MULTIMEDIA CONFERENCE ACROSS WIDE-AREA NETWORKS

Jože Rugelj†, Adriano Endrizzi

Joint Research Centre of CEC
21020 Ispra, VA, Italy

†CEC-DG XII grant holder,
on leave from J. Stefan Institute
Ljubljana, Yugoslavia

ABSTRACT

The design of a multimedia interactive conferencing system based on wide-area networks as a communication infrastructure is discussed. Co-operation activities take place by sharing application packages installed in remote servers and workstations. The necessary upgrades to the existing communication protocols which make them support multicast connections are described in detail.

1 Introduction

The technical problems related to the distribution of the information in an electronic conference environment can best be understood by analogy with the traditional face-to-face conference. The speaker's voice and the visual information he is presenting are carried by the "ether" of the conference hall to the ears and eyes of the participants. The information is sent only once and is received simultaneously by all attendees. Such 'ether'-like capability is not to be found in the available computer-to-computer communications procedures. Although many local area networks and satellite networks utilize broadcast links as their basic communication media, the available higher level protocols deal with point-to-point communication only.

For this reason the designers of the electronic conference software systems adopt mixed solutions which see the participating workstations be connected through a mesh of point-to-point liaisons. Such liaisons are offered by the underlying transport and network services and the broadcast communication is carried out by the attached host computers that are charged with the task of the duplication and the retransmission of the data according to needs.

It is our belief that, in order to make the operation of real-time computer supported conferencing systems efficient, broadcast communication should be offered at the network and transport level. This is particularly true in case wide-area networks are involved. Indeed, since charges increase with the volume, the distance and the number of liaisons, the duplication and the retransmission of the same data should be avoided as much as

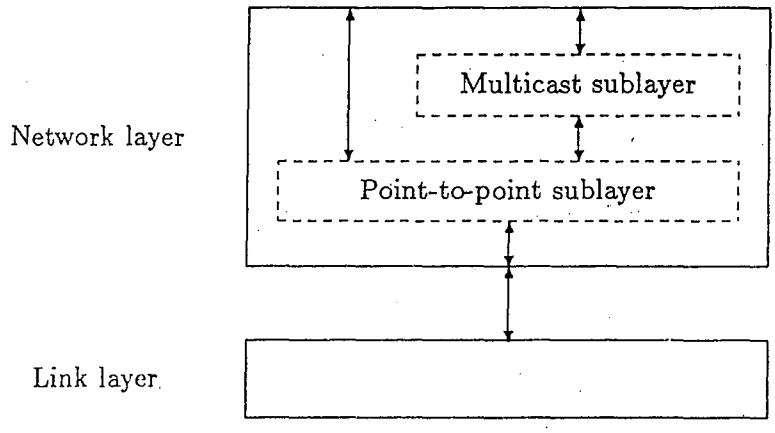

Figure 1: Multicast sublayer

possible. If adequate broadcast services at the network level would be available, important reductions in transmission delays would be achieved, too.

The arguments above strongly suggest the definition of enhancements to the existing communication protocols which make them support broadcast liaisons as well as point-to-point liaisons.

2 Supporting multicast communication

In order to support multicast liaisons across wide-area networks, new software modules need to be introduced in the nodes of the communication network. According to the ISO-OSI seven layer structure (fig. 1), those modules could be grouped into a sublayer in between the network and transport layer. Assuming that the network provides a basic X.25 point-to-point packet switching service, additional primitives are offered to the transport layer to deal with the multicast liaisons.

In order to cope with the rather complex problem of the synchronization of distributed processes, a special primitive is also introduced at this sublayer. The solution of the general synchronization and communication issues in a distributed environment is based on ideas and experiences gained with the DUAL network machine, designed and used at the Joint Research Centre of CEC Ispra [Endr87].

The list of advanced primitives includes:

- CREATE_MCAST_CONN(set_of_nodes): conID;
- DELETE_MCAST_CONN (conID);
- CREDIT (conID, value);
- TEST&SET (var): result;

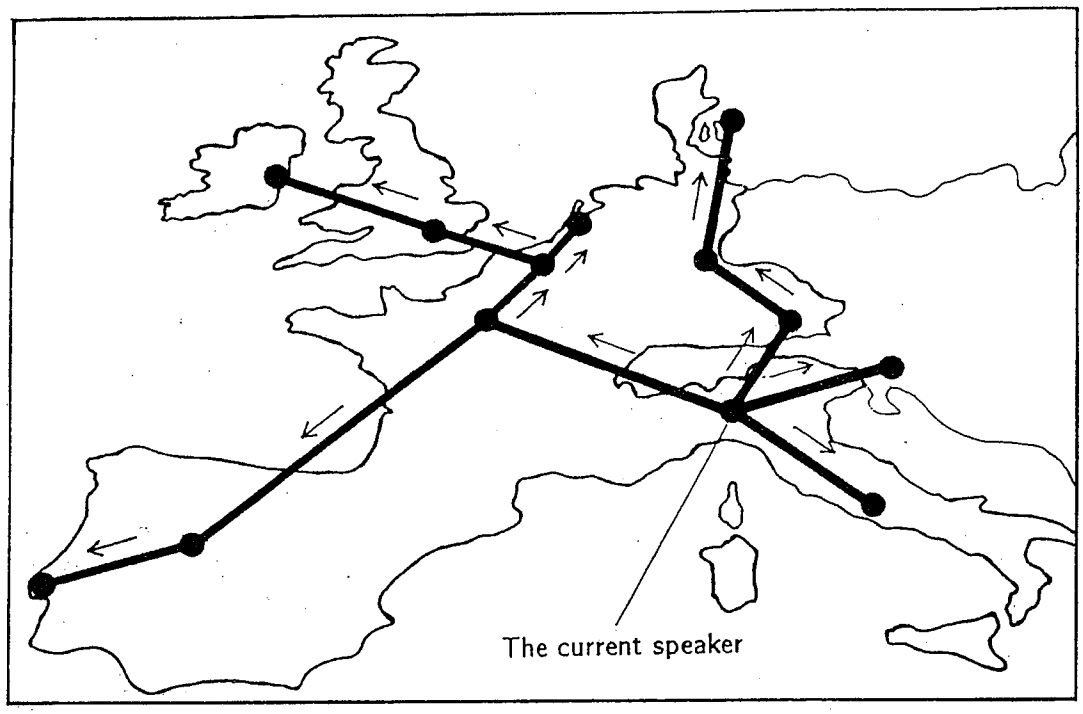

Figure 2: Packet flow in the multicast environment

Following the request for the creation of a multicast connection, a set of processes in the multicast sublayer of the specified nodes is gradually started. Those processes execute a distributed algorithm aimed at the construction of a minimum spanning tree of virtual connections connecting the specified nodes. The algorithm is started by the node to which the request for the multicast connection is issued. In each step of the algorithm, the closest node to the existing tree is selected from the set of the specified nodes, and a virtual connection is established between the selected node and the tree. The new node is then added to the tree.

The purpose of the minimum spanning tree algorithm is to connect n nodes by means of $n-1$ connections which are selected out of the $n(n-1)/2$ possible ones. The selection criteria takes into consideration the characteristics of the individual connections in such a way that the minimum of a predefined cost function can be achieved. Parameters influencing the cost function are for example the length, the load and the reliability of the individual connection. Some of those parameters are dynamic in nature. Their values are drawn from statistics about network utilization which take into account point-to-point traffic as well as multicast and other types of traffic.

All nodes that are already connected to the tree, collaborate in the execution of this algorithm. The algorithm stops when all the specified nodes become connected to the tree. The resulting tree constitutes the basic communication path on which the multicast transmission takes place. On the reception of an incoming data packet, a node makes copies of it and send them on the outgoing branches. Since the established tree is the minimum spanning tree, multicast is achieved with the minimum number of duplications and retransmissions. Transmissions delays are also reduced because a certain amount of parallelism can take place.

The request for the deletion of a multicast liaison is broadcast to all nodes involved with

it. When such a request is received, the virtual connections belonging to the supporting tree are cleared and all processes devoted to it are stopped.

In an established multicast liaison, the transmission of data units from one or more senders to the receivers must be accompanied by a consistent flow of credit units in the opposite direction. The purpose of this flow control mechanism is to limit the production rate of the senders to the level allowed by the consumption rate of the slowest receiver. This protects the network resources from the congestion caused by data units in excess which accumulate inside the network and cannot be delivered.

The envisaged flow control mechanism is such that the existence of multiple receivers is transparent to the senders. The main problem with the multicast flow control, when more than one sender is active, is the determination in each node of the amount of credits to be back-propagated to the senders.

At the time a multicast liaison is first established, a certain amount of credits *no_of_cred* is allocated in each node for each virtual connection to the other nodes. Assuming there are *no_of_send* senders, the network must be able to accept (*no_of_cred* * *no_of_send*) packets in advance. The arrival of a data packet on a virtual connection causes the corresponding *no_of_cred* to be reduced by one unit. On the other hand, the arrival of a credit message causes this number to be increased by one unit. Any node on the path to the sender may back-propagate the received credits to the other nodes only if the minimum *no_of_cred* increases. This makes the slowest receiver or the slowest part of the network regulate the transmission speed of the sender.

The *Test&Set* indivisible operation is not part of the data transmission services. It has been introduced in support of the synchronization mechanisms which are so essential to the higher level procedures of distributed processing. It has been included in this level of the architecture, because there is where it can be implemented most efficiently.

For each multicast liaison, a number of named variables are allocated, each variable representing a non-sharable resource. The *Test&Set* operation makes use of the same spanning tree that supports the multicast liaison. Whenever a *Test&Set* operation is issued, a broadcast wave is generated from the source node to the other nodes. If the broadcast wave can reach all other leaves of the tree without encountering a similar wave originating from another source, then the operation is considered successful and the associated resource can safely be given to the requesting application. In its way to the leaves, a successful wave locks the associated variable in all visited nodes. If an already locked node is encountered, this node will produce a negative wave whose purpose is to undo the effects of the original wave.

3 The model of the conference system

As reported in the relevant literature ([Stef87], [Saka88], [Tani88], [Bonf89]), the existing conference systems consist of networked workstations that are driven by a common "conference manager". The conference manager module runs on a predefined server node of the network and is responsible for all conference services including the broadcast of the I/O messages to all participants. The conference manager represents an instance of a more general concept which is referred to as "application sharing" ([Sari85]). This consists of making an application, which is usually conceived to serve one user at a time,

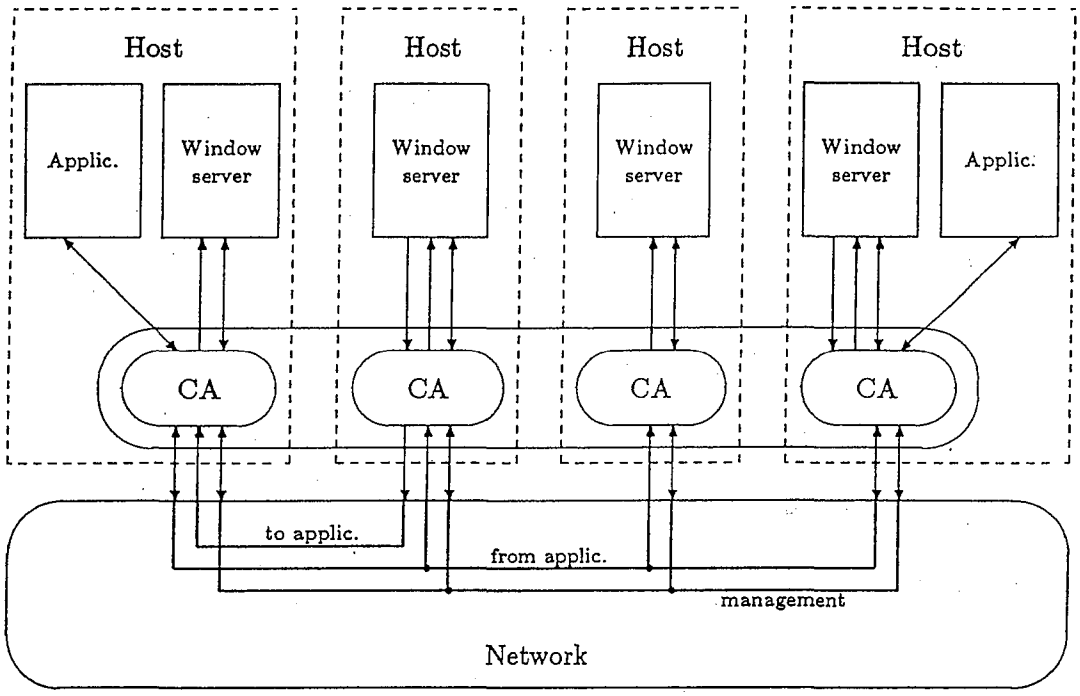

Figure 3: The model of the conferencing system

visible and accessible to other users. The process of application sharing clearly involves the multicast transmission of the I/O activities as well as the serialization of the accesses requested by the different users to the application itself. Our approach to the conference system and, more generally, to systems for the co-operative work is based on the concept of application sharing and, in particular, is focused on the clear separation of the I/O and synchronization aspects from the more application-oriented aspects. The idea is to provide each workstation and server in the network with a "conference agent" module. The distributed conference agent modules co-operate in the management of the communication between, on the one hand, the requested applications and, on the other hand, the end-users. With the introduction of the conference agents, existing application such as productivity tools, accesses to data bases and other familiar packages can easily be included in the conference environment.

The conference agent has three types of interfaces. The first interface exchanges messages and commands with the window manager of the workstation (i.e. X-Window, NeWS, ...) and the other multi-media I/O drivers. This allows, on the one hand, to capture the input from the active user and to multicast it to the other participants and, on the other hand, to direct the received data to the proper window or multimedia device. The second interface is devoted to the exchange of commands and results with the applications activated in support of the co-operative work. Commands originate from the currently active user who is usually located remotely from the corresponding application. The received results need to be broadcast to all participants. Finally, the conference agent has an interface with the local transport service that supports the necessary communication liaisons with all other conference agents across the available networks.

As shown in fig.3, there are three types of communication liaisons. The first type is represented by a point-to-point connection from "the current speaker" to the application.

The second is a multicast connection from the application to all participants. The third type of liaisons includes a multicast connection between the co-operating conference agents.

The participant to a conference is presented with a set of windows, one window per each shared application. Additional windows are needed for the activation of new applications, for the management of the conference itself, for the control of the microphones and loudspeakers, the shared blackboard and other multimedia devices.

The user who first starts the conference has special rights and duties. He acts as the chairman of the conference and, in particular, he defines the list of the participants, their access rights (i.e. speaker, observer) and the corresponding passwords.

As the result of a request for the setup of a new conference, the local conference agent contacts the conference agents of the envisaged remote partners and establishes with them the necessary multicast environment across the network.

Once the invitation to join the conference has been accepted by the participants, the chairman defines the minimum set of shared multimedia devices which are concurrently activated in the remote workstations. He then gives the floor to the first speaker who in turn activates the necessary applications. The right to access a particular application or a multimedia device in input mode is circulated amongst the participants according to rules of fairness or is dictated by the chairman.

The status of the participants and the queue of requests to be satisfied is reported in the window devoted to the management of the conference.

4 Conclusion

In order to make multimedia conference system successful, powerful networks, remarkable software engineering efforts as well as considerations on human factors are required. Moreover, indications which can only emerge from detailed experiences with real products are to be included.

As a contribution to the process towards the setup of conference systems, attention has been paid to the underlying communication infrastructure and its interfaces to the existing software applications and multimedia presentation devices. The proposal has been made to enhance network and transport services in order to support the required multicast and synchronization facilities. A possible architecture of the distributed conference system has been proposed based on the concepts of application sharing and co-operating conference agents. The announced high speed wide-area networks are expected to offer a new dimension to the potentialities of co-operation technologies across long distances [Endr89].

On a more limited scale, some experiences are carried out at the JRC-Ispra involving the use of Unix systems, X-Window and the DUAL backbone network. Such network has already been provided with a mix of X.25, multicast and synchronization features. Progress now is made in the development of the user agent functionality inside the workstations. We plan to produce a demonstration scenario that hopefully can give indications as regards the applicability of the envisaged approach to wide-area networks.

References

[Sari85] S. Sarin, I. Greif: "Computer-based Real-Time Conferencing Systems", *IEEE Computer*, October 1985, pp. 33–45

[Endr87] A. Endrizzi: "The DUAL Backbone Network: Distributed and Parallel Processing on a Large Scale", *Computer Networks and ISDN Systems* 14, 1987, pp. 373–381

[Stef87] M. Stefik et. al.:"Beyond the chalkboard: Computer support for collaboration and problem solving in meetings", *CACM 30*, January 1987, pp. 32-47

[Saka88] S. Sakate, T. Ueda: "Multiparty Desktop Conference System", *IEEE Proc. International Zurich Seminar on Digital Communications*, March 8–10, 1988, pp. 21–27

[Tani88] H. Tanigawa, Y. Hayashi, M. Matsumoto: "Multipoint communication control for document-oriented teleconferencing", *IEEE Proc. International Zurich Seminar on Digital Communications*, March 8–10, 1988, pp. 29–35

[Bonf89] A. Bonfiglio, G. Malatesta, F. Tisato: "Conference Toolkit: A framework for real-time conferencing", *Proc. EC-CSCW '89*, September 13–15, 1989

[Endr89] A. Endrizzi: " Electronic Conference and Parallel Processing across High Speed Networks", *RARE User Meeting on High Speed Networking*, Bruxelles, February 28, 1989