Constructing a Distributed Multimedia Joint Viewing and Tele-Operation Service for Heterogeneous Workstation Environments

Gabriel Dermler[1], Thomas Gutekunst[2], Bernhard Plattner[2]
Edgar Ostrowski[3], Frank Ruge[3], Michael Weber[4]

University of Stuttgart[1]
IPVR
Breitwiesenstrasse 20-22
D-70565 Stuttgart, Germany
dermler@informatik.uni-stuttgart.d400.de

Swiss Federal Institute of Technology[2]
Computer Engineering and Networks Laboratory (TIK)
ETH-Zentrum, ETZ
CH-8092 Zürich, Switzerland
{gutekunst, plattner}@tik.ethz.ch

Technical University of Berlin[3]
PRZ, Sekr. MA 073
Straße des 17. Juni 136
D-10623 Berlin, Germany
{ostrowski, ruge}@prz.tu-berlin.d400.de

German Research Center for Artificial Intelligence (DFKI)[4]
KIK-TEAMKOM
Postfach 1150
D-66041 Saarbrücken, Germany
mweber@dfki.uni-sb.de

Abstract

JVTOS (Joint Viewing and Tele-Operation Service) is an advanced teleservice to support cooperative work over distance which allows distributed users to work in a collaborative fashion with multimedia. JVTOS comprises facilities for session management, floor control, multimedia application sharing, telepointing, and audio/video communication. It provides generic support for cooperation-aware multimedia applications.

JVTOS offers services for multimedia collaboration across high-speed networks and is primarily aimed at running in heterogeneous workstation environments comprising different hardware platforms and also different operating and window systems.

This paper describes the design of JVTOS as well as its implementation on different platforms which is currently under development.

1 Introduction and motivation

Synchronous collaboration usually takes place with all participants being in the same room. Participants are engaged in a face-to-face dialog in which black- or whiteboards, overhead projectors and other equipment are frequently used. Many CSCW applications (computer-supported cooperative work applications) mirror such meeting rooms with the users sitting in the same room in front of their computers. High-speed networks will make a distributed version of this approach possible, both technically and economically. Multimedia information may be transferred via the network offering a joint usage of applications, as well as audiovisual communication facilities. Users may thus attend a "virtual" meeting without even leaving their office, being able to retrieve informaton from their personal or corporate databases.

CSCW tools may be *cooperation-aware,* i.e. specifically designed to be used in groups, such as a conferencing system supporting several users; alternatively, a CSCW tool may be a coordinating environment in which *cooperation-unaware* single-user applications are being run in a group context [6] [15]. Independently of which approach is taken, both kinds of CSCW applications require a clear distinction between a private and a group context, but still it will be necessary to integrate both workspaces as seamlessly as possible.

Using a homogeneous hardware and software platform and a corresponding CSCW software is the usual approach to achieve a good integration of the group and private workspaces. This paper, however, discusses how to construct a tele-cooperation environment, called "Joint Viewing and Tele-Operation Service" (JVTOS), which allows for multimedia collaboration even in heterogeneous workstation environments comprising different hardware platforms and also different operating and window systems.

JVTOS is issue of work package 4.2 within RACE II project CIO (R2060) [3]. Specifically, JVTOS will support Sun Sparcstations with SunOS/Solaris, the Apple Macintosh with MacOS, and the IBM PC with MS-DOS and Windows.

Both a reference model and a conference model for JVTOS have been presented in [10] and [13] respectively.

A service overview is given in section 2. Section 3 focusus on user-level services, while section 4 discusses key aspects of the lower-level services, including continuous media support. Section 5 sketches the state of the implementation and section 6 concludes this paper.

2 Service overview

JVTOS comprises four user-level services as well as two low-level services (figure 1). The low-level services are hidden from the JVTOS user whereas the user-level services are accessible through respective user interfaces.

The two low-level services, the *Audio/Video Communication Service* and the *Multimedia Interstream Synchronization Service*, target the specific requirement of distributing multimedia information. The user-level services are *Session Management*, *Multimedia Application Sharing*, *Picturephone* and *Telepointing*. Above these, JVTOS is open towards integration of other user-level services.

The *Session Management Service* (SMS) administrates and runs sessions. Sessions are the frame in which collaboration takes place. The SMS also manages the other user-level services and is thus the major control of the entire JVTOS teleservice.

The *Multimedia Application Sharing Service* (MASS) allows the distribution of single-user applications to be viewed by all session participants. The MASS distributes the application windows to all users, which may use different hardware platforms and window systems since the MASS comprises translators to mediate between these different platforms.

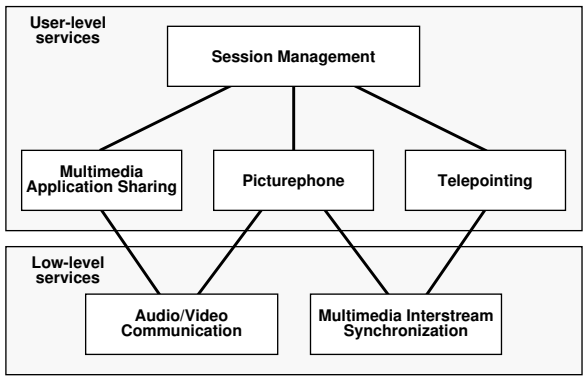

Figure 1: JVTOS service overview

The *Telepointer Service* (TPS) allows a session participant to move a telepointer in shared windows being visible to all other session participants, whether this telepointer owner currently holds the floor or not. The TPS also distributes the floor holder's mousepointer.

JVTOS also provides generic support for collaboration-aware multimedia applications by offering the session management, floor control and telepointer services to them. Such an application is the *Picturephone* which allows the session participants to communicate audiovisually to each other.

3 User-level services

3.1 Session Management

The Session Management Service provides functions for session and user management. Such functions are e.g. opening and closing sessions. Users may be invited by the session chairman, or they may request to join or leave an existing session.

The roles of participants are also handled by the Session Management Service: active participants (contributors) might become passive participants (observers) and vice versa, the session chair might be shifted.

The Session Management Service includes a Management Information Base and a Floor Control Service.

The *Management Information Base (MIB)* holds session-static as well as session-dynamic information. Static information relates to JVTOS users, such as their names and addresses. Dynamic information relates to session-specific data, such as the current set of participants and their roles (e.g. the current floor holder). The MIB can be accessed by all services to get relevant information.

The *Floor Control Service (FCS)* requests the Multimedia Application Sharing Service to allow a session contributor to provide input to a shared application. The concept of floor control is separated into mechanism and policy [7]. The floor control mechanism handles the low-level activities of passing the floor and maintaining a synchronized event stream for all participants. The floor control policy comprises a set of rules governing the floor control, i.e. determining how the floor is requested and granted. Each session can use a different floor control policy which can be changed at any time whereas JVTOS uses a fixed set of floor control mechanisms to implement different floor control policies.

The Session Management Service controls the other user-level services not only by starting and terminating the respective service, but also by transfering dynamic session-specific information such as changing the group of participants or shifting the floor. A generic set of control primitives provides this functionality.

3.2 Multimedia application sharing

The Multimedia Application Sharing Service [14] allows *cooperation-unaware single-user multimedia applications* to be shared among several heterogeneous workstations. The terms "cooperation-unaware" and "single-user" denote that the applications were actually constructed for a single user only and hence are not aware of being run in a group context. Multimedia applications may handle text and graphical information, still pictures, moving pictures (video and animation), and sound [12]. According to [1], audio and video are referred to as *continuous media*.

Compared to specialized cooperation-aware applications, the application sharing approach has several advantages: First, users are not required to learn new applications for cooperative work, they can share the applications they are used to. Secondly, the sharing service does not need to be modified to support new applications. Finally, applications are taken "as is", i.e. third-party applications can be shared without any modification.

The Multimedia Application Sharing Service allows users to jointly and simultaneously view the output of multimedia applications visible through windows located on each user's display. Windows visible to each user are referred to as *shared windows*. Applications of which the output is jointly viewed are called *shared applications*. *Floor control* is used to determine which user is allowed to direct input to a shared application. The right to direct input to a shared application is denoted by the *floor*, the user currently allowed to do so is called the *floor holder*. Input from users not holding the floor is dropped.

Heterogeneity support. In the context of application sharing, support for heterogeneous platforms means that (1) application output may be presented on any platform, (2) application input (which is subject to floor control) may come from any platform, and (3) a shared application may execute on any platform.

Service architecture. Figure 2 depicts the architecture of the Multimedia Application Sharing Service for the scenario of one multimedia application being shared among two terminals. Terminal denotes the set of a user's input/output facilities such as keyboard, mouse, display, and audio/video input/output devices. Interaction between the application and the native terminal is intercepted by the sharing service and distributed to all involved terminals.

Xv [4] has been chosen as the basic interchange protocol for application data. It is an extension of X [16] and allows to control still and motion video. X is a network-transparent, device-independant windowing and graphics system currently supported by most leading workstation manufacturers.

The Multimedia Application Sharing Service is implemented on the basis of a distributed X multiplexor [2] which has been extended for video support and heterogeneous platforms. The *pseudo server* is the entity which

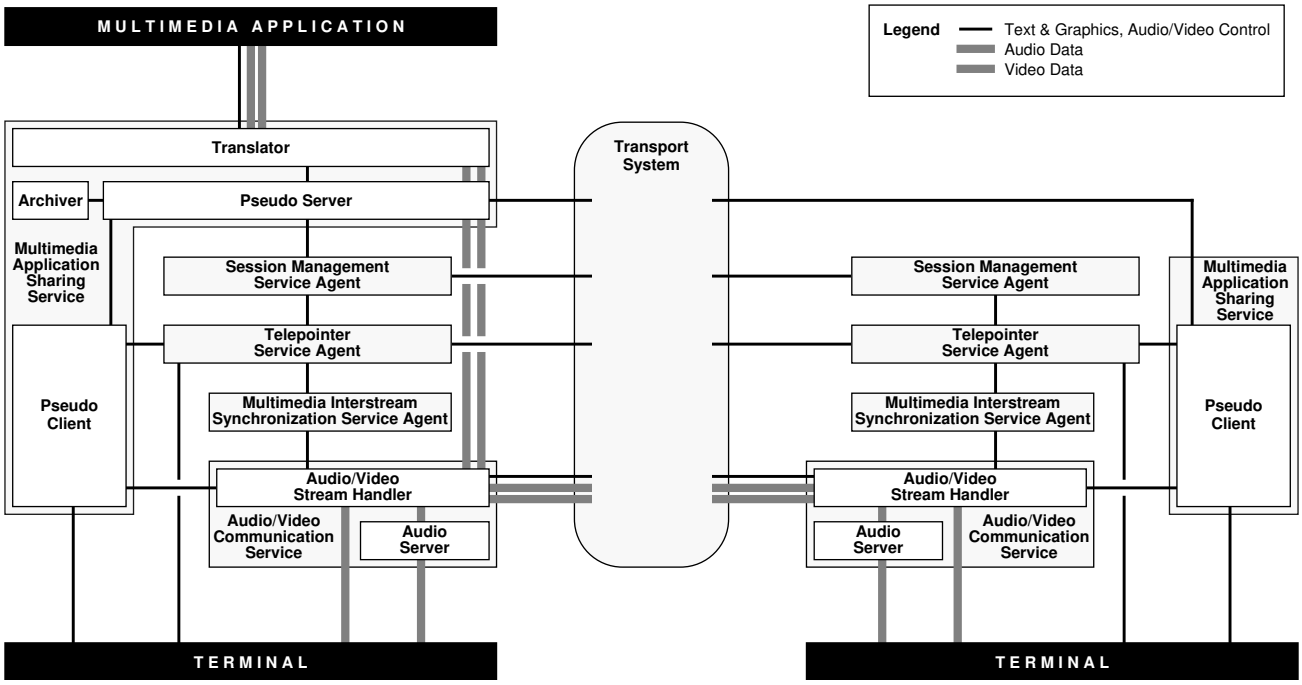

Figure 2: Application sharing architecture

multiplexes the Xv protocol data stream. There are *pseudo client*s for each terminal which map the respective Xv protocol data stream to the local characteristics. For non-X windowing systems (i.e. MacOS and MS-DOS/Windows), *translators* [17] map non-Xv protocols to the Xv protocol. On the Sun platform, there is no translator required. The *archiver* keeps track of Xv resources created during a session which enables the sharing service to integrate new participants into an ongoing JVTOS session and thus allows for dynamic user participation.

Note that pseudo server, translator, and archiver are related to the shared multimedia application. Therefore they run only once whereas there are as many pseudo clients as participants in the session.

Translators. Translators map non-Xv protocols to the Xv protocol whereas commercially available X servers are extended for Xv to non-Xv conversion. On the Macintosh platform, application output is translated by intercepting calls to the Macintosh Toolbox and translating them into Xv requests which are sent to the pseudo server. Xv events (application input) are translated into Macintosh events which are subsequently read from the event queue by the applications as their input. Translation on the PC platform is still an open issue.

Multimedia support. Multimedia applications have to use well defined interfaces in order to be shared. Table 1 lists the operating system extensions supported by JVTOS.

As continuous media data streams are time-critical, the Multimedia Application Sharing Service only *sets up and controls* continuous media data streams. The data is actually transported by the *Audio/Video Communication Service (AVCS)*. Pseudo interfaces handle the supported audio/video sharing access points for the specific workstations. Application requests related to continuous media data streams are converted into commands for the AVCS.

Sun Sparcstation	Xv (Video Extension to X), audio device (“/dev/audio”)
Apple Macintosh	QuickTime (selected components)
IBM PC	MME (Multimedia Extension to Windows), MCI (Media Control Interface)

Table 1: Supported OS extensions

3.3 Picturephone

JVTOS includes a picturephone, which provides interpersonal audiovisual communication for all JVTOS users. During a session, the Picturephone will probably run most of the time. The Picturephone must be smart enough to allow other multimedia application to run in parallel without degrading their quality.

In the first version, the Picturephone uses the AVCS and its integrated stream handlers for audio and video. Therefore, the Picturephone does not process any continuous media data. A later version may have own facilities for audio and video processing and displaying, e.g. for mixing or switching audio channels.

The first version supports two-way audio and two-way video streams between all participants of a JVTOS session. The Picturephone specifies media sources and destination endsystems and asks the AVCS to transfer these streams to the remote sites. Processing of audio/video data is done by the AVCS whereas control of the media sources and sinks is done by the Picturephone.

As JVTOS is capable of running multi-party sessions, the Picturephone provides stream control mechanisms to prevent the participant from both non-intended information missing as well as information overflow. Consequently, the Picturephone provides parallel output of multiple video and audio streams (i.e. multiple windows and audio mixing) as well as options for selecting individual sources of information (e.g. switching on a per-channel basis). Nevertheless, a complex user interface has to be avoided since JVTOS wants to support interpersonal communication in a session. It is not our intention to build a general videoconferencing service although this could be covered as well with the AVCS.

On each platform, the user interface will be adapted to the native look-and-feel of the target window system, but the functions offered will be identical. The Session Management Service supplies information about the members of a session enabling the Picturephone to adjust itself according to changes during a session.

Although the design of this application has not been completed yet, we have experienced the needs for some functions from multipoint video conferencing prototypes used with JVTOS in different scenarios.

For all session participants, status information is shown indicating the connectivity, visibility and audibility. There is a gain control for each audio channel and also a gain control for the sum of all audio channels. Each video window also displays an indication to which session participant it belongs. If the underlying hardware supports adjustments for contrast, hue, and brightness, the JVTOS user will be able to modify these values via the user interface.

3.4 Telepointing

Communication between JVTOS users, as described so far, allows for sharing information generated by an application as well as discussions via the Picturephone. JVTOS enhances this communication by providing globally visible

pointing tools, termed telepointers. Each user may own a set of telepointers and move these on his display. A user's telepointer becomes visible to others as soon as it enters a shared window, i.e. as soon as it could point to an object which is jointly viewed and possibly referred to in an audio/video conversation.

The service realizing the telepointers is the Telepointer Service [9]. Its purpose is to monitor movements of pointers owned by a user and to display these movements locally and at remote user sites. In order to keep end-to-end delays minimal, a decentralized implementation is pursued. The fact that this can require a large number of connections is ameliorated by the use of multicast channels provided by the transport system [5]. Overhead is further eliminated by restricting the update interval of telepointers to be above a certain threshold.

The Telepointer Service relies on support from other JVTOS services (figure 2). From the session management it receives addressing data of session participants. From the sharing service it receives the list of shared windows. Thus, the service is able to determine when a telepointer is to be "shared" and when so, with whom.

In order to preserve efficiency, telepointers are implemented as separate shaped windows. Moving a telepointer means moving a window in front of a shared window containing the object of discussion. This solution requires a close correlation between a telepointer and the shared window state concerning e.g. its position, size, or visibilty. The solution for the required correlation may vary on the different platforms.

In X, anchoring a telepointer as a child of the shared window where it is located, proves suitable. All correlation is done efficiently and automatically by the X server. In addition, moving a telepointer between windows is supported by the reparenting concept. A similar solution seems tractable for the PC platform. In contrast, the Macintosh solution will first have to realize a correlation mechanism between telepointers and shared window states since the Macintosh Toolbox lacks the concept of a window hierarchy.

4 Low-level services

4.1 Audio/video communication

One of the main goals of JVTOS is to provide identical views of multimedia applications to the participants of a JVTOS session. These "views" have more than the two dimensions of a typical computer display: e.g. stereo audio gives a user the impression of space as well as time. Therefore in some media (such as audio, video, and animation) time is inherent and in others it is not (such as text and graphics). Continuous media require much more precise handling of presentation timing than the other media.

In order to allow joint viewing/listening of multimedia applications, there are three steps to be performed for the audio and video output of an application: (1) fetch the audio/video data, (2) distribute the data to the other terminals, (3) play/display the the audio/video data.

Audio/Video Communication Service (AVCS). The Audio/Video Communication Service [11] fulfills these steps under the control of the user-level services. It hides the complexity of audio/video processing in the workstation behind an application programming interface (API) which provides simple functions to transfer continuous media data streams from a local source (e.g. a microphone) to a remote sink (e.g. a loudspeaker). Although the AVCS might be directly used by any application, the design focus was on the support for multimedia application sharing. The AVCS is capable of multipeer communication, negotiates codings, and sets up transport connections to remote AVCS agents.

Multimedia operating system extensions. Multimedia services dealing with continuous media for input or output have to interact with the operating system in some way. Ideally, all operating systems (OS) would provide the same continuous media interface capable of handling time and different audio and video codings. But in practice, we find many different interfaces supporting different types of continuous media. For JVTOS, we had to decide which continuous media interfaces to support on each platform, similar to the choice of window systems to support. Table 1 lists the multimedia OS extensions supported by JVTOS.

As these extensions do not provide enough support for tele-cooperation services (e.g. output of multiple audio streams), JVTOS also implements basic multimedia functionalities:

- *Audio and video stream handlers* supporting intrastream and interstream synchronization,
- an *audio server* providing for duplication, conversion, and mixing of multiple audio data streams.

With the AVCS, the user-level services still use the specific extensions of the operating system for local operations, but the AVCS solves the heterogeneity problems for the communication between remote systems.

Application Programming Interface (API). The AVCS provides an interface for (1) user agent registration, (2) connection establishment, (3) connection release, and (4) connection control.

The services which use the AVCS (i.e. the Multimedia Application Sharing Service and the Picturephone) are called user agents (UA). In order to use the AVCS, a user agent must have registered with the local AVCS agent. The main service of the AVCS is the provision of audio/video connections to remote service users.

The user agents control connections by manipulating attributes. These are used to tell the local AVCS agent what to do (e.g. to associate the output of a video connection with a region of a window). Most attributes are also passed to the remote AVCS agents and to the peer user agents (e.g. to change the video encoding during the lifetime of a connection).

Examples for attributes are: type (AUDIO, VIDEO, ...), source AVCS agent address, source video port (window id, ...), video encoding (MOTION_JPEG, MPEG_VIDEO, ...), bandwidth, transport delay, and transport system hints (USE_MULTICAST, ...).

Multimedia interstream synchronization. The Multimedia Interstream Synchronization Service (MISS) implements a central instance on each JVTOS endsystem which controls the sink playout time with respect to the presentation time of the source endsystem. It is explained in chapter 4.2 in more detail.

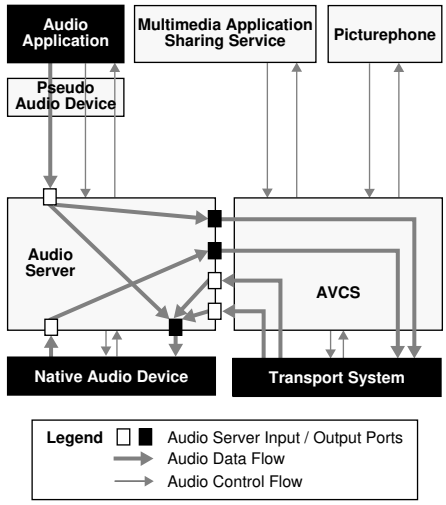

Figure 3: Audio sharing scenario

Advanced transport system. The knowledge about the structure, the coding and the timing of a continuous media data stream enables JVTOS to choose a transport service with suitable quality of service (QoS). Important QoS parameters that can be adjusted are delay and delay jitter, bandwidth, TSDU size, and error tolerance. In addition, the AVCS can reserve resources in the endsystems to avoid both bottlenecks and waste of resources. In order to provide QoS guarantees to the application, the underlying transport system [5] is based on XTP (Xpress Transfer Protocol).

Audio server. The audio server is a module which allows multiple access to the audio hardware. It can be understood as a switch panel, where an input port (application data or analog/digital converter output) can be connected to one or multiple output ports (e.g. applications). This implies that mixing has to take place whenever two or more sources meet at an input port. Multiplexing has to take place if several data streams leave an output port. The audio server supports a set of different audio codings for input and output ports. If interconnected input and output port codings do not match, audio coding (G.711, G.721, G.723) conversion has to be done.

Figure 3 shows the audio configuration of a scenario where one audio application is being exported to a remote user, one audio application is imported from a remote user, and the Picturephone accompanies this session.

The Multimedia Application Sharing Service and the Picturephone both have control over the respective AVCS connections (e.g. connection setup and close, coding, and gain control). Both are AVCS user agents and work concurrently without any knowledge of each other. The audio server automatically does duplication respectively mixing of the audio data on the input and output ports as needed.

Format conversion of audio streams is done on the way from one audio server port to the other. As we do not require our target systems to have special hardware support for this purpose, the audio server will also include a pure software solution for this task. The implementation will do the conversions in realtime, though we expect considerable reduction of the audio quality for some conversions.

4.2 Synchronization

The group-oriented nature of multimedia communication in a JVTOS system implies a multitude of data streams which have to be presented in a synchronized fashion to the session participants. We distinguish three levels of synchronization requirements:

Intrastream synchronization. This type of synchronization refers to each involved continuous media data stream such as audio and video. The goal is to match generation and consumption rates of connected source and sink pairs and eliminate end-to-end transmission jitter down to acceptable values (20 msec for audio and 40 msec for video). In JVTOS, intrastream synchronization is assigned to the service handling the corresponding stream, i.e. the AVCS. The latter delays data presentation, if jitter is detected and pauses or skips data presentation if a rate mismatch has occured [8].

Interstream synchronization. Multimedia implies that information is carried not only in single streams, but also implicitely in the temporal relation between presentations of data of different streams. Therefore, all information generated by a JVTOS user, when addressing other users via the Picturephone application or the telepointing facility) has to be in synchrony when presented at a remote site. The same applies to the output of a shared multimedia applica-

tion. The goal is achieved by interstream synchronization involving the Multimedia Interstream Synchronization Service (MISS).

In its current version - provided for the first JVTOS prototype - the MISS at each user site is invoked at regular intervals by the services desiring interstream synchrony, e.g. the Telepointer Service and the AVCS. Each service provides information identifying the group of streams to be synchronized (currently the identifier of the machine the stream group originates from), relevant temporal information about data units received, and its possibilities to buffer data before presentation. In turn, the MISS computes for each service an artificial delay which has to be inserted before presentation. This procedure allows to concentrate decisions on synchronization policies in the MISS and hides them from the involved services.

Conferencing Synchronization. Synchronizing the output of shared applications and the presentation of picturephone or telepointer information is another requirement, referred to in JVTOS as *conferencing synchronization*. It may come in two flavours. In the first case, user-to-user communication via Picturephone and telepointers is isochronous. This means that what someone says or shows is perceived by other users at virtually the same time. This in turn requires that output of a shared application has to be the same at all user sites in order to ensure that users mean the same if they refer to jointly viewed objects. The solution may be that the distributed MISS agents rely on a globally synchronized clock and exchange information among each other to ensure simultaneous presentation of application data.

In the second case, user-to-user communication is not isochronous. Here, a master-slave communication pattern for the Picturephone and telepointing has to be enforced, e.g. users talk one after the other exchanging explicitely the right to do so. The synchronization mechanism has to ensure in this case that a slave who is, for instance, listening to the current master, is presented with the application output later than the master. The difference would exactly correspond to the transfer delay incurred on the audio channel. This scenario can but does not have to rely on global clocks. However, it also implies a protocol between the distributed MISS agents.

The conferencing synchronization requirement depends very much on the application which is shared. For instance, viewing a static object (even if presented in a video sequence) does not call for any additional mechanism since, even if picturephone information arrives late at a user site, the object it is referring to is unchanged. Rapidly changing application output is the other extreme. Currently, we expect that cooperation via Picturephone and telepointers in relation to jointly viewed objects in most cases relate to rather static objects. Therefore, conferencing synchronization is not supported in our first prototype, leaving it to future experience to evaluate the effects of this decision.

5 Implementation work

The implementation of JVTOS is currently being progressed in a distributed fashion, i.e. at several of the involved sites. Currently, the fully distributed Session Management Service supporting several floor control policies is being implemented on the Sun platform. The Multimedia Application Sharing Service - so far without continuous media support - is available on the Sun as a distributed service, and is currently being ported to the Macintosh platform. As far as continuous media support is concerned, a simple version of the AVCS has been implemented; the current work focuses on the implementation of the full AVCS as outlined in section 4.1 and the port from the Sun environment to the other platforms. The Telepointer Service has been implemented as a standalone service, and work on the implementation of the synchronization functions is now starting.

A prototype of JVTOS has been developed for the CeBIT ‘93 fair in Hanover, capable of demonstrating the functionality of JVTOS including the Picturephone.

6 Summary and conclusions

This paper describes a design of a system capable of making an off-the-shelf application program usable in a cooperative work environment. It has to be admitted that such systems have been studied and implemented previously. Similar systems are even available commercially. So what is innovative about JVTOS? The general answer to this question is that JVTOS, in contrast to all application sharing systems known to us, does support multimedia application sharing in a heterogeneous setup. Heterogeneity is present in various flavors:

- JVTOS will run on three rather different hardware platforms using three different operating systems.
- Shared applications may be initiated on a workstation of any type. Such applications use the window system of the workstation they run on, but it will still be possible to interact with such applications using a workstation of any of the three types. The key to this functionality is our use of the Xv protocol as an interchange protocol and the concept of translators.
- Several types of networks are supported through the transport system, which additionally is capable of providing channels with characteristics adapted to the requirements of JVTOS.

Continuous media support is not restricted to the provision of just a picturephone. Instead, sharing of multimedia applications is also possible, with the implication that an-

other element of heterogeneity has to be taken care of, the handling of different kinds of multimedia toolboxes associated with the three platforms.

Some of the decisions taken during the JVTOS design will have to justify their viability during the implementation and use of the system. We decided to transport multimedia control information in an extension of the X protocol; a different approach would have been to use a specific control protocol for continuous media data streams - note that we took this approach with audio support, as Xv has no support for audio.

We are also interested to study the impact of the distributed implementation of the Multimedia Application Sharing Service on system configurability on the one hand and performance on the other hand. We expect that allocation of the two main components of the Multimedia Application Sharing Service (pseudo server and pseudo client) will yield different results with respect to performance; thus a fine tuning of the overall system behaviour may be achieved by careful allocation of these components.

Acknowledgements

The work discussed in this paper was performed in the context of the RACE II project CIO (R2060) and specifically within the work package 4.2, in which many of the ideas put forward were born. We would like to acknowledge the contributions of all involved partners that were left unmentioned in this text.

References

[1] David P. Anderson, Ramesh Govindan, George Homsy, Robert Wahbe: "Integrated Digital Continuous Media: A Framework Based on Mach, X11, and TCP/IP". *Technical Report. University of Berkeley, California.* 1990.

[2] John Eric Baldeschwieler, Thomas Gutekunst, Bernhard Plattner: "A Survey of X Protocol Multiplexors". *ACM Computer Communication Review, Vol. 23, No. 1 (April, 1993)*. New York, 1993.

[3] Wulfdieter Bauerfeld: "RACE-Project CIO (R2060): Coordination, Implementation and Operation of Multimedia Tele-Services on Top of a Common Communication Platform". *Proceedings, International Workshop on Advanced Communications and Applications for High Speed Networks (IWACA '92), pp. 401 - 405*. München, 1992.

[4] David Carver: "X Video Extension Protocol Description (Version 2)". *Digital Equipment Corporation, Workstation Engineering/Project Athena, Contrib Section of MIT X11R5 Distribution*. 1991.

[5] Paul Christ, Ilka Miloucheva, Andreas Rozek: "Multimedia Communication Platform: Transport Service Programmer's Interface (TPI)". *Internal Report of RACE/CIO WP 3 (Deliverable 21)*. Stuttgart, 1993.

[6] Stefan Cronjäger, Walter Reinhard, Jean Schweitzer: "Functional Components for Multimedia Services". *Proceedings, International Conference on Communications (ICC '93)*. Geneva, 1993.

[7] Terrence Crowley, Paul Milazzo, Ellie Baker, Harry Forsdick, Raymond Tomlinson: "MMConf: An Infrastructure for Building Shared Multimedia Applications". *Proceedings, CSCW '90*. Los Angeles, 1990.

[8] Gabriel Dermler: "Design of JVTOS: Synchronization". *Internal Report of RACE/CIO WP 4.2*. Stuttgart, 1993.

[9] Gabriel Dermler: "Design of JVTOS: The Telepointer Service". *Internal Report of RACE/CIO WP 4.2*. Stuttgart, 1993.

[10] Gabriel Dermler, Konrad Froitzheim: "JVTOS - A Reference Model for a New Multimedia Service". *Proceedings, 4th IFIP Conference on High Performance Networking (hpn '92), pp. D3/1 - D3/15. Edited by A. Danthine, O. Spaniol*. Liège, 1992.

[11] Gabriel Dermler, Edgar Ostrowski, Frank Ruge: "Design of JVTOS: Proposed Implementation for the Audio/Video Communication Service". *Internal Report of RACE/CIO WP 4.2*. Berlin, 1993.

[12] Konrad Froitzheim, Edgar Ostrowski, Nelson Pires: "Multimedia Applications and how they Interact with Workstation Hardware and Operating Systems". *Internal Report of RACE/CIO WP 4.2*. Ulm, 1992.

[13] Thomas Gutekunst, Thomas Schmidt, Günter Schulze, Jean Schweitzer, Michael Weber: "A Distributed Multimedia Joint Viewing and Tele-Operation Service for Heterogeneous Workstation Environments". *Proceedings, GI/ITG Workshop on Distributed Multimedia Systems, pp. 145 - 159. Edited by W. Effelsberg, K. Rothermel*. Stuttgart, 1993.

[14] Thomas Gutekunst, Bernhard Plattner: "Sharing Multimedia Applications Among Heterogeneous Workstations". *Proceedings, Second International Conference on Broadband Islands. Edited by O. Spaniol, F. Williams.* Athens, 1993.

[15] Alex Jarczyk, Peter Löffler, Gerd Völksen: "UPN Study: Computer Supported Cooperative Work". *Internal Report, Siemens AG*. München, 1992.

[16] Oliver Jones: "Introduction to the X Window System". *Prentice-Hall*. Englewood Cliffs, 1989.

[17] Miroslav Vodslon: "Feasibility of Translating Native Windowing Systems to X Window". *Internal Report of RACE/CIO WP 4.2*. Berlin, 1992.