Adding Scalability to Transport Level Multicast[1]

Markus Hofmann
Institute of Telematics, University of Karlsruhe
Zirkel 2, 76128 Karlsruhe, Germany
Phone: +49 721 6083982, Fax: +49 721 388097
E-Mail: m.hofmann@ieee.org

Abstract: Driven by the capabilities of modern high speed networks, a new generation of distributed systems is emerging, that will be able to support a wide range of applications in distributed environments. Such systems are expected to consist of a large number of geographically dispersed components interacting via communication networks. Therefore, forthcoming communication systems must scale well in respect to both, number of users and number of connections. Powerful and scalable control mechanisms are essential to overcome the implosion problem and to realize efficient error recovery. This paper presents a new control scheme suitable for multicast communication in large-scale groups and describes its integration in a well-known transport protocol.

Keywords: Reliable Multicast, Implosion Control, Local Groups, Large-Scale Networks, Dynamic Groups

1. Introduction

The availability of high speed networks and recent enhancements concerning processing capacity has stirred excitement among the computerized community. Emerging networks and communication services make the support of superior distributed applications technically feasible, while increasing demand on communication in distributed environments has made it necessary. Some of these new applications, such as collaborative work, video-conferencing, or information dissemination, need to distribute user data among multiple participants. At the same time, widespread availability of IP multicast [6] and development of applications deployed in the MBone [11] have considerably increased the geographic extent and the size of communication groups. Extensive use of services like Internet radio or large-scale conferencing leads to thousands of receivers being involved in a single multicast communication. In addition, communication participants may be spread all over the world. As the size and the geographic span of communication groups increases, efficient connection management schemes including scalable error and traffic control become more and more essential. These protocol elements are placed in the transport layer of a protocol stack according to the Internet protocol architecture [3]. Recent research projects deployed new transport level protocols to meet the requirements of large-scale multicast applications in heterogeneous networks [12], [7], [10]. Some proposals only address particular aspects of group communication and focus only on some specific user environments. However, multimedia applications often have to handle several highly diverse data streams at the same time. A system supporting distributed cooperative work, for example, might offer shared use of a text editor while simultaneously providing audio and video communication between all the participants. Such applications require multipurpose and flexible communication

1 In: Proceedings of *Third COST 237 Workshop - Multimedia Telecommunications and Applications*, Barcelona, Spain, 25.-27. November 1996

subsystems. On the other hand, other proposals suffer from a high degree of complexity and are too general in some issues.

This paper outlines a solution to the problem of multicasting to large groups based on the *Local Group Concept (LGC)* [8]. Section 2 discusses the problems specific to the envisaged scenario and presents a brief summary of possible solutions. Section 3 and 4 introduce the Local Group Concept and a new communication service supporting the establishment and reconfiguration of local groups. Section 5 describes the integration of LGC into the Xpress Transport Protocol (XTP) [13], while Section 6 presents some results of a performance evaluation. Finally, Section 7 concludes the paper.

2. Control Schemes for Multicast Communication

Multicast capable networks provide efficient and scalable routing of data packets to multiple receivers. However, the bearer service provided by these communication networks does not fit the requirements of some applications. IP multicast, for example, offers an unreliable datagram service. The provision of full-reliable data transfer requires additional mechanisms in higher level communication protocols. Some kind of interaction between sender and receivers is necessary to ensure correct data delivery as well as to perform any kind of congestion or flow control. Neither of the hosts has enough information to control data streams on its own. The provision of full-reliable data transfer, for example, is based on a comparison between sent and received data. The transmitter has knowledge about which data units have been sent and the receivers about which data units have been received successfully. Therefore, the provision of a full-reliable communication service requires the transmission of receiver status back to the sender or vice versa.

So called sender-based schemes, in which the transmitter is responsible for controlling data transfer, rely on collecting status information at the sending site. Receivers transmit control units, including acknowledgments and traffic control information, back to the sender. As the number of receivers becomes very large, the multicast sender is overwhelmed with return messages of its receivers. This fact is known as *sender implosion.* The effect of implosion is twofold. Firstly, the large number of return messages results in processing overhead at the sender and, therefore, delays data transfer. Secondly, an enormous amount of control units may cause an excess of both, bandwidth and bufferspace, which in turn causes additional message losses. An optimal control procedure would reduce the number of status reports received by the sender down to one.

Other approaches use receiver-based schemes, where receivers themselves are responsible for error detection and error recovery. They need not to return status reports or acknowledgments to the transmitter. These schemes are not suitable to provide a full-reliable communication service, because the failure and dropping out of a single receiver could not be detected by the transmitter. Furthermore, flow or rate control always require some kind of status exchange between sender and receivers. This will lead to sender implosion even in the case of receiver-based reliable multicast communication.

Another issue of great importance is the development of efficient error correction schemes. Several procedures have been developed. Forward error correction (FEC), for example, has been proposed to be suitable for real-time applications since it allows error recovery without adding any delay associated with retransmissions. However, FEC does not prevent sender implosion caused by messages necessary for traffic control. Furthermore, adding redundant control information wastes bandwidth even when no or just a few errors occur. The sender must add enough control information to enable correction of all errors. While receivers in heterogeneous networks have highly diverse error characteristics, it is not adequate to choose a single fixed level of redundant coding. Such a fixed coding level may be excessive for some receivers while it may be insufficient for others. Other schemes use retransmissions to close the gaps in the data stream of the receives. Most of these protocols use Go-Back-N or selective repeat to retransmit lost and corrupted data. Receivers do request missed data directly from the transmitter without any consideration of network topology and current network load. In the case of group communication, it is also possible to exchange data with other receivers. It is preferable to request lost and corrupted data from a group member placed next to the host which is missing some information. An optimal error correction scheme would stimulate retransmissions of missed data units by the receiver located closest to the failing host. This would minimize transfer delay and network load.

According to the intention of the developers, several multicast protocols using different procedures have been designed. Some protocols, such as MTP (Multicast Transport Protocol) [1] and RMP (Reliable Multicast Protocol) [14] realize a centralized control scheme to provide a totally ordered multicast delivery. The central component controlling data transfer may become a bottleneck when dealing with large communication groups. Therefore, these protocols will not scale well with the number of receivers. The first protocol defining some kind of implosion control was the Xpress Transfer Protocol (XTP) [5], [13]. It defines two heuristics called *damping* and *slotting*. These algorithms suppress redundant control messages by multicasting return messages to the whole group. Recently, SRM (Scalable Reliable Multicast) [7] enhances the original XTP mechanisms and adapts them to the Internet environment. It implements receiver-based error control and is, therefore, not able to provide a full-reliable service. To avoid a flood of retransmission requests, SRM uses timers carefully set and adjusted to the current network load. However, the correct setting of timers will be no easy task for highly dynamic networks with quickly changing load and frequently changing network structure.

The implosion problem could also be solved by defining so called local groups. A new concept has been defined using a hierarchy of local groups to relieve the multicast transmitter of the responsibility for dealing with all status messages from individual receivers. This concept and some related work will be introduced in the next section.

3. Implosion Control Based on Local Groups

To avoid sender implosion and to realize a more efficient error recovery, the *Local Group Concept (LGC)* [8] splits the global communication group into separate subgroups. These subgroups will combine communication participants within a local region, forming so called *Local Groups*. Each of them is represented by a *local Group Controller* that collects status information from the members of its local group. The local Group Controller evaluates these return messages, combines them into a single control packet and transmits it back to the multicast sender. Local Group Controllers also support the provision of local retransmissions and local message processing. They are able to perform and to coordinate retransmissions of lost and corrupted data within their subgroup. This reduces delay caused by retransmissions across wide area links and decreases the load for sender and network. The integration of message processing capabilities into local Group Controllers reduces the implosion problem of multicast traffic and error control for large groups. Local Group Controllers evaluate received control units and inform the multicast sender about the status of the local group. This includes error reports as well as parameters to control data flow. Parallel processing of status reports and their combination into a single message per local group relieves the multicast sender as it reduces the number of control units to be evaluated at the sending side.

In each local region, one of the receivers is determined to function as local Group Controller. The dedicated system has to collect control messages from all the members of its subgroup and has to forward them to the multicast sender in a single composite control unit. Controllers of subgroups are also responsible for organizing local retransmissions. After evaluating received status messages a local Group Controller tries to transfer lost data units to all the receivers that have observed errors or losses. To retransmit data units a local Group Controller can use either unicast or restricted multicast transmission. This decision may be static or dynamically based on the number of failed receivers. If a controller itself misses some data units, it will try to get them from another member of its local group. Therefore, a multicast sender has only to retransmit messages missed by all members of a subgroup. Local retransmissions lead to shorter delays and decrease the number of data units flowing through a global network.

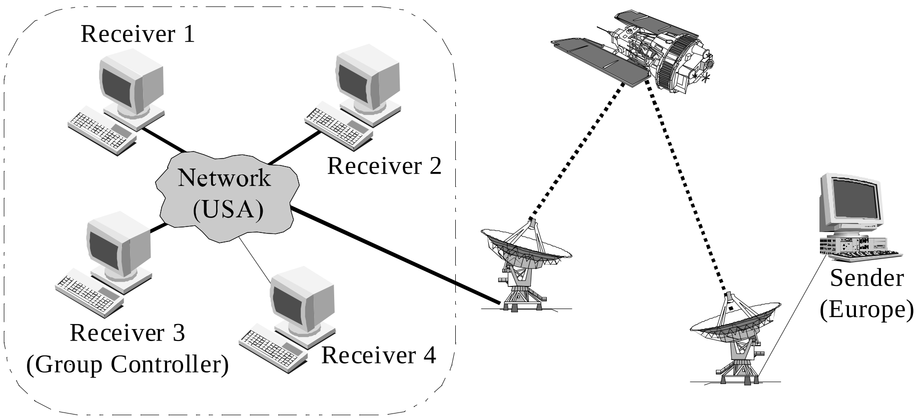

Fig. 1. Example for the Definition of Local Groups

An example scenario illustrating the basic idea and the advantage of this concept is given in Figure 1. A multicast sender located in Europe communicates over a satellite link with four receivers, which are spread over the USA. The satellite link is characterized by high transfer delay and high carrier fees. Therefore, it is desirable to reduce data traffic over this link. In this type of scenario it is useful to combine all four receivers into a single subgroup. One of the receiving hosts has to function as the controller of the subgroup. In this case, local retransmissions do not traverse the transatlantic link. This reduces transfer delay and global network load.

The resulting data streams are shown in Figure 2. The transmitter multicasts data units directly to all group members using a multicast capable delivery service (1). The local Group Controllers are kept out of the outgoing data path avoiding an extra handling of data units at each level of the hierarchy. Therefore, local Group Controllers need not to store data fragments, reassemble complete data units, interpret and forward them. Instead, multicast forwarding is done by the delivery service in a more efficient way. After receiving a status request, regular receivers transmit control messages to their corresponding local Group Controller (2). The controller combines the status reports into a single control unit and sends it to the multicast transmitter (3). Therefore, it could be said that the Local Group Concept causes some kind of triangulation of data flow.

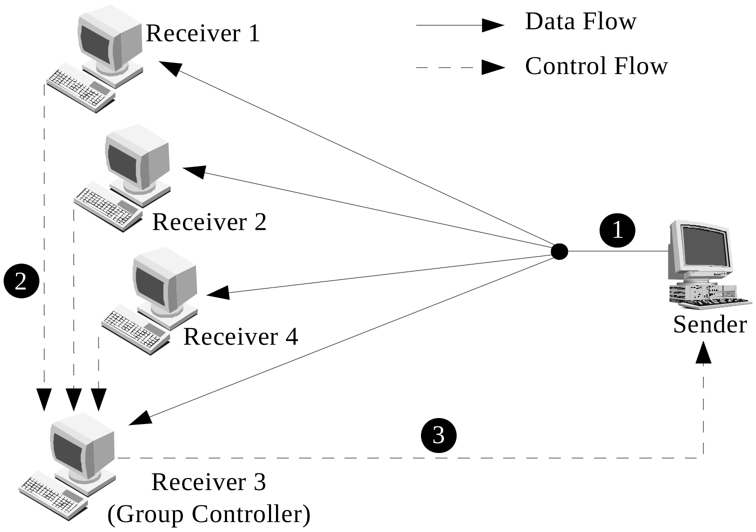

Fig. 2. Triangulation of Data Flow

The Local Group Concept also includes mechanisms to increase fault tolerance [8]. The Fail Stop of a local Group Controller, for example, is handled by dynamic reformation of Local Groups. Of course, actions performed after the failure of a communication participant depend on the given group semantic. If the communication user requires a full-reliable service, the multicast connection will be closed due to the failure of a group member. In the case of semi-reliable service, the actions to be performed depend on the attributes and the role of the failed receiver. Dynamic group reformation is also used to adapt the group structure to the current network load and to the current state of all communication participants. For further improvements in respect to transfer delay and implosion, local groups are organized in a hierarchical structure.

In the given example, the decision to combine all four receivers into one single subgroup was based on the intention of minimizing transfer delay. In other scenarios, it may be appropriate to minimize other parameters or to use a combination of several metrics. The next section discusses several aspects of structuring global communication groups.

4. GDS - A Service to Establish Local Groups

The efficiency of a concept like LGC is mainly influenced by the decision where to place the functionality of local Group Controllers. It can be integrated in several types of communication nodes, for example network switching entities, dedicated servers or regular hosts. The choice of a site for a local Group Controller will be determined by several criterion. It will be influenced by the network topology, the current network load, the capabilities of all the communication nodes and by the application specific requirements.

4.1. Placement of Local Group Controllers

The integration of controller functionality in network switching entities (e.g. switches, gateways, routers) seems to be very attractive. It allows buffering of data packets and gathering of status information along the path between transmitter and receivers. Controllers placed within network switching entities will always be located upstream in relation to the receivers. This is desirable, because downstream controllers will miss the same packets as receivers located closer to the transmitter. On the other hand, this approach will not scale well with the number of multicast connections. Additional per-connection processing and buffering is required and additional state information has to be managed at the switching entities. Controlling data flow requires examination and evaluation of individual packets. Therefore, switching entities must reassemble and evaluate packet headers and control messages before forwarding them. This increases transfer delay, especially in cell-oriented networks such as ATM. As the number of multicast connections increases, a switching entity coordinating retransmissions, processing acknowledgments and forwarding packets for several multicast connections will soon become a bottleneck. It is preferable to parallelize the handling of different multicast connections and to distribute it to several communication nodes. The CP protocol mentioned in [12], for example, uses local exchanges to combine status messages of receivers within a local region. This implies modifications of internal network switching entities. The entities have to examine and interpret the protocol control information included in data units, meaning that part of the protocol must reside on these systems. For that reason, it is difficult to update the protocol and to introduce a new protocol version.

Another solution would be the establishment of dedicated servers to coordinate multicast communication. However, this requires installation and maintenance of additional network components. The static and centralized character of this approach causes additional problems in highly dynamic networks and communication groups.

For the reasons listed above, LGC integrates the functionality of local Group Controllers into regular hosts. In principle, every communication participant is able to perform the tasks necessary for controlling a subgroup. One member per local group is designated as the active controller. The selected station represents the local group and performs local acknowledgment processing and local retransmissions. Over and above that, the flexibility of this solution permits to realize the approaches described above. If an intermediate system or a dedicated server subscribes to the multicast group, it will be able to take the role of a local Group Controller. Because the placement of local Group Controllers is dynamic in LGC, it is necessary for joining receivers to detect existing Local Groups and to chose an appropriate one. The selection of the "best-fit" local group is supported by a new service, which is introduced in the following paragraph.

4.2. Selection of Local Group Controllers

The establishment of local groups is done during connection setup. Before receiving any data, new communication participants have to subscribe to a nearby local group in respect to a chosen metric. The suitability of a certain metric, such as delay, bandwidth, throughput, error probability, reliability, carrier fees, or number of hops between two nodes, depends mainly on the application using a communication service. While an interactive application may wish to minimize transfer delay, a user transferring files is interested in reducing the financial cost of a transfer. Of course, it could also be suitable to combine several metrics and to weight them according to the intention of the service user. The establishment of local groups is also influenced by the structure and the type of a network used for data exchange. In wide-area networks or interconnected LAN/WAN environments, it is suitable to divide global communication groups into several clusters depending on the location of each host. In local environments, it is more convenient to establish logical subgroups according to the capacity of each communication participant. Inefficient receivers, for example, are associated with different subgroups to avoid asymmetrical and unfair load of Group Controllers.

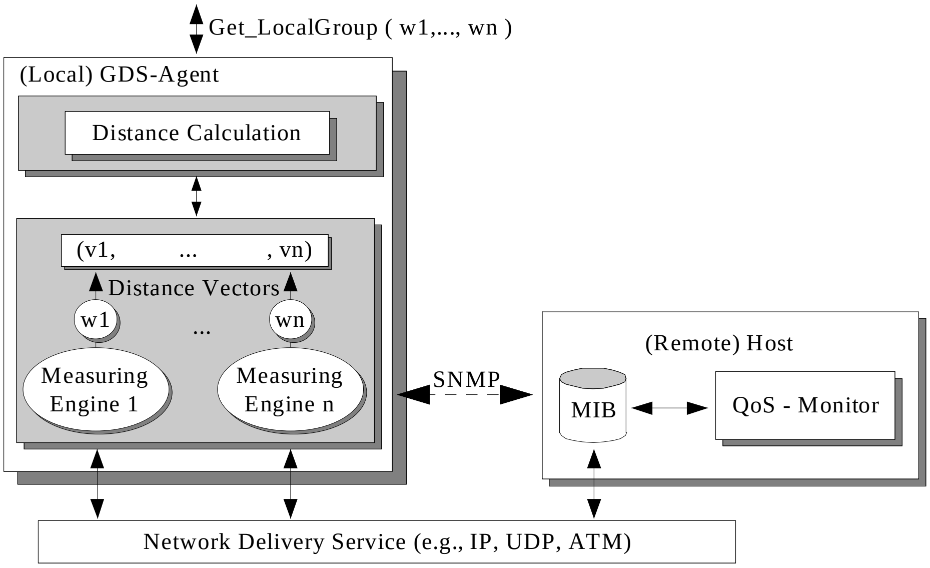

Fig. 3. GDS Architecture

The establishment of local groups is supported by a novel communication service, named *Group Distance Service (GDS)*. It provides mechanisms to find a nearby local group for each joining communication participant. Therefore, every host supporting the Local Group Concept incorporates a so called GDS Agent. As shown in Figure 3, a GDS Agent consists of several measuring engines. Each of them gauges the distance $d(i)$ to nearby located local Group Controllers based on a certain metric. One engine, for example, computes the number of hops between two hosts by implementing an expanded ring search. Another engine determines the transfer delay between two hosts based on the measured round trip time. Other engines may use SNMP (Simple Network Management Protocol) [4] to request the (financial) cost for data transfer across a certain link.

As illustrated in Figure 3, a search engine has been defined accessing the Management Information Base (MIB) of remote QoS Monitors. The Quality-of-Service (QoS) monitors store all QoS-related information (e.g. transfer delay, throughput) in this MIB [9]. A GDS agent may query the remote MIB via SNMP and use the information obtained from the remote QoS Monitor to select an appropriate local Group Controller.

Each measured value $d(i)$ is multiplied with a metric-specific weight $w(i)$ that has been defined by the service user according to its requirements. A user interested in minimizing transfer delay, for example, weights transfer delay relatively high compared to other metrics. To consider only one single metric $m(i)$, all weights are assigned a value of "0" with the exception of weight $w(i)$. The weighted values define a n-dimensional distance vector $v = (v(1), \ldots , v(n))$, where $v(i) = d(i) * w(i)$. The GDS Agent computes such a distance vector for each local Group Controller in its neighborhood. Therefore, it assigns each local Group Controller a member of a n-dimensional codomain. The joining receiver is assigned the zero vector. The overall distance between the new receiver and each local Group Controller is now defined to be the euclidean distance between their corresponding distance vectors.

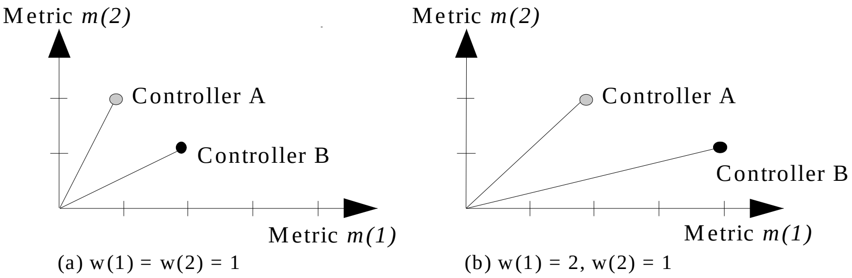

Fig.4. Impact of Different Weights on Distance Vectors

An example illustrating the dependency between the weight of metrics and the overall distance is given in Figure 4. The distance of Controller A has been computed to be 1 based on metric $m(1)$ and to be 2 in respect to metric $m(2)$. The distance of Controller B has been calculated to be (2,1).

If the user assigns the same weight $w(1) = w(2) = 1$ to both metrics, the distance of both controllers will be fixed to the same value (see Figure 4a). If the user prefers minimizing metric $m(1)$, it may assign the weights $w(1) = 2$ and $w(2) = 1$. This results in a smaller distance for Controller A, as illustrated in Figure 4b.

The GDS Agent chooses the local Group Controller "closest" to the joining receiver and returns its identity. If there is no suitable local Group Controller, the GDS user will get the information to establish a new Local Group and to appoint itself as its controller.

5. Adding Scalability to XTP Multicast

The generic character of the Local Group Concept allows an easy integration of the concept into existing multicast protocols without the necessity of substantial modifications. Therefore, LGC could be used to improve the scalability of well-known communication protocols. It is not necessary to introduce new protocols nor to establish any additional network components. This section proposes some extensions to the *Xpress Transport Protocol (XTP)* [13] to enhance its scalability based on the Local Group Concept.

XTP defines a set of protocol mechanisms whose functionality is orthogonal to one another. The intention of its design has been the provision of some basic functionality, from which the user can compose a service according to its requirements. Support for multicast communication and group management has been part of XTP since revision 3.6. First ideas for minimizing control traffic from the receivers resulted in damping and slotting algorithms to avoid implosion. Instead of returning control units exclusively to the multicast transmitter, receivers send them after a random delay to the whole group. Therefore, every group member receives this message and skips its status report if the incoming control unit corresponds to its personal state. This mechanism might reduce the number of acknowledgments to be processed by the sender. But in large-scale global networks, the usefulness of multicasting control units is very questionable. A large number of receivers together with this mechanism may cause a flood of control units all over the global network. In addition, the suppression of status messages from the receivers prevents the support of a full-reliable data transfer. In revision 4.0 [13], the mechanisms were totally revised. The suppression of status messages was removed and procedures to uniquely identify each communication participant were introduced. An addendum to the XTP 4.0 Specification [2] introduces modified multicast mechanisms to allow a more efficient mapping between control units and the corresponding communication participants. However, as the number of receivers grows larger, the sender will soon be overwhelmed with return messages of the receivers. To overcome the implosion problem and to improve the efficiency of error recovery, the Local Group Concept described in this paper has been integrated into XTP. There are some minor changes necessary, while the principle design of XTP is kept intact.

The notation used in this paper follows the definitions given in [2]. The abbreviations used are:

Kg	transmitter's local key for the multicast group
Kg'	transmitter's local key for the multicast group as a return key
Kr	receiver's local key
Kr'	receiver's local key as a return key
Ki	key assigned by the transmitter to uniquely identify receiver i
Ki'	the same as a return key
TAg(m)	multicast transport address of the group

TAt(u)	unicast transport address of the transmitter
TAr(u)	unicast transport address of the receiver
DA	destination address field in an address segment
SA	source address field in an address segment
Ag(m)	multicast delivery service address of the group (e.g. IP multicast address)
At(u)	unicast delivery service address of the transmitter (e.g. IP address)
Ar(u)	unicast delivery service address of the receiver (e.g. IP address)
dest	destination address used by the delivery service
src	source address used by the delivery service

First of all, a joining receiver has to determine an appropriate local group. It will get the identity of the local Group Controller either by using the Group Distance Service or manually from the service user. If no appropriate local group is found, the new receiver appoints itself as controller of a new group. In this case, it joins the association by multicasting a JCNTL packet according the XTP Specification. The packet sequence and the procedures will be the same as for regular receiver-initiated multicast as defined in [2]. Otherwise, the joining receiver registers itself at the chosen local Group Controller by sending a LCNTL packet to the controller. The format and the function of a LCNTL packet is very similar to the JCNTL packet defined in XTP. The difference between these two packet types enables a Group Controller to distinguish between control units used to register a receiver at the local group and a JCNTL used to register a host at the multicast transmitter. The LCNTL packet is assigned packet type 9 and it has the layout illustrated in Figure 5. It is derived from the JCNTL packet by adding a lookup segment.

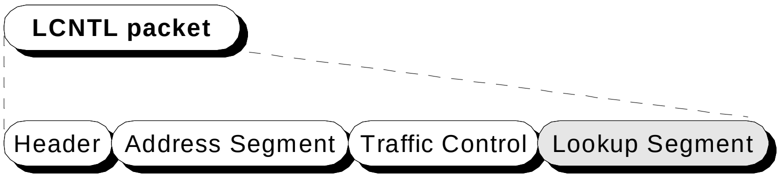

Fig. 5. Format of a LCNTL Packet

As defined in XTP, mapping incoming packets onto an appropriate context requires either a return key ("abbreviated context lookup") or the regular key and the source address of the sender ("full context lookup"). Because data packets in LGC are multicasted by the transmitter, new receivers have to learn the multicast key and the address of the multicast transmitter. Local Group Controllers include this information in the LCNTL answer in response to a register request. It is stored in the Lookup Segment of the LCNTL. This segment contains all the information necessary for a receiver to associate multicast packets from the transmitter with the correct local context. The Lookup Segment format is shown in Figure 6.

Instead of defining a single lookup scheme for a certain delivery service, LGC provides parametric lookup information where one of several formats can be used. The *lkey* (lookup key) field contains the key associated with the multicast context, the transmitter's local key for the whole multicast group. Therefore, the field is 8 bytes wide. The *lformat* (lookup information format) field is 1 byte wide and

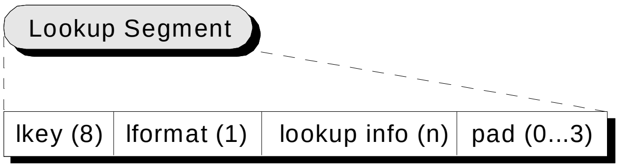

Fig. 6. Fields of the Lookup Segment

identifies the format of the lookup information included in the *lookup info* field. The length of the lookup info field depends on the chosen lookup information format. At the moment, two formats according to Table 1 are defined.

Lformat		***Lookup Information***
Decimal	*Hex*	
0	0x00	empty
1	0x01	delivery service address of transmitter (IP Format)

Table 1 - Lookup Information Formats

When a receiver wishes to join an existing multicast group, the packet sequence is as follows. First, the receiver sends a LCNTL packet to its local Group Controller to register itself and to learn about the multicast key and the unicast address of the multicast sender. The receiver needs this information to associate multicast packets from the multicast sender with the appropriate context. The LCNTL sent by the joining receiver includes the following fields:

```
dest   =  Ac(u)      // delivery service address of controller
src    =  Ar(u)      // delivery service address of receiver
key    =  0          // defined by XTP
alloc  =  n          // current window size for the receiver
xkey   =  Kr'        // receiver's local key as a return key
DA     =  TAg(m)     // transport address of multicast group
SA     =  TAr(u)     // transport address of receiver
Tspec  =  spec(r)    // traffic specifier offered by the receiver
```

This LCNTL must have the SREQ set according to the XTP semantic. The local Group Controller checks the (local) join request and the Tspec included in the packet. If the request is denied for some reason, it will send a DIAG packet. Otherwise, it will transmit a LCNTL with the following fields:

```
dest   =  Ar(u)      // delivery service address of receiver
src    =  Ac(u)      // delivery service address of controller
key    =  Kr'        // receiver's local key as a return key
xkey   =  Ki'        // key assigned by the controller to uniquely identify
                     // the receiver (as a return key)
DA     =  TAr(u)     // transport address of receiver
SA     =  TAc(u)     // transport address of controller
Tspec  =  spec(m)    // current traffic specifier for multicast group
```

linf	=	[At(u), Kg]	// lookup information necessary to associate // multicast packets with the multicast context

The lookup segment of the packet includes all the information necessary to perform a full context lookup for multicast data packets and to map them onto the appropriate context.

After the sequence has completed, the local Group Controller knows the key (Kr') to be used when sending control messages to the new receiver, and the transport level address (TAr(u)) for this receiver. Therefore, the local Group Controller is able to uniquely identify the new member of the group. Until now, the transmitter has no knowledge about the identity of the new receiver. It is up to the local Group Controller to ensure that all the members remain active, and that everyone gets all of the data. The local Group Controller is able to detect the failure of a receiver and it is able to ensure correct data delivery to all of its members. In the case of an error (e.g., a lost packet or a failed receiver), the local Group Controller will operate according to the given reliability class. If the reliability class of the association is "full-reliable", for example, the local Group Controller will abort the association. Otherwise, it may drop the failing host. Data flow towards single receivers can be controlled by defining "roles" or "attributes" for each receiver. The local Group Controller will control data flow according to these attributes. For example, return messages from a receiver with the attribute NOFLOW will not be used to calculate flow control parameters for the local group.

In some situations, it may be wishful to let the transmitter know the identity of all receivers. Therefore, the transmitter should be able to learn about all the group members if it so wishes. The need to identify each single receiver is indicated by the selection of a specific reliability class (RCLASS). The reliability class is part of the traffic specifier as defined in [13]. It is included in JCNTLs, LCNTLs, and TCNTLs. We have introduced an additional reliability class (RCLASS 5) indicating the need to uniquely identify all communication participants. If the transmitter has chosen this reliability class for a certain association, it will get the identity of all receivers from the local Group Controllers. An additional key exchange is not necessary, because the transmitter will never get back control messages directly from the members of a local group.

During data transfer, every multicast packet sent from the transmitter to the whole group contains the multicast key and will be addressed to the multicast group:

dest	=	Ag(m)	// multicast delivery service address of the group // (e.g. IP multicast address)
src	=	At(u)	// network address of transmitter
key	=	Kg	// transmitter's local key for the multicast group

The transmitter cannot use a return key, because receivers will use different keys to identify their local context. Therefore, targets have always to perform a full context lookup to map the packets onto the appropriate context.

Control messages from receivers to their corresponding local Group Controller are sent using the return key assigned by the local Group Controller during the join progress:

dest	=	Ac(u)	// network address of controller
src	=	Ar(u)	// network address of receiver
key	=	Ki'	// key assigned by the controller to uniquely identify // the receiver (as a return key)

Because targets use a return key, the local Group Controller can use abbreviated context lookup to map to the appropriate context.

Communication between local Group Controllers and the transmitter follows the regular XTP specification.

To request missed or corrupted data packets from members of the local group, a controller sends an ECNTL packet directly to one of the targets that have successfully received the packet. A group member receiving an ECNTL evaluates the *span* fields and transmits the requested data packets directly to the local Group Controller.

6. Performance Evaluation

Analytical methods have been applied and simulations have been performed in order to evaluate the benefits of the Local Group Concept. The simulations have been based on BONeS/ Designer, an event-driven network simulation tool by Comdisco. The simulation scenario consists of a sender and various receivers which are connected to a network (similar to the scenario in Figure 1). The sender is linked to the network across a backbone link. A transfer delay of 20 ms for the backbone link was assumed. The transfer delay within the network was assumed to be 2 ms. Both values were obtained from *ping* measurements in a real environment. The error probability was set to 10^{-1}, which is not uncommon for highly loaded internetworks. Other simulation parameters were data rate, processing delay, status request rate and burst length. Packet losses were simulated using a Markov-Model. For traffic generation, a constant data source has been used.

The evaluation examined the impact of group size on average transfer delay and network load. The values obtained for the Local Group Concept have been compared to the corresponding results for a common sender-based technique using multicast retransmission. The results show significant improvements for the new concept compared to common multicast techniques (Figure 7). More detailed information concerning the simulation scenarios, the simulation parameters and the simulation results can be found in [8].

Another interesting feature of the Local Group Concept is the partial independence between network load and number of receivers. Analytical methods have been applied to a scenario similar to that one described above. The number of receivers varied from 1 up to 1000. We assumed 50 additional receivers at the sending side of the wide area link. For the common, sender-based technique, the ratio of retransmissions and total data traffic over the wide area link depends directly on the number of recipients. The Local Group Concept, in contrast, is not influenced by the number of participants and shows very low values compared with the results of the other techniques [8].

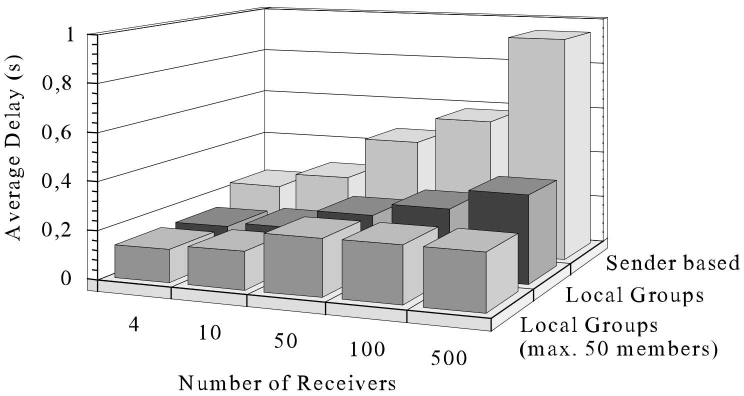

Fig. 7. Impact of Group Size on Transfer Delay

The independence between network load and the number of receivers simplifies resource reservation for highly dynamic communication groups. The bandwidth required on certain links can be reserved regardless of the current or the future group size. This simplifies the handling of multimedia data flows.

7. Conclusion

Common multicast protocols are a significant improvement over simple point-to-point protocols. However, most of them are not suitable for the case where the transmitter has to handle data flow to a large number of receivers. To avoid sender implosion and to increase efficiency of error recovery, the Local Group Concept (LGC) has been introduced. It aims for the provision of a scalable, retransmission-based error recovery in heterogeneous networks. The establishment of local groups is supported by the Group Distance Service that permits an application depended evaluation of several metrics. Performance evaluation including simulation and analysis of network load shows for the Local Group Concept significant improvements compared to common multicast control schemes. The benefits are achieved without the necessity to modify internal network equipment such as ATM switches or IP routers.

At the moment, work is going on to implement a GDS Agent on top of IP. It will be part of a complex test-suite for group communication in the Internet. The test-suite will be used to investigate the dependency between group structure (including placement of local Group Controllers) and communication performance. Furthermore, simulations of more complex scenarios will be finished and evaluated soon.

8. Acknowledgments

The author would like to thank J. William Atwood from Concordia University Montreal for valuable discussions. Special thanks to Stoyan Kenderov and other members of the LGC Working Group for various suggestions concerning the simulation models and the design of the Group Distance Service.

References

[1] S. Armstrong, A. Freier, K. Marzullo: *Multicast Transport Protocol.* RFC 1301, February 1992.

[2] J. W. Atwood, O. Catrina, J. Fenton, W. T. Strayer: *Reliable Multicasting in the Xpress Transport Protocol.* Proceedings of the 21st Local Computer Networks Conference, Minneapolis, MN, October 1996.

[3] B. Carpenter: *Architectural Principles of the Internet*. RFC 1958, June 1996.

[4] J. Case, M. Fedor, M. Schoffstall, C. Davin: *Simple Network Management Protocol (SNMP).* Network Working Group, RFC 1157, May 1990.

[5] G. Chesson, ed.: *Xpress Transfer Protocol Specification, Revision 3.6.* Protocol Engines, Inc., Available from XTP Forum, Santa Barbara, 1993.

[6] S. Deering, D. Cheriton: *Multicast Routing in Datagram Internetworks and Extended LANs.* ACM Transactions on Computer Systems, 8(2):85-110, May 1990.

[7] S. Floyd, V. Jacobson, S. McCanne, C. Liu, L. Zhang: *A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing.* Computer Communication Review, Vol. 25, No. 4, Proceedings of ACM SIGCOMM'95, August 1995.

[8] M. Hofmann: *A Generic Concept for Large-Scale Multicast*. In: Broadband Communications (ed.: B. Plattner), Lecture Notes in Computer Science, No. 1044, Proceedings of 1996 International Zurich Seminar on Digital Communications, Springer Verlag, February 1996.

[9] M. Hofmann, C. Schmidt: *A Flexible Protocol Architecture for Multimedia Communication in ATM-Based Networks*. Proceedings of 18th Biennial Symposium on Communications, Kingston, ON, Canada, June 1996.

[10] H. Holbrook, S. Singhal, D. Cheriton: *Log-Based Receiver-Reliable Multicast for Distributed Interactive Simulation.* Computer Communication Review, Vol. 25, No. 4, Proceedings of ACM SIGCOMM'95, August 1995.

[11] V. Kumar: *Mbone: Interactive Multimedia on the Internet*. New Riders Publishing, Indianapolis, Indiana, USA, 1995.

[12] S. Paul, K. Sabnani, D. Kristol: *Multicast Transport Protocols for High Speed Networks.* Proceedings of International Conference on Network Protocols, Boston, 1994.

[13] W.T. Strayer, ed.: *Xpress Transport Protocol Specification, Revision 4.0.* Available from XTP Forum, Santa Barbara, USA, March 1995.

[14] B. Whetten. T. Montgomery, S. Kaplan: *A High Performance Totally Ordered Multicast Protocol.* Submitted to INFOCOM'95, April 1995.