication needs of these systems.

The rest of this paper is organized as follows. A classification of the different information types and their requirements are given in Section 2. Section 3 gives a short overview of the proposed P2P technology and Section 4 describes the proposed design. The related work is briefly mentioned in Section 5 and Section 6 concludes the paper and identifies areas of future research.

2 Collaborative Virtual Environments

2.1 Basic Concepts

CVEs can result in complex systems that are very costly to maintain and operate in presence of large users population. A useful mechanism to cope with the scalability problem is the concept of *Locales* [2]. Locales are defined as integral partitions of the whole world. This way they provide more manageable portions of content to deal with. Neighbor Locales are linked to enable the transition among them.

Table 1. Content classification

Synthetic Content Type	*Information Type*		*Driving Requirements*	*Communication Pattern*	*Update*
• Environment Description	Static		Storage Space	Locally Accessed	Pull
• Events	Dynamic	Transient	Latency, Ordering	(Multicast) Streaming	Push
• State		Persistent	Maintenance, Consistency	Unicast Fetching	Pull

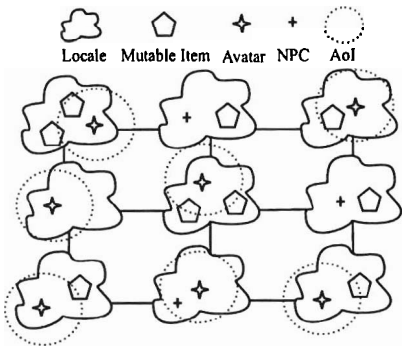

Figure 1. Modeling Concepts in Virtual Environments

Similarly, users' ability to experience the surrounding world is limited to reflect reality. In most approaches, a so-called *Area of Interest* (AoI) is defined as the spatial area of the world that a user can experience. In many cases, AoI is equivalent to the Locale that intersects with their position (additionally could be also the neighbor Locales). Users' presence in the virtual worlds is modeled using *avatars*, a graphic representation.

In many cases Virtual Environments contain intelligent entities that interact with the users actively when they approach them. Such entities are called *Bots* (or *Non Player Characters* (NPCs) in game terminology). NPCs differ from normal objects (e.g. those that compose the landscape of the virtual world) in two main aspects: (i) they have a state that can be modified in an interactive way by the users and (ii) they can stimulate events that are experienced by users in the vicinity. In addition to NPCs, a number of passive objects might have a state that could be modified by user actions. Such objects are called *mutable*.

Figure 1 illustrates the aforementioned concepts. In this figure only neighbor Locales are connected, however, based on the rules that govern the virtual world, non-neighbor Locales can be connected as well (i.e. through teleport devices). Moreover, mutable objects can belong to more than one Locales or can be moved to any of them.

2.2 Content Description and Characteristics

CVEs are usually composed of multiple media sources such as synthetic graphics, animations, audio and video. Standardized documents such as MPEG-4 Systems [10] describes the scene composition specifications. Moreover, MPEG-4 DMIF [11] describes an integration framework to deliver the media sources to the destination end-points. COSMOS [5] is a reference implementation of the aforementioned specifications that integrates the rendering of synthetic graphics, animations, audio and video and employs multicast mechanisms to deliver the content among the collaborating end-points.

The media sources that compose the rendered scenes have heterogeneous characteristics and delivery requirements. This work focuses on the maintenance and delivery of *synthetic graphics media sources*. Synthetic media are described by a number of hierarchically correlated objects (e.g. using the Virtual Reality Modeling Language (VRML) or the BInary Format for Scenes (BIFS) [10]). While VRML and BIFS describe the objects that compose the rendered scenes, the rendering itself is implemented with graphics packages such as OpenGL or Java3D [4].

Table 1 provides a coarse classification of the synthetic graphics content, expanded to include the related characteristics and delivery mechanisms.

- *Environment Description.* The synthetic media content describing CVEs is statically defined. Such content includes immutable landscape description. Landscapes can be defined by vendors that provide the applications or can be user-defined, created by usually available editors. Role Playing Games are common examples of such applications (though the technology used is usually proprietary).
- *Events.* Another important information type (related to the synthetic media content) is the captured interaction of the users with the content itself and their navigation inside the world. Depending on the application, delivery of such events can become very demanding in terms of tolerated latency. However, concepts such as AoI and Locales reduce significantly the generated network overhead, making their implementation more feasible. Multicast-based communications mechanisms are common in delivering such content.

Users vicinal to the event sources subscribe to receive the transient information.

- *State.* An additional dynamic information type is the announcement of the active event sources so that users can subscribe to receive the event streams and the state of (possibly inactive) mutable objects. Mutable objects description is initially included and retrieved similarly to the static content introduced above. However, users can interact with the mutable objects and modify their state. While the interaction itself is described by a series of events available to the subscribers of that source, the result is important to be captured and be available to users beyond the spatiotemporal limits of the Locales and time, even when no active user will be interacting with that object. Such a spatiotemporal evolution from transient to persistent state information is described in Figure 2. As an interesting example, consider a snowed landscape where a couple of avatars is walking on. The two users can immediately observe their footprints left on the virgin snow via the streaming of the generated events. Assume that shortly after those two avatars left the specific Locale a new one enters that area. The marked footprints on the snow must be available for observation.

This work focuses on maintaining and providing this type of information. It should be noted that such type of information triggers the subscription of a listener to transient information generated by the media source, if the source is active.

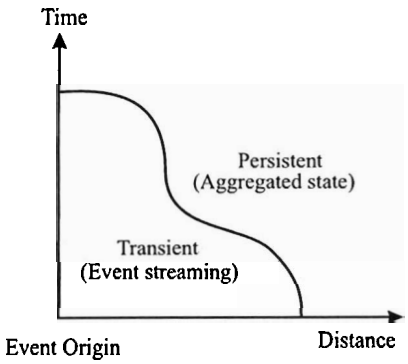

Figure 2. Evolution of Event Handling

2.3 Usage Patterns

A very important factor in designing an optimal architecture for CVEs is to take into account the way information is required at and updated by the participating users. As it has already been mentioned, Locales partition the virtual world in order to reduce the required information that has to be delivered to each end-point. However, as users move from Locale to Locale, they should be informed about the status of all mutable objects that belong to the entering Locale. Frequently changing Locales increases significantly the information that should be delivered, introducing this way a potential bottleneck for centralized approaches. In such scenarios, P2P approaches can perform more efficiently assuming an appropriate overlay network design. On the other hand, mutable objects states are updated at rates that satisfy the needs of the application, whenever users interact with them.

Summarizing, it is common that while update operations are performed on single objects, read operations are grouped to all the objects of a Locale.

3 P2P Networks

P2P networks provide an alternative communication paradigm with respect to the widely used client-server one. A large number of design approaches has been proposed by the research community on how to construct such networks. An interesting direction is based on DHTs.

3.1 Basic Concepts and Characteristics

Distributed Hash Tables (DHTs) are distributed data structures that define efficient communication patterns among the involved nodes. Like standard hash table structures, DHTs relate stored items to keys. Items can be retrieved given their key value. DHTs provide mainly very efficient lookup operations. Well known approaches such as Chord [16] or Pastry [14] require a logarithmic number of steps with respect to the size of the network to find any indexed item in the worst case. However, the aforementioned solutions require that the participating nodes maintain a number of connections that increases also logarithmically with respect to the network size. Moreover, homogeneity in the capabilities of each peer is assumed.

As it has been observed by a number of studies (i.e. [15]) the majority of the users in P2P file sharing applications participate for short time periods only. This causes additional concerns on the cost of maintaining such a network. Adaptive approaches such as Omicron [6] can deal better with this phenomenon. However, there are no studies that capture the behavior of users in very large scale environments. In any case, users represent less reliable components than dedicated servers and this fact should be taken into account in the design phase of the systems.

3.2 Omicron

Omicron *(Organized Maintenance, Indexing, Caching and Routing for Overlay Networks)* is a P2P overlay net-

work aiming to address issues of heterogeneous, large-scale and dynamic P2P environments. Its hybrid, DHT-based approach makes it highly adaptable to a large range of applications. Omicron deals with a number of conflicting requirements, such as scalability, efficiency, robustness, heterogeneity and load balance.

The rational in Omicron's approach is to reduce the high maintenance cost by having a small, constant number of connections and routing table sizes per peer (at least for the majority of them), while still performing lookup operations at low costs. For this reason the usage of appropriate graph structures (such as de Bruijn graphs [7]) is suggested. However, while the small number of connections reduces the operational cost, it causes robustness problems. To address this issue, clusters of peers are formed with certain requirements on their stability over time. In order to maintain their stability, new joins are directed to the least stable clusters. When the stability of a cluster gets below a certain threshold, a merging operation takes place between the unstable cluster and a neighbor cluster.

As it is shown in Figure 3 the introduced de Bruijn graph based overlay is applied at the inter-cluster organization. Inter-cluster routing of messages is based on shift operations performed on the cluster GUID to select the neighbor cluster whose GUID matches best to the requested key. The operation is repeated until the final destination is reached.

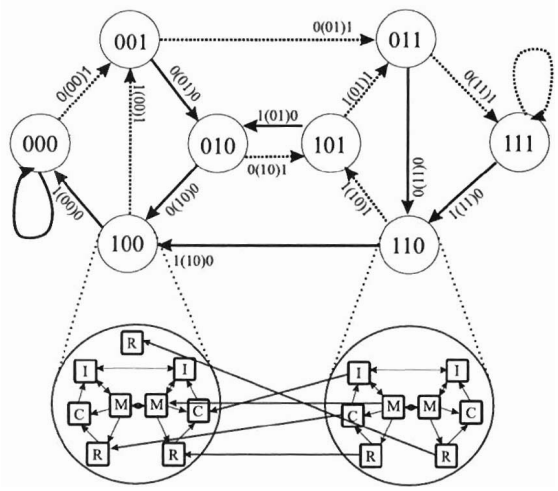

Figure 3. Omicron Overlay Network

Going a step further, a role-based scheme is introduced to deal with the heterogeneity of the peer capabilities and user behavior. This scheme fits the contribution of each node to its resource capabilities and aims at the maximization of the cluster efficiency by providing appropriate incentives to peers to take a certain role. The identified roles are based on the core overlay operations, namely routing (R), maintenance (M), indexing (I) and caching (C). An example illustration is given in Figure 3. The pictured de Bruijn graph consists of 8 nodes, each with two outgoing connections.

A dual identification scheme has been introduced for Omicron with a number of advantages. Clusters are assigned a *Globally Unique IDentifier* (GUID) that is used to route requests over the network. Advertised items are assigned a GUID and are located at the clusters whose GUID matches best. Moreover, peers are assigned their own GUID to trace their actions in the system.

In addition, an incrementally expandable algorithm has been designed to adapt the exponentially growing de Bruijn graphs to the incrementally expandable P2P systems. Detailed description of the Omicron mechanisms and algorithms are provided in [6].

4 Design Overview

This system provides distributed maintenance of mutable shared information. In this section we describe the proposed structure, how it fits with the common usage patterns of state information in CVEs and its properties.

4.1 System Structure

The fundamental entity of the proposed structure is the cluster of peers offering object-level maintenance to replace the requirement for centralized responsible entities[1]. However, only a subset of capable peers are assigned the ***Storer*** role (as it is described in Section 3.2, each peer is assigned a set of roles it can handle best according to its resource capabilities and the observed user behavior). Such a selective assignment reduces significantly the cost introduced by peers with limited resources.

Stored items are objects that have a mutable state, modifiable by multiple users. Mutable objects can either be related to a set of Locales or can be global (though, commonly objects belong to a single Locale). Taking into account the relationship of the mutable objects and the Locales the following use cases on requesting the maintained information have been identified:

- Users join the system and request the state information of the mutable items that do not belong to any Locale.
- Users enter a new Locale and request the state information of the mutable items that belong to that Locale.

The critical design aspect is how to assign object maintenance responsibilities to clusters. Based on the analysis of the common usage patterns, the following object distribution policy is proposed.

[1] Though, centralized entities might be required in the bootstrap phase. Additionally, they can take over in highly unstable periods.

Mutable objects are uniquely identified by a GUID. Similarly, Locales are identified by their own GUID. These identifiers are used to map the related object information to the Omicron DHT. The Locales-related items contain indexes of the location of the mutable items. Since it is a common use case to retrieve all the mutable items' state when entering a new Locale, it is more efficient to group the storage of the items that belong to a single Locale together and access all of them in a single operation.

However, it is important to distinguish the potential situations that could arise based on the relative populations of the involved entities. We assume that the number of mutable items is much larger than the number of Locales. The number of users is certainly larger than the number of clusters (based on the stability requirements of clusters). The unknown entity population ratio is between Locales and clusters. In the following, $|\mathcal{L}|$ represents the number of the locales and $|\mathcal{C}|$ the number of clusters.

- $|\mathcal{L}| < |\mathcal{C}|$. Assuming that the number of items is much larger than the number of Locales, it is suggested to divide further each Locale into *Partitions*. Each Partition is identified by a PartitionID. In order to map an object to a cluster, the concatenation of the related LocaleID and PartitionID is used to identify the responsible cluster for each Partition.
- $|\mathcal{L}| > |\mathcal{C}|$. Here, a straightforward mapping function applies. The cluster with an identifier that matches best with the equal size prefix of the LocaleID is selected.

A special case are items that do not belong to any Locale. To address this issue, a hierarchical composition of the Locales is proposed. The highest level structure represents the entire world to which the global items are assigned. Similarly to the normal Locales, higher-level entities are uniquely identified and mapped on the overlay.

Similarly to the information retrieval use cases, when users interact and change the state of mutable objects, they also update the stored values. Updates may occur at full event rate as responsible clusters could be subscribers to their assigned objects or at a slower rate when it is acceptable by the requirements of the application. In order to reduce the load of the Storers and distribute more evenly the cluster responsibilities, peers assigned the Cacher role can subscribe to the sources of interest and aggregate locally the information. The updated object states are integrated back to the Storers at much slower intervals.

4.2 Properties

A number of desirable properties are provided by this design:

- **Scalability.** Based on Omicron properties, lookup operations for any item requires a number of messages at most logarithmic in the number of clusters.
- **Load balance.** Work is distributed among the clusters and furthermore, responsibilities are assigned inside the clusters to distribute the effort for every common operation (routing, indexing, storing, maintaining, aggregating).
- **Self organization.** Self stabilization procedures are in place to reorganize the membership of the clusters, the inter-cluster organization and the assigned responsibility roles.
- **Support for heterogeneity.** Extending the Omicron approach to include the storage functionality, the proposed system can handle efficiently participants with diverse capabilities.
- **Efficiency.** Provided that the Omicron clustering mechanism assigns a set of peers to each operation, consumers can select cluster members that minimize network overhead (by selecting peers that are physically located close).
- **Persistence.** Replication of storing responsibilities and assurance of clusters stability ensure the persistent maintenance of the stored information.
- **Security.** Data integrity against malicious data modifications is protected by setting quorums of peers with equal roles (Indexers, Storers).

4.3 Open Issues

Although replicating and distributing information enables the system to achieve many of the properties introduced in Section 4.2 this results in a consistency issue. Ideally each copy of an information unit in the system is identical. However, since the state of the information changes through updates (as it is the case with mutable objects), this state change has to be reflected in every copy. Conventionally this is ensured at the protocol level by transmitting the state change to every note hosting a copy in the system [17]. So called integrity conditions ensure that the information has been correctly received by the relevant (sub-)set of notes. Strong integrity implies that all copies have to be successfully updated. However, since this is not always possible (especially in distributed autonomous systems), consistency can be maintained by ensuring that a majority of nodes (quorum) present in the system, receives updates correctly [13]. A newly joining node (or a node coming back into the system) requests the information status and based on the replies from the other nodes it can decide what the most actual information is. Using the same method each peer can decide about the status. However, this would cause undue communication overhead.

Currently it is being researched how to set integrity conditions, how to propagate the information and how to deal with unstable system conditions.

5 Related work

Lately a significant research effort has been focused in designing massive multiplayer games based on P2P technology. The game industry seems to be an important driving factor for the future research on this field.

In particular, Knutsson et al developed SimMud, an experimental prototype on top of FreePastry [12]. FreeMMG is an open source initiative to support massively multiplayer games. [8]. Mercury is a scalable publish/subscribe system for Internet games [3].

On the other hand, more traditional approaches require multicast infrastructure. Beacons [2] are multicast-based solutions that enable the location of objects by advertising the information of the Locales that encapsulate them. Also MPEG-4 DMIF suggest the creation of general-purpose multicast channels to advertise other, specific to a certain content multicast channels.

6 Conclusions and Future Work

A new design to deal with persistent state information in CVEs has been presented. The P2P paradigm, as it has been lately explored in large scale file sharing applications, can be adapted effectively to many of the network aspects of CVEs.

The proposed approach decouples users' location from maintenance responsibilities. This way even underpopulated Locales are sufficiently maintained. A number of important requirements are met, such as scalability, load balance, support for heterogeneity, persistence, etc. Consistency is an open issue to be explored further.

CVEs require a number of different media sources. Dealing with all of them poses the need for an adaptive set of delivery (and maintenance) mechanisms. Media and user interaction event streaming are still challenging areas for further research. The expandable role-based architecture of Omicron can fit well to future extensions to deal with the aforementioned media.

References

[1] ANSI/IEEE. 1516-2000 High Level Architecture (HLA), 2000.

[2] J. W. Barrus, R. C. Waters, and D. B. Anderson. Locales and Beacons: Efficient and Precise Support For Large Multi-User Virtual Environments. *IEEE Computer Graphics and Applications*, 16(6):50–57, November 1996.

[3] A. R. Bharambe, S. Rao, and S. Seshan. Mercury: A Scalable Publish-Subscribe System for Internet Games. In *ACM Netgames2002*, April 2002.

[4] V. Darlagiannis, R. Ackermann, A. E. Saddik, N. Georganas, and R. Steinmetz. Suitability of Java for Virtual Collaboration. In *Proc. of Net.ObjectDays'2000*, pages 71–78, October 2000.

[5] V. Darlagiannis and N. D. Georganas. Virtual Collaboration and Media Sharing using COSMOS. In *Proc. 4th WORLD Multiconference on Circuits, Systems, Communications & Computers (CSCC 2000)*, July 2000.

[6] V. Darlagiannis, A. Mauthe, and R. Steinmetz. Overlay Design Mechanisms for Heterogeneous, Large Scale, Dynamic P2P Systems. *Journal of Networks and System Management*, 12(3):371–395, 1 2004.

[7] N. de Bruijn. A combinatorial problem. In *Proceedings of the Koninklije Nederlandse Academie van Wetenshapen*, pages 758–764, 1946.

[8] C. G. Fabio Cecin, Jorge Barbosa. FreeMMG: An Hybrid Peer-to-Peer, Client-Server, and Distributed Massively Multiplayer Game Simulation Model. In *2nd Brazilian Workshop on Games and Digital Entertainment (WJogos'03)*, November 2003.

[9] ISO/IEC. 14772: Information technology – Computer graphics and image processing – The Virtual Reality Modeling Language (VRML), 1997.

[10] ISO/IEC. 14496-1: Information Technology Coding of audio-visual objects, Part 1: Systems, 1999.

[11] ISO/IEC. 14496-6: Information Technology Coding of audio-visual objects, Part 6: Delivery Multimedia Integration Framework, 1999.

[12] B. Knutsson, H. Lu, W. Xu, and B. Hopkins. Peer-to-Peer Support for Massively Multiplayer Games. In *The 23nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2004)*, March 2004.

[13] A. Mauthe. End-to-End Support for Multimedia Multipeer Communications. PhD thesis, Computing Department, Lancaster University, November 1997.

[14] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, pages 329–350, 2001.

[15] S. Saroiu, P. K. Gummadi, and S. D. Gribble. A measurement study of peer-to-peer file sharing systems. In *Proceedings of Multimedia Computing and Networking 2002 (MMCN '02)*, 2002.

[16] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 149–160. ACM Press, 2001.

[17] P. Triantafillou and C. Neilson. Overlay Design Mechanisms for Heterogeneous, Large Scale, Dynamic P2P Systems. *IEEE Transaction in Software Engineering*, 23(1):35–55, January 1997.