Constructing an X-based Teleconferencing System*

Feng-Jian Wang , Shun-Ting Lu†, Jing-Hwa Liang

Institute of Computer Science and Information Engineering
National Chiao Tung University
Hsinchu, Taiwan, R.O.C.

Abstract

A teleconferencing system making use of computers and network communication lets people participate in a conference without staying together. During the conference, participants communicate with one another through electronic devices such as terminal, mouse, ..., etc, and network. In this paper, we present the constructing of a teleconferencing system based on a standard software system, X/Window System. It discusses an X-based system model for tools necessary in a teleconferencing system, then analyzes the concurrency controls and communication mechanisms among these tools. A prototype for text-based conference called CTUTC for chairperson-members is presented since X provides dynamic-window configuration. Our model provides a framework from which an iconic-based generating system is possibly built.

1. Introduction

A teleconference [6,11,12] is a conference where the participants are distributed in different places. It makes ideas to travel instead of people so that it can be made more available and easier to participate in for those who are located in remote regions or cannot easily travel. For example, a text-based teleconferencing system provides a shared document of text, which is displayed on the window of each workstation for discussion and modification. The system can also provide some facilities such as a voting system, private communication channels for participants, and other media to smooth the interaction during the conference.

One obstacle [3] on the design of teleconferencing systems is building from the scratch. X-window systems [5,13], providing a set of standard user-interface routines based on window, relax developers of network-based applications from detailed designs and implementations of network I/O. A window in X is either a regular window which can be moved, iconized, and resized, or a pop-up window which can be moved, but neither iconized nor resized. A tool interacting with user can be designed with an independent window. A system of I/O intensive application may be thought as a composition of several window-based tools. Many vendors are now embracing X/window as a platform-independent window system standard.

To build a teleconferencing system on X, four aspects at least need be considered: (1) how to organize and smooth the interaction of window-based tools? (2) how to manage the shared document? (3) what is the communication protocol between two window-based tools? (4) how to integrate multi-media tools?

In this paper, we present the constructing of an X-based teleconferencing system. A teleconference in our approach is modeled as communications among participants and window based tools. The teleconferencing system allows that only one tool accepts participant's input with other tools idling, closed, or interacting with other node through network. This is accomplished by using event-driven method, where an event occurs when an input is sent to a tool from either device or network. The characteristics of shared documents and application tools were carefully analyzed. With the help of communication buffers and related functions supported by X, a tool is defined with specific communication missions easily.

Based on the model, we developed a prototypical system, called CTUTC. CTUTC is composed of identical modules of which each runs on a participating SUN workstation. CTUTC lends itself well for presentation by providing a shared document, displayed in a controlled window on each workstation(site). The shared document allows controlled modification so that participants can discuss and modify the document consistently. CTUTC supports a voting system which forces the recipient to

* This research was supported partially by the National Science Council, Taiwan, Republic of China under contract NCS80-0408-E009-28

† Mr. Lu is now working at Telecommunication Laboratories, Directorate General of Telecommunications, Ministry of Communications, Taiwan, R.O.C.

0-8186-2755-7/92 $03.00 © 1992 IEEE

reply within a fixed time interval. It allows private communication by broadcasting cut-paste buffer. It also provides voice media, from which a participant can express his idea in natural way. Moreover, it allows people to join or leave while the conference is in progress.

2. Event driven model

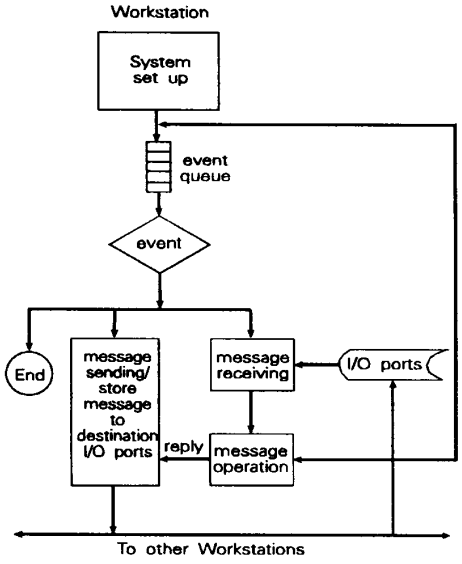

Figure 1 : The overview of an X-based teleconferencing system model

In an X-window system, inputs of a workstation, from devices or network, are managed by X server. An input event invokes X-server to dispatch the input to its corresponding tool; this model of behavior is called *event driven*. The main part of an application program is an event loop that waits for next input to do corresponding action. An X-based teleconferencing system can be thought as a system of identical components, one per node, of which each is composed of window-based tools interacting with participant or other tool(s) through the help of X server. Its flow chart is shown in Figure 1, where the system set-up module boots the first tool of the system through an event.

The classifications of events, in a teleconference of chair-member model, are shown as tree structures in figures 2 and 3. An internal node in a classification tree represents a type (class) of event, while a leaf represents the window-based tool sending or receiving the instance of its father node. The father of an internal node represent its super type (class). Figure 2 shows the types of events in chairman node, and figure 3 for member node.

The communications in our teleconferencing system are done, in fact, through these leaves, called information-exchange (*info-exchg*) tools. Info-exchg tools include:

(1) tools manipulating the object being discussed, such as graphics or text editor for shared document,
(2) tools providing private communication, such as message sent WIN,
(3) tools as electronic devices, such as counter of votes, and
(4) tools of other media, such as video displayer and speaker [10,11].

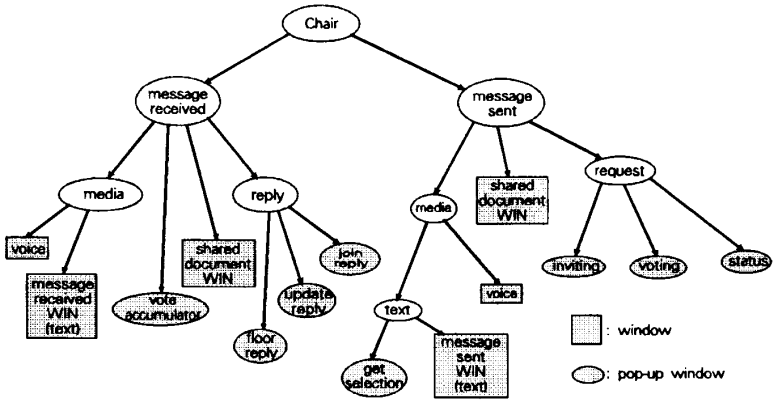

Figure 2 : Event classification of chairperson

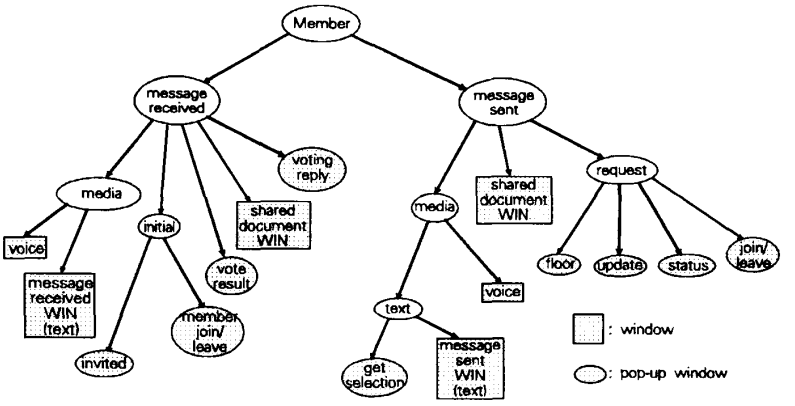

Figure 3 : Event classification of member

3. The relationships among window-based objects

From the viewpoint of window, each info-exchg tool can be deemed as a window-base tool, i.e., a tool associated with a base window. To smooth interactions, several window-based tool may be coordinated into a window-composite tool, and two types of window-based tools, menu and button, are introduced. For example, shared document WIN, message sent WIN, and message received WIN are coordinated in a window-composite tool in our CTUTC.

There may be more than one window-based tools opened in X, but only one accepting input from device at a time. The others are disabled from device, but not network. The tool grabbing the accepting right is said to be *active*. A tool is activated only when it is selected, and is suspended automatically when another one is selected. A tool is a component of another one, if it is invoked by the later. For example, a tool booted (invoked) due to the selection of a menu item or button is a component of the later. The window-base tools within a window-composite tool are the components of the later too. A menu terminates (quits) when its item component is invoked, but other tools co-exist with their component(s). When a window-composite tool is invoked, its components are opened and suspended. Its components are terminated when it is quit.

A window-based tool may, in some cases, disable others from doing I/O. For example, the "voting-reply" tool may disable others except media tools, when a voting takes place. Participants are thus forced to vote immediately since they can hardly do anything else.

These window-based tools can be defined as abstract data types according to the following properties:

(1) window functions,
(2) component tools,
(3) capability of disabling others,
(4) remote communication protocols, and
(5) concurrency control of multiple accesses.

For example, a shared document of text can be defined as an object allowing the following editing capabilities in chairman-members mode: (1) all participants can look at any part of the document using functions of cursor movements, (2) chairman is the only one that can do the modification, and (3) each modification is propagated to the affected area in displayed window of each participant right away.

A teleconferencing system will then be constructed as the composition of instances of these data types. A example of object-oriented construction for such kind of distributed systems is XMVC [15].

4. Issues in communications and concurrency control

4.1 The communications using cut buffers in X

Each X provides eight cut buffers, i.e., eight I/O ports opened for participants communication. One can send his messages to the recipient by putting the messages into any empty I/O port of the latter. One, perhaps the most serious, problem of using cut buffers directly is that if two or more messages are put into the same I/O port, the previous message may be overwritten by the latter. This problem can be solved by dividing the problem into the following two cases: (1) no more than nine persons participate in a teleconference. (2) more than nine persons participate in a teleconference. The solution of applying a B-tree like structure is as follows:

(1) No more than nine participants,

In this case, each I/O ports of a participant is used

to store messages from a distinct participant. Message overwritten will never occur if the sender waits until his previous message are consumed.

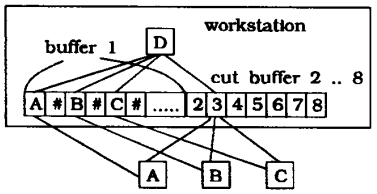

Figure 4: Using a dedicated cut buffer to represent status of transmission

(2) More than nine participants,

If more than eight participants sending messages to the same person simultaneously, there will be no unique I/O port for each participant. Our approach to this problem is to use the first I/O port of each machine as the status of transmission and name it *status-buffer*. The data in the status-buffer are the ID of senders and the number of their messages been received (see Figure 4). Senders are divided into seven groups, of which each uses the same I/O port to store their messages. After sending message through the I/O port, sender must read the status-buffer to verify the receiving. If the message is received, the recipient will increase the receiving number in status-buffer by one. Otherwise, the number does not change and the message must be sent again.

Because X guarantees the delivery of a message to an I/O port, the message will reach the destination eventually. However, that a participant need to access the I/O ports at least twice for each message transmission costs extra. The system need rearrange the participants ID for best allocation of cut buffers when a site enters or quits during the conference.

4.2 Management of a Shared Document

Each participant in a teleconference can view shared document in his shared-document window. Whenever the shared document is updated, all other shared document windows will also reflect the changes. The notification of change can be done explicitly (by voting) or implicitly (by authorized people). It is not allowable that more than one users update the shared document at the same time. These two approaches, implicit and explicit, of managing shared document and display window are used in both conference modes: "chairperson-members" and "non-chairperson", where the former uses implicit approach more and the latter uses voting for majority consensus [2] more.

4.2.1 Chairperson-members Mode

A chairperson is the manager of the shared document; he is responsible for passing control (permission to update the shared document) to and taking it away from the other participant(s). In other words, he determines who has the control of modifying the shared document.

4.2.2 Non-chairperson mode

In this mode, a token is used to prevent two or more participants from updating the shared document simultaneously. To prevent a participant from holding the update right too long, a time-out mechanism is booted. A token is invulnerable in a period after it is given to a node. It becomes vulnerable after time out.

Status-buffer can be used to implement the token. The status-buffer of an appointed workstation can be expanded one byte or longer to represent the status of token. Whoever wants to update the shared document must first check the status of token in that machine. If the token is free, the token requestor can change the token status by encrypting the byte. If more than one request the token at the same time, only one can get the token. The others know the winner by reading the status some time latter. The floor control can be done likewise.

This approach may, however, be unfair. If many participants want to grab the token concurrently, the participant at that machine has better potential because his request need not go through the network. One solution to this problem is to store the token in one machine within a fixed time interval. After the time interval, the machine should pass the generation right of token to other machine.

In the chairperson-members mode, the chairperson may always hold the shared document of newest version. The member who gets the updating right need only update the document of this version (with the help of chairperson). In non-chairperson mode, one who intends to update the shared document need find the shared document of newest version. Besides, both modes accept modification only after the agreement of majority, gained by means of voting system.

4.3 The Voting System

A voting system is able to identify the variance in participant's attitudes on given issue rapidly. It needs to consider several critical factors such as "nominal or anonymous", "weighting", "time out", "single/multiple choices", and so forth. Algorithms 1 and 2 present a general solution for a voting system. Algorithm 1 lets the

chairperson choose a voting mode among the followings: "nominal/anonymous", "weight/none", "single/multiple choices", "disable other tools", "what percentage of votes are admitted to agree the decision", "how many proposals will be agreed", and "how long is the time out". The voting message is then broadcasted to all the participants. Algorithm 2 lets a participant vote according to the voting conditions he received. Once a participant has voted or time out occurs, his vote is replied to chairperson site for accumulation. As soon as all votes are received, the voting result is broadcasted to each participant in summary format. Both algorithms are based on X in LAN; they are not suitable in the application of WAN.

Algorithm 1: Chairperson issue a voting, collects votes, and broadcast the results.
Input: The voting conditions, lists.
Output: The voting result.

1. Choose a voting mode according to the following conditions:
   condition 1: Nominal/Anonymous
   condition 2: Single/Multiple choices
   condition 3: Weight/None
   condition 4: Disable other tools/None
   condition 5: What percentage of replying votes are admitted to agree the decision?
   condition 6: How many proposals will be agreed?
   condition 7: How long is the time out?
2. Input the voting lists;
3. Pack the voting lists and conditions;
4. Wait for replying, accumulate the replying votes according to the voting mode;
   4.1 If condition 2 = "Single" m=1
   Else m= the # of multiple choices;
   4.2 Extract the participant's name and his vote(s): ($V_1 \cdot \cdot V_m$) from the replying messages;
   4.3 If condition 1 = "Nominal"
   List the participant's name and his vote(s) in the voting result table;
   4.4 If condition 3 is set ($V_1 \cdot \cdot V_m$) * "Weight";
   4.5 Add ($V_1 \cdot \cdot V_m$) to the catalogs which they belong to;
5. Broadcast the voting result according to condition 5 and 6.

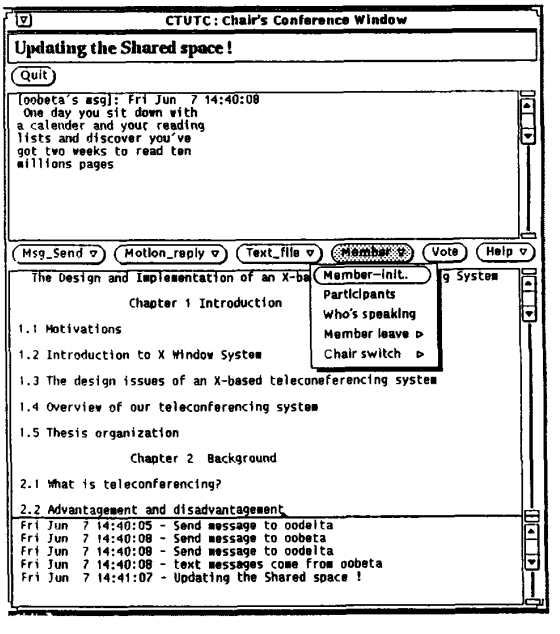

Figure 5: The CTUTC user interface for the chairperson in a teleconference.

Algorithm 2: Participant replying to a vote.
Input: The vote(s).
Output: The decision made by the participant.

1. Extract the voting messages (voting conditions and lists);
2. Set up the voting tool according to the voting conditions;
3. Start the timer;
4. Vote according to the voting conditions:
   4.1 If condition 4 is set. Disable other tools;
   4.2 If one has voted before time out; Return his vote;
   Else Return default;
5. Wait for the voting result;

Note that each voting is given a time period in Algorithm 2. If participant does not vote by time out, a default is set as if he votes. As soon as the voting is done, the system will receive a voting result broadcasted from Algorithm 1.

Voting delay is a key problem. One way to this problem is using two-phase voting: the chairperson sends the voting message, then starts the timer after he gets the acknowledgement of all voting messages. However, the delay may occur after a participant replies his vote. There is, in fact, no upper bound on how long a delay may be, although it is very short in general. Another

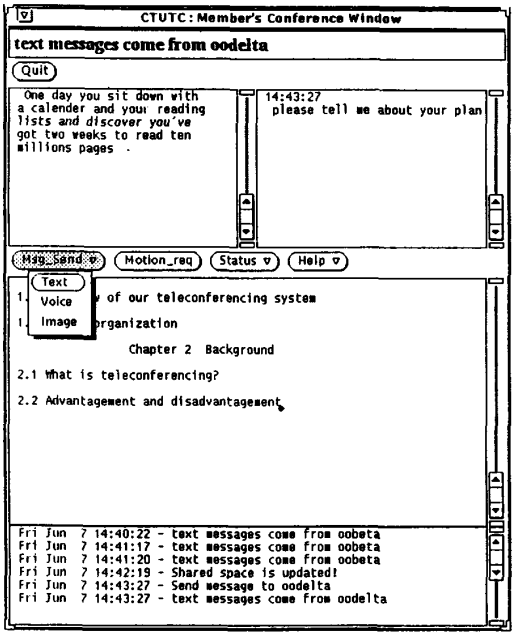

Figure 6. The CTUTC user interface for the participant in a teleconference.

method skips this problem by starting the timer along with the voting message so that a participant is forced to vote (by allowing hardly nothing) within a fixed time interval. Network traffic load is thus reduced dramatically when voting is held.

5. Our Teleconferencing System -- CTUTC

Based on the above discussion, a prototypical teleconferencing system has been developed by means of X/View toolkit [9,14] and run on several SUN workstations connected by Ethernet LAN. This system, called **CTUTC** system, contains two portions in each workstation: one for chairperson and the other for member, and lends itself well to presentations and conferences. The media includes text and voice, while they are transmitted via the same network.

CTUTC system provides text-based windows as in Figure 5 and Figure 6 for participants to hold a presentation. The chairperson oversees and conducts all activities in the conference. He has the right to determine who can join the conference, updates the shared document, and issues a voting. He is the only participant that can terminate the conference. Other participants have right to consult with the chairperson to update the shared document. They can communicate with each other through cut buffers and *shared window*. With the cut buffers, a participant is also free to move information from and to any window (including the window developed by other tool). CTUTC has the following significant features:

(1) *Simplified convening procedure*: During the set-up process, the chairperson is in charge of choosing and notifying the person whose name are in a candidate list to join the conference. During a conference, chairperson is notified to decide the acceptance of new participant(s). Besides, a participant is free to leave or return by sending message to the system.

(2) *Suitable for presentation*: A *shared window* is like a chalkboard or a projector which presents information to all participants. The content of this window represents the shared information of a conference. Whenever the chairperson scrolls it, all views in participants will be noticed for modification. If a modification is done, the message of the action is sent to all participants right away for voting.

(3) *Simplified text transmission*: A participant can send text message, a region selected by pointing device, to receivers specified.

(4) *Conference log*: Each message from network is an event, e.g., "conference initiation", "participants' communication", "voting", "motion request/reply", "who joins/leaves", ..., *etc.*, and is listed with occurring time in the *conference log window* .

(5) *Presentation of status information*: A small summary window indicates what the conference is about, who the participants are, who the chairperson is, and which participant is speaking now. Besides, there is a one-line status window indicating an important event, such as "voice is coming", "voting is taking place", "shared information is updated", "a participant is leaving or joining", ..., *etc.*

(6) *Effective voting tool*: When the chairperson issues a voting, a "*voting reply*" tool displaying the vote-concerned messages is booted to ask the participant to vote 'Yes' , 'No' or, 'Don't care' by time-out. In order to make the vote effectively, it holds the participant's attention by disabling other tools. The voting result is broadcasted right after all participants vote.

(7) *Voice system*: By using soundtool [13], sender

transmits digitized voice, and receiver plays back by converting digitized data into analog signals (See Figure 7). The soundtool is a X/View demonstration program that allows recording, playing, and simple editing of audio data.

Figure 7. The overview of voice system.

6. Conclusion and future work

In this paper, we have discussed the most important issues for constructing an X-based teleconferencing system in LAN. Our approach is based on event-driven model, a scheme that integrates window-based tools for application programs. Besides, a prototypical X-based multimedia teleconferencing system, CTUTC, on a set of workstations connected by LAN is presented. Text-based presentation is efficient in CTUTC, but not voice transmission since it is not a real time operation.

There are still some interesting topics to be studied: multimedia applications and iconic-based teleconferencing generator. Yet this field is still very young, researchers are immersed in to solve the technical difficulties:

(1) Multimedia system [4,7,8,10,11]. We may integrate data, voice, and image communication over high-speed networks.

(2) Iconic-based teleconferencing generator [1]. Data types (see Section 3) of window-based teleconferencing tools can be represented as icons with certain attributes. It is possible to build a teleconferencing system by selecting icons with proper attribute values.

References

[1] Chang, S., "Visual Language: A Tutorial and Survey," *IEEE Software*, January 1987, pp. 29-39.

[2] Coulouris, G., and Dollimore, J., *Distributed Systems Concepts and Design*, Addison-Wesley, 1988.

[3] Dennis, A. R., *et al.*, "Information technology to support Electronic Meetings," *MIS Quarterly*, Dec. 1988, pp. 591-625

[4] Jayant, N.S. and Christensen, S.W., "Effects of Packet Losses in Waveform Coded Speech and Improvements Due to Odd-Even Sample-Interpolation Procedure," *IEEE Trans. Comm.*, Vol.COM-29, Feb. 1981, pp. 101-109.

[5] Jones, O., *Introduction To The X Window System*, Prentice-Hall, 1989.

[6] Kelleher, K., and Cross, T., *Teleconferencing Linking People Together Electronically*, Prentice-Hall, 1985.

[7] Kraemer, K.L., and King, J.L., "Computer-Based Systems for Cooperative Work and Group Decision Making," *ACM Computing Surveys*, Vol. 20, No. 2, June 1988, pp.115-145.

[8] Mukherjee, B., "Integrated Voice-Data Communication over High-Speed Fiber Optic Networks," *IEEE COMPUTER*, Feb. 1991, pp. 49-57.

[9] Nye, A., *XView Programming Manual*, O'Reilly & Associates, Inc. 1990.

[10] Poggio, A., Garcia, J.J., et al., "CCWS: A Computer-Based, Multimedia Information System," *IEEE COMPUTER*, October 1985, pp.92-103.

[11] Sakata, S., "Development and Evaluation of an In-House Multimedia Desktop Conference System," *IEEE Journal on Selected Areas in Communications.*, Vol. 8, No. 3, April 1990, pp. 340-347.

[12] Sarin, S., and Grief, I., "Computer-based Real-Time Conferencing System," *IEEE COMPUTER*, Oct. 1985, pp. 33-45.

[13] Scheifler, R., and Gettys, J., "The X Window System," *ACM transactions on Graphics*, Vol. 5, No. 2, April 1986, pp. 79-109.

[14] Sun Microsystems, Inc., *XView 1.0 Reference Manual: Summary of the XView API*, August 1989.

[15] Wang, F. J., *et al.*, "XMVC - an X version of MVC" to appear in *IEEE 3rd workshop on Future Trends of Distributed Computing Systems*, 1992.