The HyperVerse - Concepts for a Federated and Torrent-Based "3D Web"

Jean Botev, Alexander Höhfeld,
Hermann Schloss, Ingo Scholtes
Systemsoftware and Distributed Systems
Computer Science Department
University of Trier
Email: {botev, hoehfeld, schloss, scholtes}@syssoft.uni-trier.de

Markus Esch
Faculté des Sciences, de la Technologie
et de la Communication
University of Luxemburg
Email: markus.esch@uni.lu

***Abstract*— The vision of a "3D Web" as a combination of massive online virtual environments and today's WWW currently attracts a lot of attention. While it provides a multitude of opportunities, the realization of this vision on a global scale poses severe technical challenges. This work-in-progress paper intends to point out some of the major challenges and highlights key concepts of an infrastructure that is being developed in order to meet them. Among these concepts, special emphasis is put on the usage of a two-tier Peer-to-Peer approach, the implementation of Torrent-based data distribution and the development of a graded consistency notion. The paper also briefly presents the current state of a prototype implementation that is being developed in order to validate these concepts and evaluate alternative approaches.**

I. Introduction

Although various forms of distributed virtual environments currently attract a lot of attention, most of today's representatives are proprietary worlds that are hosted in a centralized fashion. When considering the vision of something one might call the "3D Web", in section II we find that - in contrast to today's precursors - decentralized approaches are required in order to provide a distributed persistent virtual environment on a global scale. With the HyperVerse project we intend to investigate technologies that are appropriate to realize such a scenario.

A global scale "3D Web" offers a variety of obvious interesting opportunities, like immersive mass events and the facilitation of real-time interaction between users based on advances in human interface technologies. Another interesting scenario arises from the observation that mobile devices are becoming increasingly location-aware and network-capable while at the same time getting smaller and cheaper. Thus it seems reasonable that in the future more and more everyday objects will feature such capabilities as well as all kinds of sensor technology. Combining these developments with a "3D Web" would allow more and more real-world objects to possess a real-time virtual representation, giving users intuitive means to remotely access various kinds of information on their state. In this respect, the current surge of geo-referenced information accessible via virtual globes like "Google Earth"[1] or "Virtual Earth"[2] is a first glimpse of what is yet to come. In the course of this paper, a persistent virtual environment which provides these opportunities at a global scale will henceforth be called "HyperVerse".

While its opportunities sound alluring, it is obvious that the realization of a HyperVerse scenario poses severe technical challenges. In the context of this paper, we restrict ourselves to briefly mentioning some of the - according to our opinion - most important questions:

- How can scalability suitable for the provision of a global scale HyperVerse scenario be achieved?
- How can such a scalability be combined with interactivity, consistency and persistency?
- How can client resources be utilized in a way that unburdens core network resources?
- Which cross-layer aspects can be identified that are induced by HyperVerse-specific communication patterns?
- How can the client-side fan-in problem in densely populated regions be resolved?

The HyperVerse project aims at the creation of a federated infrastructure meeting these requirements. In section II we will present some of the key concepts the project relies on. Section III will give a brief description of the current state of a prototype implementation that is being developed in order to evaluate these concepts.

II. Concepts of a Federated HyperVerse Infrastructure

In order to retain the decentralized nature, scalability, independence and reliability of the WWW and the Internet in general, for the provision of a HyperVerse scenario it is not eligible to rely on centralized server farms that are controlled by a single instance. Thus we embrace Peer-to-Peer (P2P) technologies in order to support the targeted global scale. For a multitude of reasons we do not aim at a pure P2P approach but rather use a two-tier architecture consisting of a highly structured federated backbone and a loosely structured P2P client overlay. The main reason for this is the expected churn rates in the HyperVerse scenario. Since we envision

[1] http://earth.google.com

[2] http://www.microsoft.com/virtualearth

a lightweight client software that is used in a way that is similar to today's Web browsers, clients will most likely exhibit exceedingly high churn rates as well as heterogeneous capabilities. Accordingly the P2P topology used for clients needs to be highly churn resilient in the face of global-scale user numbers. At the same time HyperVerse scenarios require massive amounts of persistent data to be reliably and efficiently accessible. Unlike in today's Massive Multiplayer Online Games (MMOGs), this data is required to be totally dynamic and cannot be predistributed with clients. We tackle this problem by using a massive amount of public servers that are supposed to be comparably reliable and host data in a federated manner. Public servers resembling today's Web Servers, we do not require them to be under control of any centralized authority. For their provision we rely on the incentive of being able to publish information.

In accordance with [11], by explicitly distinguishing between these two classes of peer participants we exploit their different properties in order to provide better reliability, availability and scalability of the whole system. The comparative low churn rate of public servers can be utilized by using highly structured P2P overlay networks. This provides for an efficient and reliable data retrieval. For clients, less structured topologies seem to be appropriate (see figure 1). The BitTorrent protocol [5] has proven to be valuable for the scalable distribution of huge files and is extremely resilient against churn [2]. In particular this resilience does not depend on the number of peers. The following section will provide more details on the application of similar approaches to distributed virtual environments.

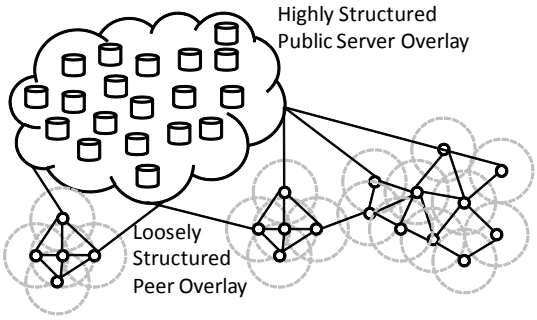

Fig. 1. Two Tier HyperVerse Infrastructure

A. Torrent Based Data Distribution

In order to be able to describe the Torrent-based distribution of information, we give a short description of the dynamic space-based interest management model implemented in the current HyperVerse prototype in so far as it affects concepts described in the following paragraphs. For the same reason, we also give a short description of the caching scheme that has been implemented in order to exploit data locality inherent to virtual worlds. Throughout the paper we refer to the terms object and terrain data as mesh, texture and meta information that are associated with dynamic 3D objects as well as comparably static world terrain.

a) Interest Management: Given an avatar's 3D position p in the virtual world, we differentiate between its Field of View (FoV) and Area of Interest (AoI). Assuming a maximum view distance d, an avatar's FoV is described by a sphere with radius d around p. The AoI is another sphere with radius $d + \Delta$ $(\Delta \geq 0)$ around p. Initially all objects and terrain data within the AoI sphere around p are retrieved. In order to mitigate the effects of retrieval latency, we use another sphere with radius $d + \Lambda$ $(\Lambda < \Delta)$ around p. The user can move within a distance Λ around p (see e.g. position p' in figure 2) without requiring further object retrieval. Whenever the avatar has moved more than Λ away from the position p, around which the AoI has been retrieved (see e.g. position p'' in figure 2) - a new AoI centered around the current position will be set. At this point, information on all objects and terrain within the new AoI need to be retrieved, more precisely only on those that haven't been in the AoI before. The introduction of the threshold Λ allows for more time for requesting these data since the users FoV is still $\Delta - \Lambda$ away from regions for which no information has been prefetched. Accordingly, the choice of the parameters Λ and Δ influences retrieval frequency, the amount of data present within the AoI as well as the retrieval latency that can be tolerated without having visual effects. Both parameters can be chosen by clients according to their individual capabilities.

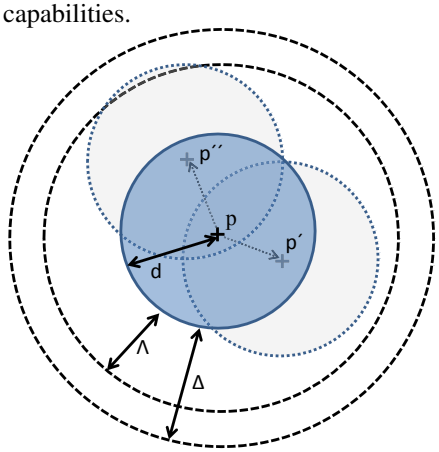

Fig. 2. A 2D projection of Area of Interest (AoI) and Field of View (FoV)

b) Data Locality in Virtual Environments: Looking at the pattern of access to object and terrain data induced by a supposed primary continuous movement through virtual worlds, one recognizes both temporal and spatial locality of reference with respect to the world's geography. Due to the avatar's movement, there is a higher probability for objects near the avatar's FoV to be accessed in the future - a fact that is allowed for by prefetching data from within the AoI as described above. One aspect of temporal locality in virtual worlds refers to the fact that recently accessed objects remain in the users FoV for some time and therefore will be accessed repeatedly. Another aspect is caused by frequent visits to the user's favorite venues. A certain user exhibits e.g. a higher probability of frequently accessing information residing in distinct areas within the virtual world. Even in today's WWW

one recognizes that most users repeatedly check an individual working set of favorite information resources, be it subscribed feeds, news portals, Blogs or community Websites. In today's WWW, in order to save redundant transmission, these locality aspects are allowed for by caches at various stages like user agent caches or caching proxy agents. In distributed virtual environments, data locality can be exploited by applying caching techniques as well. The rendering process involved in clients implicitly requires objects and terrain data to be cached locally at least as long as these data concern objects in the user's FoV. While such a cache can be based on a simple LRU strategy, the further exploitation of locality and optimization of the cache hit rate requires more advanced concepts. The current prototype HyperVerse browser (being described in section III) uses a multi-tier caching strategy in which object information within the FoV are held in graphics memory. This is backed up by a fixed-size in-memory cache that uses a combined LRU, geographical distance and object size metric as a replacement heuristic. It is especially important to consider the in-memory size of cached objects since it might involve huge variations and discarding big objects bears a greater potential for wasting network resources: Wrongly ejecting e.g. a few Kilobyte-sized object is doubtlessly less troublesome than spuriously discarding one being several Megabytes in size. The in-memory cache is backed up by a permanent storage cache file with configurable size. The maximum size of this cache can most likely be amply chosen in order to optimize performance and unburden network resources.

Caching is complicated by the fact that objects must be allowed to dynamically change at any time. This may occur following a user interaction with an object or abruptly in a scenario of augmented virtual real-world object representations that actively push state updates triggered e.g. by sensors. Due to the usage of caching, at a given time several copies of data may reside in the caches of different HyperVerse browsers. In order to keep these cached data coherent, a Publish/Subscribe paradigm maintaining subscriptions to objects residing in a client's cache is used. Dynamic and asynchronous changes of objects are actively pushed to subscribers in order to maintain cache coherency. Depending on the cache-tier, different update strategies are used. Updates of objects that reside in graphics or system memory (i.e. especially those in the FoV) are pushed to subscribers instantly. For objects residing in the disk storage cache file, the first update of an object sets a dirty-bit and the subscription is canceled. Dirty-marked objects are actively refreshed by clients as soon as they enter the AoI, preventing needless communication on objects that cannot instantaneously enter the FoV. Rather than requiring information sources to send unicast messages to all subscribers or relying on multicast facilities of lower protocol layers, the Torrent-like scheme that will be described in the next section can be used for a scalable dissemination of update messages.

c) Application of Torrent Principles: The number of users residing in a certain region of the virtual world and thus requesting the same object and terrain data is hard to predict and possibly massive. Thus it is clear that their distribution as well as the maintenance of subscriptions poses scalability problems. We particularly address a future scenario in which clients at the network's edges have high uplink bandwidths. Due to the multi-tier caching scheme in combination with the Publish/Subscribe pattern used, objects in memory as well as unmarked objects in the disk cache are known to be up-to-date. Furthermore, within highly populated regions this information is available redundantly. In order to utilize these resources we apply a distribution scheme similar to the BitTorrent protocol. The basic idea is that each HyperVerse client makes accessible cached information in a Torrent-like manner, i.e. data are split into individually addressable pieces. When a user requests data of a certain region, a number of other clients can be used to concurrently transfer pieces of data that are already present in their caches. For this the distributed backbone consisting of public servers (being described in section II-B) keeps track of the clients' AoI. According to the interest management model described above and by choosing an appropriate value for Λ and using an adequate federation scheme for the backbone service, the update frequency in individual servers can be kept manageable.

Clients use the backbone service in order to retrieve a (size-constrained) subset of nearby peers along with their AoIs. Knowing that peers at least contain an up-to-date version of objects within their AoI, a client uses this information in order to compute the coverage of its own AoI by peer AoIs. In a first step peers are used to retrieve meta-information on all objects that reside in areas that are covered by peers' AoIs. From this information the redundancy degree of objects within different portions of the covered area is computed based on the peers' AoI. Pieces of objects are then retrieved concurrently from all peers whose AoI contains the object's position. AoIs in low-density regions may contain portions uncovered by peers. Objects from within such portions will be retrieved from the backbone service which at the same time serves as initial seed for the Torrent-based distribution of data. An optimum peer selection strategy is still being investigated. In the current version random peers are used although ideally the peer subset should be chosen in a way that optimizes AoI coverage as well as object redundancy. Without going into further detail, a Publish/Subscribe extension to this Torrent-based scheme can be realized by clients propagating object piece updates to peers that have recently requested a given object. Since objects consist of many pieces, the probability of not receiving any piece update is comparably low and can only occur if all peers of which the object pieces have been received are offline. If at least some piece updates have been received, any remaining pieces can be actively requested by updating the peer set after a certain threshold.

B. Federated Backbone Service

As motivated earlier in this section, a two-tier P2P approach has been chosen in order to allow for lightweight clients and handle varying churn rates. Public HyperVerse servers can be thought of as a kind of federated "3D Web Servers". Objects can be published by adding a new public server to the

backbone federation or by uploading them to existing servers - their providers possibly demanding a fee for this service. By this means, public server providers are incentivized in a way that is similar to today's WWW and abides existing business models. The responsibilities of public servers consist in tracking client AoIs and thus connecting nearby peers for a Torrent-based distribution and a P2P exchange of movement information. Public servers also serve as initial seeds for all objects they contain. Initial seeds of world-specific terrain data are redundantly and equally distributed among all servers.

It is clear that the federation scheme underlying the public server backbone needs to be organized in a way that balances load between them. Apart from this it must provide simple facilities to perform range queries in order to efficiently retrieve the peer sets of client AoIs. Although to date it is not yet clear which of them will be actually chosen, several promising candidate technologies have been identified. Among them the *P-Grid* [1] system represents an efficient structured overlay network based on the concept of a distributed trie. It achieves highly efficient lookup operations and guarantees load-balancing of storage and query load. By preserving key order it furthermore supports range queries. Another interesting fact is that approaches relying on a space-based federation of public servers can capitalize on research stemming from the field of mobile ad hoc networks. One federation scheme we are currently testing for applicability is e.g. loosely based on the distributed and scalable GRID location service [13]. Moreover we investigate in how far techniques and methods of swarm intelligence are applicable in order to realize a decentralized and self-organized network of public servers. This includes the evaluation whether decentralized control algorithms for public servers can be combined with efficient routing protocols in order to support the requirements mentioned above.

C. Individual Dynamic Instantiation

The Torrent-based distribution of 3D data described above is suitable to mitigate the problem of distributing the same object or world data to a massive number of clients resulting from high avatar densities in confined virtual regions. Under such circumstances, the problem of mutual visibility and thus exchange of motion information remains. Apart from exhibiting a fan-in problem at clients, it also hampers usability since from a user's perspective it is not reasonable to visualize an unlimited number of nearby avatars. This problem already occurs in today's MMOGs, so the countermeasure of so-called instantiated sessions has been developed in this domain. Using this technique, several separated and dynamically instantiated "copies" of the same region are created, each copy allowing a certain maximum number of users. This not only improves usability but also scalability since instances can be handled separately on machines in the provider's server farm. Within the HyperVerse, this is not a desirable solution since it precludes interaction between users residing in different instantiated sessions. Accordingly, we investigate how a dynamic and individual instantiation based on an avatar's *social bias* can be provided. In this context, the social bias is auto-generated information on an avatar's position in the social network based on its history of interaction with other avatars. Using this information, rather than separating instances, it is necessary to provide a dynamic *individual instance* for each user in a scalable way whenever avatar density in a certain region exceeds a certain threshold. In each individual instance, only relevant avatars shall be visualized, thus separating out and/or coarsely summarizing dispensable information. Interaction across these instances is possible since each avatar is present in individual instances of all user's that share a similar social bias. Non-overlapping transitive relations are not approached at this stage though.

Extending 3D space by additional dimensions which allow an avatar's social bias to be encoded as position in a multidimensional space seems alluring. Using an appropriate encoding, a simple Euclidean distance between these positions could be interpreted as "social connectedness". The interest management scheme depicted above could then be extended to use higher dimensional FoV and AoI sphere abstractions. Unfortunately it is easy to see that social relationships between an unlimited number of users cannot be represented by positions in any finite-dimensional metric space, since it would require encoding a potentially unlimited amount of information in a fixed-size position vector. The usage of non-metric topological spaces and an appropriate definition of closeness appears to hold the key for a scalable implementation but requires further research.

D. Multilevel Consistency

The consistency of distributed data and state is an important issue which needs to be addressed at the design stage of any distributed environment. We classify consistency problems into *data consistency problems* arising from data replication and *update propagation problems* resulting from the distribution of global state representations among adjacent users.

Consistency in centralized environments is comparably easy to achieve because only a single copy of the data or state representation exists on a central instance (server). As motivated in section II, there are several arguments which suggest that centralized approaches are not suitable for a global scale virtual environment. Accordingly, we expect such environments to be based on decentralized federated infrastructures. For the sake of scalability it is hardly possible to provide strict consistency for the whole virtual world targeting global scale user numbers. Therefore we introduce the term *world partition* representing a designated virtual region along with all users within. Hereby the virtual world can be arbitrarily subdivided in several world partitions which are not necessarily disjoint. We define the *weight* of a world partition as a function $w : \mathbb{R} \times \mathbb{N} \rightarrow \mathbb{R}$ of the region's area and number of users within. In order to guarantee consistency within these world partitions, we use a multilevel consistency model. Besides the given application scenario, the weight of a world partition is decisive for the guaranteed consistency level. The maximum consistency degree that can be provided is reciprocally proportional to the weight of the a world partition. That is, in

a "lightweight" world partition we ensure higher consistency levels than in a "heavier" one. We are confident that this relaxation of the consistency notion is expedient for massive virtual environments since e.g. small consistency variations of avatars in a user's FoV become less perceptible and thus crucial the more avatars are visible.

III. A Prototype HyperVerse Implementation

In this section we will briefly describe the current state of a HyperVerse prototype that has been implemented based on the aforementioned concepts. It consists of a lightweight 3D browser which is based on a network-aware engine capable of retrieving, caching and rendering 3D terrain and object data from a Web Service based backbone service. The HyperVerse browser as well as the underlying engine serve as a prototype reference implementation which we use to evaluate different approaches. In order to perform "simulations" using the actual implementation while at the same time saving resources, the actual 3D visualization can be detached from the browser. In this mode, the client is remotely controllable via a Web Service. This can be used in order to create a number of client instances on a test cluster, manage them and retrieve measurands via a centralized controller based e.g. on real-world mobility models that are available from the MANET research community [12].

a) HyperVerse Browser: The HyperVerse browser is a thin-client interactive 3D application combining concepts known from today's Web browsers (like e.g. Bookmarks) and virtual globes (like e.g. the use of geo-referenced objects, geography-based navigation, satellite imagery and topographic height data) with an avatar-based interaction and navigation known from MMOGs. Based on widespread and cheap game console hardware - namely the Bluetooth-connected Nintendo Wii controller, head-tracking facilities have been implemented. By this means, an immersive 3D experience and fine-grained interaction between avatars can be provided. Figure 3 shows the user interface of the HyperVerse browser. Although we currently use a virtual world that is based on real world topographic and imagery data, the browser does not depend on this. The backbone service providing appropriate data, any other virtual world that uses either a spherical or cartesian coordinate system can be used. Communication between the browser and the backbone service as well as other clients is based on Web Services and defined by descriptive interfaces, thus making the used protocol traceable and furthering interoperability. The HyperVerse browser is available at[3].

b) HyperVerse Engine: The network-aware HyperVerse engine is based on Microsoft DirectX and the .NET framework and is capable of rendering XML-based Collada[4] models. Being a powerful and wide-spread intermediate format, Collada is most prominently used as a 3D inlet of the Google Earth KMZ4 format with a large pool of available models. Furthermore the engine supports the rendering of SRTM[5] terrain, Blue Marble[6] as well as Landsat[7] topographic imagery data. It contains basic dead reckoning technologies for real-time rendering of object and avatar movement in the face of network delay, basic geodetic mathematics utilities as well as an implementation of the space-based interest management model and the caching scheme described in section II-A.

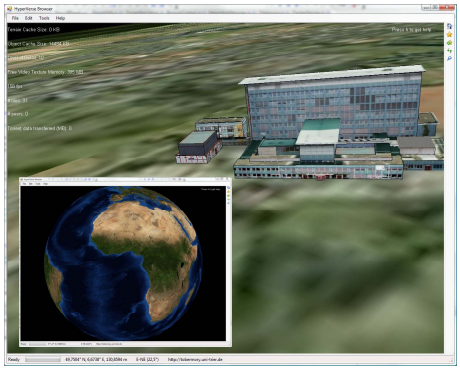

Fig. 3. HyperVerse Browser Application

c) HyperVerse Backbone Service: Since candidate technologies for the federated backbone service are currently under investigation, for the time being only a simple Web Service based Tracker and initial seed service has been implemented. It serves real-world topographic and imagery data in different levels of detail as well as object data. According to the Torrent-scheme described in section II it also keeps track of client AoIs and is used by the HyperVerse browser in order to retrieve peers whose avatar's AoIs subtend that of the local avatar.

IV. Related Work

Similar to our notion, [14] claims that peer-to-peer architectures are suitable for supporting distributed virtual environments (DVE). In order to investigate this type of architectures and to simulate large-scale DVEs in an efficient way, the authors propose a distributed simulation platform that provides appropriate performance metrics and contains all the elements involved in real DVE simulations. Techniques resembling our Torrent-based data distribution have been introduced to 3D virtual environments by [8]. The authors propose the P2P *3D Streaming* framework *FLoD*. It allows clients within a virtual environment to retrieve relevant data from nearby clients while minimizing perceived transmission delay. An important contribution of FloD is its support for progressive meshes and textures by defining so-called base and refinement pieces. By prioritizing base pieces, the rendering process can start before all object pieces have been received. The overlay topology of FloD is based on the Voronoi-based *VON* [7] scheme. Evaluations of FloD have shown that P2P 3D streaming is much more scalable than client-server approaches. Solipsis [9] is another massively shared P2P virtual reality system that addresses global-scale user numbers. In order to tackle the scalability problem and heterogeneous access device

[3]http://hyperverse.informatik.uni-trier.de
[4]http://collada.org
[5]http://www2.jpl.nasa.gov/srtm
[6]http://earthobservatory.nasa.gov/Newsroom/BlueMarble/
[7]http://landsat.gsfc.nasa.gov/

capabilities, it provides adjustable data flows based on varying awareness radii.

Similar to our notion of providing consistency within virtual environments, Myriad [15] permits transient inconsistency, thus relaxing resource requirements in collaborative virtual environments. [10] considers consistency aspects in distributed virtual environments and introduces an approach based on global timestamps. In [6] multilevel consistency addressing the replication techniques of world data is introduced. *CyberWalk* [4] is a DVE using an on-demand transmission approach for the distribution of the virtual environment to the clients. Similar to FloD, CyberWalk uses a multiresolution caching mechanism that reduces model transmission and rendering times by employing progressive models. Network delay is mitigated by providing a caching and prefetching mechanism. Moreover, it allows a client to continue to operate, at least partially, when a network connection is unavailable.

The paper [3] examines an architecture that supports persistent game state in public server based multiplayer games. In opposite to our notion, public servers do not provide a single virtual environment but are separated in the sense that they provide local per-server virtual worlds for a very limited number of users. All servers share certain persistent game items and character capabilities which are contributed and controlled by the game publisher in a centralized fashion.

V. Conclusion and Future Work

In the course of this paper we presented key concepts of the HyperVerse project in which we investigate how global-scale persistent virtual environments can be provided. In order to guarantee persistent information without putting scalability at stake, we have chosen to apply a two-tier hybrid P2P approach by combining a WWW-like federated public server backbone with a scalable Torrent-based data distribution. While our approach resembles the one described in [8], the main difference is the abdication of highly structured client overlay topologies. Since we explicitly address a scenario with Browser-like clients, we assume churn rates to be much higher than that of any prevalent Peer-to-Peer applications. In order to mitigate this problem, we use a highly structured federated backbone acting as Torrent Tracker and interconnecting clients to a loosely structured and highly churn resilient overlay. Another noticeable difference is that due to the propagation of peers' AoIs information, clients are able to locally decide which peers contain required data pieces without having to actively send requests to nearby clients.

The main contribution of introducing a Torrent-based approach to massive virtual environments is the mitigation of Flash Crowds - a spontaneous surge of interest in a certain region of the HyperVerse. The BitTorrent protocol has proven to successfully address this problem in the context of distributing large files in today's WWW. By means of considering virtual geography and locality aspects that are inherent to virtual worlds, we argued that massive virtual environments can benefit from a Torrent-based data distribution much in the same way. In order to overcome the Flash Crowd related client-side fan-in problem, we intend to use a dynamic instantiation approach that is based on the relationship between users and does not preclude interaction across instances.

While we cannot yet provide a definite answer to the question which federation scheme shall be used for the massively distributed public server backbone service presented in section II-B, we identified some candidate technologies as well as general directions of research. As a next step, some of these technologies need to be evaluated. Being a promising approach, a P-Grid-based backbone service is being implemented and will be available soon for further evaluations. In the course of this paper we have also given a brief overview of our approach to consistency handling which is characterized by a twofold relaxation of the consistency notion. We argue that this relaxation is suitable for massive virtual environments in the sense that it allows a minimization of sensible effects while being conducive to scalability.

References

[1] K. Aberer. P-Grid: A self-organizing access structure for P2P information systems. *Sixth International Conference on Cooperative Information Systems (CoopIS 2001)*, 2172:179–194, 2001.

[2] A. Al-Hamra, A. Legout, and C. Barakat. Understanding the properties of the bittorrent overlay. Technical report, INRIA Sophia Antipolis, France, 2007.

[3] C. Chambers, W. chang Feng, and W. chi Feng. Towards public server mmos. In *Proceedings of 5th ACM SIGCOMM workshop on Network and system support for games (NetGames '06)*, page 3, New York, USA, 2006.

[4] J. H. P. Chim, R. W. H. Lau, H. V. Leong, and A. Si. Cyberwalk: a web-based distributed virtual walkthrough environment. *IEEE Transactions on Multimedia*, 5(4):503–515, 2003.

[5] B. Cohen. Incentives build robustness in bittorrent, 2003. citeseer.ist.psu.edu/cohen03incentives.html.

[6] J. C. de Oliveira. Issues in large scale collaborative virtual environments. http://citeseer.ist.psu.edu/oliveira01issues.html.

[7] S.-Y. Hu, J.-F. Chen, and T.-H. Chen. Von: a scalable peer-to-peer network for virtual environments. *Network, IEEE*, 20(4):22–31, July-Aug. 2006.

[8] S.-Y. Hu, T.-H. Huang, S.-C. Chang, W.-L. Sung, J.-R. Jiang, and B.-Y. Chen. Flod: A framework for peer-to-peer 3d streaming. In *The 27th Conference on Computer Communications (IEEE INFOCOM '08)*, 2008.

[9] J. Keller and G. Simon. Solipsis: A massively multi-participant virtual world. In *PDPTA*, pages 262–268, 2003.

[10] S.-J. Kim, F. Kuester, and K. H. Kim. Towards enhanced data consistency in distributed virtual environments. volume 5006, pages 436–444. SPIE, 2003.

[11] J. Kubiatowicz. Extracting guarantees from chaos. *Commun. ACM*, 46(2):33–38, 2003.

[12] J.-Y. Le Boudec, S. PalChaudhuri, and M. Vojnovic. Perfect simulations for random trip mobility models. *Proceedings of the 38th Annual Simulation Symposium 2005*.

[13] J. Li, J. Jannotti, D. S. J. D. Couto, D. R. Karger, and R. Morris. A scalable location service for geographic ad hoc routing. In *Proceedings of the 6th annual international conference on Mobile computing and networking (MobiCom '00)*, pages 120–130, New York, USA, 2000.

[14] S. Rueda, P. Morillo, and J. M. Orduna. A Peer-To-Peer Platform for Simulating Distributed Virtual Environments. In *Proceedings of the 13th International Conference on Parallel and Distributed Systems (ICPADS '07)*, 2007.

[15] B. Schaeffer, P. Brinkmann, G. Francis, C. Goudeseune, J. Crowell, and H. Kaczmarski. Myriad: scalable vr via peer-to-peer connectivity, pc clustering, and transient inconsistency. In *Proceedings of the ACM symposium on Virtual reality software and technology (VRST '05)*, pages 68–77, New York, NY, USA, 2005. ACM.