INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

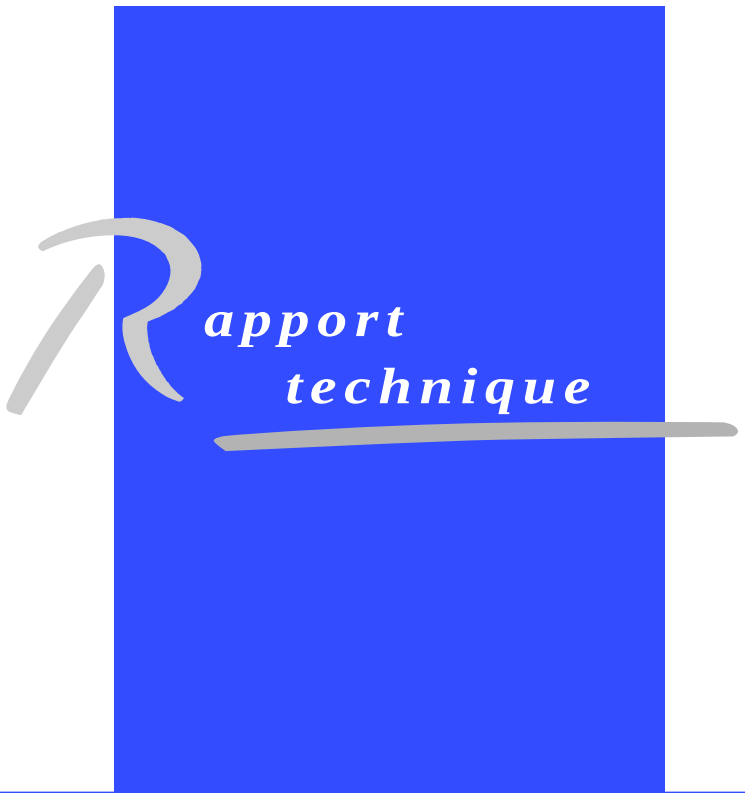
V-EYE
A Very Large Scale Virtual Environment for multimedia conferencing

Thierry Parmentelat — Alexis Gourdon — Thierry Turletti — Élodie Larreur

N° 0296

May 2004

Thème COM



ISSN 0249-0803

V-eye
A Very Large Scale Virtual Environment for multimedia conferencing

Thierry Parmentelat* , Alexis Gourdon† , Thierry Turletti‡ , Élodie Larreur§

Thème COM — Systèmes communicants
Projet Planete

Rapport technique n° 0296 — May 2004 — 22 pages

Abstract: This paper describes a prototype application named V-EYE that implements a Large Scale Virtual Environment (LSVE), combining a 3D world, textual messages, multimedia communications and High Definition video.

When developing V-EYE, our primary goal was to create a platform suitable for easily experimenting multimedia transmission, with a large number of participants, and on heterogeneous IP networks, e.g. combining a very powerful backbone such as VTHD, together with wired LANs, as well as WLANs in the IEEE 802.11 family. In particular, this platform is very useful for evaluating the scalability of transmission protocols and the support of differentiated services for multimedia applications.

V-EYE uses the SCORE communication architecture. In this approach, agents perform a real-time mapping of the geographic area into spatial areas whose sizes depend on the density and location of participants; each of these cells is associated with a multicast group. In this way, each participant can focus on his area of interest, and receive only the relevant traffic.

Key-words: multimedia conferencing, IP multicast, scalability, virtual environment, RTP reflector, High Definition Video

* INRIA Sophia Antipolis, Thierry.Parmentelat@sophia.inria.fr
† INRIA Sophia Antipolis, Alexis.Gourdon@sophia.inria.fr
‡ INRIA Sophia Antipolis, Thierry Turletti@sophia.inria.fr
§ FT R&D Lannion, elodie.larreur@rd.francetelecom.com

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : +33 4 92 38 77 77 — Télécopie : +33 4 92 38 77 65

V-eye
Un environnement virtuel à très grande échelle pour la conférence multimédia

Résumé : Nous présentons dans ce papier une application expérimentale baptisée V-EYE qui implémente un monde virtuel à très grande échelle, et qui combine un monde en 3D, des messages textuels, des communications multimédia et la Vidéo Haute Définition .

Ce développement a pour but principal de créer une plateforme qui permette d'expérimenter commodément des transmissions multimédia avec un grand nombre de participants et sur des réseaux hétérogènes, avec par exemple une dorsale très rapide comme VTHD, des résaux locaux filaires ou sans fil de la famille 802.11. En particulier, cette plateforme est très utile pour évaluer les capacités de passage à l'échelle des protocoles de transmission, et le support de services différenciés pour les applications multimédia.

V-EYE repose sur l'architecture de communication SCORE. Dans cette approche, des agents assurent le découpage en temps réel de la zone géographique en zones dont la taille dépend de la position et de la densité des participants; chacune de ces cellules est associée à un groupe multicast. De cette manière, chaque participant peut se concentrer sur sa zone d'intérêt, et ne recevoir que le trafic correspondant.

Mots-clés : conference multimédia, multicast IP, passage à l'échelle, environnement virtuel, réflecteur RTP, vidéo haute définition

Contents

1 Introduction

We first give an overview of the application capabilities (section 2), and then describe the adopted software architecture (section 3), that is vastly based on publicly available pieces of software, and give indications on the platform requirements.

Almost all communication within V-EYE use IP multicast. We give further details on the SCORE component (section 4), that is central with respect to the multicast usage strategy.

After a section dedicated to discussing some experimental results (section 5), we provide a comparison with other related virtual environments (section 6), and conclude on potential tracks for future work (section 7).

2 V-eye features

The V-EYE (Virtual Eye) application aims at demonstrating the benefits of multicast IP streams for implementing a Very Large Scale Virtual Environment (VLSE). The goal is to communicate among a set of participants that may scale up to the several thousands. Each participant moves within a synthetic 3D geographic world, and typically sends several streams of data :

- ◇ **audio** content as captured by his microphone,
- ◇ **video** as captured by a webcam,
- ◇ textual **messages** as typed at the console, and
- ◇ **location** messages so that his moves are correctly echoed to other participants.

Figure 1: A sample snapshot of a V-EYE session

Figure 1 illustrates these functions, and the additional feature that we implemented within V-EYE, namely a virtual movie theater, that allows - provided that sufficient bandwidth is available - to watch a movie multicasted in High Definition Digital Video (*DV*)

format. This feature supports 3D-perspective display of the video content, as well as mixing of the audio soundtrack with the other participants' audio streams. All the audio streams are also rendered in 3D, so as to provide realistic indications of the source position.

Principle of locality. The *map* in the lower right corner gives an overall view of the world. It also shows, more interestingly for our purpose, the different levels of *neighborhoods*, or *scopes*, that result from the current settings. The light gray area is the one where we can have audio and video contacts, as explained below. The medium gray zone surrounding it is the space where other people can be seen in the virtual world. When in this area, the different participants in the vicinity appear as their *avatar*, together with a distinctive *face* picture that allows to recognize them. Finally the dark gray area depicts the zone from where no information is currently received.

The well-known conferencing tools `vic` (Video Conferencing) [17] and `rat` (Robust Audio Tool) [7] appear in the upper left corner. Through these tools the user can have audio and video contacts with the participants located in the light gray zone.

3 Software Architecture

3.1 Overall architecture

In our virtual world, some components are static such as the world itself, while others, like the participants, are dynamic. One possible architecture is to use a scene graph structure provided for example by a modelling language such as VRML, and to program the interactions in JAVA, like e.g. in Mimaze-3D [13].

This approach is workable as long as there are few participants in the world. We preferred to use a more flexible approach which has the following characteristics :

- ◇ Use customized version of `vic` and `rat` conferencing tools, for maximizing software re-use.
- ◇ Use MBUS [8] as the communication mechanism with `vic` and `rat`.
- ◇ Re-use as much as possible the DV-oriented code from DVTS [19] for dealing with the DV video stream.
- ◇ Use of the OpenGL library for the 3D rendering of our data, together with portable toolkits for managing OpenGL windows (GLUT and GLUI).
- ◇ Use of an Oriented-Object programming language (C++) to structure our data.

As a result, the software that implements each participant's client is depicted in figure 2.

The interfaces with the conferencing tools and with the DV viewer will be detailed in the following sections. For now, let us stress that the 3D-OpenGL representation allows a better user interactivity and higher performance, since OpenGL is a standard supported by most graphical cards. Some nice effects can be easily added such as texturing, shadows, reflects. Portable toolkits for managing our OpenGL windows (GLUT and GLUI) are also useful to easily creating a simple user-interface.

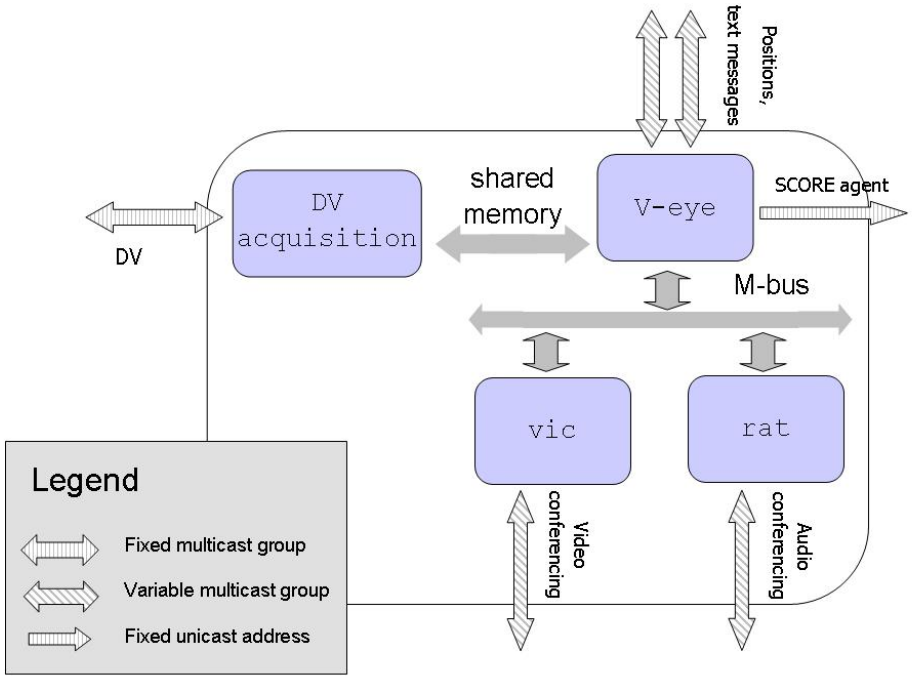


Figure 2: Internal architecture of a V-eye client instance

3.2 `vic` and `rat` conferencing tools

Taking advantage of the SCORE architecture requires `vic` and `rat` to be able to switch to an arbitrary multicast group upon request. For instance, when a participant crosses a cell boundary, the V-EYE process is notified by the SCORE agent that it should use another set of multicast groups.

This is done by the synchronization architecture provided by the MBUS (Message bus) [8]. The MBUS communication mechanism has a simple addressing and message syntax. According to the current cell we can send a message to `vic` or `rat`, telling them when to change their address group. We had to modify `vic` and `rat` so as to handle these specific new (MBUS) messages.

Another modification was performed in order to keep `rat` informed of the location of the other participants, so as to allow for a simple 3D-audio rendering (attenuation accordingly to positions and orientations of speakers and listener).

3.3 High definition Digital Video

The *DV* standard, as defined in IEC 61834, is now widely adopted in home video devices. There are currently numerous software implementations that deal with this format, most of them aiming at easing local interactions between a camera and a computer for capturing, mixing, and playing videos. In order to as well address remote exchanges, the DVTS consortium ([5]) defined and implemented an encapsulation of the *DV* format in an RTP stream ([19, 10, 9]).

As part of the DVTS system as provided at the above location, the `dvsend` tool is able to send a *DV* stream as-is within an RTP stream. Of course this stream can be sent to a multicast address if desired. Its counterpart is named `xdvshow` and allows to display the stream sent by `dvsend`.

Basically, we had to make the following changes to `dvsend` to fit our needs :

- ◇ split the *DV* content into two streams dedicated to video and audio respectively.
- ◇ transcode the audio stream to some widely available audio format, so that `rat` can play it mixed with the other participants voices. To that end we could of course take advantage of the `libdv` [12] library, whose purpose is precisely to provide easy encoding and decoding from and to the *DV* format.

In addition to that, we faced the following issue. So as to be consistent with the SCORE paradigm, these two streams should be sent over a multicast group that should vary over time so as to follow the SCORE mapping, i.e. to geographically match the movie theater location.

In order to solve that issue, we adopted the following path. We started from the tool named `rtptrans` initially part of the `rtptools` package [21]. This works the following way; it takes as an argument a set of RTP streams (i.e. IP address + even port number), and repeats every packet arriving on one of these streams to the $n-1$ other streams, implementing a multi-directional RTP reflector.

Starting from that tool, we wrote `rtpbounce` that acts exactly the same way, except that the set of target RTP streams may be modified on the fly through a small set of commands issued on the standard input.

On top of that, we wrote the `scorebounce` utility; it is a simple SCORE client that takes as an argument :

- ◇ a fixed geographical location - typically the movie theater,
- ◇ a fixed multicast group - typically the one where `dvsend` transmits the DV audio stream,
- ◇ and an "interest radius".

It then basically forks an instance of `rtpbounce` on its fixed RTP stream, and then adjusts the set of target RTP streams to a so-called zone of interest; this zone is computed as the set of cells that intersects with a square centered at the fixed location and as large as the interest radius.

The figure 3 shows how all these elements interact and how we achieve to send the DV streams to the appropriate RTP streams.

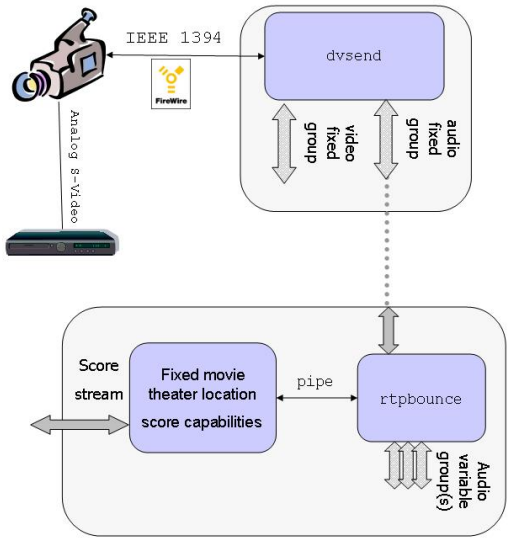


Figure 3: Putting together the `scorebounce` utility

3.4 Hardware and Software Requirements

As of today, V-EYE works on the Linux platform. Here is the list of the recommended configuration for achieving the best results in terms of responsiveness :

- ◇ **Display adapter** : V-EYE being massively based on Open-GL for 3D-rendering the virtual world, it is highly recommended to use an nvidia video adapter, because this driver offers full support for hardware-accelerated Open-GL under Linux. In fact there supposedly are such drivers for all major video adapter manufacturers, but setting up such drivers turned out to be very tedious.
- ◇ **Webcam** : the Philips webcam is preferred for good adaptation within `vic` [20];
- ◇ **Audio Adapter** : `rat` was written a few years ago and is not aware of the latest ALSA revision [1]. You thus need to use an audio adapter that comes with an OSS driver, or deploy ALSA with the OSS-compatibility feature enabled.
- ◇ ***DV*** : the host that runs the `dvsend` software for sending the movie contents needs to have a firewire port, and we had to go for the 2.4.21 kernel in order to have the DV capture work properly.

4 The Score component

In terms of network architecture, there are two extreme, and equally naive, approaches for implementing the traffic streams between actors within a VLSE :

- ◇ The **unicast-only** approach would require every participant to explicitly ask his neighbors to unicast their outgoing streams to him;
- ◇ The **single-multicast** approach is the one where every type of content is assigned a single multicast group.

Each of these methods provide its own pros and cons. The single multicast approach seems to help master the required bandwidth, but unfortunately implies an individual bandwidth that grows linearly with the total number of participants. As opposed with this situation, the unicast approach allows to implement a notion of vicinity - provided that an appropriate infrastructure is deployed - but however the bottom line is that on the uplink, each participant has to send the same stream as many times as he has interested neighbors, which again is far from optimal under some conditions.

Our goal is to build virtual worlds able to scale to potentially thousands of members or more.

- ◇ SCORE is a scalable multicast-based communication architecture for Large-Scale Virtual Environments (LSVE) on the Internet [15]. It is shown in [11] that in a group communication setting, the percentage of useless (or *superfluous*) information received by each participant increases with the number of data flows and the number of users. This is not surprising since within a VE, each participant simultaneously interacts with only a limited set of other participants. The superfluous information represents a cost in terms of network bandwidth, routers buffer occupation and end-host resources, and is mainly responsible for the degradation of performance in LSVE's.

◇ SCORE implements a transport-layer filtering mechanism with multiple agents that allows to filter out superfluous traffic before it reaches the end-host. This approach involves the dynamic partitioning of the VE into spatial areas called *cells* and the association of these cells with multicast groups. The basic idea is to dynamically partition the VE into cells of different sizes, depending on the density of participants per cell, the number of available multicast groups, and the link bandwidth and processing resources available per participant.

4.1 User satisfaction metric

Participants have limited network and CPU processing cycles resources. If the participant's area of interest is so large that the traffic he receives cannot be processed in real time, no mechanism could enable him to receive all the data he is interested in. Indeed, in this case, even if the cells he subscribed exactly match his area of interest, the received traffic exceeds his capacity. For this purpose, a user satisfaction metric is used to take into account the proportion of interesting data rate received and processed, and his capacity corresponding to the maximum data rate that he can handle (limited by his network connectivity and/or processing power). When a participant receives and processes all the data he is interested in, his satisfaction metric is maximal whatever the superfluous traffic rate. Here, the goal is not to adapt to the worst receiver in terms of network connectivity and processing power, but to maximize the satisfaction of the receiver with the lowest user satisfaction value.

4.2 Agents responsibility

In SCORE, *agents* are processes that run at different parts of the network. They are not servers, in the sense they do not aim at processing any global state for the VE, so they do not receive data traffic sent between participants. Actually, agents dynamically determine zones with the VE by considering the distribution of participants and they calculate appropriate cell-sizes according to the density of participants in each zone. Agents also have to periodically process the satisfaction of each participant according to his capacity, the size of his area of interest and the density of participants within his current zone. Once this computation is done, agents can determine new zones (or inversely they can aggregate existing zones), and modify the cell-sizes within the zones where the participants with the lowest satisfaction are located. Therefore, SCORE requires the dynamic partitioning of the VE into cells of different sizes, and the association of these cells with multicast groups. Agents have to dynamically determine appropriate cell-size values in order to maximize users' satisfaction.

During the session, agents perform the following operations:

◇ Partition the VE into several *zones*, according to the distribution of users, the users satisfactions, and the VE structure (e.g., rooms, walls, etc.).

◇ Compute the appropriate cell-size for each zone (see Figure 4).

◇ Divide each zone into cells, according to his computed cell-size, and assign a multicast group address to each cell of each zone.

◇ Inform the participants of which multicast groups they need to join in order to interact with participants located around them.

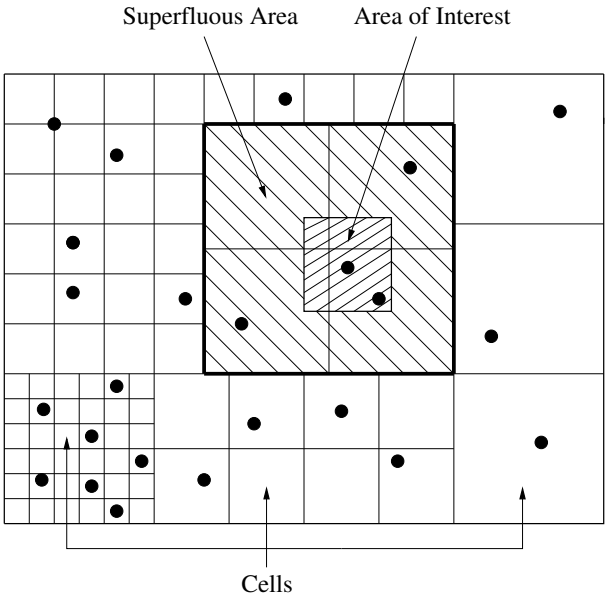


Figure 4: Partitioning with different cell-sizes

The first three operations are called the *mapping algorithm*, whereas the results of these operations as designated as the *mapping information*.

4.3 Mapping information

In order to communicate mapping information to users, i.e., the association between cells and multicast group addresses, it is necessary to find a way to identify and name these cells within the VE. First, the VE is statically partitioned into several large parts, called *start-zones*. These start-zones are actually defined according to the intrinsic structure of the VE (e.g., rooms, floor, walls, etc.) and can't ever be combined. Each start-zone is statically partitioned into indivisible *zone-units* which are the smallest unitary zones that compose the start-zone. During the session, start-zones are dynamically divided into zones which all have the same cell size. So, cells are mapping of multicast groups to a number of zone-units. As agents decide to define new zones in order to take into account changes in the distribution of participants, they identify these zones as sets of one or more contiguous zone-units belonging to the same start-zone.

To summarize, a zone is a subset of a start-zone and is composed by n contiguous zone-units ($n \geq 1$). Within a given zone, all cells have the same size but two distinct zones could have different cell-sizes.

4.4 Participants-to-Agent communication

There are several levels of communication in SCORE :

- ◇ Each participant subscribes to one or more multicast groups but sends data packets on a single group.
- ◇ Each participant is connected to a single agent, using a UDP unicast connection.
- ◇ Agents communicate with each other on a single multicast group: the *Agent Multicast Group* (AMG).

A participant has to subscribe to two different kinds of multicast groups:

- ◇ *data groups* associated to the cells that intersect his area of interest. Note that a participant only sends data to the multicast group associated to his current cell.
- ◇ *control groups* associated to the *start-zones* that intersect his area of interest. For these groups, a participant is only a receiver. Agents use control groups to send mapping information relative to the start zones. This information is periodically sent for each start-zone, and contain the mapping information for all the zones belonging to the start-zone (i.e., the cell-size for each zone and the associated multicast groups addresses).

For each of these groups, the participant has to make early *join*s taking into account his speed, and the join-latency value.

Each participant is connected to his nearest agent using a UDP connection. Each time a participant enters a new zone-unit, he sends a short message to his agent. This message contains his identity, his position in the zone-unit, his current size of area of interest and his capacity [14]. Therefore each agent is able to track the location of its connected users in the VE. In order to evaluate the density of participants within each zone, agents exchange information on the AMG multicast group.

4.5 Mapping algorithm

Agents implement a mapping algorithm to dynamically define zones in the VE, and to dynamically compute an appropriate cell-size within each zone, considering the distribution of participants and their satisfactions. To simplify the computation of mapping information, SCORE only considers square cells, with an integer number of cells per zone. Throughout the session, agents periodically compute the average density of participants per multicast group, by dividing the number of connected participants with the number of available multicast groups for the application. As participants arrive and move in the VE, agents keep track of the density of participants in each zone.

The mapping algorithm consists in the following operations:

- ◇ First, agents compute a cell-size for each zone by only taking into account the distribution of participants in the VE.
- ◇ Then, the participants with the lowest satisfaction are identified as well as their distribution in the VE. If agents detect a concentration of unsatisfied participants within a part of a zone, this zone is divided into two new zones in order to isolate these participants.
- ◇ Finally, agents can decide from time to time to aggregate contiguous zones, if the cell-sizes are the same for these zones and if they belong to the same start-zone.

5 Experimental Results

5.1 Network description and measurement tools

The V-EYE application has been tested over the VTHD network[23] (Vraiment Très Haut Débit). VTHD is a national high speed network, deployed and operated by France Telecom R&D, with the support of RNRT, an initiative of the French Minister of Industry and Research. This platform interconnects sites involved in R&D activities with access links from 1 Gbps up to 2.5 Gbps. Among others, VTHD allows for experimentation related to IP/WDM, protection/restoration mechanisms, CoS and VPN provisioning, IPv6, and Multicast. Theses services are tested with Next Generation Internet applications such as V-EYE, telemedecine applications, and GRID computing applications. VTHD supports multicast services based on both ASM and SSM model.

The main interest of experimenting V-EYE within this environment is that it involves a lot of multicast sources/receivers, related to numerous groups, inside a very high bit rate multicast environment without impacting network running and performances. The goal is also to gain a better knowledge of IP Multicast, and to test equipment interoperability inside a multi-vendor network such as VTHD.

For the sake of these experiments, we wrote a dummy V-EYE client that allows to simulate the presence of a large number of participants. These pseudo-participants - also known as Non Player Characters in the games terminology - are depicted on figure 5

As described on figure 6, this experimentation is simultaneously run on several sites connected to VTHD :

- ◇ FTR&D Lannion,
- ◇ INRIA Sophia-Antipolis
- ◇ INRIA Rocquencourt
- ◇ ENST-Bretagne in Brest

They are located inside separate autonomous systems (AS). Inter-AS multicast configuration has been setup using MSDP/MBGP peering between the RP's of the four AS's involved.

The following tasks were performed in order to setup the network :

- ◇ enabling routes advertisement inside MBGP (multicast sources and RP routers prefixes);

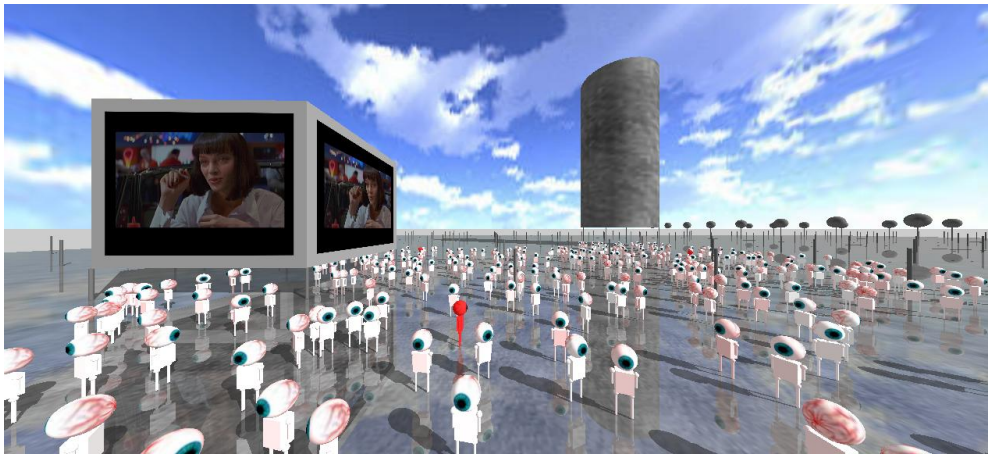


Figure 5: The VE populated with dummy participants

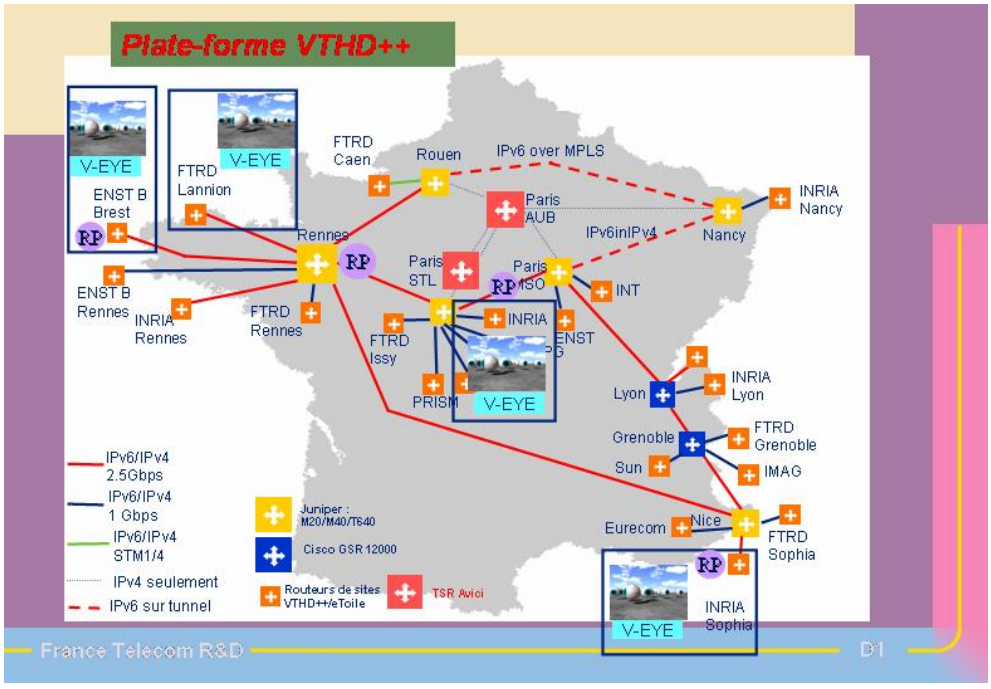


Figure 6: The VTHD network

◇ checking MSDP announces filtering;
◇ defining multicast boundaries.

From a routing point of view, MSDP/MBGP sessions establishment must be checked. Filtering issues encountered between different AS have been solved. Multicast flows are correctly transferred through VTHD network. PIM-SM states corresponding to V-EYE multicast groups are created and maintained inside the core routers in their multicast routing tables.

The measures were made thanks to a couple of Ipanema ip—e1000 passive traffic measurement devices located in the backbone routers close to the border. Being passive, these devices do not interfere with the application traffic; they perform basic identification and timing tasks and send the collected data to a central server (IP-boss) located in the network back office, that computes the resulting measurements, including traffic rate and loss ratio, one-way latency and jitter.

Additionnally, for gathering information on the border networks, we chose to run the IP multicast Monitor named MultiMON[18],

MultiMON is a monitoring tool that collects, organizes and displays all the IP multicast traffic present at the location of the MultiMON server. While MultiMON is a general purpose multicast monitoring tool, it is especially intended to monitor multicast traffic on local network segments and should assist a network administrator in managing the traffic on an Intranet.

Multimon is built on a client/server basis which allows the data collectors (called servers) to be distant from the GUI front end displays (called clients).

When the server is started, a window pops up and shows the detected multicast sessions identified by host/port number, incoming byte count and session type.

On initial startup, network monitoring does not take place; only when a client is requesting data will tcpdump be started, and multicast traffic monitored. When a client is initially started, the client main window displays the total bandwidth occupied by the multicast traffic and provides a graphical (pie chart) breakdown of the traffic by application type.

5.2 Experiment scenarios

End-to-end testing has been realized in order to validate V-EYE multicast software from each site. The following tests have been performed :

◇ the participants can see each other appear in the graphic environment, and can communicate using textual messages, provided they are located in the 'medium gray' range;
◇ they can use remote conference (audio and video) if there are located in the same cell;
◇ each V-EYE host runs multiple dummy eyes, so as to create multicast traffic on many multicast groups;

Additionnally we performed QoS measurements (delays, packets loss, jitter, rate). A few simple scenarios have been defined in order to be able to correlate the data gathered with well-defined actions from the participants, e.g.

- ◇ whether the participants are grouped in the same cell or hanging around the world;
- ◇ whether there are dummy participants present, and if so whether they are concentrated in a given area or scattered throughout the world;
- ◇ whether there is a DV flow flowing in the backbone.

In the following sections (5.3 and 5.4) we will comment the results obtained in two specific runs among the whole collection of data.

5.3 Experiments with audio traffic

First consider an audio flow sent by a V-EYE participant in Lannion - see figure 7.

We can notice that peaks (770 ms of delay, 0.5% of packet loss and 80 ms of jitter) observed for RAT flow match with movements inside virtual world at 09:33, 09:53 and 10:08. Actually, when leaving the departure cell, the participant can't communicate anymore with others since multicast group has changed, unless he is located close to another one during those movements. The same behavior can be noticed for `vic` multicast flows.

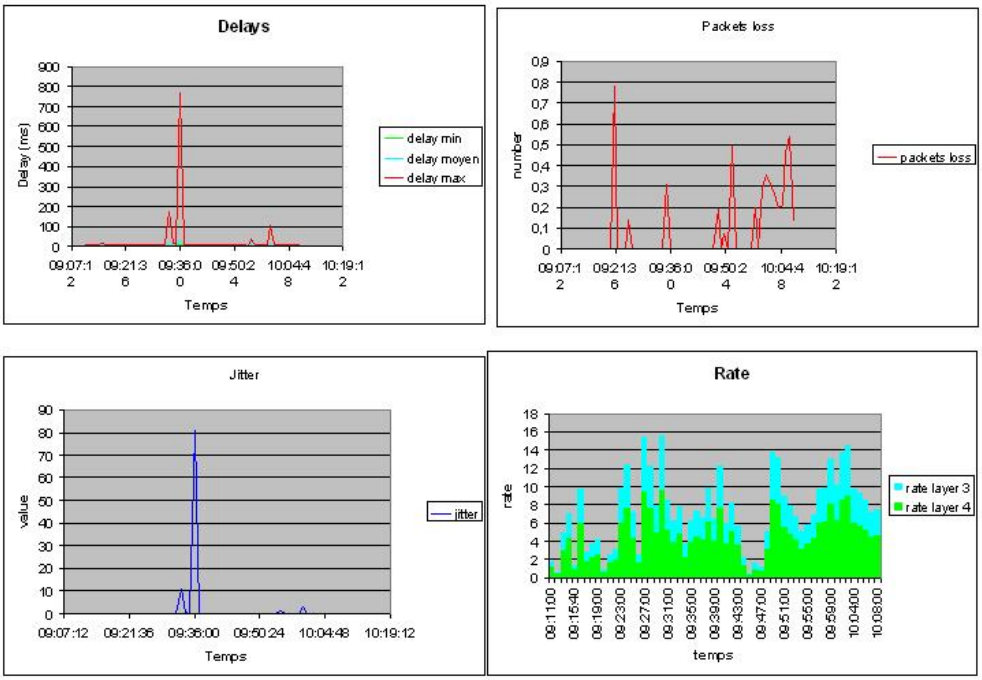


Figure 7: Network characteristics during audio sessions

5.4 Experiments with DV video traffic

If we now have a look on QoS results for a DV flow sent by a Sophia terminal, depicted in figure 8. we observe that bit-rate is quite constant (about 28 Mbits/s): it only decreases when Lannion participant's eye leaves the departure cell (where Digital Video movie is displayed) at 16:18 and 16:23. The participant can't watch the movie anymore if it not located inside its field of view. This fact also explains why there are some delays, packet losses and jitter at the same time.

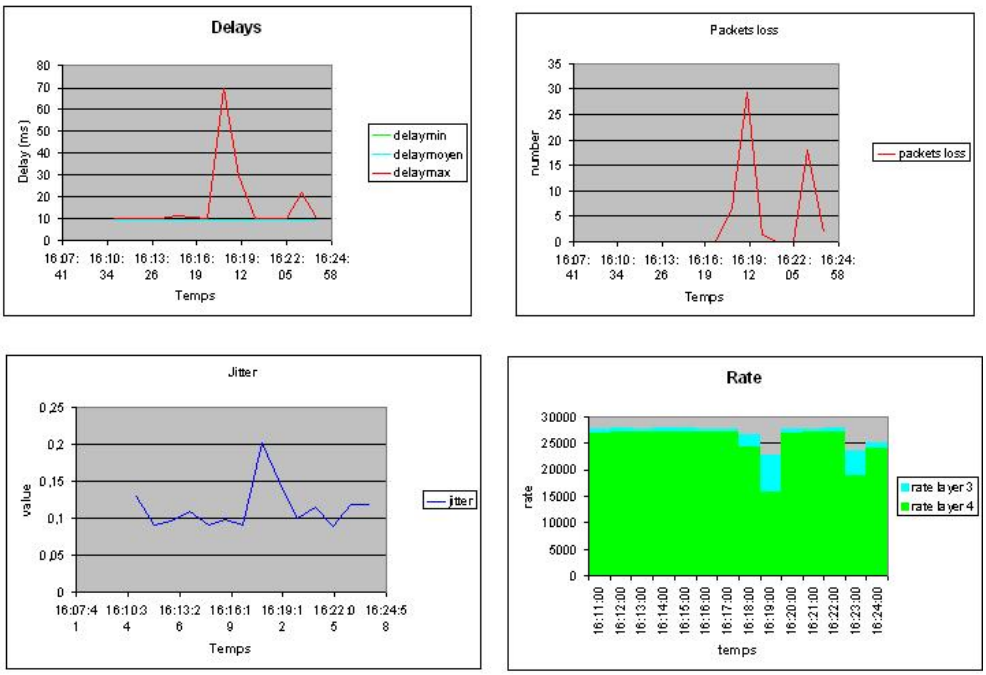


Figure 8: Data gathered during DV session

5.5 Large population

Globally, the traffic related to dummy clients does not disturb other multicast flows reception.

Referring to our test scenario, dummy participants scenarios involve:

- ◇ 200 dummy avatars generated from Lannion terminal and located in the departure cell, which simultaneously join active multicast group (225.1.5.12)
- ◇ 200 other dummy avatars generated from Sophia terminal and scattered everywhere inside graphic environment, which join random multicast groups.

On the one hand, Lannion dummies don't generate additional traffic (since they just join the same existing multicast group). On the other hand, Sophia dummies are physically scattered inside virtual world, which implies a great number of new receivers for new multicast groups. Network equipments inside VTHD must maintain the multicast routing table ((S,G) states). Thus, multicast traffic inside VTHD network is not directly proportional to the number of receivers.

Those QoS measurements shows a successful network delivery of multicast UDP packets. Furthermore, no packet loss could be detected for non-real traffic, such as text message or geographical flows.

This first multicast test campaign is globally successful since each site can now use V-EYE and communicate with other participants through VTHD IP network.. As a result, other VTHD-connected entities interested in this attractive experimentation have recently join us.

6 Comparison with other Virtual Environments

A large number of Virtual Environments (VE) applications have been designed so far. Here we mainly focus on VE applications that uses multicast communication. Several architectures such as NPSNET [16], DIVE [3], MASSIVE-2 [6], GREENSPACE [22] and SPLINE [2] have already been designed using multiple multicast groups. NPSNET [16] is the first Distributed Interactive Simulation (DIS) application to use IP multicast protocol. The NPSNET VE is partitioned into hexagonal cells which are associated with multicast groups. DIVE [3] uses a replicated world database and peer-to-peer communication. To update an object, a message is sent to all peers to update their own replicas accordingly. To reduce network traffic, the objects in the DIVE virtual world are hierarchically composed and can be associated with a set of hierarchical multicast groups. MASSIVE-2 is a collaborative virtual environment that allows users to interact with each other over a combination of graphics, audio and text media. It uses both a client/server architecture and multicast. The server provides the participants with an initial point of access to the VE and entities are replicated on demand and are associated with multicast groups. In GREENSPACE [22], each participant transmits its state using multicast and a lightweight server assigns multicast addresses and informs participants of changes in the VE. In SPLINE [2], the VE is divided into several zones (or *locales*), which have arbitrary size and shape and are associated with multicast groups. It uses a distributed architecture in which each node maintains a partial copy of the VE corresponding to the area of interest.

The department of Defense has been pursuing its own architecture, called HLA [4], for virtual environment interoperability which has been recently adopted by the IEEE. HLA filtering mechanisms are based on DIS experience with multicast and use the concept of

routing spaces. A routing space is made of *subscription* regions corresponding to member's expression of interest and *update* regions that express what a member is able to produce; regions are rectangle areas in the routing space. While the DIS protocol is closely linked with the properties of military units and vehicles, the rationale behind HLA is that it could be used with different types of simulations (not only military applications) and it is targeted towards new simulation developments.

None of these different works have presented an architecture to dynamically partition the VE into multicast groups, taking into account the density of participants per cell and the participants' capacities. V-EYE is one of the first application to implement such a scalable architecture for a large scale virtual environment.

7 Conclusion and Future Work

We described how the SCORE general communication framework can effectively help in implementing traffic segmentation, and thus achieving a very high level of scalability in a VLSE. We also outlined how such a rather volatile, location-dependant, addressing scheme, can be effectively coupled with SCORE-unaware applications, such as the DV subsystem, for a smooth integration of both approaches.

The implementations and experimentations described in this paper are all based on the ASM flavor of IP multicast. We are currently working on a SSM-based approach. SSM was initially introduced so as to ease multicast deployment at the network layer, shifting the burden onto the application, notably for sender sources management. Although the SSM model may at first glance look very more appropriate for streaming than for interactive communications, we believe it is possible to take advantage of the central role of the SCORE communication architecture and to build an SSM-based VLSE with a very affordable amount of changes at the application layer.

So far, we could not find a stable version of the Linux kernel that is compliant with IGMP-v3 and PIM-v2, while Windows xp claims to have this capability natively. This is why the V-EYE application is currently being partially ported on this SMM-capable platform.

Additionally, there being no evidence of IP multicast being one day available on the Internet in the large - even under its presumably more manageable SSM form - many approaches have been proposed recently in the so-called application-layer multicast arena. Those approaches generally rely on an underlying overlay network such as the ones introduces in peer-to-peer technologies, and we are considering to study the feasibility of a SCORE-like model in such a pseudo-multicast environment.

References

[1] *Advanced Linux Sound Architecture*. http://www.alsa-project.org/.

[2] J. W. Barrus, R. C. Waters, and D. B. Anderson. Locales: Supporting large multiuser virtual environments. *IEEE Computer Graphics and Applications*, pages 16(6):50–57, November 1996.

[3] C. Carlsson and O. Hagsand. Dive - a multi user virtual reality system. In *Proceedings IEEE VRAIS*, Seattle, Washington, September 1993.

[4] J. Dahman, J. R. Weatherly, and F. Kuhl. *Creating Computer Simulation Systems: An Introduction to The High Level Architecture*. Prenctice Hall, 1999.

[5] *DVTS, Digital Video Transport System*. http://www.sfc.wide.ad.jp/DVTS.

[6] C. Greenhalgh. Dynamic, embodied multicast groups in massive-2. Technical Report NOTTCS-TR-96-8 1, University of Nottingham, 1996.

[7] V. Hardman and M. Iken. Enhanced reality audio in interactive networked environments. In *Framework for Interactive Virtual Environments (FIVE)*, 1996.

[8] D. Kutsher J. Ott and C. Perkins. The message bus : A platform for component-based conferencing applications. In *The CSCW2000 workshop on Component-based Groupware*, 2000.

[9] K. Kobayashi, A. Ogawa, S. Casner, and C. Bormann. Rtp payload format for 12-bit dat audio and 20- and 24-bit linear sampled audio. *RFC-3190*, 2002.

[10] K. Kobayashi, A. Ogawa, S. Casner, and C. Bormann. Rtp payload format for dv (iec 61834) video. *RFC-3189*, 2002.

[11] B. N. Levine, J. Crowcroft, C. Diot, J. J. Garcia-Luna-Aceves, and J. F. Kurose. Consideration of receiver interest in content for ip delivery. In *Proceedings IEEE INFOCOM*, 2000.

[12] *Quasar DV Codec: libdv*. http://libdv.sourceforge.net/.

[13] E. Léty, L. Gautier, and C. Diot. Mimaze, a 3d multi-player game on the internet. In *Proceedings 4th International Conference on VSMM (Virtual Systems and MultiMedia)*, Gifu, Japan, November 1998.

[14] E. Léty and T. Turletti. Issues in designing a communication architecture for large-scale virtual environments. In *Proc. of the 1st International Workshop on Networked Group Communication*, Pisa, Italy, 17-19 November 1999.

[15] Emmanuel Léty, Thierry Turletti, and François Baccelli. Score : a scalable communication protocol for large-scale virtual environments. *to appear in IEEE/ACM Transactions on Networking journal*, June 2004.

[16] Michael R. Macedonia, Donald P. Brutzman, Michael Zyda, David R. Pratt, Paul T. Barham, John Falby, and John Locke. NPSNET: A multi-player 3d virtual environment over the internet. In *Symposium on Interactive 3D Graphics*, pages 93–94, 210, 1995.

[17] Steven McCanne and Van Jacobson. vic : A flexible framework for packet video. In *ACM Multimedia*, pages 511–522, 1995.

[18] *Multicast Traffic Monitoring Tool.* http://www.merci.crc.ca/mbone/MultiMON/.

[19] Akimichi Ogawa, Katsushi Kobayashi, Kazunori Sugiura, Osamu Nakamura, and Jun Murai. Design and implementation of dv based video over rtp. In *Packet Video Workshop*, 2000.

[20] *Linux support for Philips USB webcams.* http://www.smcc.demon.nl/webcam/.

[21] Henning Schulzrinne. *RTP tools.* http://www.cs.columbia.edu/IRT/software/rtptools.

[22] P. Schwartz, B. Campbell, S. Tanney, S. Yen, and T. Shen. Virtual playground: Architectures for a shared virtual world. *ACM Symposium on Virtual Reality Software and Technology*, November 1998.

[23] *Le projet VTHD++.* http://www.vthd.org.

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-0803