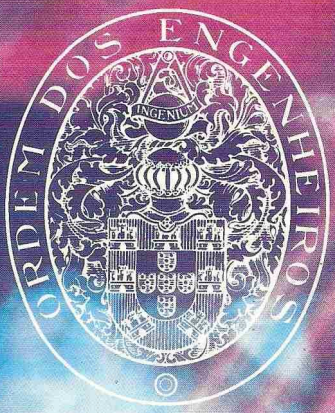
# 2.º ENCONTRO NACIONAL DO COLÉGIO DE ENGENHARIA ELECTROTÉCNICA




ORDEM DOS ENGENHEIROS

14 e 15 DEZEMBRO 1995
INSTITUTO SUPERIOR TÉCNICO

# DISTANCE TRAINING OF X APPLICATIONS THROUGH A MULTIMEDIA CO-AUTHORING ARCHITECTURE

**Adérito Marcos, Benjamim Ferreira, Christoph Hornung**
Fraunhofer Institute for Computer Graphics
Wilhelminenstr. 7, 64283 Darmstadt, Germany
{marcos, benjamim, hornung}@igd.fhg.de

## Abstract

Distance Training (DT) of applications is essentially a process of CSCW (Computer-Supported Cooperative Work). Its main goal consists of maintaining a training session among several geographically distributed participants who may also be located in the same room.

The aim of this paper is to propose a global cooperative model for distance training of X Windows applications. Our considerations are based on the experience and results acquired throughout the development and usage of our own prototype: the VirtualX system. It supports the different phases of a well defined training cycle by providing specific features of application-sharing together with group coordination. A complete multimedia co-authoring environment is available for the whole training cycle, including mechanisms of computer-conferencing (text, audio and video communication).

**Keywords**: CSCW, Distance Training, Multimedia Technology, Application-Sharing, Co-Authoring Process

## 1. INTRODUCTION

Computer-Supported Cooperative Work has emerged, over the last decade, as an increasingly important multidisciplinary field, which has been promoting the development of systems able to computationally support team work in areas and activities where the traditional face-to-face meetings or common communication means (mail, fax, telephone, etc.) were mainly used.

Some wide studies carried on the human and technological requirements of different work domains have validated the importance of group work in a large range of activities, specially if they are computer supported. CSCW comes to be fundamental in all those cases where team members need to work together but have the problem of being geographically distributed [14].

Several CSCW models and systems have been proposed within the research community as well as to the industry. Gradually, some of them are being successfully used under real conditions situations.

On the other hand, the increased evolution occurred on the multimedia technology has been enabling new ways of information manipulation in terms of storing, retrieval, combination and end-user presentation. Multimedia technology enhances a best expressiveness of the concerned information, and above all, supports forms of intuitive remote user communication through video, audio and text channels. Actually, most of the today's cooperative environments are being implemented over multimedia platforms, specially when involving editing features. These environments are normally defined as belonging to the Cooperative Multimedia Systems class.

Cooperative Multimedia strategies support the building of one of the most important CSCW environment classes and central subject of this paper - the Distance Training systems. The DT system's main goal is to maintain a training session among several geographically distributed participants, connected over network and who may also be located in the same room. These systems must found solutions to problems such as: supporting the coherence of the training information (material) through the distributed system by managing possible conflicts between versions at local workspaces and, above all, promote the necessary mechanisms for the inter-group awareness and integrity.

Usually, along with the distributed training process runs an additional co-authoring environment sustaining a conforming process of brainstorming/idea discussion. This co-authoring is basically a cooperative multimedia editing environment which permits the participants to compose together different sort of documentation (co-authored) or simply keep up a computer-conferencing

(multimedia communication plus exchange of data). Indeed, it is our intention in this paper to propose a global solution to support Distance Training along with a detailed overview of a concrete system.

This paper is organised as follows: we start by giving an overview of the CSCW systems. Next, we expose the VirtualX system, i.e., the distributed architecture and the Co-authoring/DT processes. Finally, an overview of some representative related work and an outline of the future work directions as also some conclusions, make up the second part of the paper.

## 2. THE SCOPE OF CSCW

In order to easy the categorising of the distinct nature of the computational cooperative work, CSCW systems can be organised through two main characteristics:

i) The form of interaction
ii) The geographical distribution

These characteristics are able to catalogue the great variety of the existing CSCW systems and are useful in our current case to classify the Distance Training environments.

### The Form of Interaction

CSCW systems are essentially concerned with supporting groupwork, where group members are cooperating to solve a particular problem, or range of problems. People cooperate in a variety of manners depending on a range of circumstances. They can either interact and cooperate *synchronously* or *asynchronously*. The synchronous interaction demands the presence of all the involved users while the asynchronous cooperation occurs in an off-line mode (over long time period) and does not require the simultaneous interaction of the participants. Using this classification, we can stress three main categories of systems [16]:

i) *Purely synchronous systems*
This type of systems need the simultaneous presence of all users. In fact, the users are sharing the same time slice while working together. Real-time conferencing/co-authoring systems are the typical examples of this form of interaction.

ii) *Purely Asynchronous systems*
Asynchronous systems are conventionally implemented to permit cooperative work without the presence of all group members. Participants with different roles can independently contribute to a common objective while the system executes the coordination of the different inputs being generated. Examples of asynchronous interaction systems are (between others): cooperative message systems and traditional conferencing systems.

iii) *Mixed*
Mixed systems can actually support the two above type of interaction. They allow on-line cooperative sessions to take place while maintaining asynchronous facilities. Usually, a central asynchronous conferencing server is implemented and augmented to provide real-time conferencing processes. In fact, most of the today's cooperative systems integrate *mixed* interaction.

### The Geographical Distribution

Traditionally, CSCW systems were strictly related to distributed environments. Besides, one of the main CSCW technology advantage is providing means to support groupwork over remote distances. In this way, participants who might be geographically distributed can execute tasks together without leaving their desks while keeping on using all their related local facilities (e.g. machine, software environment, archives, etc.). This avoids travel expenses and permits a best continuity of the work. For instance, participants are managed to use shared computational facilities instead of using the traditional face-to-face meetings means - paper, mail, fax, etc., which improves the performance of the work.

However, some recent research work has also included the support of computational face-to-face meetings (in the same room). This introduces a new type of CSCW systems - the co-located ones. Besides, in most of the cases the division between remote and co-located is as much a logical as a physical one and is related to the accessibility of users to each other rather than their physical proximity.

Accordingly, we can divide CSCW systems, concerning about their geographical distribution, into four main categories [17]:

i) *Co-Located*
Purely co-located systems require the local presence of all users. This class of system normally takes the form of a purpose built meeting room with a large projected computer screen and a number of personal computer linked by a local area network [10].

ii) *Virtually co-located*
This type of systems are virtually co-located in terms that they do not require participants to be in the same room. This is often completed by the use of Multimedia technology, allowing real-time audio and video links to be maintained. Systems in this category include real-time multimedia conferencing environments.

iii) *Locally Remote*
Locally remote systems are those providing high-bandwidth real time accessibility between users often using shared screen techniques. Argumentation and co-authoring systems and real-time conferencing facilities can be considered in this way.

iv) *Remote*
Purely remote systems are those that assume the existence of only minimal accessibility between

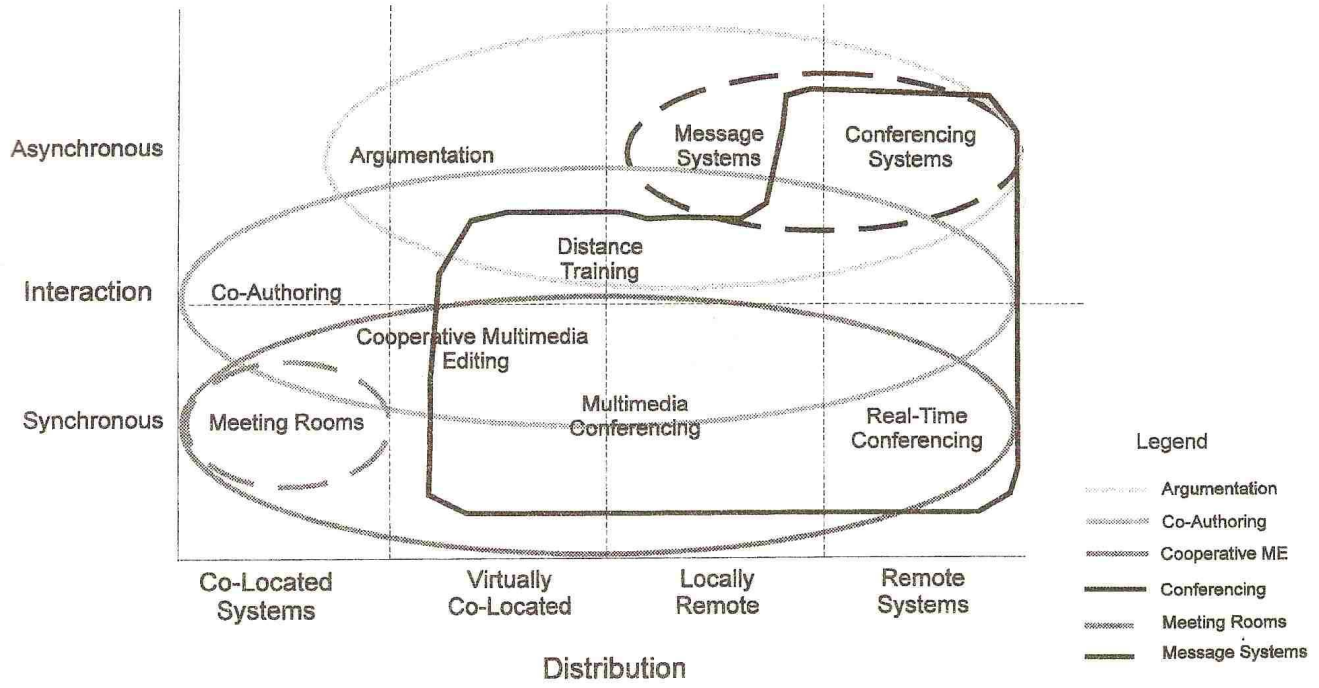



Fig.1: A classification space for CSCW systems.

users. Usually, these systems implement very large distance cooperation where participants do not have the possibility to maintain real-time communication. Message systems and computer conferencing systems which assume only rudimentar „dial-in" mechanisms are included in this category.

### CSCW systems

In this subsection we expose an overview of the four principal classes of CSCW systems. They can be classified through a combination of the two above categories. Furthermore, they represent as much a generic division of the today's CSCW systems as being a rigid classification [14].

i) *Message Systems*
These systems are those related/evolved to/from the traditional electronic mail environments which permit distributed users to exchange textual messages each other. Some recently developed systems incorporate also the possibility to include multimedia messages, as well as, executable programs.

ii) *Computer Conferencing Systems*
These systems are also related to electronic mail environments, but presenting a more complex management of the messages and concerned group. A conference manager controls the actual topic, the users roles and interaction rights. Additionally, the development of reliable high speed communications has led to the emergency of real-time conferencing facilities, allowing users to communicate in real time using audio and video media.

iii) *Meeting Rooms*
This class of cooperative systems has been developed in recent years and generally consists of a typical automated meeting room, with a large video projector, a computer (or network of computers), video terminals, a number of individual input/voting terminals, and a control terminal. These features are mainly designed to help people during the traditional face-to-face meetings.

iv) *Co-Authoring Systems*
This is a general class of systems mainly related with cooperative editing systems, where different authors are discussing together while generating a common co-authored document. This resultant document is the final product of a process of argumentation or co-authoring.

The Fig. 1 exposes the different classes of CSCW systems concerning about the form of interaction and geographical distribution.

Nevertheless, cooperative applications are often also divided between *cooperation-unaware* versus *cooperation-aware* ones. Collaboration-unaware systems are the typical standard applications designed for the use by only a single user. To use these single-user applications in a group environment, a special application sharing mechanism or conference manager is necessary, in order to allow multiple users to manage together the application. The application still remains a single-user application in terms of simultaneous input while the output is multiplexed to multiple terminals.

Only one person at a time can access the application. When that person has finished the input, other participants in the group can work with the application.


Fig.2: Collaboration-unaware versus collaboration-aware.

A special floor control mechanism handles the users input. The term collaboration-unaware results from the fact, that the application sharing mechanism encapsulates the single user application in a way that it is unaware of the multiple users in the group (see Fig. 2).

Collaboration-aware systems are especially designed from scratch to support a specific cooperative (team work) activity. They are therefore aware of the multiple users in the group and can maintain simultaneous input from multiple users. The above considered CSCW systems are mostly related with collaboration-aware systems.

Taking the especial case of Distance Training under the above four classes of systems, it can be defined as being a sub-class of co-authoring plus a hybrid combination of real-time and multimedia conferencing. On one hand, DT may involve on-line tasks where all or part of the team are simultaneously sharing the same objects and on the other hand, situations can occur where asynchronous private work is taking place. Additionally, application sharing mechanisms can also be needed if the training process involves the distribution of single-user tools. Therefore, DT systems are generally hybrid environments.

About the team itself, it includes usually only one chairman (the Coordinator) and several participants (the Learners). Additional social roles can be adopted, such as commentators who have the responsibility to follow and comment on the running training process. These comments can be further used for evaluation purposes.

Accordingly, three main phases of the training cycle can be established:

1) Training session performed by the Tutor. The Learners follow the Tutor explanations with very few interventions.

2) Individual work of the Learner (e.g., completing exercises). However, the Tutor arbitrates the process.

3) Cooperative work of a sub-group of Learners (one or more) with the Tutor (e.g., correcting exercises). The Learners can communicate with each other only with the Tutor agreement.

Based on these three phases, other sub-phases can be found, such as: private work of a sub-group of members or a common brainstorming process where everybody has freedom to intervene.

## 3 THE VIRTUALX PROTOTYPE

VirtualX is a multimedia co-authoring environment (based on X Windows/OSF/Motif) developed to support distance (or local) training of X applications. It allows teams of two to six elements who might be geographically remote located and connected over network to share X applications following a pre-defined training framework.

Users can compose together co-authored multimedia documents or simply use a graphical brainstorming area to sketch while maintaining a discussion through real-time multimedia communication.

VirtualX provides group awareness by allowing the traditional paradigms of cooperative editing: personalised multiple cursors, WYSIWIS (What You See Is What I See), social roles, users' identification, tele-pointing, multi-user interface, multi-user communication. The media available for communication are text, audio and video, and for common editing exist text and graphics. Additional annotation features are available allowing users or more precisely commentators to perform comments on the co-authored documentation and shared applications by using text, graphics and audio.

VirtualX is implemented over a centralised distributed architecture while two especial components are supporting the cooperative multimedia editing and application sharing environments.

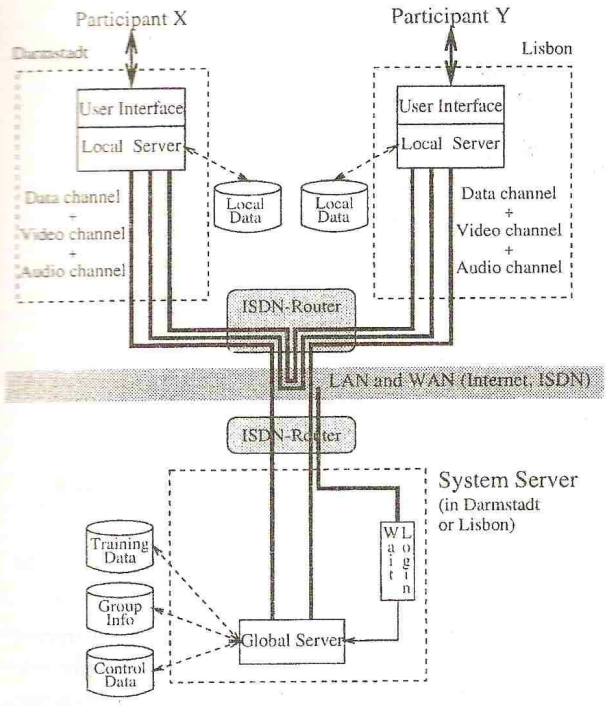



Fig.3: An overview of the main distributed architecture.

### The Cooperative Architecture

The VirtualX architecture concerns about the system organisation in terms of distribution of its physical processes and information among the different machines, as well as the way the communication is enabled. This architecture was designed having as main goal the support of the necessary means for the pretended cooperative work.

Consequently, since a global coherence maintenance must be supported, we have adopted the client/server architecture as the primary platform for the subsequent components. A central process, the *global server*, holds the responsibility to synchronise all the information flow generated within the system while each local user process maintains an external user interface.

A session manager implements the application sharing process and a transparent sketching manager supports the co-authoring environment.

### The Main Platform

As above mentioned, the main platform is a client/server distributed architecture, where a central process (the *global server*) co-ordinates all the information flow generated in the system. This information relates to editing actions, accessing rights to shared data, login requests, inter-group communication, control data of the shared applications or any information concerning about active users or state of the environments.

Usually, stable information (e.g. completed co-authored documentation) must be independently preserved from inconsistent or any new undesirable changes. Keeping it in the *global server* comes to be the adequate solution. Moreover, having only one external central process facilitates the synchronisation of the users' actions and consequently the maintenance of the global system coherence.

On the other hand, when dealing with private work, changes being generated by each user are strictly concerned with its local context, i.e., there is no need to distribute them (at least to other learners who are also engaged with their private tasks). Therefore, for efficiency reasons, users' workspaces shall maintain, more or less independently, these changes in local copies.

Accordingly, we have decided to integrate the above aspects in our distributed architecture. Thus, the *global server* coordinates/controls globally the distributed system while local structures of the users' workspaces are maintained by local servers. This assigns a hybrid character to our main platform. During the login time, each user starts a local process, which establishes a communication link with the *global server* (which is automatically started if it does not exist). Next, the *global server* sends all the information concerning about the current training or co-authoring session (if they do exist) necessary for the new user process to register itself as one more participant in the group session.

Each user process supports several structures needed to sustain locally a version of the shared objects (co-authored documentation or shared applications). Also updated copies of the editing state of the other users are kept, which permits to pursue a strong awareness within the group. The architecture also allows each user process in the user workplace to connect to another workplace via text, audio and video channels.

All the connections between the *global server* and the users' processes and between the users' processes themselves are made using Ethernet-LAN functions (TCP/IP) or ISDN-WAN [22].

The video communication uses the JPEG codec standard (see [6]) for the video frames, and VideoPix hardware or IndigoVideo on Sun or SGI workstations respectively. For audio, we provide crossplatform usability, by using a common intermediate exchange format and realising a number of on-line bi-directional converters supporting different code formats (see [23]).

Finally, the text communication follows an improved version similar to the UNIX/Talk feature. The Fig. 3 shows an overview of the main distributed architecture.

### The Transparent Sketching manager

The transparent sketching manager implements a cooperative multimedia editing environment allowing the different group participants to manipulate together a common document. Each user process holds a WYSIWIS local copy of the document while the *global server* synchronises all the editing actions in the system.

The transparent sketching manager establishes the „graphic part" of the co-authoring environment by allowing users to cooperatively sketch over a background image.

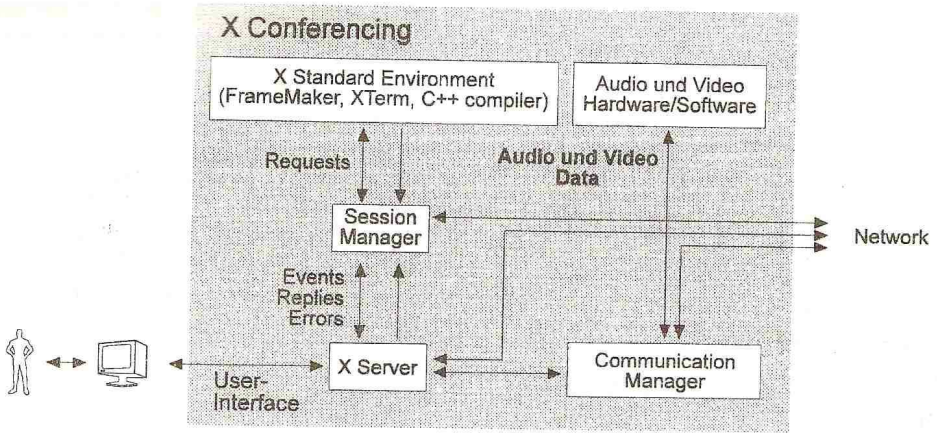



Fig.4: A single X conferencing environment. Different X applications are being controlled while and independent module supports the video/audio communications.

The coherence maintenance is supported through the synchronisation performed by the *global server*. Every editing action is sent to the *global server* and only after its agreement is definitively executed (drawn). The „agreement function" follows usually the identity behaviour, i.e., actions coming are synchronised using a FIFO (First-In-First-Out) scheme and distributed to all user processes (including the owner). However, there are situations where the „agreement" depends upon the editing rights of the user sender, editing state or any global system state variable. For instance, attempts from one user to change a protected private transparency of another one, will not be accepted.

**The X Application-Sharing component**

The VirtualX uses the MIT's X Windows system [18] where individual X applications (properly the X clients) can be shared. X is gaining ground as a standard windowing system and is currently supported by most leading workstation manufacturers including Sun, DEC, IBM, HP, Aple, AT&T or SGI.

The reason lies in its design, which appear as network/device transparent and offers a pleasant graphic user interface. X fully achieves its transparent and distributed goal, in the sense that a client, running in one machine, can work with different displays, i.e., interact remotely with different machines, which are possibly built by different manufacturers. For this reasons, X has been, for example, a particularly popular vehicle for experimentation with, and development of, shared window systems. However, the X Window System was not originally designed to support conferencing, i.e., fostering collaboration among groups of geographically separated individuals. For instance, while one user is moving a window, another one could simultaneously try to resize it. This leads to an abnormal situation for which the X servers are not prepared for [5].

In order to provide multi-user access to X single-user applications, multiplex mechanisms to distribute the output X protocol of the application to different screens, as well as, filter strategies to control/synchronise the several inputs coming from the users, must be available in the environment.

Usually, the sharing process itself defines different forms of editing rights. There are cases where only one user can manipulate the shared application while the remainder group members are simple observers. On the other hand, there are activities where the sharing process allows a complete collaborative editing, i.e., all the users have editing permission. We must observe that the shared X application is completely unaware of the multi-user accessing.

There are several features that can be provided by X multiplexors. Most of these programs support a feature called floor control. When large numbers of people are using a shared application, it is useful for one person at a time to have control of the program. This prevents two or more people from simultaneously clicking on different menu items and creating confusing results. Only the participant who has the „floor" can send commands to the application. Programs that have floor control also have mechanisms for passing the floor to others and queuing up to get the floor.

The VirtualX architecture integrates an independent component - the *session manager*, which is a X multiplexor providing floor control mechanisms.

In fact, the „virtual" designation of VirtualX results from the fact that the applications being shared just truly exist, as a duality client-server, in one machine, i.e., in the host one. All the remote users see the performed simulation in their local X servers. These remote servers, are induced to act on these applications like normal X ones. The main requirements are focused in the interception, translation and distribution of X protocol performed at the host machine. These X protocol packets flow between each shared client and all the involved X servers. The user process at the host machine creates for each shared application a link to each remote X server. Then, the X multiplexing process can freely run without the ever significant delay imposed from a *global server* control (see Fig. 4).

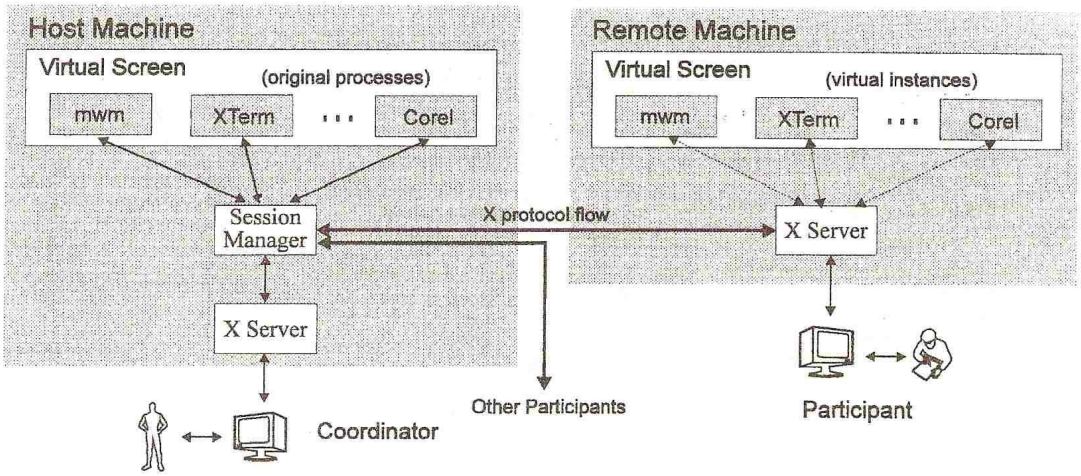



Fig.5: A distributed X conferencing. The X applications being controlled exist as processes only in the host machine. The remote machine holds simple instances of those applications.

Consequently, the *session manager* of VirtualX implements a satisfactory answer for the following requirements:

- the interception of the traffic between clients and servers and the methods to examine it and route it to the proper destinations;
- the procedures to translate X protocol requests and replies (Requests, Replies, Events and Errors) intended for one X destination (application or server) to be sent to another one (possibly remote);
- the reverse translation process for events and replies coming from multiple servers to the same single client;
- the procedure to overcome the differences in the operational characteristics used by different X servers;
- the problem of communicating with processes on different hosts that may use different byte orders to represent integer quantities;

The floor-control mechanism is usually transparent to the users. These are normally only aware of their editing rights and act in conformity. However, the system will always keep up an internal floor control because of the single-user characteristic of the shared applications.

In fact, there are three modes of user interaction control concerning a related shared application:

1) *explicit-fixed*
   Only one user can manipulate the application. Other users have only the possibility to observe (useful for on-line training sessions);
2) *explicit*
   Only one user can edit at a given time, i.e., the application is locked. However, the floor is „passable", i.e., another user can ask and get it for edition (token mechanisms);
3) *implicit*
   This mode allows simultaneous users editing.

Even so, the system will need to transparently maintain an explicit mechanism in order to avoid inconsistent editing sequences to arrive;

The quality of the system is mainly measured by factors of editing freedom. When permitted, users must have an ease application access.

### The Multimedia Co-Authoring Environment

The co-authoring process is commonly related with cooperative editing, where different authors are discussing together (arguing) while generating a common *co-authored* document. In VirtualX this resultant document can be of text or graphics. Consequently, the environment is designed to hold two cooperative editors - text and graphics (raster and 2D).

The text editor supports shared editing of a especial file - the framework. This framework keeps textual descriptions of the actual training session and helps out the group in behaving in accordance with the established tasks.

The co-authoring environment improves the DT process by providing an additional support of group work. It can be dealt as a completely independent module. The participants can ever decide to establish a brainstorming for simple discussion and/or produce a more structured document, even if no training activities are taking place.

### Transparent versus Cooperative sketching

When working together following a common goal, users usually need to have periods of private work, necessary, for instance, to complete some individual tasks or simply because the tasks of the other participants are currently not relevant. Also there are situations where one user may want to share his/her work with the current work of a specific sub-group of participants and simply observe, the current actions of another sub-group.

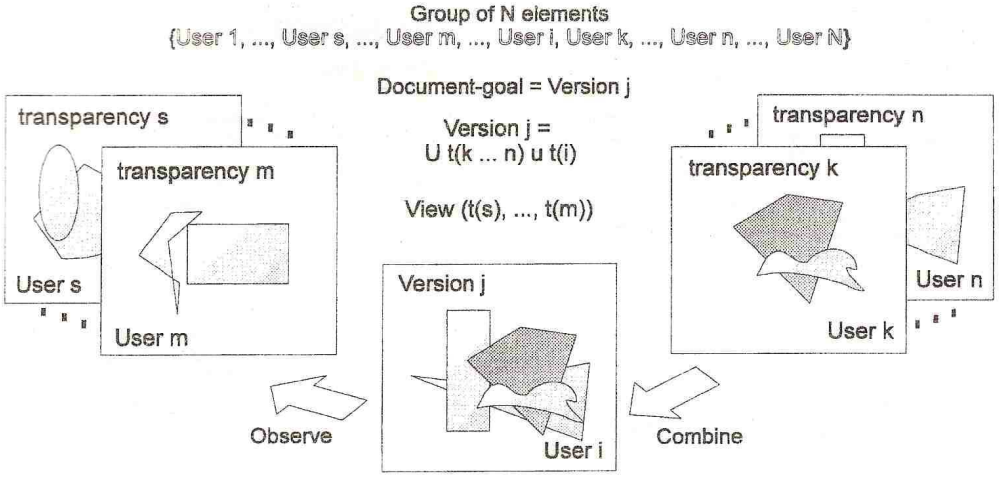



Fig.6: The transparent sketching mechanism.

This guides to the definition of *private view* or *transparency* of a given user as being his/her current editing state. The *transparency* can be observed and combined with other *transparencies* but only the user owner can change it. The *combination* of one or more transparencies generates a *version* of the common document, i.e., the document-goal (see Fig. 6).

In face of these aspects of the group work, we have decided to include in the co-authoring environment of the VirtualX two editing alternatives:

i) true cooperative
ii) semi-private

The *true cooperative* mode allow users to sketch simultaneously by combining their changes in a same common drawing, i.e., there is no private work.

The *semi-private* terminology is applied in the sense that the sketch can be view-selected. That is, users are sharing a common background while working in their own private *transparencies* and additionally having the opportunity to see and share *transparencies* of other users. This can generate different *versions* of the document-goal, which might be stored as part of a report and be subsequently recovered.
The multimedia editing process includes strongly the WYSIWIS and multiple cursors strategies.

**The Training Process**

As the DT can be defined as being a set of tasks performed by a group of people, it must include forms of group articulation in order to permit a best performance of the work. One response is the definition of social roles, which might be used to coordinate the group interactions.

In regard to the specific characteristics of the DT team and the possible different needed social roles, we have decide to have the following ones:

i) *Coordinator* - chairs the training session and has rights to change the roles of the any other participant.
ii) *Participant* - active team member contributing on the session as learner.
iii) *Commentator* - contributes to the session by commenting on the work going on with technical, quality or efficiency considerations. The system does not support any well-defined commenting task.

The social role of each of the participants in the meeting is defined during the login time of each one of the tele-conferencing environments. The whole training process is co-ordinated by using these established *social roles*.

Training processes include generally the three above mentioned main phases. These phases are quite suitable for the specific case of X applications training.

Accordingly, we have decided to have the following training cycle phases:

1) On-line training session of a set of X applications performed by the *Coordinator*. The *Participants* can however under agreement perform interruptions;

2) Individual work of the *Participant* (e.g., completing exercises). The *Coordinator* arbitrates the process;

3) Cooperative work of a sub-group of *Participants* (one or more) with the *Coordinator* (e.g., correcting exercises). The *Participants* can communicate with each other only under *Coordinator* agreement;

4) Private work of a sub-group of Participants. It is similar to phase two (2), however involving a sub-group;

5) Global brainstorming process involving the whole group. All group members (including the *Commentators*) can freely communicate and contribute to the training process;

Certainly, the existence of different levels of editing freedom does not imply any loss of control from the *Coordinator*. It has been proved that as in real live, groups require regularly the presence of a chairman.

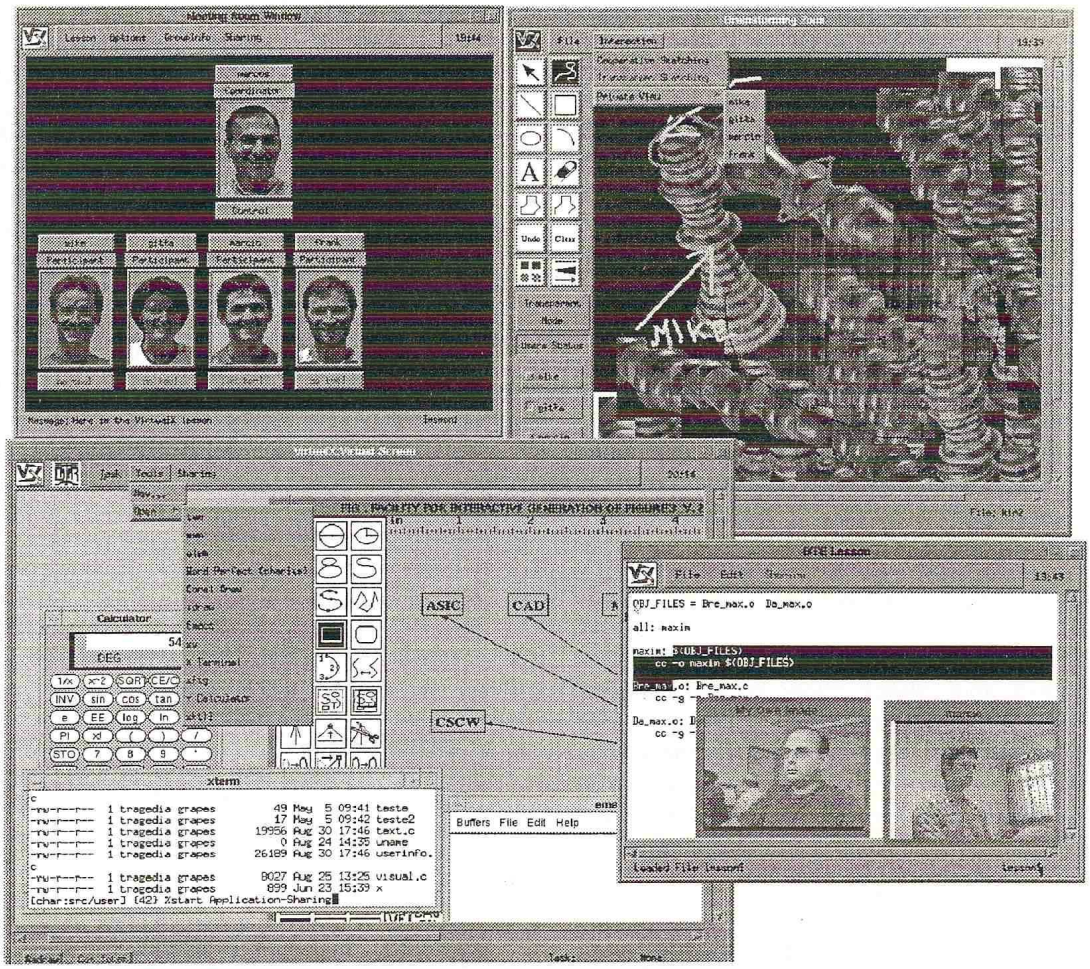



Fig.7: An overview of the VirtualX main windows. The meeting room window (top-left); the transparent sketching environment (top-right), the virtual-screen window (bottom left) with several X applications running and being controlled; the framework text editor and two video windows.

**The Class Room Environment**

As a training session can involve not only one but several applications, we have implemented the interface for the applications as a common drawing area holding a particular window manager (also cooperatively controlled) in order to support the collaborative window manipulation of the applications. This interface uses *virtual-screen* strategies (see [12]). A virtual-screen has dual identities. On one hand, it is a common window which users can open, close, move, resize, iconify, and deiconify. On the other hand, a *virtual screen*, like a real screen, is used as a container for a group of windows and when accompanying with a window manager, it allows users to manipulate individual windows in it.

In fact, the virtual-screen window holds a complete work environment, which increases the traditional two (2D) dimensional windowing screens to two and haft (2.5D). The *virtual-screen* strategies allow users to append to their original working environment, a maximal of five (5) new independent working environments (with their own window managers).

The top-level window of the class-room interface integrates a hierarchical organisation of the current participants along with other relevant information, such as: members photograph, name, editing cursor, social role, current opened communication channels, token holder, etc. This information elements play a crucial role to enforce group awareness (see Fig. 7).

## 4. RELATED WORK

Computer-Based DT can be defined as an extended form of computer conferencing. However, as mentioned, DT of applications involves a hybrid process of CSCW, i.e., it combines different aspects of group work.

On one hand, it involves real-time conferencing and co-authoring. On the other hand, forms of application-sharing, asynchronous group-interaction or private work are also fundamental to support a satisfactory DT environment.

In spite of the considerable work done in these different areas of CSCW (see [3]) only few cases have tried to joint them all in a common DT platform. Also exists a precise division between systems/solutions for application-sharing concerning about the workstations (Ws) and personal-computer (PC) platforms.

Our concrete case relates only to the Ws platforms since we are dealing with X applications.

An important representative of the DT systems is jointX [20]. It implements, for SUN-Workstations, an application-sharing environment able to multiplex X applications over long distances and using ISDN (Integrated Services Digital Network ) network. Users can share one application at time and communicate with each other using multi-point (n:n) audio connections. To access the shared applications there are implicit as also explicit floor control mechanisms. The systems allows additionally users to share private applications with a pre-defined chairman or coordinator.

jointX integrates however a poor support for group work, such as: multi-cursors, comments and annotations, group information, specific multi-user interfaces or multimedia editing. It is rather a pure application-sharing environment.

X TerminalView (XTV) [1] is another rather mentioned application-sharing system. It implements a X multiplexor plus a conferencing interface which allows users to share X applications among remotely-located workstations running X and interconnected by the Internet. This system permits simultaneously the control of several applications and uses a virtual-screen window as a container for them. XTV does not implement audio communication nor video. There is only one text communication channel available.

Examples of multimedia co-authoring systems are CoMEdiA [21] and SketchPad [19]. Both systems support cooperative multimedia editing through most of the traditional CSCW paradigms. However, both cases do not implement application-sharing and can be defined as pure collaboration-aware systems. While CoMEdiA offers a range of four different editing media (text, audio, raster graphics, 2D graphics and video), SketchPad implements the notion of different transparencies over a graphical background (raster image. Multi-(point-to-point) multimedia (text, audio and video) communication is available in both systems.

IVS [24], Communique! [11], MisterCool [9] or MMConf [2] are quite successful real-time Multimedia Conferencing systems. They support independent audio and/or video communication modules, which can be, more or less, generically integrated in more complex cooperative systems.

Altogether, we realise that most of the today's systems are stressing, more or less, only one sub-group of the cooperative work aspects. This tends to unsatisfactory solutions, especially when considering the hybrid cooperative process of Distance Training.

We argue that only an integrated solution covering as much as possible the different forms of cooperative work is able to support with success DT.

## 5. FUTURE WORK

The VirtualX system is currently being presented in its third stable version. The system is being used under real conditions by some research institutions in context of the COBRA-3 (COoperation in Bureau, Research and, Administration) project (see [4]). In spite of the tremendous work already done, we still have some considerable objectives to complete in this area.
One important facility to be integrated is the capability to have late joining. Late joining relates to all those cases when someone joins the session after it has already begun. The system must assign to the later user process the current session state and provide mechanisms for a fast integration. This comes to be critical if we take into account that the only way to uphold the current state of the X session, is storing all X protocol messages already produced (or at least up to 60% of it). For instance, a half hour session of „normal" editing generates circa 20MBytes of compressed X protocol data. Mechanisms to codify, compress, store, retrieve and re-play X protocol must be found.

Another important point being currently investigated, is the support of asynchronous cooperative multimedia editing (ACME). The fundamental advantage of such ACME facilities is to provide means to remote users to view and change with their own editing, an previous (already executed) multimedia editing session. This represents a form of meta-editing of multimedia editing sessions, in terms that the sessions themselves are dealt as editing objects. ACME is useful for long (geographic) distance collaboration where group members are distributed among different time zones, or also when for some reason the on-line work is not possible.

An ACME solution has to combine: forms of logical representation of editing actions, mechanisms to codify X protocol messages, multimedia annotations, hyper-organisation of objects and transfer mechanisms over network, to enhance an interactive codify, storing, retrieval, re-generation/re-visiting and re-editing of multimedia editing sessions. Then a training session can be codified and subsequently displayed and again changed during a different time slice.

Finally, we want to improve the cooperation mechanisms and integrity of the system in order to be used as a generic editing platform for CSCW.

## 6. CONCLUSIONS

The Distance Training of applications as a CSCW process, can be of an remarkable usefulness, specially when considering all those cases involving geographically distributed teams. DT is in fact inheriting most of the advantages related to the CSCW technology.

For instance, organisations are often confronted with the reality of having high amounts of travel expenses due to small face-to-face meetings (less of three hours)

occurring at distant places. Most of these meetings are usually involving discussion and information exchange processes where participants are only playing observer roles. In the same way, face-to-face training processes integrates commonly long periods of time with only one active contributor.

In this paper, we have proposed a global solution to support DT through a multimedia cooperative architecture. We began with a general introduction on the CSCW, by exposing the importance of this new multi-disciplinary subject in the today's research and industry community. A global overview of the principal CSCW system categories, the most important classes and a brief classification of the DT process made up the second section of the paper. Next, our own DT prototype was in detail described. The cooperative architecture, the co-authoring environment, as well as, the training were exposed. Finally, some relevant related work as also a description of some of our future work directions were outlined.

**Acknowledgements.** We thank Prof. José L. Encarnação and Prof. Manuel Próspero Dos Santos for the opportunities given, the students Markus Schmitz and Hung Le-Viet for the help in the implementation. This work is partially funded by a PRAXIS XXI-JNICT grant (CIENCIA BD/2663/93-IA).

## References

[1] Abdel-Wahab H.M., Feit M.A., "XTV: A Framework for Sharing X Window Clients in Remote Synchronous Collaboration", in Proc. of IEEE Conference on Communications Software: Communication for Distributed Applications & Systems, Chapel Hill, North Carolina, USA, pp.159-167, Apr. 1991.

[2] Barbora L.O., Georganas N.D., "Multimedia Services and Applications", European Trans. on Telecommunications, Vol. 2, No. 1, Jan./Feb. 1991, pp. 5-19.

[3] Pål S. Malm, "The unOfficila Yellow Pages of CSCW - Groupware, Prototypes, and Projects", Univ. of Tromsø, Office adr: Norwegian Telecom Research, P.O.B. 1156, N-9001 TROMSØ, NORWAY, Fax: +47-776-10262, e-mail: pall.malm@tft.tele.no, available via anonymous ftp from "gorgon.tft.tele.no" (192.135.199.112) as "/pub/groupware/cscw_yp.*".

[4] Encarnação J., Hornung CH., Berichte zu den Definitionsphasen von COBRA-1 und COBRA-3", IGD-Darmstadt, Jun. 1994.

[5] Ferreira B.G., "Transparent Sketching and Distributed Virtual X Environment in Cooperative Multimedia Systems", IGD-technical report, Darmstadt, Aug. 1995.

[6] JPEG Technical Specification", Joined Photographic Expert Group ISO/IEC, JTC1/SC2/WG8, CCITT SGVIII, Aug. 1989.

[7] Johnson P., Tjahjono D., "Improving Software Quality through Computer Supported Collaborative Review", in Proc. of ECSCW'93, Milan, Sept. 1993.

[8] Harrison W., Ossher H., Sweeney P., "Coordinating concurrent development", in Proc. of CSCW'90 (1990), ACM.

[9] Hornung Ch., Jäger M., Santos A., Tritsch B., "Cooperative Hypermedia: an Enabling Paradigm for Cooperative Work", The Visual Computer: an International Journal of Computer Graphics, Vol. 6, pp. 39-45.

[10] Kræmer K.L., Kling J.L, "Computer Based Systems for Cooperative Work and Group Decision Making", ACM Computing Surveys, Vol. 20, No. 2, June 1988.

[11] Communique! - „Turn Your Desktop Into A Global Conference Room", InSoft, Inc., Executive Park West One, Suite 307, 4718 Old Gettysburg Road, Mechanicsburg, PA 17055, Fax: 717-730-9501, info@insoft.com.

[12] Lin J., "Virtual Screen: A framework for Task Management", Dept. of Computer Science. University of North Carolina at Chapel Hill, and published at: "The X Resource 1", Winter 1992, pp. 191-198.

[13] Marcos A., "A Distributed Environment to support Cooperative Software Development", in Proc. of 5th IFIP HPN'94 (Conference on High Performance Networking), Grenoble, June 1994.

[14] Marcos A., "Cooperative Multimedia Support for Software Development and Distributed Training of X Windows Applications", Technical University of Darmstadt, Dept. of Computer Science, and also available as IGD-technical report (95i004-FIGD), Darmstadt, April 1995.

[15] Marcos A., Hornung Ch., "Using Multimedia to support Cooperative Software Development", in Proc. of VI Portuguese Conference on Computer Graphics, Braga, Fev. 1994.

[16] Rodden T., "A Survey of CSCW Systems", Dept. of Computer Science, Lancaster University, 1992.

[17] Rodden T., Blair G., "CSCW and Distributed Systems: The Problem of Control", in Proc. of ECSCW'91, Amsterdam, 25-27th Sep. 1991.

[18] Scheifler R.W., Gettys J., "The X window system", ACM Transactions on Computer Graphics, pp. 79-109, May 1986.

[19] Schendel M.G., Noll S., Rix J., "Distributed SketchPad System", in Proc. of the COMICS'91 Workshop, Chateau-de-Bonas, Gers, France, Jun. 10-12, 1991.

[20] jointX - Application Sharing System, Sietec Systemtechnik GmbH & Co. OHGCompetence Center Multimedia (CCMM) Nonnendammallee 101 D-13629 Berlin, Phone: +49-30-386-28244, Fax: +49-30-386-23780, e-mail: ccmm@kom.sietec.de.

[21] Santos A., Tritsch B., "Using Multimedia to support Cooperative Editing", in Proc. of EUROGRAPHICS'93, September 1993, Barcelona.

[22] Santos A., Marcos A., "An Algorithm and Architecture to support Cooperative Multimedia Editing", in Proc. of 4th Workshop on Future Trends of Distributed Computing Systems, September 1993, Lisbon.

[23] Tritsch B., Hornung Ch., "Cooperative Multimedia on Heterogeneous Platforms", in Proc. of Dagstuhl Workshop on Multimedia System Architectures and Applications, Dagstuhl, 1992.

[24] Turletti T., "H.261 software codec for videoconferencing over the Internet", (No. N 1834), Institut National de Recherche en Informatique et en Automatique.