Combinatory Multicast for Differentiated Data Transmission in Distributed Virtual Environments

Andrés Quiroz Hernández and Helmuth Trefftz Gómez
{aquirozh, htrefftz}@eafit.edu.co
EAFIT University
Medellín, Colombia

ABSTRACT

The shortage of multicast addresses is a factor that limits their use in Internet-based applications. Since multicast plays a very important role in supporting resource heterogeneity among users of applications with a high demand of computational resources, such as Distributed Virtual Environments, the sparing use of these addresses is of great importance. This paper suggests a scheme for the combination of multicast groups for differentiated data transmission, based on layered multicast, with which the number of different levels of service that can be provided to users of a shared virtual environment with a fixed number of groups is multiplied. For applications where the levels of service correspond to different transmission rates, the scheme also significantly reduces bandwidth utilization. We present a basic method for its implementation and describe preliminary tests to prove its functionality.

1. INTRODUCTION

With the rapid growth in the field of Distributed Virtual Environments (DVEs), both in the proliferation of applications and in the complexity of the shared virtual worlds, the search for greater efficiency in the communication of shared data is ongoing. This need is due in part to the problem of dynamic shared state, which is defined as the tradeoff between the consistency or synchronization of two or more views of a shared object or world and the effective transmission rate of the messages necessary to maintain the state of this object updated among the users (this is called the *consistency-throughput tradeoff* [9]). According to this theorem, it will always be impossible, because of communication latency, for two geographically separated users to visualize the exact same state at the exact same time, if this state is constantly changing. The theorem can also be expressed in terms of the need to share the available bandwidth (which can encompass all available resources) between messages to update the shared state and messages to maintain the consistency of the different views. This is why the optimization of communication architectures and protocols for DVEs and their adaptation to the heterogeneous resources of their users is of utmost importance as these systems share more information, involve more complex computations, and are subject to greater real-time constraints.

Several methods have been devised to deal with the adaptation to users' limited and heterogeneous resources. In general, these are based on the selective transmission of data to different groups of users or on downgrading such data to different degrees, in either the update frequency of the shared state or in the detail or resolution of the description of this state, according to user interest, location, and/or capability. In other words, they are based on differentiated data transmission. This allows the optimization of the use of resources, in such a way that update and control messages can be received and processed on time, and, thus, a synchronized, if not completely congruent, view of the virtual world can be maintained. Macedonia and Zyda [4] make a complete account of the most representative methods and systems that apply these principles.

Independently of the particular method used, it is necessary to determine the communication protocol that will allow the differentiated data transmission to the network clients. With unicast protocols, such as TCP or UDP, it is necessary to maintain a flow with each separate client, where each flow can be adapted in content and quality to the specific needs of a particular client. Examples of tools that use this scheme are found

in [2, 3]. Although this kind of communication allows for perfect adaptation to the particular needs of each client, it is limited in terms of scalability for placing too much load on the server[1] (there are, however, methods for server distribution that deal with this problem [1]). On the other hand, IP multicast has become a commonly used tool for the implementation of DVEs because of its unique capacity to reach a particular subset of network clients and its publisher/subscriber functionality, which allows clients to select the information that they want to receive [4, 9]. This way, the server can maintain a single flow for each different quality of data, assigning to each a different multicast group, so that each client can subscribe to the one that best conforms to its particular conditions. For example, in a system that provides different multicast groups for different geographic regions, a client must subscribe to the group that corresponds to its present location in order to receive only the updates that are within its user's sensory range. Of course, this scheme is less flexible in its adaptation to user needs because generally a fixed set of possibilities is offered to clients from which to choose.

One of the problems associated with this type of solution is the relative scarceness of multicast addresses; the range of multicast addresses goes from 224.0.0.0 to 239.255.255.255, and there are several ranges within this one that are reserved by the IANA (Internet Assigned Numbers Authority) [9]. Applications that are deployed in the Internet are, thus, in risk of running into conflicts with other applications that use the same multicast addresses. To minimize this risk, the use of multicast groups in the application must be conservative, which places a greater restriction on the amount of differentiated flows that can be offered.

This paper suggests a scheme for multicast utilization that grants greater differentiation in the levels of service that can be offered, especially when dealing with update rates, using less multicast addresses than would be necessary with existing methods. The emphasis of this work is on presenting the theoretical basis of this method, and on the aspects that should make its implementation plausible. Some tests were also conducted on a local area network in order to verify certain functional aspects. The next section describes some previous work on which our method is based, the Switchboard Architecture and layered multicast. Section 3 explains the alternative to layered multicast that we propose, and Section 4 shows our method of implementation. Section 5 then describes the tests that were carried out and the conclusions drawn from them. Finally, several possibilities for further work are discussed.

[1] Here we refer to the server as the source of data, which can also be a user of the virtual environment in a peer-to-peer architecture. However, we make an emphasis on client/server architectures because we consider they are better suited for differentiated service.

2. BACKGROUND AND SUPPORTING THEORY

2.1 SWITCHBOARD ARCHITECTURE

The Switchboard Architecture (SA) [8] is a generic client/server architecture that allows quality differentiation for the transmission of different data types in a DVE. Basically, it consists of a server, which implements the switchboard, and a set of *N* clients. Dividing object information into different data types, the server receives separate updates for each data type from each user. It then provides different levels of downgrading of the data according to its type and organizes the new update flows into different transmission plugs, implemented as multicast groups, as can be seen in Figure 1.

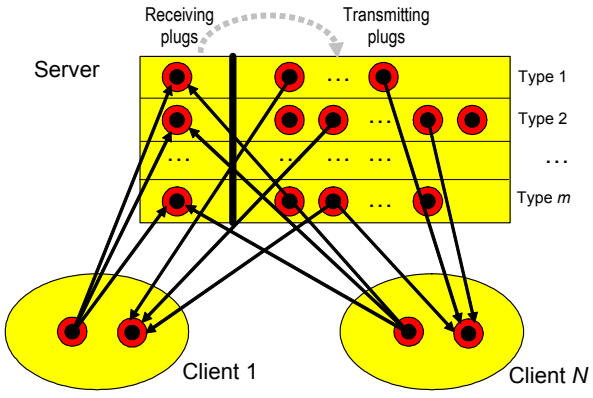

Figure 1. Switchboard Architecture layout

This configuration is especially suited for providing different frequency levels or rates because it can be programmed to perform this conversion on any type of data flow. Other quality transformations usually require more

specific procedures. The server's task, as it is implemented in [7], is to duplicate the signal to relay and apply different levels of buffering, so that each resulting flow has a different rate. Since this involves duplication of the data, it is very likely that there will be redundant packages in the different flows for the same data type, and, consequently, the quantity of service levels that can be provided is equivalent to the number of multicast groups that must be used. This type of utilization of multicast groups is called *simulcast* [5], since each client subscribes to only one group per data type, according to the level of service required.

2.2 LAYERED MULTICAST

Layered multicast [5] is a data distribution scheme that is commonly used for the distribution of video. As with other data differentiation transmission techniques, it makes use of a set of multicast groups that carry different information from which clients can choose. However, the splitting of information into these groups is made so that the groups are ordered, with data in a higher level depending on data from lower levels. This way, the subscription to a multicast group only makes sense if subscriptions to the levels below it exist. This hierarchical organization is very useful for data that can be progressively refined, and it can achieve significant savings in bandwidth, since there is no redundancy in the information that is transmitted through each of the multicast groups. For video, for example, successive refinements in color or resolution that would be of no use by themselves can be applied to a basic signal. Formally, the conditions for layered multicast are the following:

$$C_i \cap C_j = \varnothing, \ \forall i \neq j \qquad (1)$$

$$\mathrm{S}(n) = \{C_i \mid i \leq n\} \qquad (2)$$

Here, each set C_i represents the content of a multicast group or layer that has a level of service i, and $\mathrm{S}(n)$ is the set of groups to which a client must subscribe to receive a service level n.

While the first condition is fundamental for the application of this scheme, if bandwidth saving capability at the server is considered, then it is evident that the second condition is not always necessary and could be relaxed. For example, while still maintaining a hierarchical organization of information, there could be layers of equal hierarchy that could be included alternatively in the subscription set, or non-essential intermediate layers that could be excluded from it. Since condition (2) implies a one-to-one relation between the multicast groups and the levels of service, its relaxation can increase the number of service levels that can be offered with the same number of groups. In addition, rate differentiation is not hierarchical and is not suited for this scheme, although Shirmohammadi and Georganas propose a message hierarchy for collaborative applications in [6]. Because of the above, we propose a generalization of layered multicast, applicable especially to rate adaptation.

3. COMBINATORY MULTICAST

So far, as is explained by Jacobson et al. in [5], the type of organization considered for multicast groups can be either cumulative, which corresponds to layered multicast, or independent, i.e. simulcast, like the one used in the SA. Both have a one-to-one relation between the groups and the service levels offered. However, according to the reasons presented in the previous section, when information is not hierarchical, there are many other possibilities for its combination. The general condition that would replace layered multicast's condition (2) is the following:

$$\mathrm{S}(n) = \left\{C_i \mid \sum i = n\right\} \qquad (2b)$$

The potential benefit of this formulation is that, with it, a system can support more levels of service than the number of multicast groups used. Therefore, the server's coverage of users' needs can be improved, without overusing bandwidth because of redundancy or requiring too large an amount of multicast addresses. However, precisely because there are many possibilities for defining the content of the groups that can be combined, the implementation method of this scheme is of special importance in the achievement of a real benefit.

4. IMPLEMENTATION ISSUES

Combinatory multicast requires changes in the transmission and reception layers of the server and clients, respectively, in the SA. Basically, the signal from the server must be multiplexed,

with the basic multicast groups serving as the separate channels. At the server, a de-multiplexer would take each packet from the original flow and direct it to a different multicast group. The packets would travel in their separate groups along the network, and, at the client, a multiplexer must reconstruct the flow, according to the level of subscription, so that the relative order of the packets is maintained. Figure 2 illustrates this process.

It must be noted that, unlike layered multicast, there cannot be dependencies among the packets that make up the data flow, since there is no guarantee that the client will be subscribed to any particular group. Because of this, every update must make an absolute description of the shared state (see [6]). In order to separate the packets among the multicast groups, a sequence must be defined so that each group has a given rate. This is not a trivial problem, for several reasons. First, it is necessary to choose the groups' rates so that they can be combined to obtain the greatest amount possible of different resulting rates, based on the source's original transmission rate. Secondly, given a set of groups for one data type, the packets must be distributed in such a way that each group effectively transmits with the assigned rate and maintains certain uniformity in the time between messages. The system designer must make sure, therefore, that the original flow has enough messages to support each one of the individual flows, and that the assignment of packets to groups is periodic. Finally, it is possible for the server to receive updates for each type at variables rates, and, thus, for the multiplexing procedures to have to be adaptive in order to maintain the offered service levels.

For this paper, we assume that transmission rates are fixed and known ahead of time. This can be ensured by setting update periods where the current state of an object is reported regardless of whether there have been any changes. In order to determine the number and levels of service of the basic multicast groups, we propose a roughly exponential pattern, which is used in the mint values of some currencies, i.e. 1, 2, 5, 10, 20, 50, 100, and so on. The combination of these values covers intermediate values well. For a given transmission rate, multicast groups can be defined with rates taken from consecutive values of this pattern, so that their sum does not exceed the given transmission rate. It is possible to assign the same rate to more than one group, as long as the assignment contributes to the distribution of all updates among all the groups

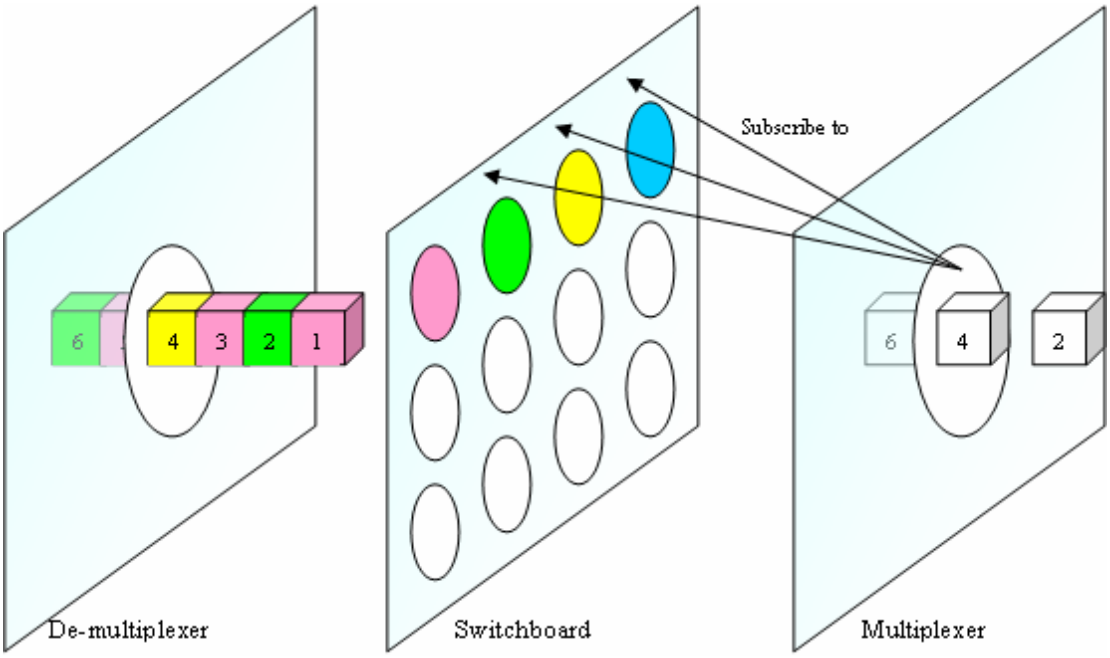

Figure 2. Procedure for message transmission through combinatory multicast groups. The server (left) distributes the packets of a flow according to the rate of each group, while the client (right) must ensure the ordering of the packets that it receives through the different groups to which it is subscribed for a particular data type.

and to the achievement of the highest possible number of combinations. For example, it is justified for an original flow of 30 updates per second to generate groups of 1, 2, 5, 10, and 10 updates per second.

Since the pattern is roughly exponential by a factor of two, given a maximum service level of L and if the rates of the groups are not repeated, a coverage of 80% of all inferior rates can be achieved with a number of groups in the approximate order of $log_2 L$. Although for large values of L the combination of many multicast groups would be needed for certain levels, it is not expected for the transmission rates of DVEs to be excessively high due to limitations in users' perceptiveness.

For the appropriate distribution of packets among the multicast groups, a unit of time must first be chosen for reference so that the number of updates that are expected in that unit of time can be determined. If the previous step was done optimally, then the sum of the rates of each of the combinable groups should be equal to the received rate. Then, on a sequence with the number of updates received in the unit of time, a pattern can be defined to assign update packets to the groups, in decreasing order of frequency, in order to obtain periods that are as uniform as possible. This ordering is suggested because it is usually easier to accommodate small amounts of packets to low frequency groups when the rest of the packets have been assigned, than it is for the converse case. In any case, the lack of uniformity in the periods of service levels is an inherent problem to this combinatory scheme, because different combinations will result in different orderings of updates in time anyway. Although this problem could be solved by applying a filter to give the signal a uniform period, in exchange for greater latency in the updating of the shared state, we are interested in observing in our tests the visible effect in the movement of objects in the virtual environment.

The technique for multiplexing packets received through separate multicast groups at the client depends on the policy for the handling of the shared state. There are two basic ways to proceed. On one hand is a system that always displays the most recent update received, discarding others that are previous to one already displayed. In this kind of system, the most important thing is to maintain a view that is as up-to-date as possible and in-sync with the remote source. On the other hand is a system that keeps a queue of out-of-sync messages, while it waits for the next in the sequence. The emphasis of this policy is on the sequence of actions that lead to a given state. It must be parameterized by the maximum length of the queue, which indicates how long the system must wait for an update before it is considered lost. The selection between these techniques is critical for this type of system because it is expected that the existence of several multicast groups for a single flow increases the probability that the packets will arrive in disorder at their destination. Because of this, the application of the first policy might result in an unnecessarily excessive degradation of the update flow. Taking some time to order the updates, a number of them that would otherwise be discarded can be displayed, but the appropriate length of the queue must be carefully defined, because unacceptable latency could be incurred if there is packet loss.

5. TESTS AND RESULTS

The tests were carried out on a prototype of a virtual environment for remote collaboration around an object of interest. The graphical interface of this tool is shown in Figure 3. There are three types of shared data: the updates for the virtual object that is being observed and manipulated by the collaborators, the updates for the 3D arrows that are used as avatars for the collaborators in the virtual world, and video updates, which in this case consist of a series of still frames.

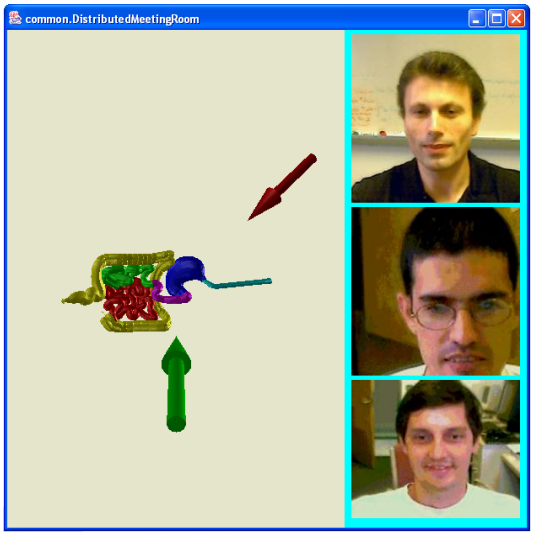

Figure 3. Application for remote collaboration

There is currently an implementation for this tool based on the SA that uses simulcast to provide levels of service of 1, 2, 5, 10, and 20 updates per second for each data type. It is therefore assumed that the flow from each source will be the maximum offered, i.e. 20 updates per second. Following the methods from the previous section for a combinatory multicast scheme, the same number of multicast groups will be used, at the same basic rates. The pattern for the distribution of each of the 20 updates per second among the multicast groups is shown in Table 1, where each column contains the rates of the groups to which an update is assigned.

Note that the group that delivers 20 updates per second is assigned all packets in the flow, and, because of the redundancy with respect to the other groups, is not combinable with them. It was included because without it the maximum rate could not have been offered, unless two extra groups of one update per second each were created. This shows that there are multiple ways to define the multicast groups, and that the application of the methods presented in the previous section does not guarantee the best choice. With this group definition, however, every rate between 1 and 20, except for 4, 9, 14, and 19 updates per second (80%) can be obtained. For comparison, the same service levels were implemented with the original simulcast Switchboard Architecture, for which 16 separate multicast groups are required.

The first thing that can be observed is the reduction of the maximum use of bandwidth at the server, which is obtained directly from the almost total reduction of redundancy in the transmitted flows. Figure 4 shows that in the combinatorial multicast scheme only the original flow of 20 updates is sent per second, plus, in this case, an additional 20 updates per second of the redundant channel, while in the original scheme the amount of packets equivalent to the sum of all the service levels is being sent per second. The dotted line points out the level attainable if all redundancy is eliminated with combinatorial multicast.

The idea behind the tests is to be able to observe the behavior of the client application as it makes successive subscriptions to all service levels using both schemes. It is assumed that the processing capabilities of the machines do not influence the test results, i.e. enough processing capability is available to execute all updates.

With regard to the visualization of the movement of the objects on the screen, the lack in uniformity of the period of some levels provided by combinatory multicast is barely noticeable, and mostly at the lower rates. With optimum network conditions, there was no problem with the synchronization of messages at the client. With heavy network traffic on the local network as a source of congestion, there was packet loss in both cases (this loss was verified by making the update queue at the client sufficiently long). As can be seen in Figure 5, there is no clear tendency that allows the conclusion of greater update loss because of the existence of the multiple channels per flow of combinatory multicast.

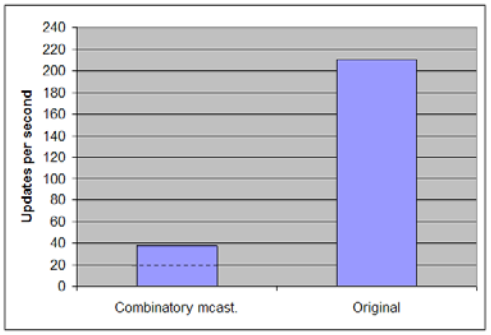

Figure 4. Maximum bandwidth utilization levels of each of the schemes. Both allow the reduction of utilization by not transmitting updates on channels that have no subscriptions.

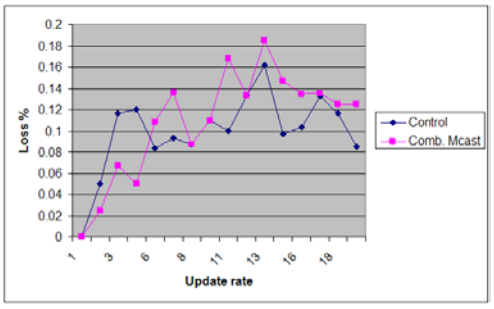

Figure 5. Update message loss comparison in both transmission schemes under conditions of congestion.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
10	5	10	2	10	5	10	1	10	5	10	-	10	5	10	2	10	5	10	-
20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20

Table 1. Pattern for the assignment of packets to multicast groups.

6. CONCLUSIONS AND FURTHER WORK

This paper defined combinatory multicast as a scheme for the differentiated transmission of data in Distributed Virtual Environments for the potential economization of multicast addresses and server bandwidth. The conditions and the methods necessary for its implementation were described, and its functionality on local area networks was demonstrated.

The tests that were carried out are still preliminary, and, as has been mentioned, are only good for proving functionality, but not performance. Only with tests on a wide area network or on the Internet, where multiple routing possibilities exist, can the real effect of update disorder on the visualization and synchronization at the client be observed. In addition, the optimality of the proposed methods has not been established, and, therefore, their refinement or redesign can be the subject of further study. For some systems, it may not be possible or practical to assume constant update rates, because of which the formalization of the methods for online application is also necessary. As they are defined, these methods are suited for programming, but for highly dynamic cases the study of more versatile procedures is worthwhile.

In any case, the saving of multicast groups accomplished by this scheme compared to previous methods with regard to the levels of service offered is significant. Although for a single data type a difference of one or two updates per second may not be noticeable by the user, for systems that seek optimum machine performance with scarce resources, the possibility provided can be of great importance.

REFERENCES

1. FUNKHOUSER, Thomas A. Network Topologies for Scalable Multi-user Virtual Environments. In: IEEE VRAIS, 1996.
2. FUNKHOUSER, Thomas A. RING: A Client-Server System for Multi-user Virtual Environments. In: ACM SIGGRAPH Symposium on Interactive 3D Graphics, 1995. p. 85 – 92.
3. KAZMAN, Rick. Making WAVES: On the Design of Architectures for Low-end Distributed Virtual Environments. In: Proceedings of IEEE Virtual Reality Annual International Symposium, 1993. p. 443 – 449.
4. MACEDONIA, Michael R. and ZYDA, Michael. A Taxonomy for Networked Virtual Environments. In: IEEE MultiMedia, Vol. 4, No. 1, 1997. p. 48 – 56.
5. MCCANNE, Steven; JACOBSON, Van; and VETTERLI, Martin. Receiver-driven Layered Multicast. In: Proc. ACM SIGCOMM, 1996. p. 117 – 130.
6. SHIRMOHAMMADI Shervin and GEORGANAS, Nicolas D. An end-to-end communication architecture for collaborative virtual environments. In: Computer Networks, No. 35, 2001. p. 351 – 367.
7. TREFFTZ, Helmuth; MARSIC, Ivan; and ZYDA, Michael. Handling heterogeneity in networked virtual environments. In: Presence: Teleoperators and Virtual Environments, Vol. 12, No. 1, 2003. p. 37 – 51.
8. TREFFTZ, Helmuth. System-wide Constraints and User Preferences in Collaborative Virtual Environments. Doctoral dissertation, Rutgers University, New Jersey, 2002.
9. ZYDA, Michael and SINGHAL, Sandeep. Networked Virtual Environments: Design and Implementation. New York: Addison Wesley, 1999.