The London Travel Demonstrator

Anthony Steed
Department of Computer Science
University College London
Gower St, London, UK, WC1E 6BT
+44 020 7388 7665
A.Steed@cs.ucl.ac.uk

Emmanuel Frécon, Anneli Avatare
Swedish Institute of Computer Science
Box 1263
SE-164 29 KISTA
+46 8 633 15 34
emmanuel | anneli@sics.se

Duncan Pemberton, Gareth Smith
Computing Department
Lancaster University
Lancaster, UK, LA1 4YR
+44 1524 593801
pemberto | gbs@comp.lancs.ac.uk

ABSTRACT

Travel can be a stressful experience and it is an activity that is difficult to prepare for in advance. Although maps, routes and landmarks can be memorised, travellers do not get much sense of the spatial layout of the destination and can easily get confused when they arrive. There is little doubt that virtual environments techniques can assist in such situations, by, for example, providing walkthroughs of virtual cityscapes to effect route learning.

The London Travel Demonstrator supports travellers by providing an environment where they can explore London, utilise group collaboration facilities, rehearse particular journeys and access tourist information data. These services are built on the Distributed Interactive Virtual Environment (DIVE) software from SICS. In this paper we describe how the application was built, how it exploits the underlying collaboration services, and how the platform provides for scaleability both in terms of the large extent and detail of this application and in the number of participants it can support.

Keywords

Collaborative virtual environments, travel applications, large-model support, real-time rendering.

1. INTRODUCTION

With collaborative virtual environment (CVE) toolkits becoming more common over the past few years, much research activity has focused on providing scaleable systems. By scaleable we mean both that the system supports large numbers of users and that models that are large in size and extent can be accessed.

We have taken the approach of extending an existing CVE system, DIVE [3], developed at the Swedish Institute of Computer Science, to support scaleable applications through a number of platform level extensions. Thus we have attempted to provide scaleability without compromising the basic collaboration services that exist within the platform.

1.1 Demonstrator Scenario

In order to demonstrate the scaleability of the resulting platform we describe an application based around a scenario of virtual travel to London. The broad aim of the demonstrator is to present an application to a group of users enabling them to specify and rehearse a meeting of any sort. This includes supporting the selection of features at a given location (Hotels, Conference Venues, etc.), using both abstract and facsimile based information visualisation approaches. The demonstrator provides users with the ability to navigate through a large virtual cityscape (representing a real location). Their navigation is aided by a number of dynamic information visualisation systems. A suite of collaborative features aids the users in constructing, rehearsing and participating in both virtual and real meetings.

The demonstrator consists of four main geometric and functional layers:

- A 16x10km geometric model of the centre of London
- Collaboration services for use by groups
- Tourist information data visualisation service
- Simulations of public transport and crowds

The demonstrator integrates all these services into a single coherent environment.

1.2 Background

Large numbers of users are usually supported by partitioning the world into regions that are partially or totally occluded so that events passing between those regions might be discarded or filtered [9][1][10]. These regions may be tile based or may be described on actual visibility relationships between areas of the environment. In Section 4.3 we will describe how such techniques have been employed within the DIVE platform in order to support large numbers of users in complex interactive environments.

There are two aspects to supporting a geometrically complex and extensive model: maintaining a continuous view of the model for the user and managing a model that might be too large to maintain in memory [8]. Many techniques exist for speeding up rendering of large models by partitioning models into separate occluded areas or by pre-processing models to extract important objects for a particular viewpoint (e.g. [7]). However such techniques are often only applicable to well-structured models. In the worlds we are supporting, the models often use a wide variety of scene graph structures and do not lend themselves to pre-processing. However we have taken some of the core ideas behind frame-limited rendering and visibility partitioning and

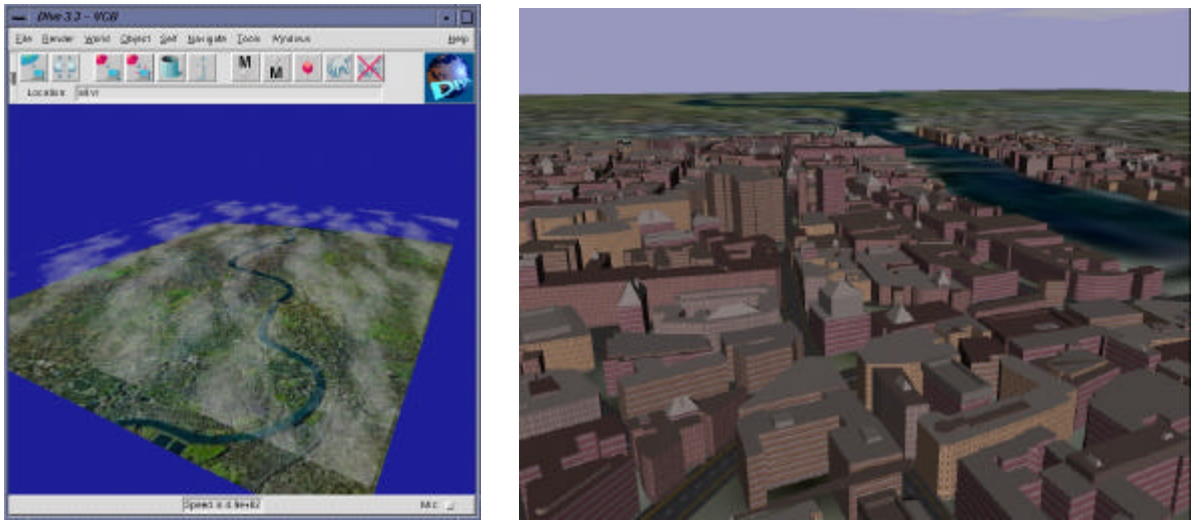

Figure 1: (a) The London model from the south-west. (b) A typical view of the model from just above ground level

have made them platform level services that any application can take advantage of (see Section 4.2).

1.3 The COVEN Project

The Collaborative Virtual Environments (COVEN) Project brings together twelve academic and industrial partners with a wide range of expertise in CSCW, networked VR, computer graphics, human factors, HCI and telecommunications infrastructures [11]. COVEN is developing platforms for large-scale virtual environments, and is demonstrating applications in the general area of virtual travel. There are two main demonstrator themes, the *citizen application*, a travel agency service for the general public, and the *business application*, a series of scenarios for the professional user. The London Demonstrator described in this paper is the culmination of the business application theme.

The project has undertaken research in novel interfaces for virtual environments, human representation, crowd simulation, collaboration services and network architectures for CVEs. The DIVE platform extensions described in this paper are motivated by much of that research.

2. PLATFORM AND SERVICES

2.1 DIVE Platform

DIVE [3], developed at the Swedish Institute of Computer Science, is an experimental platform for the development of virtual environments, user interfaces and applications based on shared 3D synthetic environments. DIVE is especially tuned to multi-user applications, where several networked participants interact over the Internet. Started as a lab tool in 1991, DIVE has evolved into a well-developed system running on many platforms[1].

2.2 Basic Services and Extensions

The demonstrator relies on many of the basic services provided by the DIVE platform. These include support for live spatialised audio, support for web integration and scripting using the TCL language [12].

The most recent version of DIVE, 3.3X, includes the scalability extensions described in this paper. These are: subjective views, high-level application events, lightweight groups, rendering extensions and database extensions. Although the extensions have been made to support this application, they have been written in an application independent way and should be applicable to other large applications.

COVEN has also contributed to other extensions that are not detailed in this paper. These include user-interface improvement, wider geometry format support, mesh optimisation and compression, plugin interface and Java API.

3. APPLICATION COMPONENTS

3.1 Model

The basic element of a travel demonstration must be a representation of the destination. Our model of London comprises a 16x10km segment of the centre of London that is generated from 2D vector, building height and 2D image data from the Cities Revealed data set[2]. The resulting model consists of about 160MB of VRML data, not including texture map data.

The original maps are stored as 2D DXF files with several layers each containing a different type of building structure. We are primarily interested in building outlines, roads centre and building spot height layers. The conversion stage consists of the following major stages [19]:

- Identifying and repairing building outlines

[1] DIVE runs on many UNIX platforms and on Windows NT. DIVE 3.3X was recently made available at *http://www.sics.se/dive*

[2] The Cities Revealed data set is licensed from The GeoInformation Group.

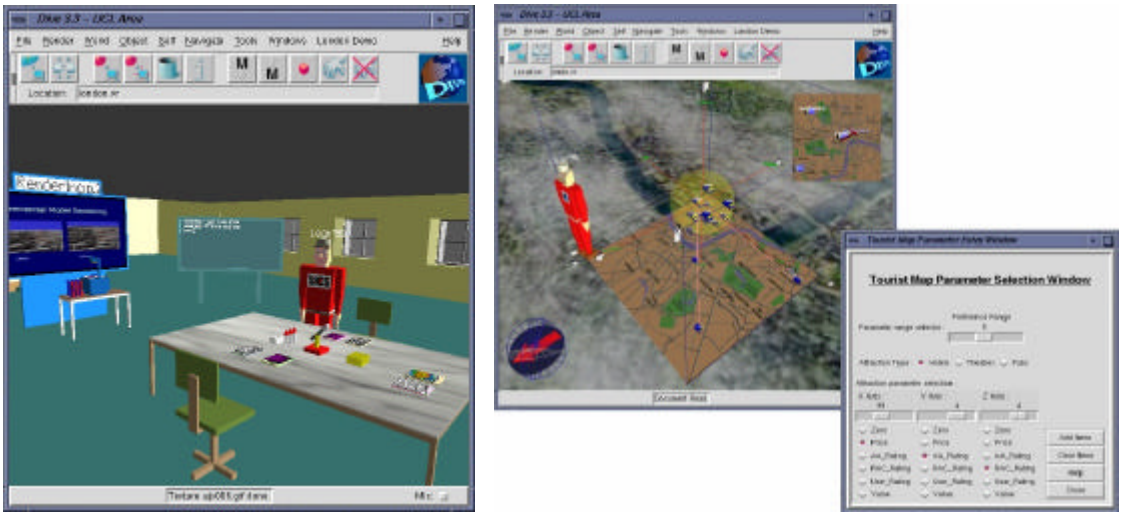

Figure 2: (a) One of the conferencing suites inside the Pearson Building model (b) The tourist data visualisation tool. The personal map can be seen on the top right of the main view window

- Building extrusion and roof-fitting
- Identifying and connecting road centre lines
- Fitting roads between buildings
- Creating pavements

Each building is composed of two or three layers, entrance, middle and optional upper level and each has a roof. The entrance and upper level are tiled laterally, but are repeated only once vertically and have a fixed height. The centre layer is tiled and/or slightly stretched to fill the intervening gap. Textures are grouped together into matching sets and the set to apply to a particular building is chosen by a few simple heuristics: buildings with a small footprint are more likely to be brick, tall buildings are likely to office blocks, and so on. The resulting model is stored as a VRML1.0 model in 160 1km square tiles, with each tile being built upon a segment of the Cities Revealed aerial photograph for central London. Figure 1 shows an overview of the model and an example view from ground level.

A few of the buildings in the model have been modelled at a higher level of detail. In this demonstrator we used UCL as a target destination for travel, though the services described in later sections could be integrated in other parts of the model as appropriate. The Pearson building is modelled in great detail, with approximately 100 rooms, some containing furniture or conferencing facilities (see Section 3.2). Figures 2(a), 3(a) and 3(c) show some examples views of this section of the model.

3.2 Collaboration Services

Conferences and meetings enable people to exchange information and resolve issues in an optimal way. Virtual conferencing is an interesting application area that allows reducing costs and saving time by enabling distributed users to participate without the need for (normally lengthy) travelling. However, in some situations, physical contact is an essential human factor that can not be replaced. In the London demonstrator, we address this issue by allowing users to collaboratively plan a real meeting from their separate geographical locations. The virtual meeting will take place in a model of the building that will host the final meeting, so that prospective visitors may get acquainted with the physical building. The demonstrator uses a COVEN plenary meeting that would take place at UCL as an example. The meeting takes place in the virtual room that represents the targeted meeting room. As a result, users will be able to rehearse their way to the main conference room.

In this case the supporting tools can aid in the preparation of the meeting. For example, an agenda for the real meeting can be prepared virtually. Other cases include ensuring the correct size of room for the number of participants, and if breakout sessions emerge is there sufficient space/rooms for interested parties. Providing the users with a representation of the actual intended meeting space enhances such organisation aspects. To support such a wide range of interactive meetings, a suite of collaborative tools is offered.

The chosen metaphor consists of implementing virtual counterparts of the different items that can generally be found in meeting rooms. The meeting rooms stress a natural point where the participants can get together, generally a table surrounded by chairs. In real meeting situations notebooks, hands-out, and slides are used, the rooms of the London demonstrator contain text- and WWW- based counterparts that allow for collaborative note taking and multimedia document visualisation. Virtual documents are “personal” in the sense that they can be used by one person at a time. However, placing them on a virtual overhead projector allows for simultaneous visualisation at all connected sites. Additional tools include virtual folders, mailboxes, text communication board, PostIt notes, etc. Finally, we use the metaphor of a virtual microphone to enforce turn taking in collaborative situations that require a single speaker and several listeners. One of the conferencing suites is depicted in Figure 2. More detailed information about the conferencing tools and their implementation can be found in [4].

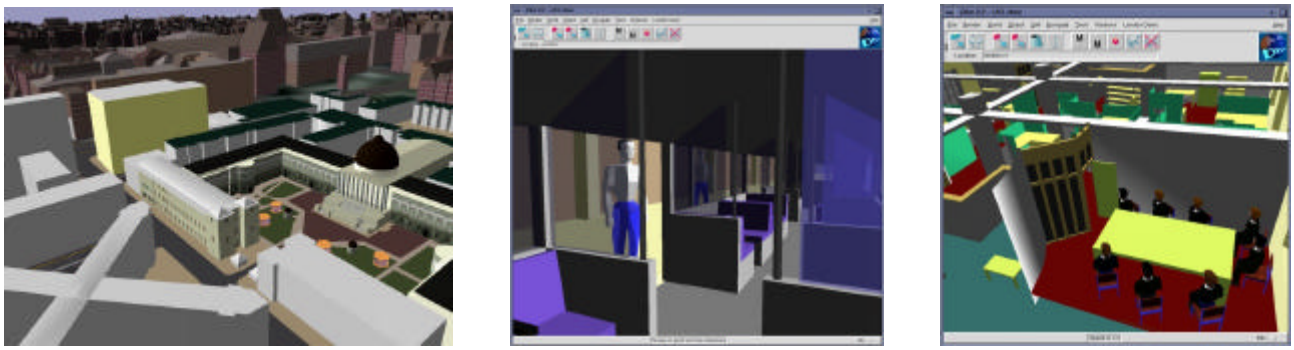

Figure 3 (a) Overview of the UCL front quad. The Pearson Building is the below and to the left of the center of the screen. (b) View from a London Underground train. Two members of a crowd are visible. (c) Cut-away view of Room 127 in the Pearson building showing the audience simulation situated in a seminar room.

3.3 Data Visualisation

The DIVE city visualisation tool has been developed to help users select places of interest according to predefined requirements. To enable this the tool retains a database of attributes (e.g. hotel prices, star ratings, etc.) for each attraction type. Therefore a database entry is maintained for each instance of the attraction containing numerical measures for each attribute. The tool then compares this to user requirements selecting the most appropriate attraction according to the user's selected acceptable range.

In this demonstrator attractions may be chosen from pubs, hotels, and theatres with the tool showing the appropriate attractions' location on the city map in the base of the visualisation tool. From this users may add attractions that relate to their requirements to a selected list. This is performed after they are happy with the results of the search. Subsequently a user may then decide to go on and look for other types of nearby attraction, the distance between the two affecting their selection of hotel, for example.

For instance, consider a scenario where a visitor to the city requires a hotel and theatre within a reasonable physical distance of each other. The visitor only has a certain amount of money to spend on the hotel, and requires it to have a reasonable AA & RAC "star" rating. Similarly for the theatre they require one with an adequate seating capacity to accommodate their party, and a short distance between the theatre and the hotel. The city visualisation tool can be used to help support this decision process.

Using the city visualisation tool in the above scenario the user may specify the attributes that a suitable hotel must posses. For example, the user may select the world's X-axis to reflect the hotels' RAC rating, the Y-axis to reflect the price, and the Z-axis to indicate the AA rating (see Figure 2(b)). The acceptable range of values that users are willing to allow the tool to use as a criterion to select hotels should then be given to the tool. Appropriate hotels that meet the user needs, within the chosen selection range, are then added to the visualiser showing their location on the visualiser's base map. These may then be added to the user's selected list of attractions. Next the user may then choose to look at theatres. Again they will select an acceptable selection range for the tool's theatre selection criterion, and input their requirements for the theatre (e.g. price, number of seats, etc.) into the tool setting the attributes to be reflected on the 3D-graph axis shown in the world. After completing these steps the user should amend the suggested theatres to their selected list of hotels.

Upon updating their selected list the user may then turn on a personal map that they can take away from the visualiser and navigate around the city model to look at a 3D scale representation of their chosen locations (in this case in London) (see Figure 2(b)). Both the visualiser's base map and the user's personal map show the locations of the attractions in the city. Therefore the visualisation tool has helped the user choose the most appropriate hotel that not only fits their selection requirements (as all of the hotels on the selected list should do), but also one that is within an acceptable distance from the theatre. The user can visually assess these distances from the spatial information provided on the base map. If more information is required about a particular attraction to help the user make their final selection, then the user may click on the attraction's icon on the map of the visualiser to display detailed information from the database relating to the item.

Clicking on an attraction icon in the personal map will transport the user to that location in the city model (the visualiser is itself located above the city). The selected attractions' location in the city are represented via a large "pin" in the city model so that the selected locations stand out while navigating the large city model. The personal map remains with the user no matter where in the city the user is located. A "you are here" pointer on personal map shows the direction and position the user is currently situated in respect to the city map and their selected attractions. Throughout the user may choose to transport to another selected attraction by clicking on icons in their personal map.

The tool providing the interactive elements and user interface has been implemented in DIVE with TCL/TK.

3.4 Simulations

The demonstrator includes three real-time simulations that enhance the travel scenario. The first is a simulation of journey on the London Underground that arrives at a station close to the

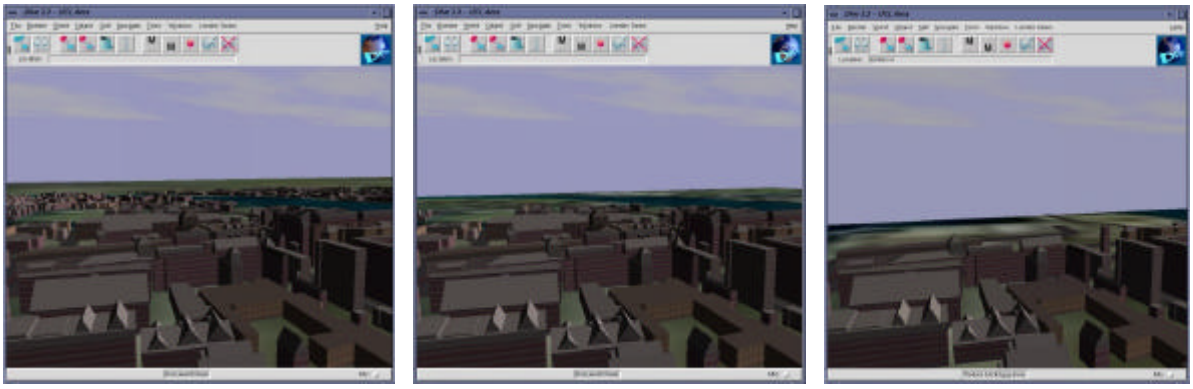

Figure 4 Example of constant frame-rate rendering. From left to right: view on Onyx at 10fps, view on Onyx at 20fps, view on O2 at 5fps.

UCL campus (see Figure 3(b)). This allows a UCL visitor to rehearse the actual route from the station to UCL and through the computer science department. Models of Heathrow Terminal 3, the Heathrow Express train service and Paddington Station will be integrated in the near future, and this will allow participants to experience the journey from the airport to our department.

The second simulation is of crowds of people in the local area. These can assist the lost traveller since they indicate major routes through the city. Currently the simulation focuses on a small crowd of 20-30 participants who follow routes in around the local London Underground station and UCL campus. A couple of members of a crowd can be seen through the window of the train in Figure 3(b).

The final simulation component is of an audience in a seminar room [13]. This has been integrated within one of the Pearson building seminar rooms (see Figure3(c)). The purpose of this simulation is to enable talk rehearsals for prospective conference attendees, or for support of those who suffer from a fear of public speaking.

All three simulations use holders (see Section 4.3) to limit event distribution to only those interested participants that are close to the simulation.

4. SCALEABILITY SUPPORT

Scaleability is enabled through extensions in three areas: extended facilities for event and scene graph management, extensions to the renderer to support constant frame-rate rendering and database extensions.

4.1 Extended Facilities

4.1.1 Subjective Views

In traditional virtual environments each user is present in the same virtual world, albeit from a different viewpoint. Users cannot tailor their representation of the virtual scene or the degree to which they are aware of other user's activities. This is somewhat analogous to early 2D shared multi-user interfaces where users were each presented with the same application views. However, research in 2D interfaces has shown a strong trend to support individual tailoring of the shared views to reflect user demands. These lessons from 2D interfaces have been integrated into DIVE, allowing individual users to have more control over their view of the virtual environment. This provides users with the ability to tailor the environment to suit their working needs, and enables applications to address users on a more individual basis.

These considerations have led to the implementation of subjective views [14] in the DIVE platform, together with their control from various interface levels such as the file format, the behaviour interface and low-level C programming. Subjective views allow tweaking the representation of individual objects for (groups of) individual users. Additionally, subjective modifiers describe independent effects that may be applied to any object's appearance. Examples of such modifiers are:

- *Normal*: View the object normally, i.e. no modifications;
- *Invisible*: The object should not be visible to this user;
- *Transparent*: Make the object transparent;
- *Wireframe*: Show the object in wireframe mode;
- *Dim*: The object should be visible but less obvious, such as made darker;
- *Bright*: The object should be presented to the user and emphasised within their visualisation by means such as increasing its luminosity.

In the London demonstrator, we have used subjective views extensively to perform visibility switching on a per-user basis. As with most features of the platform, the subjective views mechanism is interfaced with DIVE/TCL. This makes it possible to use complex predicates to toggle object visibility at run-time.

4.1.2 *High-level Application Events*

DIVE is event based. Modifications applied to the database will result in network messages that are distributed to connected peers. An event will be generated both at the sender and the receivers. Applications process can register to be notified of particular events. In addition to a set of predefined system events, such as user interaction, object collision, property manipulations, DIVE offers a framework to generate and intercept application-level events. These events can be triggered on any object of the shared environment. They are typed using a string

and are supported by streaming mechanisms that let applications to decide upon their content and organisation. These semantic-rich events can be used to send commands to applications, together with arguments and, thus, let applications define a standardised communication protocol.

4.1.3 Lightweight groups

For data communication, any object of the DIVE database hierarchy can be associated with a multicast group, called a lightweight group. When a modification message concerning an object is to be sent, we ascend the database hierarchy, starting from this object and as soon as a multicast group is found, this will be used as a communication medium. If no group is found during the ascent, DIVE will use the default world group associated to the top-most object. If the multicast group associated to an object is a null group, that part of the hierarchy will be local and modification messages will thus not be distributed at all.

Lightweight groups can be associated orthogonally to the hierarchy tree; i.e. several branches can be associated with the same group. The lightweight group mechanisms are interfaced and controlled from a high level of abstraction using the DIVE/TCL scripts. This provides the platform with a flexible tool that makes it possible to experiment with a variety of application-dependent distribution schemes without requiring a hard coding of the semantics into the platform.

DIVE sends continuous streams of data (i.e. audio and video) using unreliable multicast. DIVE associates separate audio and video multicast groups to each world. Additionally, lightweight audio and video groups can be defined. The logic of lightweight audio and video group selection is similar to that described above for lightweight object groups.

4.2 Rendering Extensions

The London model is massive and from any viewpoint many hundreds of thousands of polygons maybe seen. We have experimented with renderers that aim to give a constant frame-rate experience so that visual continuity can be maintained.

Although a technique such as occlusion culling would be eminently suitable for street level locomotion about the city, occlusion is not a general technique applicable to all models, and even in this demonstrator we find that most time is spent above roof height. For these reasons, we did not choose to integrate occlusion culling into the platform because of its specificity.

A reasonable and more generally applicable technique is to optimise the depth of the far clipping plane so that a frame-rate target is met. When the rendering completes within the required time the clipping plane is slowly moved outwards and conversely the clipping plane is moved inwards when the rendering overshoots the desired target. The change in depth is damped so that the frame-time does not oscillate and objects do not pop on and off.

This technique requires no prior knowledge of model structure, although it is more successful when there are large numbers of small or mid-sized objects rather than a few large objects. Figure 4 shows three example of the frame-rate limiting in action. Figure 4(a) and 4(b) were taken on a R10000 Onyx with 196M of ram and Infinite Reality graphics system. Figure 4(c) was taken on an R5000 O2 with 128M ram.

For the densely populated interior areas of the model, specifically the Pearson building of UCL, we have used an explicit, conservative visibility technique. The model is not suitable for computed visibility solutions such as those developed by Funkhouser [7] since the model is not structured with the required identification of cells and portals, nor would it be at all simple to identify them automatically. However it is very easy to identify large areas of mutual occlusion such as separate floors and groups of rooms by hand. We have integrated tools into DIVE itself that allow objects to be grouped together by hand and identified as "cells". These cells can be turned on or off depending on the user's location using the subjective views capability (see section 4.1.1). Although the resulting visibility relationships are much more conservative than an analytic solution, they are simple for the scene author to describe, and since this facility is available through DIVE/TCL, the types of visibility relationship can be based on complex predicates.

The challenge we are now facing is incorporating other extensions such as incremental rendering, visibility culling, image based rendering and incremental level of detail control into a single renderer in a generally applicable way [16].

4.3 Database Extensions

Given the size of the base model, only a small portion can be can be kept in memory or rendered at one time so it must be stored in paged tiles. The tiling provides a natural scoping of world events since tile proximity can be used as a gross indication of mutual awareness. Also, the tiles themselves are static objects in the environment since we don't expect the buildings themselves to be changed, or at least not very frequently.

These types of consideration led to the development of a world structuring technique called *holders*. Holders are a database abstraction that builds on top of the low-level lightweight groups described in section 4.1.3. A holder is a unsynchronised portion of the hierarchy that carries a URL or file information and a multicast address. Holders are not loaded through the world database mechanisms but are explicitly loaded by each peer. This relieves another peer from having to serialise and send the state of that portion of the scene graph. Events that are then generated within the part of the hierarchy below a holder are only distributed to parties connected to that holder.

Holders make it also possible to avoid distributing network messages to other participants by using null groups. In this case, all connecting peers will read the associated URL, and no local change messages will be communicated. However this does not mean that these portions of the hierarchy are static. An event originating in the holder can be routed via a node at a higher level of the hierarchy, outside the holder, and peers can on reception of this high-level message make local changes to their own copy of the holder as needs be. Such high-level events, described in section 4.1.2, can encapsulate complex behaviour as described in Figure 5 and we have used them for this purpose in a number of situations. Note that holder connection and management is interfaced to DIVE/TCL, which allows describing their behaviour using scripts.

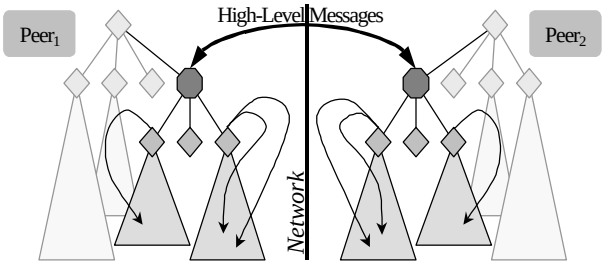

Figure 5: Semantically enriched application-level events can be used together with holders to cut down radically network traffic in some situations. This figure exemplifies how a few high-level messages can generate a cascade of database modifications simultaneously, instead of generating many network packets describing the modifications.

For this demonstration, each map tile is placed under a separate holder. The holders are loaded when the user collides with a bounding sphere. Since this behaviour is scripted in DIVE/TCL, more sophisticated look-ahead paging could be performed. Individual tiles are loaded incrementally in order to maintain a constant frame-rate by not locking out the rendering process. Holders are also used within buildings where large areas of occlusion occur as a first, crude visibility partitioning. In that case, holders are used together with subjective view mechanisms (Section 4.1.1) that toggle the visibility of large parts of the model for individual users. Finally, holders are used to encapsulate the simulations. For example, the crowd simulation is contained within a holder. A high-level application event (Section 4.1.2) is used to communicate current positions of the crowd members, but each peer independently animates the walk cycle, since there is no need to synchronise these animations between peers.

All files used to initialise the content of the holders used in the London demonstrator have been saved in the DIVE binary format, a general format that had to be developed to support this application. The binary format sits on top of the hierarchy serialisation mechanisms for socket communication and this format both significantly improves file-loading time and saves disk space.

5. TESTING AND EVALUATION

The types of service that are integrated into the current DIVE platform result from previous studies of people using CVE systems [17][18]. The London Demonstrator has been refined both by our experience in a series of network trials and focused usability evaluations.

5.1 Network Trials

Network trials form a large part of the COVEN Project's work. From March to August 1999 roughly 20 network trials were undertaken, with the number of participants varying from 4 to 16.

Figure 6 shows a snapshot from one of these trials involving eight participants taking part in a one-hour session based on a conference planning scenario. The eight participants were composed of four groups of two with groups in London, Nottingham, Lancaster and Stockholm. Participants were using Linux, SGI Impact and SGI O2 computers. The trial was held over the Internet in mid-afternoon when congestion would have been expected to be reasonably high. The details of how such trials were supported using the facilities of the DIVEBONE, an application layer multicast bridge, can be found in [6].

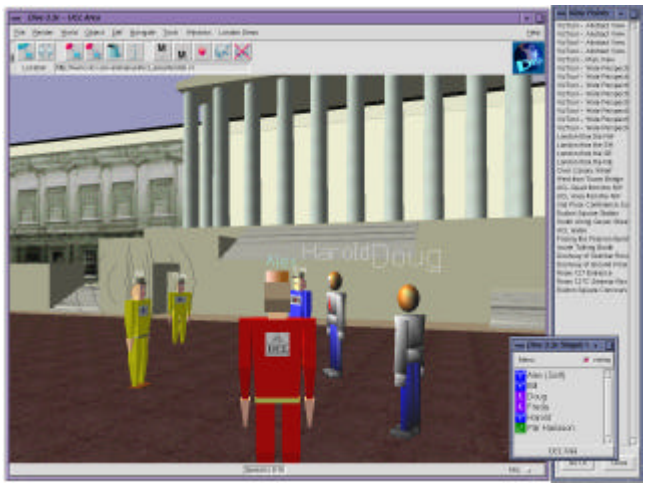

Figure 6: A group of users meeting in UCL's front quad.

This trial was very successful, in that the application supported the eight mutually aware participants and they were able to access the collaborative facilities provided. Despite the range of machines used, the scalability extensions ensured that the experience was interactive and responsive for all participants. Indeed larger numbers of users were supported on other occasions, and we fully expect to be able to support a few tens of people simultaneously in this application. With most other CVE systems and with previous versions of DIVE, it simply wouldn't have been possible to host such an application because of the lack of rendering and database control with extremely large models.

5.2 Usability Evaluation

The format and tasks undertaken during the network trials are designed to answer certain questions about the usability of the 3D user interface and application components. We have found, for example, that the spatialised audio greatly enhances the ability of the users to communicate with one another since speakers can be identified by audio location.

We have, in parallel, also been applying prototype methods for usability inspection of CVE applications [15]. These have served to improve aspects of the interface such as object representation, menus, text chat interface, usage of private audio groups and avatar representation.

Certain facilities such as the audience simulation are the subject of more in-depth studies [13], and the general area of small-group collaboration is one that we have been studying using elements of this application [17][18].

6. CONCLUSIONS

The COVEN project has investigated several aspects of CVE design, implementation and evaluation [11]. The resulting experience has been applied to the design and creation of the

COVEN/DIVE platform that integrates many novel services and enables a large class of CVE application that wasn't previously possible. With the London Demonstrator we have shown new levels of scaleability whilst retaining core collaborative services.

- The demonstrator successfully integrates several applications into one coherent space. The model itself provides a space within which to integrate other travel information or data visualisation services.
- The resulting application is interesting from a user's perspective and provides broad functionality to support a general class of travel applications.
- Scaleability is achieved both in terms of numbers of users and size of the model
- Rendering and database extensions are integrated at a system level so are generally available to other applications and other large models.
- Scaleability support is provided in a flexible manner through the scene scripting language.

Future work will concentrate on further scaleability extensions. Foremost amongst these will be the integration of further geometric visibility techniques, more advanced rendering management and geometry streaming.

7. ACKNOWLEDGMENTS

This research was funded by the European ACTS programme (COVEN AC040). We would like to thank all the project members whose work is reflected in the design and implementation of the London Demonstrator. We would also like to thank those who took part in the network trials and those who participated in the usability inspections. The crowd simulation was implemented by Bernhard Spanlang whilst he was visiting UCL.

8. REFERENCES

[1] Barrus, J., Waters, R. and Anderson, D. Locales: Supporting Large Multiuser Virtual Environments, IEEE Computer Graphics & Applications, Vol. 16, No. 6, 1996, 50-57.

[2] Floyd, S., Jacobson, V., McCanne, S., Liu, C.-G. and Zhang L. A Reliable Multicast Framework for Light-Weight Sessions and Application Level Framing, in Proceedings of ACM SIGCOMM 95 (New York, ACM Press, 1995), 242-256.

[3] Frécon, E., and Stenius, M. DIVE: A scaleable network architecture for distributed virtual environments, Distributed Systems Engineering Journal, Vol. 5, No. 3, 1998, 91-100.

[4] Frécon, E., Avatare-Nöu, A. Building Distributed Virtual Environments to Support Collaborative Work. 1998 ACM Symposium on Virtual Reality Software and Technology (VRST'98), Taiwan.

[5] Frécon, E., and Smith, G. Semantic Behaviours in Collaborative Virtual Environments, Proceedings of 5th Eurographics Workshop on Virtual Environments, 1999.

[6] Frécon, E., Greenhalgh, C. and Stenius, M. The DIVEBONE - An Application-Level Network Architecture for Internet-Based CVEs, ACM Symposium on Virtual Reality Software and Technology (VRST '99). University College London, London, Uk. December 1999.

[7] Funkhouser, T.A., Sequin, C.H. and Teller, S.J. Management of Large Amounts of Data in Interactive Building Walkthroughs, Proceedings of 1992 Symposium on Interactive 3D Graphics, Computer Graphics, 11-20, ACM Press

[8] Funkhouser, T. Database Management for Interactive Display of Large Architectural Models, Graphics Interface '96, Canadian Human-Computer Communications Society.

[9] Greenhalgh, C. Large Scale Collaborative Virtual Environments, to be published spring 1999, London, Springer-Verlag.

[10] Macedonia, M., Zyda, M., Pratt, D., Brutzman, D. and Barham, P. Exploiting Reality with Multicast Groups: A Network Architecture for Large-scale Virtual Environments, in Proceedings of IEEE Computer Graphics & Applications (1995), 38-45.

[11] Normand, V. *et al.* The COVEN Project: Exploring Applicative, Technical and Usage Dimensions of Collaborative Virtual Environment, Presence: Teleoperators and Virtual Environments, 8(2), 1999.

[12] Ousterhout, J. Tcl and the Tk toolkit. Addison-Wesley, Reading, MA, 1994, ISBN 0-201-63337-X.

[13] Slater, M., Pertaub, D. and Steed A. Public Speaking in Virtual Reality: Facing an Audience of Avatars, IEEE Computer Graphics and Applications, 19(2), March/April 1999, p6-9.

[14] Smith, G. and Mariani, J. Using Subjective Views to Enhace 3D Applications. ACM Symposium on Virtual Reality Software and Technology (VRST '97). ACM Press. Swiss federal Institute of Technology (EPFL), Lausanne, Switzerland. September 1997. Pages 139-146.

[15] Steed, A. (ed) Usability Evaluation of the Online Demonstrators, COVEN Deliverable 3.5A, available at http://www.cs.ucl.ac.uk/research/vr/Coven.

[16] Steed, A. and Frécon E. Building and Supporting Large-Scale Collaborative Virtual Environments, Proceedings of UKVRSIG'99.

[17] Steed, A., Slater, M., Sadagic, A., Bullock, A. and Tromp, J. Leadership and Collaboration in Collaborative Virtual Environments, Proceedings of IEEE Virtual Reality '99 Conference, Houston, Texas, March 1999.

[18] Tromp, J., Steed, A., Frécon, E., Bullock, A., Sadagic, A. and Slater M., Small Group Behavior Experiments in the COVEN Project, IEEE Computer Graphics and Applications, Vol. 18, No.6, Nov/Dec 1998, 53-63.

[19] West, J. MSc Thesis, Department of Computer Science, University College London, 1998.