Paper title: Multicast Communication for Scalable Distributed Virtual Environments

Authors: Renata Cruz Teixeira
Otto Carlos Muniz Bandeira Duarte E-mail: otto@gta.ufrj.br

Affiliation: Universidade Federal do Rio de Janeiro

Address: Grupo de Teleinformática e Automação – GTA
COPPE - Programa de Engenharia Elétrica
Universidade Federal do Rio de Janeiro
P. O. Box 68504 - 21945-970 - Rio de Janeiro - RJ - Brasil

Phone number: +55 (21) 260-5010 ext. 240

Fax number: +55 (21) 290-6626

Abstract This paper describes a platform for distributed virtual environments on the Internet, presenting the principles that have guided the platform design, together with an architecture and a protocol for this platform. The platform employs the division of the participants of the virtual environment into multicast groups in order to achieve scalability. We investigate scalability and group exchange dynamism of the platform via simulation. The results show that the division into multicast groups reduces the number of messages exchanged, and the delay both to enter and to exchange groups. They also show that the proposed platform provides a good interactivity.

Event: General Conference

Multicast Communication for Scalable Distributed Virtual Environments

Renata Cruz Teixeira — Otto Carlos M. B. Duarte
`renata@gta.ufrj.br` — `otto@gta.ufrj.br`

Grupo de Teleinformática e Automação (GTA)
Universidade Federal do Rio de Janeiro
COPPE/EE - Programa de Engenharia Elétrica
P.O. Box 68504 - 21945-970 - Rio de Janeiro - RJ - Brazil

Abstract

This paper describes a platform for distributed virtual environments on the Internet, presenting the principles that have guided the platform design, together with an architecture and a protocol for this platform. The platform employs the division of the participants of the virtual environment into multicast groups in order to achieve scalability. We investigate scalability and group exchange dynamism of the platform via simulation. The results show that the division into multicast groups reduces the number of messages exchanged, and the delay both to enter and to exchange groups. They also show that the proposed platform provides a good interactivity.

1 Introduction

Distributed virtual environments (DVE) aim at providing users with a sense of realism by incorporating realistic tri-dimensional graphics and stereo sound to create an immersive experience. In these environments, also called virtual worlds, geographically dispersed users can interact with each other and with the environment as they interact in the real world. The integration of immersive graphics with the ability to have multiple users share information and manipulate objects in the environment makes virtual reality systems a powerful tool. These systems are being used for education and training, for engineering and design, for commerce, and for entertainment.

For all their potential, DVEs are difficult to implement correctly or effectively for a large number of users. The main limitations to scale distributed virtual reality systems are the restrictions of the communication system [1]. A DVE must contend with data losses, limited bandwidth, and delay.

In these environments, each participant has a graphical representation, called an avatar. As a user moves within the virtual environment, his avatar current state must be updated so that other users will visualize the avatar in the correct location. If a user picks up an object in the environment, other users need to be notified that the object is now being carried. These updates prevent inconsistent representations of the virtual environment among different hosts. Thus, each participant must continuously send state updates to all other ones sharing the same environment, what causes a great volume of data exchanged.

Nevertheless, a participant does not interact with all other ones in the world simultaneously. For example, in a large DVE, all the participants rarely stand in the same room. Instead, users tend to scatter into different sub-groups or teams, each located in a different part of the virtual world. Thus, exchanged messages may be filtered, defining a scope where participants can interact with each other, this scope is called interaction scope. Some proposals [2, 3, 4] were based on data filtering to optimize the use of communication system resources, translating each interaction scope into a multicast group. Since participants may move around the world, group partition does not remain static. Therefore, multicast protocols to support DVEs must allow dynamic group exchanges. Besides, the delay introduced with these group exchanges should be minimum, for the purpose of avoiding interruptions in user's perception of the virtual environment.

Research into large-scale virtual worlds has been growing significantly. The NPSNET [2], developed by the Naval Postgraduate School, is a simulation system for military training based on the distributed interactive simulation (DIS) protocol, and the logical partition of the world into classes. Each class has an area of interest manager which is responsible for it. This reduces the computational load in hosts and the information exchanged on the network. The communication for the Distributed Interactive Virtual Environment (DIVE) [3] uses a reliable multicast protocol with negative acknowledgement (NACK). The DIVE consider a multicast group per world, hence there is no data filtering within a world. The Model, Architecture and System for Spatial Interaction in Virtual Environments (MASSIVE) [5] system uses a spatial model for data partitioning among clients. The SPLINE (Scalable Platform for Large Interactive Networked Environment) [4] splits the world into locales, which can be processed separately.

This paper describes a platform for distributed virtual reality systems on the Internet. This platform provides a consistency maintenance and distribution service to virtual reality applications. The paper discusses the impact of participants division into multicast groups on these systems, focusing on the tradeoff between the number of groups and the number of group exchanges needed. This compromise was investigated via simulation, measuring the delay to exchange groups and enter the virtual world, the number of messages exchanged, and the number of group exchanges.

This paper proceeds as follows: section 2 describes platform design issues, the architecture, and the protocol proposed to it. Section 3 presents the simulation model and the infrastructure. Section 4 discusses simulation results, which evaluate the impact of the distribution in the platform. Finally, section 5 concludes.

2 The Proposed Platform

The proposed platform aims at supporting distributed virtual reality applications, providing a distribution and consistency maintenance service on the Internet. The application provides the scene for the virtual environment, the avatars, and information about the environment structure. The platform is responsible for all interactions in the virtual world, assuring world consistency.

2.1 Platform Design

The platform divides participants into multicast groups according to the structure defined by the application. Consistency control is assured inside each group by a special participant called leader. The leader is responsible for controlling all participants entrances, departures, and group exchanges (handoffs) within his interaction scope. In addition, the leader controls access to virtual environment objects.

In order to maintain world consistency, communication among group participants must be reliable. Multicast protocols, however, offer limitations to the number of users when reliability is needed [6]. Therefore, the proposed platform reduces reliable communication dividing virtual world operations in critical operations, which need reliability, and non-critical operations. Operations like entering the virtual environment, departing, and exchanging groups are classified as critical operations, while state updates are non-critical operations.

In summary, scalability is achieved by the distributed consistency control, which is performed by the leaders, and the reduction of reliable data exchange, by the division of operations.

The platform uses three communication types with different features. Non-critical messages require minimum delay and all participants within a scope receive it. Critical operations demand reliable message exchanges, which may be unicast, to deliver the database, for example; or multicast, as when an avatar is sent to a group. In order to support all these requirements, the use of three different transport protocols gathered with a network service that provides multicast delivery is needed. The IP Multicast [7] allows hosts to join and leave groups at any time, that is, the membership of a group is dynamic. So, it was used together with the UDP for the transmission of non-critical messages. For the critical multicast messages the transport protocol employed was the SRM (Scalable Reliable Multicast) [8], a reliable multicast protocol with negative acknowledgments, aiming at scalability. The TCP is used to exchange messages between two participants reliably.

2.2 Platform Architecture

The platform architecture is presented in figure 1. All end hosts in the DVE use this structure, except leaders which use, in addition, specific control modules. Virtual environment data are partially-replicated in the database (DB), which contains the scene and avatar information. Updating this database, users modify the state of the virtual world.

The modules that maintain the environment consistency are: the virtual environment module, and the group controller module of the leader. On the participant, the virtual environment module sends updates performed in its local DB to all other participants, and receive others' updates. On the leader, the group controller manages group joins and leaves.

Direct interactions between users and interactions with the environment are managed by the module interaction controller. This module is responsible for setting up a connection between the participants, so that they can interact directly through the multimedia module. This multimedia module adds a necessary sense of realism to the virtual environment, providing direct communication through gesture, typed text, or voice. The

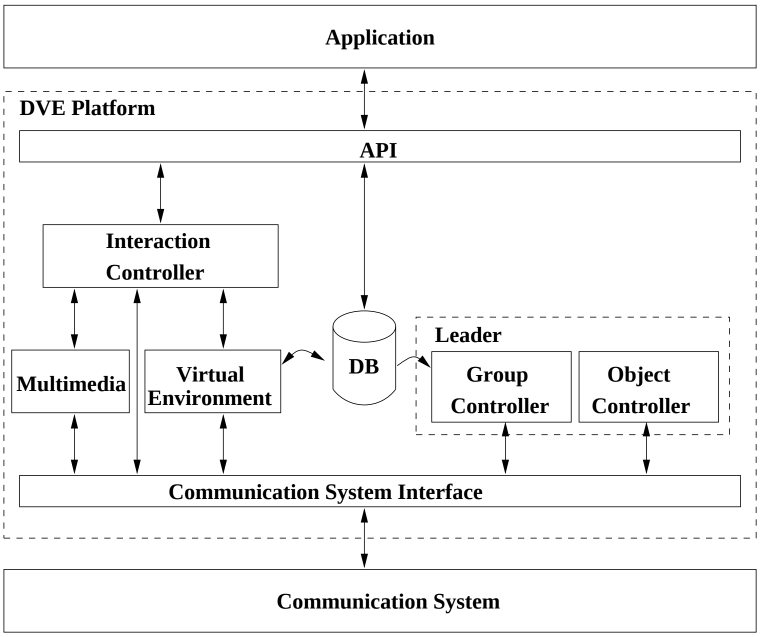

Figure 1: The proposed platform architecture.

interaction controller is also responsible for the concurrency control to access objects in the world. This control is performed together with the object controller module, which contains permissions of access to all objects managed by the leader.

2.3 Protocol Description

The platform includes interactions both among users and with the virtual environment, and the control required to maintain world consistency. This paper main focus is the control of virtual environment information in order to achieve consistency maintenance. This control includes functions that determine system scalability – world entrances and departures, state updates, and handoffs. Therefore, the protocol description presented here concentrates in these functions. Table 1 shows the protocol data units (PDUs).

Each virtual world participant is a member of a multicast group. This group is determined according to its position in the virtual environment. All leaders belong to a control

Table 1: Protocol Data Units.

PDU	*Purpose*
JR	Request to join a group
JI	Indicate a new avatar joined the group
JC	Confirm a join request, sending the DB
LR	Leave the group
FNL	Indicate that a participant should initiate a handoff
ES	Update avatar state

multicast group, which has a known address. Thus, to begin a session, i.e., to enter the virtual world, participants multicast a JR PDU to this address requesting to join a group. The JR PDU carries the initial position and the information of the new participant avatar. Upon receipt of a JR, all leaders verify whether the position requested is inside its interaction scope. The leader responsible for this position answers the request by sending a JC PDU, which carries the DB containing the current state of its scope. After the delivery of the DB, the leader multicasts to all members of its scope a JI PDU with information of the new participant.

After the entrance phase, participants exchange state updates, multicasting ES PDUs to their interaction scopes. An ES PDU reports avatar position and behavior information. The leader, as any other participant, receives every state update of all members of its scope. When it detects that a participant is crossing the boundaries of its scope, the leader sends a FNL PDU to that participant, indicating that a handoff must be started. Thus, the participant multicasts a JR PDU and restarts the another phase. After receiving a JC from its new leader, the participant leaves its old group multicasting a LR PDU. It is important to remark that handoffs are managed by platform modules without the knowledge of the application. To finish a session, a participant multicasts a LR PDU to its scope, requesting the removal of its avatar from the environment.

3 Simulation

The platform was simulated in order to verify the impact of data filtering on DVEs. Simulations were based on the protocol described in section 2.3, and include operations for consistency maintenance: world entrances and departures, handoffs, and state updates.

3.1 Simulation Infrastructure

The tool chosen to simulate the platform is the *ns* (Network Simulator) [9]. The *ns* is a discrete event simulator used in the networking research community. It provides an infrastructure for network protocol simulation that has abstractions to nodes and links. This infrastructure also provides support for routing, including multicast routing. Moreover, it includes several models of common Internet protocols (e.g., TCP, UDP), plus several novel protocols targeted for eventual use in the Internet (e.g., reliable multicast).

This infrastructure is used to simulate the platform for DVEs. The transport protocols used were the UDP, the TCP, and the SRM. Some adjustments were needed in the SRM protocol in order to fit platform requirements. In the SRM, when a new user joins a session, the user will obtain the session's history from the network [8]. On DVEs, however, when a user joins a group he wishes to receive only the current state of his interaction scope. The receipt of the session's history would cause a large number of needless messages, wasting both communication system resources and processing time for these messages to be discarded by the new member. Therefore, the SRM module of *ns* was adapted not to send the session's history to new group members.

3.2 Topology Generation

Topologies to simulations were created by the topology generator GT-ITM (Georgia Tech Internetwork Topology Models), aiming at graphs characteristics that resemble those of the Internet. The GT-ITM uses a hybrid generation approach [10], the Transit-Stub model, which allows the generation of topologies using a domain structure similar to that found in the Internet.

The generated graphs have several sub-networks (stubs) interconnected by cores (transit), which represent Internet backbones. Two topologies were used in the simulations: one with 20 nodes – four in the transit plus four stubs with four nodes each; and another with 40 nodes – 8 in the transit plus 8 stubs with four nodes each. Both topologies were generated by the GT-ITM with similar link latencies and a bandwidth of 2 Mbps. The former topology is presented as an example in figure 2, where the transit nodes are circled and link labels indicate the latencies generated. In the simulations, the platform is only executed on the nodes of the stub domains. Each node of the stub has exactly one user. Therefore, in the simulation using the topology presented in figure 2 the number of participants simulated is 16, and using the other topology the number of participants is 32.

3.3 Simulation Model

The platform depends on some characteristics and information defined by the application, which are translated into parameters of the simulation model. The application must set the database containing the initial scene of the virtual environment, this database is translated into a parameter with the initial size of the database. The avatar definition is converted into a parameter that express the size of an avatar. We assume that all avatars have the same size.

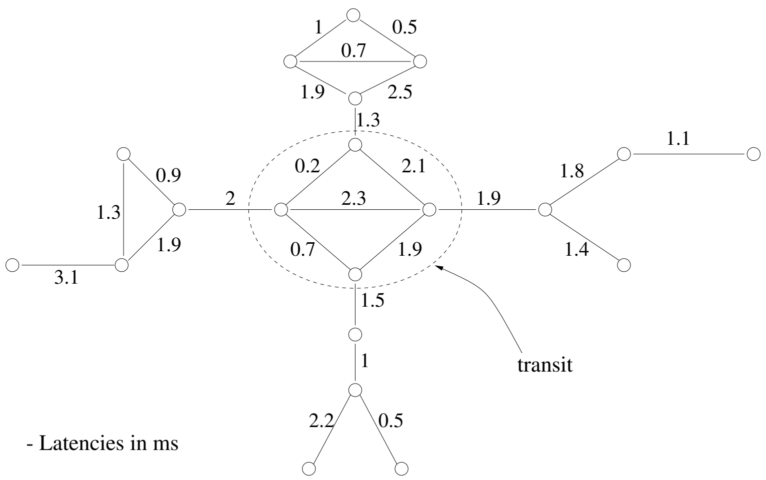

Figure 2: Topology created by the GT-ITM.

Those parameters are used for scope joins and leaves. When a new user joins a group, the database is increased by the size of the new avatar. Similarly, when a participant leaves its scope, the size of its avatar is decreased from the database size.

Leaders are randomly[1] chosen among all possible participants. The virtual environment is divided among all leaders, so that there is no intersection among the areas of the different leaders, and that the union of all areas results in the virtual world itself. The initial position of participants is randomly selected among all possible positions in the environment.

The entrance and the departure time of every participant are randomly generated, except the time of the leaders, which enter in the beginning of a session, and leave in the end of it. The leaders do not exchange groups, so there is no need to elect a new leader for a group in the middle of a session.

Participant's movement around the world is associated with a probability (P_m). Thus, with probability P_m a participant moves one position ahead, i.e., increments its position by one unit. This probability is related to the participant's speed, so, a faster participant

[1]All random values were generated using the random variables available in *ns*.

Table 2: Parameters used for the simulation.

Parameters	*Values*
DB initial size	10 k bytes
Avatar size	1,5 k bytes
Update size	100 bytes
P_m	0,2

should have a higher probability of movement. In the simulations performed, we consider that all participants have the same probability of movement. For each participant's movement, a state update message is multicasted to its scope. When participant's position cross the boundaries of its scope, the leader begins this participant's handoff.

The simulations were performed based on this model, and using the topologies described in section 3.2. The simulation parameters used to obtain the results are presented in table 2. The platform was simulated with sessions of five minutes each.

4 Results

The measurements performed aimed at evaluating the impact of data filtering in distributed virtual environments, analyzing some features to achieve more scalability and less interruptions in perception of the virtual environment. In order to assess how the division into scopes affects scalability, the delay to enter the world and to exchange groups, and the number of messages exchanged were measured. Interruptions are evaluated by the number of handoffs in a session, and the time spent in each handoff.

In the simulations performed, the space of the virtual world, i.e., the number of possible positions remains, while the number of groups varies. So, when the number of groups increases, the area of each group reduces. When there is only one group, the virtual environment has one leader for all participants. So, the platform acts as a centralized system, overloading the leader. There is no data filtering inside the virtual world. The world is not partitioned, there is no group exchanges, and hence there is no interruptions in perception of the world.

On the other hand, when the number of groups is greater than one, consistency control is distributed among leaders. The larger the number of groups, the fewer participants are controlled by each leader, the greater the distribution of this control, and the smaller the number of update messages exchanged. Nevertheless, there are more interruptions in the environment due to more frequent handoffs.

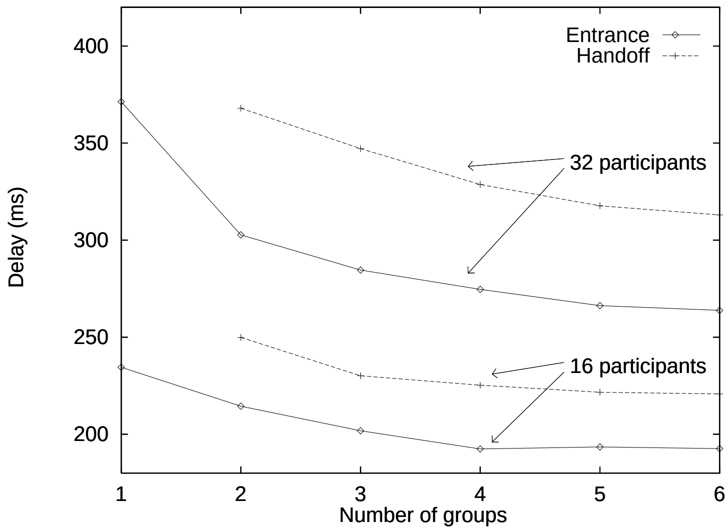

Figure 3: Delay to enter the virtual environment and to do a handoff.

Figure 3 shows simulations comparing the average delay to enter the virtual world and to do a handoff for 16 and 32 participants. It is important to remark that there is no delay associated with handoffs when the number of groups is equal to one. The difference between handoff and entrance delays for each topology is due to the delay in notifying the participant that it must leave the group and start the entrance phase.

When the number of groups increases, entrance and handoff delays decrease. This behavior is similar for both topologies, and is caused by the distribution among leaders of the responsibility to send the database to participants. With fewer participants in each scope, leaders become less overloaded and can answer join requests faster. This distribution is essential to the scalability of the environment, because when the number of participants increases, the division into groups keeps the processing load approximately constant in each leader. The main element that affects both delays is the delivery of the database by the TCP, because of transmission delay. Hence, a compression algorithm could be used in order to keep this database as small as possible.

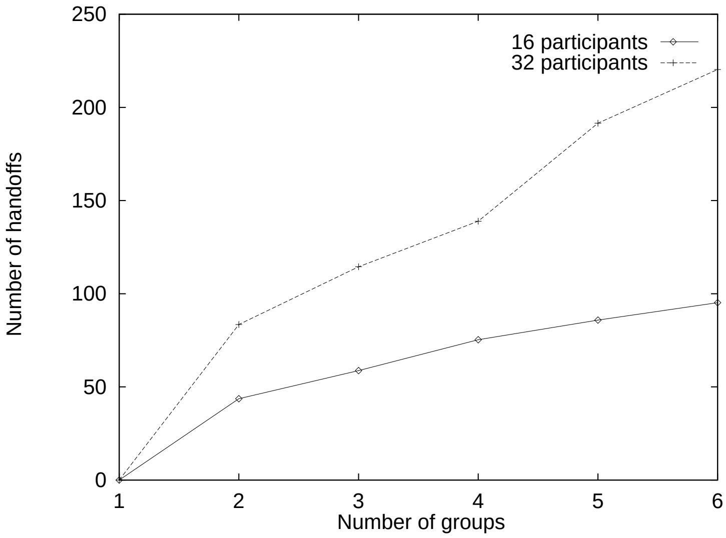

Figure 4: Total number of handoffs for all participants.

Figure 4 shows the total number of handoffs accomplished by all participants together on a session. The number of handoffs is greater for 32 participants than for 16. This is due to the fact that this number is calculated to all participants, but the number of handoffs per participant depends on the size of the area of each group. When the number of groups increases, the area associated with each group decreases. As the probability of movement remains for all performed simulations, the average speed of participants remains too. Therefore, when the number of groups increases, the frequency of handoffs increases.

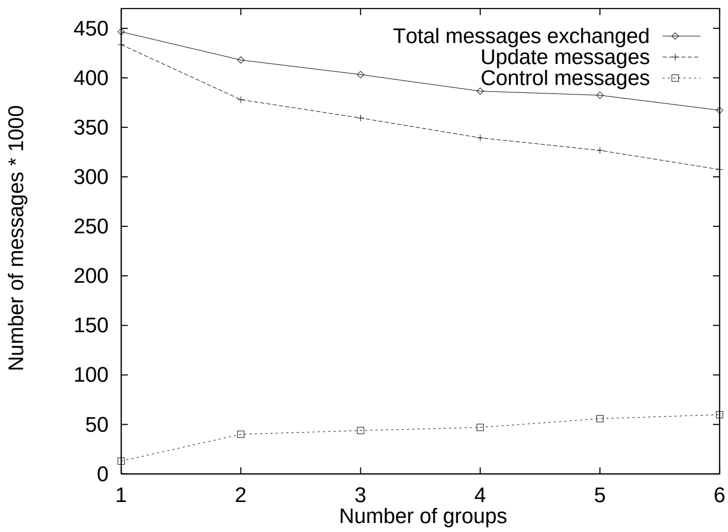

Figure 5: Number of exchanged messages for the topology with 32 participants.

Figure 5 shows the number of messages exchanged for 32 participants – the behavior for 16 participants is analogous. While the number of groups increases, the number of update messages decreases. The interaction scopes become smaller, hence the update messages are delivered to fewer participants, optimizing the use of communication system resources. Simultaneously, the frequency of handoffs increases, and so the number of control messages to perform these handoffs. The reduction of update messages, however, is more significant than the increase of exchanged control messages, what can be confirmed by the total messages exchanged, which decreases when the number of groups increases.

5 Conclusion

This paper discussed the use of multicast protocols for distributed virtual reality applications, and its implications. Multicast communication is essential to implementation of data filtering, allowing more scalable DVEs. The proposed platform aims at increasing scalability by the distribution of the consistency control among leaders, and the reduction of the exchange of reliable messages.

Simulation results show that the division into groups, using multicast protocols, cause a reduction in delay both to enter and to exchange groups, and in the number of messages. This reduction in delay benefits participants interactivity, and decrease probability of inconsistencies. There is a tradeoff between this gain in interactivity and the interruptions generated by handoffs when the number of groups increases. Nevertheless, for the simulations performed, the interruptions were not significant, standing for less than 1% of the total time of a session for each participant. Virtual environment division into groups plays an important role in the system scalability. Despite the increase in control messages, this division reduces the total number of messages and the load in each leader, enabling more participants sharing a virtual environment.

In summary, the proposed platform, using a distributed model among leaders and a division of the environment in separated scopes, allowed a good interactivity without harm to perception of the virtual environment.

References

[1] M. R. Macedonia and M. J. Zyda, "A Taxonomy for Networked Virtual Environment", *IEEE Multimedia*, vol. 4, no. 1, 1997.

[2] M. R. Macedonia, M. J. Zyda, D. R. Pratt, D. P. Brutzman and P. T. Barham, "Exploiting Reality with Multicast Groups", *IEEE Computer Graphics and Applications*, pp. 38 – 50, Sept. 1995.

[3] O. Hagsand, "Interactive Multiuser VEs in the DIVE System", *IEEE Multimedia*, vol. 3, no. 1, pp. 30–39, 1996.

[4] J. W. Barrus, R. C. Waters and D. B. Anderson, "Locales: Supporting Large Multiuser Virtual Environments", *IEEE Computer Graphics and Applications*, vol. 16, no. 6, pp. 50–57, Nov. 1996.

[5] C. Greenhalgh and S. Benford, "MASSIVE: A Distributed Virtual Reality System Incorporating Spatial Trading", in *Proceedings IEEE DCS'95*, Vancouver, Canada, 1995.

[6] S. Pingali, D. Towsley and J. Kurose, "A Comparison of Sender-Initiated and Receiver-Iniciated Reliable Multicast Protocols", *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 3, Apr. 1997.

[7] S. Deering, *RFC 1112: Host Extensions for IP Multicasting*, Aug. 1989.

[8] S. Floyd, V. Jacobson, C.-G. Liu, S. McCanne and L. Zhang, "A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing", *IEEE/ACM Transactions on Networking*, vol. 5, no. 6, pp. 784 – 803, Dec. 1997.

[9] K. Fall and K. Varadhan, "NS Notes and Documentation", tech. rep., The VINT Project, Nov. 1997.

[10] E. W. Zegura, K. Calvert and S. Bhattacharjee, "A Quantitative Comparision of Graph-Based Models for Internet Topology", *IEEE/ACM Transactions on Networking*, vol. 5, no. 6, pp. 770 – 783, Dec. 1997.