A Scalable SSM-based Multicast Communication Layer for Multimedia Networked Virtual Environments

Thierry Parmentelat, Laurentiu Barza, Thierry Turletti, Walid Dabbous

To cite this version:

Thierry Parmentelat, Laurentiu Barza, Thierry Turletti, Walid Dabbous. A Scalable SSM-based Multicast Communication Layer for Multimedia Networked Virtual Environments. [Research Report] RR-5389, INRIA. 2004, pp.18. inria-00070614

HAL Id: inria-00070614
https://hal.inria.fr/inria-00070614

Submitted on 19 May 2006

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

A Scalable SSM-based Multicast Communication Layer for Multimedia Networked Virtual Environments

Thierry Parmentelat — Laurentiu Barza — Thierry Turletti — Walid Dabbous
INRIA Sophia Antipolis

Surname.Name@sophia.inria.fr

N° 5389

November 2004

Thème COM

Rapport de recherche

ISSN 0249-6399 ISRN INRIA/RR--5389--FR+ENG

A Scalable *SSM*-based Multicast Communication Layer for Multimedia Networked Virtual Environments

Thierry Parmentelat , Laurentiu Barza , Thierry Turletti , Walid Dabbous
INRIA Sophia Antipolis
`Surname.Name@sophia.inria.fr`

Thème COM — Systèmes communicants
Projet Planète

Rapport de recherche n° 5389 — November 2004 — 18 pages

Abstract: This paper describes a communication layer which is suitable for real-time multimedia virtual environments (VEs) that need to achieve high scalability such as computer games involving multimedia exchanges. This approach is based on the Single Source Multicast (*SSM*) model as the underlying network infrastructure.

Though the *SSM* model might, at first glance, look more appropriate for streaming-like applications, we show that it is possible to take advantage of such a communication layer to solve the new issues that the application layer has to address in the *SSM* environment. We even outline that *SSM* has benefits over Any Source Multicast (*ASM*), in that it provides greater flexibility in flow management.

As opposed to alternative techniques such as peer-to-peer overlays, the use of network-layer multicast performs very well in terms of latency. Our demonstration application embeds multimedia conferencing capabilities to demonstrate that real-time and massively multi-player applications can be successfully implemented on top of this communication layer.

Key-words: virtual environment, scalability, interactive multimedia, conferencing, SSM, multicast, latency, peer-to-peer, adaptive filtering

Une architecture de communication basée sur SSM-multicast pour les environnements virtuels multimédia à très grande échelle

Résumé : Ce papier décrit une couche de communication adaptée aux environnements virtuels incluant des échanges multimédia et aux contraintes temps-réel qui en résultent, tout en permettant de passer à une échelle arbitrairement grande. Cette approche utilise le modèle "Single Source Multicast" (*SSM*) comme infrastucture de communication réseau.

Alors que modèle *SSM* peut, à première vue, sembler plus approprié aux applications non-interactives comme la diffusion en ligne, nous montrons qu'il est possible d'en tirer profit d'une telle couche de communication pour résoudre les nouveaux problèmes qui se posent à l'application dans le monde *SSM*. Nous établissons même que *SSM* a des avantages par rapport au modèle traditionnel "Any Source Multicast" (*ASM*) en ceci qu'il permet davantage de finesse dans la sélection des flux.

Contrairement aux techniques alternatives que pourraient constituer les réseaux pair à pair, le multicast natif réseau se comporte très bien en termes de délais de transmission. Notre application de démonstration contient des fonctionnalités de conférence audio et vidéo, de manière à démontrer que des applications multimédia massivements multi-joueurs peuvent être implémentées avec succès en utilisant notre couche de communication.

Mots-clés : environnement virtuel, passage à l'échelle, multimédia interactif, conférence, SSM, multicast, délai, pair-à-pair, filtrage adaptatif

1 Introduction

Networked Virtual Environments have become very popular over the last few years. Numerous applications of this concept can be found, in particular in the online game industry, and in military simulation tools[11].

With the raise of capabilities of the average home computer, these applications tend to provide multimedia functions as an inter-player communication medium. On the other hand, this implies additional complexity in the communication module of the VEs.

Networked Virtual Environments have the following communication requirements:

- ◇ **Scalability.** In multiparty interactive gaming, many users communicate and react in real-time, evolving in a very large world; we want this world to be able to accommodate for a very large number of players. In addition, actual multimedia communication is considered to take place among many small groups of users.
- ◇ **Heterogeneity.** As an additional constraint, it is very likely that a large set of users will exhibit quite various capabilities in terms of both computing power and network bandwidth; the communication model must thus provide with mechanisms (i) for fine-grain filtering of incoming flows, and (ii) for adaptation to each player's capabilities.
- ◇ **Low delays.** By nature, multimedia flows such as audio and video streams have real-time requirements. These constraints are especially hard when the interactive dimension is present.

The communication infrastructures underlying networked virtual environments range in a wide spectrum, from a pure traditional client-server approach, passing through multicast-based approaches, up to more modern approaches that rely on peer-to-peer overlays.

The traditional client-server approach generally exhibits a limited level of scalability. Of course the server-side infrastructure can take advantage of clusters or farms. In the case of latency-demanding applications, like games in the "Quake" style, known as First Person Shooter, it is very common to see the server infrastructure organized into subgroups, each handling a specific game area. In any case, whatever the adopted optimizations, the total amount of resources required on the server side still grows linearly with the number of players.

A more recent approach involves P2P overlays[7], and takes advantage of all the connected computing capabilities to achieve scalability in a very elegant way. However, such a scheme is still a very advanced research topic, and many issues need to be addressed before an operational implementation can be envisioned. In addition, as far as multimedia traffic is considered, the major concern is related to the overall delay that a media traffic is likely to be subject to, for three major reasons :

- ◇ First the overlay arrangement is computed at a purely application layer, that has no or limited means to infer the actual underlying network topology.
- ◇ Second, the routing functions are implemented in clients that are located at the edge of the network. For these two reasons, the resulting flow is likely to involve a total

number of routers that greatly exceeds the optimal route that a shortest path routing protocol would produce.

- ◇ Third, the client performs the routing function at the application layer, which requires data packets to go through the whole network stack before being forwarded to the next hop.

For these reasons, it is tempting to consider an intermediate approach between these two extrema, i.e. client-server and P2P. Since the network-based multicast protocols were mostly motivated by multimedia traffic in the first place, it is tempting to take them into account when designing a communication layer.

As described into further details in section 2, a number of works have studied the communication layers for large-scale virtual environments based on multicast as the underlying network infrastructure. Multicast addresses the real-time requirements of the VE, thanks to its ability to build shortest-paths trees from the source to the receivers.

Historically, the first multicast model introduced in the Internet was *ASM* (Any Source Multicast), that is available on most academic networks, and that is also used internally on some commercial networks. However, *ASM* has the following drawbacks in our perspective :

- ◇ There is in general no multicast connectivity between commercial networks, given the complex stack of protocols that need to be deployed, and for this reason *ASM* cannot be relied on for building an Internet-wide solution.
- ◇ Though *ASM* looked a rather appropriate network-layer solution for LSVE communication, this model exhibits scalability problems, because of the reduced address space, thus limiting the maximum number of multicast groups an application can use.
- ◇ In addition, *ASM* is not suitable for fine-grain filtering : there is no access control for the senders or the receivers of a multicast group.

The IP multicast community is currently rather in the favor of the *SSM* model with the hope to make network administration easier. In this paper, we describe a communication architecture built on top of *SSM*, named SCORE-*SSM*.

The benefits of our approach can be summarized as follows :

- *SSM* is seldom considered for this kind of application, mostly because as compared with the former *ASM* model, the application now needs to handle the IP-address of its connected parties, and this is an extra burden that the application needs to cope with. In our approach, the communication layer takes care of the issue, and transparently provides the application with the IP-addresses of the players in its vicinity, modulo a locally tunable 'area of interest'.

- In addition, the application can now implement very fine-grain filtering of its incoming flows. In some situations it may even become very confortable for the application to take advantage of groups that are *not only* based on locality. This may be the case if you need to implement communications with a remote group, e.g. in military environments. This may also be the case in very crowded worlds, where you want to focus on someone's speech among several other discussions. Such non-local groups are perfectly feasible within the SCORE-*SSM* approach.

This paper is structured as follows. Section 2 presents the state of the art in this area. Section 3 provides an overview of SCORE-*SSM*, while section 5 presents comparative experimental results.

2 Related Work

From the network point of view, the *SSM* model is situated between the *ASM* model and the point-to-point communication model (i.e. one-to-many communication is situated between many-to-many and one-to-one communications). An insight comparison between *ASM* and *SSM* at the protocol level can be found in [2]. The challenges raised for a smooth integration of *ASM* and *SSM* are studied in [1].

Worth being mentioned too, a related work [15] has proposed the use of proxies in order to provide an *ASM* service model on top of *SSM*. However, this model is fitted for applications which do not have high performance requirements, and would not fit our strong real-time constraints, because it introduces extra complexity at the network layer. We also think that backward compatibility with *ASM* applications is not required. In addition this model would only allow for rough data-filtering, in much the same way as *ASM* did.

A large number of Virtual Environments (VE) applications have been designed so far. Here we focus on VE applications that use multicast communication. Several architectures such as NPSNET [10], DIVE [4], MASSIVE-2 [6], GREENSPACE [13] and SPLINE [3] have been designed using multiple multicast groups. NPSNET-IV was the first VE application to use the IP multicast. The VE is statically divided into hexagonal regions and a group leader maintains a list of all the entities within the region. Each avatar sends data to the multicast group associated with the zone that covers its current location. To receive data, the avatar subscribes to the multicast groups associated with the zones overlapping its area of interest. SPLINE uses "beacon servers" to learn about the avatars situated nearby in the VE. Each beacon server has a multicast group associated in order to receive data about the avatars located in its corresponding region. MASSIVE-2 implements a self-configuring hierarchy of multicast groups with aggregated data transmissions, from children to parents, in the hierarchy. In DIVE, the objects of the VE compose a hierarchy. A set of hierarchical multicast groups is associated to this structure. In GREENSPACE, each avatar transmits its state using multicast and a lightweight server assigns multicast addresses and informs avatars of changes in the VE. The department of Defense has been pursuing its own architecture, called HLA [5], for interoperable virtual environment, which has been adopted by the IEEE. It specifies functional requirements, but not the hardware, software or the network architecture. Furthermore, HLA defines a value-based filtering which use the concept of routing spaces. None of these different works have presented a mechanism to dynamically partition the VE into multiple multicast groups, taking into account the density of avatars per cell and capacities of avatars.

3 SCORE-SSM

In this work, we focus on games where the avatars only communicate with their closest neighbors (in terms of Euclidian distance in the virtual world), the communication held in other parts of the VE represent unwanted (or superfluous) traffic. It has been shown in [8] that in a group communication setting, the percentage of superfluous information received per avatar (or player) increases with the number of data flows and the number of players. Indeed, within a VE, each player simultaneously interacts with only a small subset of players. 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 this section we present SCORE-*SSM*, a new communication layer based on SSM which allows a large number of heterogeneous avatars to be simultaneously connected to the same virtual environment. We have used this communication layer to develop a demonstration application (named V-Eye, see a snapshot in figure 1) that implements a Large Scale Virtual Environment, combining a 3D world, textual messages, audio and video communications and High Definition video. This application is extensively described in [12] and turned out to be a very helpful tool for validating and experimenting our communication layer.

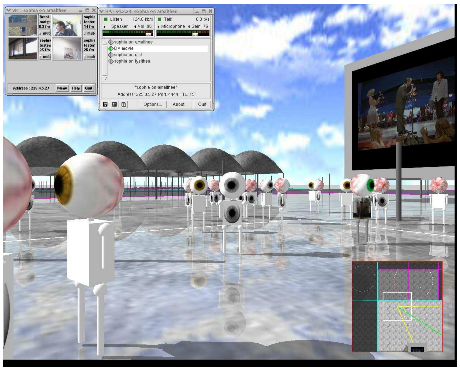

Figure 1: A sample snapshot of the LSVE application

3.1 Responding to the communication requirements of LSVEs

In order to fulfill the communication requirements of the LSVEs, the communication layer has to be scalable, to support heterogeneous users and to respond at the multiparty requirements.

Scalability and heterogeneity requirements are dealt through filtering and adaptation: SCORE-*SSM* implements a transport-layer filtering mechanism with multiple agents allowing to filter out superfluous traffic before it reaches the end-hosts. This approach involves the dynamic partitioning of the VE into spatial areas called *zones* and the association of these zones with multicast groups. The basic idea is to dynamically partition the VE into zones of different sizes, depending on the local density of avatars in the VE, the number of active agents, and the bandwidth and processing resources of the players. In order to support a very large number of avatars simultaneously connected in the VE, we use a hierarchy of agents which effectuate a hierarchical partitioning of the VE into areas, upon the avatars density.

In order to fulfill the multiparty requirement, SCORE-*SSM* has to provide, for each avatar, the IP addresses of its neighbors within the VE. From an application perspective, the major difference with the *ASM* model is that it requires the application to provide the IP source address when joining a given channel. Though this has great advantages at the network layer, it clearly adds a burden on the application, and for this reason it is commonly believed that the *SSM* model is more appropriate for contexts with only a few sources, such as multimedia streaming broadcasting.

In an SSM-based environment, the mapping information between a zone and a group is not sufficient to communicate with other players. In fact, both the group and the identification of all sources (i.e., the IP addresses of players) within a zone are required. So, the SCORE-*SSM* agent layer has to provide (for each avatar connected in the VE) the IP addresses of its neighbors. Using these IP addresses, an avatar is able to subscribe to the multicast channels corresponding to these sources. To provide the neighborhood information, we have extended this IP address discovery mechanism to simultaneously transmit an approximate position of the neighbors in the VE. This additional information allows an avatar to perform a better filtering and to receive only data flows from the neighbors situated in its area of interest. In this case, the filtering mechanism ensures that no superfluous data is received by the avatars.

In a nutshell, the SSM-based communication protocol exchanges more signalling data (i.e. the avatars' IP addresses, VE coordinates) as compared with the *ASM* case, with the benefit of no superfluous data received by the avatars and the precise identification of the data flows (a different multicast channel is used for to every data flow).

3.2 Filtering mechanism

SCORE-*SSM* implements a two-level filtering mechanism. First, the VE is broken up into non-overlapping zones, according to the position and the density of the avatars in the VE. An avatar receives the approximate positions of the other avatars situated in the VE zones intersected by its area of interest. Second, it computes its "closest" neighbors among these neighbor avatars in order to communicate with them.

The first level of filtering of SCORE-*SSM* (we call it *"agent filtering"*) is composed of an agent-based signalling overlay. The *agents* are processes that run at different parts of the network. To improve the scalability, the VE is split into zones as shown in Figure 2; an agent is responsible for the signalling in one area corresponding to one or more zones of the VE.

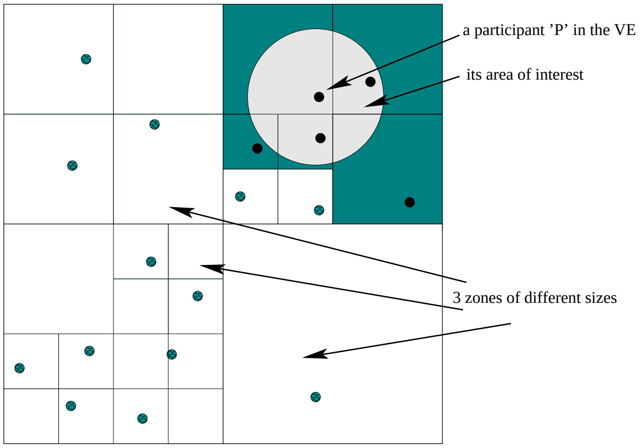

Figure 2: a VE area partitioned in zones, in SCORE-*SSM*

The size of an area is determined by the maximum number of avatars that each agent can manage simultaneously. So, a VE uses several agents which communicate together using a signalling layer. By signalling, we refer to all the network traffic required to set-up the communication between the avatars. In addition, the agents decide how to split the VE into zones: they take into account the avatars distribution in the VE, and dynamically perform a dynamic partitioning of the world into zones. If the number of avatars in the VE is very large, the inter-agent communication may become significant. In order to avoid scalability problems, the agents are hierarchically organized.

The second level of filtering (we call it *"avatar filtering"*) is done using the SSM data distribution model. Once an avatar has identified its closest neighbors, it can subscribe to their corresponding data flows. The use of *SSM* allows the avatars to communicate only with the "closest" (in terms of Euclidian distance in the VE) neighbors, without receiving unwanted data flows from the "distant" avatars. We can observe that an *SSM* channel corresponds to a specific data flow sent by a specific avatar. The distinction between the different data flows transmitted per player, can significantly improve the filtering mechanism (for example, an avatar may specify precisely what audio flows and what video flows it wants to receive at a given moment). Of course, this comes at the price of more complex filtering mechanisms that should be implemented by the communication protocol in order to precisely detect the neighbors (and the data flow types they are sending) with which an avatar exchanges information.

To recapitulate, prior to receiving any data, an avatar has first to identify the set of avatars located in its current zone. This is the role of the agent filtering mechanism. Then, the avatar chooses among the received set of neighboring avatars, a subset (the avatars located in its area of interest) with which it receives data. For every avatar in this subset, it subscribes to a number of channels (each flow is sent on a different multicast channel), according to its current interest.

3.3 Agents responsibility

The agents fulfil the following roles in the SCORE-*SSM* protocol:

- Agents may split their corresponding area into non-overlapping zones. Let S_{AG} be IP address of the agent AG and G_Z the IP multicast group address of zone Z. The zone Z will use the multicast signalling channel (S_{AG}, G_Z). If the number of avatars in a zone exceeds a given threshold[1], the agent divides this zone in smaller zones, according to the zone-partitioning strategy. In SCORE-*SSM*, the zone-partitioning algorithm divides a zone into four smaller, equal-sized zones. Note that there is a tradeoff between the number of zones created and their shapes. For example, if we divide a zone into 2 zones, this may lead to complex shapes after a number of successive division/unification operations. Moreover, if a zone has a complex shape, it may require larger signalling packets, to transmit its geometry characteristics to the avatars.
- Each agent communicates with all the avatars located in the area of the VE it manages. This is a two-way communication : 1/ the agent receives a low-rate periodical position updates flow from all its avatars (avatars use point-to-point UDP connections), 2/ the agent sends (using an SSM multicast signalling channel for every zone it manages) the approximate position of each avatar located in that zone. In addition, for each zone, it transmits the multicast channel addresses of the neighboring zones so as to allow the players to subscribe to these channels when approaching a new zone. To allow handover between different areas (see next item), the agent also transmits in boundary zones the IP address of neighboring agents.
- While moving through the VE, an avatar may enter in a VE area managed by another agent. Before entering this new area, the avatar has to retrieve the multicast signalling channel corresponding to the zone to which it is heading to. To solve these "handover" situations, each agent uses a "rendez-vous" multicast signalling channel $(S_{AG}, G_{rendez-vous})$ to periodically transmit the multicast channel addresses of the bordering zones (the zones located at the border of the area managed by the agent).

3.4 Information sent by avatars

When an avatar enters the VE, it first listens to the "start-up" multicast channel (whose address is hard-coded in the application) to get the address of an agent. Then, it contacts this agent to find out the signalling channel address corresponding to its position in the VE.

Each avatar communicates with the agent which manages its current VE area. The avatar sends a low rate flow position to keep the agent informed about its movement in the VE. As this update is done only once per second, and the avatars are moving with different

---

[1] This threshold is set by agents taking into account the average capacity of avatars.

speeds, the agents only maintain their approximate positions. In return, each avatar receives through multicast signalling channels the approximate positions of the avatars situated in the VE's zones intersected by their area of interest.

The different flows sent by an avatar (audio, video, ...) are sent on different multicast channels. For each type of data flow, a unique *SSM* multicast group address is used by all the avatars connected in the VE : "G1" for position flows, "G2" for audio flows, "G3" for video flows, etc.. This reduces the signalling overhead (as the different multicast group addresses are already known) and furthermore allows the aggregation of multicast forwarding states in intermediate routers.

Furthermore, each avatar has a mechanism to express its current area of interest taking into account different metrics such as: available capacity, interest in particular data flow and distance to the source. This mechanism is used to update the set of multicast channels an avatar subscribes to during the session.

4 Experimental results

In order to evaluate the performance of SCORE-*SSM* we have conducted a number of experiments. In these experiments, we compare the communication traffic of a LSVE using SCORE-*SSM* with the communication traffic of a LSVE deployed on top of SCORE-*ASM*. SCORE-*ASM* was built using the ASM model and we briefly present it before detailing our experiments.

4.1 A brief review of SCORE-ASM

Before detailing the experiments, we outline the main characteristics of SCORE-*ASM*.

SCORE-*ASM* implements a transport-layer filtering mechanism with multiple agents allowing to filter out the 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 avatars per cell, the number of available multicast groups, and the bandwidth and processing resources of the players.

4.1.1 Player satisfaction metric

Players have limited network and CPU processing cycles resources. For that reason, a player satisfaction metric is used so as to take into account the proportion of relevant data rate it is able to process. Here, the goal is not to adapt to the receiver with the lowest capacities, but to maximize the satisfaction of the receiver with the lowest player satisfaction value. For instance, the presence of a player using a PDA will be an incentive for SCORE-*ASM* to define small-size cells in its vicinity, so as to allow him to reduce its total incoming traffic by choosing a rather small area of interest.

4.1.2 Agents responsibility

Like in SCORE-*SSM*, in SCORE-*ASM agents* are processes that run at different parts of the network. They are not servers, in the sense that they do not aim at processing any global state for the VE, and they do not receive the data traffic sent between players.

In fact, the agents dynamically determine zones with the VE from the distribution of avatars and their satisfaction metrics, and maintain cell-sizes in each zone accordingly. This is how SCORE-*ASM* implements a dynamic partitioning of the VE into cells of different sizes, and the association of these cells with multicast groups.

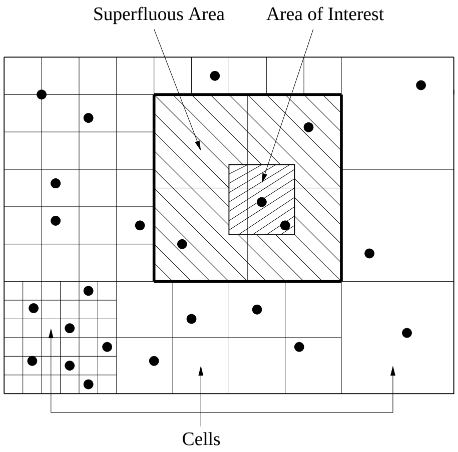

Figure 3: Partitioning with different cell-sizes

4.1.3 Players-to-Agent traffic

There are several levels of communication in SCORE-*ASM* :

- ◇ Each player subscribes to one or more multicast groups but sends data packets on a single group.
- ◇ Each player is connected to a single agent, using a point-to-point UDP connection.
- ◇ Agents communicate with each other on a single multicast group.

A player has to subscribe to two sorts of multicast groups : *control groups* are dedicated to signalling and *data groups* hold the actual application payload. For each of these groups, the avatar has to make early *join*s so as to compensate for the join-latency value, according to its own speed.

Each player is connected to its nearest agent using a UDP connection. Therefore, each agent is able to track the location of its connected players in the VE.

4.2 Detailed description of the experiments

We have compared experimental performance results obtained with SCORE-*SSM* with the ones obtained for SCORE-*ASM* [9]. Our testbed is composed by 5 NetBSD machines connected on a 100Mbps Ethernet network. A total number of 1000 avatars have been launched on these machines to generate data traffic, and one agent to handle the signalling traffic. The dimension of the VE is (1200×1200 $units^2$) and the number of available multicast groups in SCORE-*ASM* case is 144. Furthermore, to analyze the different experimentations, the following parameters have been used:

- ◇ *Partitioning policy*, either static (agent-less) or dynamic, in which case we considered a re-mapping period (RP) of 1s. This stands for the period between two different re-mappings decided by agents.
- ◇ *Area of interest* (I) expressed according to the cell area of 100×100 $units^2$
- ◇ *Avatars velocity* (V) in the VE (in units per second). We have compared two cases: $V = 10$ units/s and $V = 100$ units/s,
- ◇ *Distribution of avatars' capacity* (C): capacities are randomly selected using a uniform distribution on either the interval $[20, 40]$ or the interval $[10, 50]$ sources/s.

While moving in the VE, the avatars choose a destination, with a given probability. This probability has been chosen such that it creates "hot-spots" in the VE, i.e. places in the VE with high avatar density. When they reach their destination, the avatars compute a new destination with the same algorithm.

In the following paragraphs we analyse the curves obtained in our experiments and compare them with the curves obtained for SCORE-*ASM*.

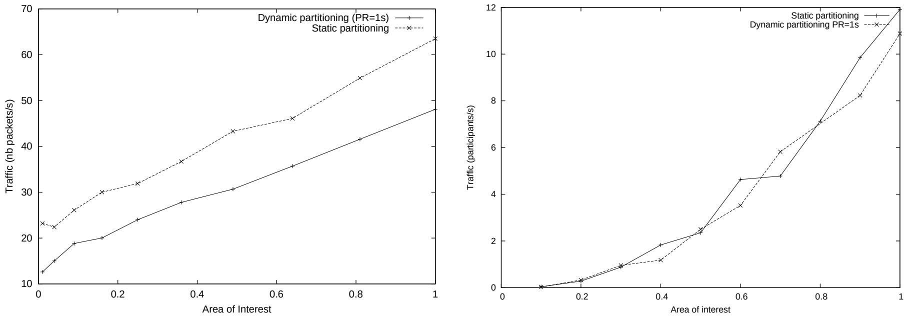

Figure 4: Received data traffic per avatar in SCORE-*ASM* (left) and SCORE-*SSM* (right)

4.2.1 Mean traffic received per avatar

In Figure 4, we compare the average incoming data rate per avatar (in sources/s) for a static partitioning scheme and for a dynamic partitioning scheme (using $RP = 1s$). This traffic is used to compute the satisfaction of the avatars. With Score-*ASM*, we observe a significant gap between the two curves: dynamic partitioning reduces the total traffic received by the avatars. This result was already established in [9], it basically means that dynamic partitioning allows to reduce superfluous traffic, i.e. traffic received from the selected multicast groups but outside the area of interest.

On the right side of figure 4, we observe that the amount of data received is similar, regardless of the VE partitioning. This shows that the Score-*SSM* filtering mechanism performs equally well in static and dynamic partitioning policies. If we compare the curves for the dynamic partitioning case in the two graphs, we observe that in Score-*ASM*, the avatars receive about 10 times more traffic for small areas of interest (I=20) and about 4 times more traffic for large areas of interest (I=100) than in Score-*SSM*. In Score-*SSM*, the total data traffic received by the avatars is generated by the players situated in their area of interest. In Score-*ASM*, the total data traffic received per player contains additionally some superfluous traffic. So, all the extra traffic received by a player which uses Score-*ASM*is generated by avatars located outside its area of interest (i.e. it is superfluous traffic).

4.2.2 Avatars distribution in multicast groups

In the ASM model, the address space of the multicast addresses is limited, so, it is crucial to efficiently use these groups. The repartition of the avatars within the multicast groups represents an indicator of the partitioning algorithm efficiency.

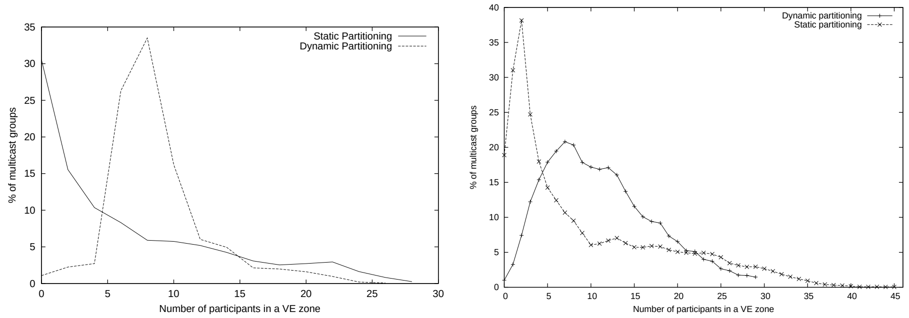

Figure 5: Avatars distribution in multicast groups with Score-*ASM* (left) and Score-*SSM* (right)

On figure 5, for a given number N of avatars on the abscissa, we can observe for each curve the percentage of multicast groups containing between N and $(N-2)$ avatars. In Score-*ASM*, the curves corresponding to the dynamic partitioning of the VE present a peak value around 7 avatars in a zone (i.e. on a multicast group), regardless of the size of the area of interest. This value is close to the ratio between the number of avatars connected (1000) and the number of available multicast groups (144). The average number of avatars per multicast group is equal to $1000/144 = 6.9$. This shows the ability of Score-*ASM* to deal with non-uniform distributions and with high-speed avatars. In the static case and on all the figures, almost 30% of the groups contain between 0 and 2 avatars. This means that a large number of multicast groups are under-exploited while these groups are a very valuable resource for the data filtering. In addition, the number of groups containing a large number of avatars is more important in the static case than in the dynamic case.

In both multicast models, when a player joins or leaves a multicast group, it uses the IGMP group membership protocol to communicate with its last hop router. As the IP router does not track the number of hosts participating in a multicast group, it must poll the hosts to determine if any of them is still active, before stopping the packets flow. So, when a player leaves, the packets are still arriving up to the host during a couple of seconds [14]. However, "leave messages" sent by the host prevent the application of receiving these packets. On the other hand, the "join" latency is fast, and takes about one RTT between the host and the first router towards the source which is not forwarding the group.

4.2.3 Subscription frequency to multicast groups

In figure 6, we compare the number of join messages per second, for Score-*ASM* on the left part of the figure and for Score-*SSM* on the right part of the figure. The experiment is done for 10 different areas of interest, for static and dynamic partitioning modes and for different avatars' speeds (10 and 100 unit/s). We observe that the avatars' speed has a direct impact over the subscription frequency, for both communication layers. In Score-*ASM*, when V=100 units/s, the join frequency is twice larger in the dynamic partitioning case than in the static partitioning case. In Score-*SSM*, we observe that the number of joins is about 6-7 times larger when the avatars are moving with a speed of 100 unit/s, than with a speed of 10 unit/s, regardless the area of interest diameter.

Comparing Score-*ASM* and Score-*SSM* curves, we note that for small area of interest diameters (I<40 units), there are three times less "join" messages in the static partitioning case and four times less "join" messages in the dynamic partitioning case. For larger area of interest diameters, (I>80 units), there are two times fewer "join" messages in the static partitioning case and three times fewer "join" messages in the dynamic partitioning case. The above differences are similar both for V=10 units/s and for V=100 units/s. The explanation of the above result is that Score-*ASM* sends more "join" messages to the agents than Score-*ASM*, because the VE is cut into a larger number of smaller zones.

4.2.4 Signalling traffic received per avatar

In Figure 7, we compare the received signalling traffic per avatar for the two communication layers. The size of the signalling packets are respectively 8 bytes for Score-*ASM* and 16 bytes in Score-*SSM*, while the UDP/IP header is 24 bytes long. Given these figures we chose to implement aggregation, i.e. to send the signalling packets representing all the avatars from a zone using a unique UDP/IP packet. All our experiments have shown

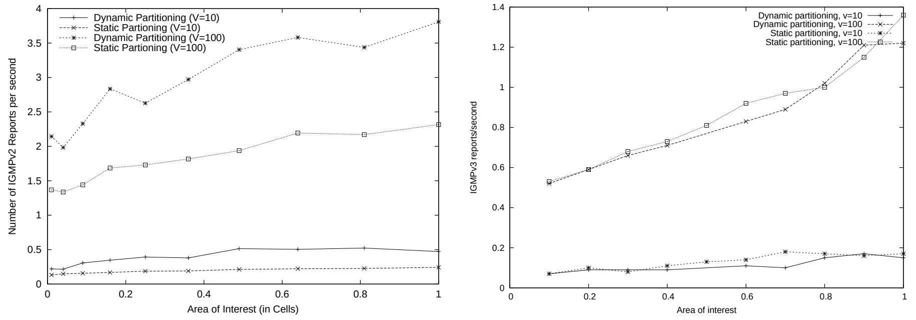

Figure 6: Comparison of subscription frequency in SCORE-*ASM* (left) and SCORE-*SSM* (right)

that when the agents partition the VE with a maximum of 30 avatars/zone, we obtain an average of 12 avatars per zone. So, the SCORE-*SSM* signalling packets average size is equal to 16*12+24 = 216 bytes, which corresponds approximatively to 7 SCORE-*ASM* signalling packets. For SCORE-*ASM*, we observe that when $RP = 1s$, the signalling is about 4 times larger than when $RP = 6s$. In addition, the signalling traffic increases very slowly with the size of the area of interest. In SCORE-*SSM*, the signalling traffic linearly increases with respect to the area of interest. If we compare the number of signalling packets received by the avatars we observe that in SCORE-*SSM* this number is much more important than in the SCORE-*ASM* case. The aggregate of the signalling information received for each zone, every second, significantly reduces the signalling traffic rate in the SCORE-*SSM* case. For example, when I=100, PR=1s (i.e., when the signalling traffic reaches it maximum in SCORE-*ASM*) we observe 1.5 packets/s for SCORE-*ASM* and 3.5 packets/s for SCORE-*SSM*. As the size of 3.5 SCORE-*SSM* packets is approximately equal to 7*3.5= 24.5 SCORE-*ASM* packets, this means that 16,3 times less bandwidth is used.

4.3 Synthesis of performance results

In a first experiment, we have examined the mean traffic received by the avatars. SCORE-*SSM* leads to a better use of network bandwidth because the avatars do not receive any superfluous data. In SCORE-*ASM*, a large part of packets received per avatar come from undesired sources.

In a second experiment, we have studied the avatar distribution within the multicast groups assigned to the zones composing the VE. This represents an indicator of the partitioning algorithm efficiency. The experiments prove that in both cases the dynamic partitioning of the VE leads to a better use of the multicast groups.

In a third experiment we have examined the avatar's subscription frequency to the zones of the VE. It is shown that regardless the partitioning mode of the VE and the avatars'

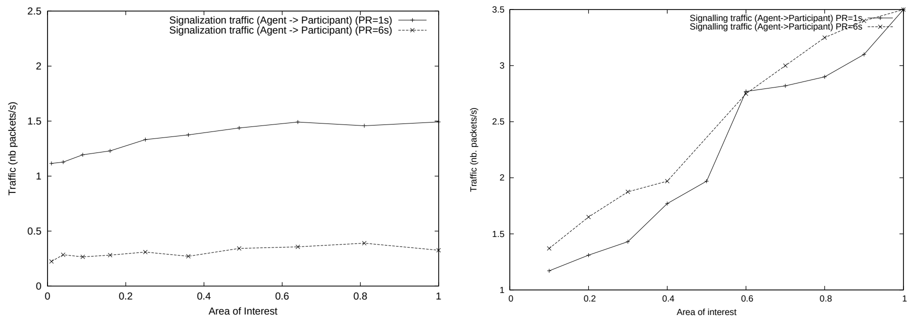

Figure 7: Signalling traffic received in SCORE-*ASM* (left) and SCORE-*SSM* (right)

speed, the avatars cross less zones while moving through the VE using SCORE-*SSM*. The zones are larger in SCORE-*SSM* compared to SCORE-*ASM* which emphasizes a better use of the multicast groups used for the grid-based filtering.

In the fourth experiment we have compared the signalling traffic received per avatar. This signalling is significantly greater in SCORE-*SSM* case compared to SCORE-*ASM*. In SCORE-*SSM*, the agents have to provide to the avatars information about their neighbors, thus the packets size is bigger and their frequency is higher compared to SCORE-*ASM* case.

The bottom line of these experiments can be stated as follows: SCORE-*ASM* generates less signalling traffic than SCORE-*SSM*, but, on the other hand, the avatars receive superfluous data which may become important in some cases. In SCORE-*SSM*, avatars do not receive any superfluous data traffic and they can decide with a finer granularity which data flows to subscribe. Although SCORE-*SSM*generates more signalling traffic than SCORE-*ASM*, the amount of extra signalling traffic is negligible when high data traffic rate such as audio/video is considered.

5 Conclusion

In this paper, we have described a scalable communication layer for multimedia VE applications that run on an *SSM*-based network infrastructure. We have compared the performance of this protocol with SCORE-*ASM* using experimentations. Although it is widely accepted that the *ASM* model is the best approach for many-to-many applications, our results show that for delay-sensitive flows, such as interactive audio and video flows, an *SSM*-based communication layer can achieve better performance than an ASM-based one at the cost of a marginal overhead in signalling, and furthermore can provide additional benefits.

References

[1] K. C. Almeroth, S. Bhattacharyya, and C. Diot. Challenges of integrating asm and ssm ip multicast protocol architectures. In *IWDC 2001*, Taormina, Italy, September 2001.

[2] Hitoshi Asaeda. Protocol analysis of any-source multicast and source-specific multicast. Technical Report RR-5080, INRIA Research Report, 2004.

[3] 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.

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

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

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

[7] Björn Knutsson, Honghui Lu, Wei Xu, and Bryan Hopkins. Peer-to-peer support for massively multiplayer games. In *Proceedings IEEE INFOCOM*, 2004.

[8] 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.

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

[10] 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.

[11] D. McGregor, A. Kapolka, Michael Zyda, and D. Brutzman. Requirements for large-scale networked virtual environments. In *Contel*, Zagreb, Croatia, 2003.

[12] Thierry Parmentelat, Alexis Gourdon, Thierry Turletti, and Elodie Larreur. V-eye, a very large scale virtual environment for multimedia conferencing. Technical Report RT-0296, INRIA, May 2004.

[13] 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.

[14] L Vicisano, L. Rizzo, and J.Crowcroft. Tcp-like congestion control for layered multicast data transfer. In *Proceedings IEEE INFOCOM*, San Francisco, CA, March 1998.

[15] Daniel Zappala and Aaron Fabbri. Using SSM Proxies to Provide Efficient Multiple-Source Multicast Delivery. In *IEEE Globecom, Sixth Global Internet Symposium*, November 2001.

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 0249-6399