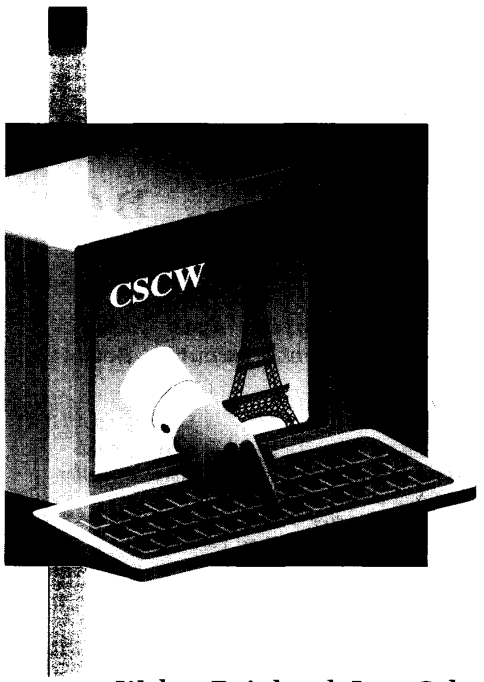


CSCW Tools: Concepts and Architectures

Walter Reinhard, Jean Schweitzer, and Gerd Völksen
Siemens Corporate Research and Development

Michael Weber, German Research Center for Artificial Intelligence

Building efficient frameworks for computer-supported cooperative work requires different approaches. The combination of CSCW transparency and awareness produces an advanced environment adaptable to specific applications

The worldwide nature of today's market has forced many companies to decentralize their organizational structures. Facing the growing complexity of commercial problems, these companies must expend considerable effort increasing the effectiveness of their production processes, augmenting their product quality, and reducing the time it takes to get their products to market. They require optimally tailored working environments, which open new application domains for computer-supported cooperative work (CSCW).

The connection of workstations into technically progressive (broadband) communication networks forms the basis for CSCW, an innovative interdisciplinary research field that is becoming increasingly important. Tools are commercially available, but they are so poorly integrated that constructing a well-tailored supporting infrastructure is far from reality.

We developed a taxonomy to evaluate CSCW tools and their relationship to collaborative processes. This taxonomy serves as a platform for specifying functional and technical requirements for CSCW systems in certain cooperative work processes. Based on this taxonomy, we follow two approaches which, when combined, provide an advanced CSCW framework for later use.

One of these approaches provides multiuser access to single-user applications. It lets users establish a group context by transparently sharing well-known single-user applications. This environment requires functionality to multiplex the output of the application to a group of users and to filter user input so that the application sees it as coming from a single user. Additional mechanisms enable users to remotely point into the shared document, and simultaneous audiovisual communication simplifies interaction on the item of interest.

The second approach is based on a network of sharable data objects accessible by various sharable tools. A virtual common workspace with a group-centered interface lets participants share the common workspace, set up conferences, see others' gestures, and hear their voices.

Integrating both approaches results in an open-system framework adaptable to specific application-domain requirements.

CSCW system taxonomy

Our CSCW taxonomy[1,2] defines and describes criteria for identifying CSCW systems and serves as a basis for defining CSCW system requirements.

Application criteria and requirements. Looking at CSCW systems from the application viewpoint, you see that certain tasks are generically present in many application scenarios. Problem-solving work, for instance, comprises such general-purpose tasks as brainstorming to generate ideas, structuring those ideas, and then evaluating them. Another cooperative tool supporting groupwork is a shared calendar, which helps manage such events as appointments, tasks, meetings, and even birthdays. Other tools help in composing and sending messages. Tools like shared editors, shared spreadsheets, and shared drawing support joint work in many domains.

Dedicated application domains require dedicated tools tailored for their use. For instance, computer-based teaching systems with remote tutors require specifically designed teaching materials. Group decision-support systems need information from the domain where they are used. To the user, a CSCW system appears complete only when specialized and generic tools are integrated.

Functional criteria and requirements. A CSCW system relates functional features with the social aspects of teamwork. Each functionality has an impact on the work behavior and efficiency of the entire group using the system. These functionalities also influence the behavior of individual group members. However, the psychological, social, and cultural processes active within groups of collaborators are the real keys to the acceptance and success of CSCW systems.

Interaction. A CSCW environment calls for certain new interaction requirements. This interaction can proceed synchronously or asynchronously, depending on the required response time. Joint editing of the same text by several authors is done synchronously, whereas sending e-mail is an asynchronous task.

Interaction can also be classified as implicit or explicit. The former relates to the object of joint work (such as shared text or images). The latter deals with direct communication among coworkers through gestures, voice, and video transfer. Interaction can be formal, if it relies on formal procedures, or informal, as in a joint session of equal partners.

Coordination. Coordination of interaction deals with communication within the group of users; it depends on the group's size and the way individual group members prefer to interact. Small groups usually need less coordination than large groups. Therefore, interaction modes change with the group's size and task.

In a brainstorming session, every group member should communicate spontaneously. In contrast, a conference format requires one person to communicate while the others listen. Hence, there are several modes of interaction control.[3] A free mode relies on agreement among the participants, while a system-based mode can be provided by the CSCW system. The latter mode uses a floor-passing strategy that grants permission to the current floor holder.

Distribution. Computers enable people to interact from remote places. A distributed working environment requires additional transfer channels for explicit interaction involving speech and gestures. Besides incurring technical problems, global distribution of cooperating partners crosses time zones and borders that pose social, cultural, and political differences (for example, languages, negotiation strategies, behaviors, metrics, and laws).

User-specific reactions. One approach for building a CSCW system is to distribute an existing single-user application via an application-sharing system. In such a system, each interaction is handled the same way regardless of individual users' roles. CSCW systems without user-specific reaction are *collaboration transparent*; for joint work, this type of system lets users work with a familiar user interface and access existing, unmodified application programs.

Certain users sometimes desire specific system reactions, depending on their individual roles or rights. This kind of CSCW system is *collaboration aware*; it "knows" the number of users and their individual roles, and is specifically developed as a multiuser application for cooperative work.

Visualization. The what-you-see-is-what-I-see paradigm determines how to visualize public data used collaboratively. Pure WYSIWIS is strongly related to collaboration-transparent CSCW systems. There are several levels of *relaxed* WYSIWIS. Users might see screens with individual layouts but identical content in shared windows, or different objects of shared data (such as different chapters of the same document in a shared editor). The application itself can provide individual views (different colors, cursor shapes, or different shared-data layouts).

Data hiding. Nobody would use CSCW if it didn't contain a way of separating private from public data. Public data should be classified as such and should be accessible to certain persons, groups, or the public in general. The granularity of data should not be restricted to file level, but to finer information units like data objects.

Status report

Our applications research focuses on three frameworks: the Joint Viewing and Teleoperation Service (JVTOS) system in the Coordination, Implementation, and Operation project on the Research on Advanced Communications in Europe program; GroupX; and the Cooperative Networking for Groups (CoNus) system.

The JVTOS works with collaboration-transparent applications and is designed to share multimedia applications among heterogeneous workstations. A first Sun/Macintosh verison is available. We plan to complete a version with full functionality by this fall.

GroupX is a Siemens MMC system developed in the German Berkom II project. With its centralized implementation, GroupX is interoperable with MMC systems of the major IT companies. It has been extended by CrystalPad, an annotation tool that works like electronic transparencies on shared applications.

CoNus is closely related to GroupX and, in fact, has been demonstrated with it in an integrated version — with CoNus the common workspace interface and GroupX used mainly as the sharing service in telecooperation sessions.

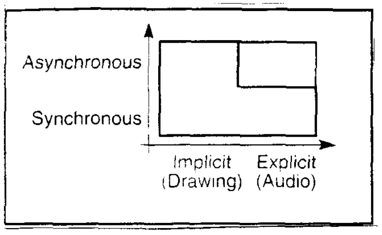


Figure 1. A 2D space of interaction criteria.

Technical criteria and requirements. Technical criteria comprise hardware, software, and network support. Hardware criteria refer to the target machine and I/O devices. Software is subclassified into the operating system, the window system, and the graphical user interface. The network aspect is subdivided into LAN, MAN, and WAN, with associated transfer protocols and rates. However, CSCW architecture can become complicated regarding data-transfer traffic, data-transfer delays, and shared-data consistency.

CSCW architecture. CSCW software systems can be subdivided into four classes of features or devices: (1) input, (2) output, (3) application, and (4) data. Each of the four can be replicated or centralized. In a replicated application, each user's machine runs the application. An electronic multiuser whiteboard provides several pens as replicated input devices, but all other features can be centralized. Centralizing the data guarantees consistency, despite high traffic to the application or I/O devices. Replicated data requires less traffic for updates, but consistency is a more difficult issue.

Flexibility

In a CSCW system, flexibility — the ability to switch dynamically between different states or modes of a CSCW tool — is a criterion rather than a property. All criteria presented within the taxonomy are evaluated on flexibility. The taxonomy can be taken as a multidimensional space, where each criterion is associated with one dimension. Each state of a CSCW system can be described as a point in this space.

Orthogonality is a prerequisite of a flexible CSCW system. A system has high-level orthogonality if the underlying concepts don't restrict each other. Low-level orthogonality exists if using one concept makes another inapplicable.

In Figure 1, the blue area represents a shared drawing tool with audio transfer. Since the tool is both implicit and explicit, it offers synchronous interaction; since it's both synchronous and asynchronous, it offers implicit interaction. Collaborative interactions occur only inside the space. The white area indicates that the tool does not support asynchronous audio transfer. A switch from synchronous to asynchronous collaboration implies that audio comments on the transferred sketch won't be possible.

Application-field flexibility. A CSCW tool that supports various applications is better than one with a specific application. Combined with an application-sharing system, single-user applications are highly flexible. Collaboration-aware CSCW tools feature the same flexibility with respect to switching applications; they can also process participant-specific information.

A modular design of applications and communication interfaces for explicit interaction guarantees dynamic switching between collaboration-aware applications. Users should be able to switch from one application to another without having to close audio and video connections.

Functional-field flexibility. Flexibility in interaction modes allows you to switch from synchronous to asynchronous mode within a cooperative work session. While preparing a document for asynchronous exchange, a mixed mode is reasonable for synchronous interaction with other participants. Further interaction modes refer to explicit and implicit communication.

The ability to run explicit and implicit communication modes concurrently is more important than being able to alternate between the two. The same holds true for interacting formally and informally. For example, for mixing formal and informal implicit interaction to help in discussing formal data (for example, a spreadsheet), it is advantageous to have informal interaction through a sketching tool.

Flexibility in coordinating a group relies on dynamic group size and variable interaction control. A flexible CSCW system should be open to any new participant. Interaction coordination is most flexible in vocal agreements, although preventing violations becomes impossible. Basic coordination via floor-passing protocols is flexible if the floor-passing mode can be changed during a session.

Technical features alone are not enough to overcome cultural differences between geographically distributed participants. Although it is not difficult to convert between different metrics or standards, other problems related to worldwide interaction (such as negotiation strategies) appear unsolvable.

A greatly relaxed WYSIWIS indicates high flexibility, since users can create individual views to shared data. The ability to switch from any level of relaxed WYSIWIS to pure WYSIWIS gives added flexibility. In data sharing and data hiding, flexibility is desirable for changing access rights so that public data can be transformed into private data and vice versa.

Technical-field flexibility. For the sake of portability and adaptability, it should be possible to perform cooperative work with different hardware platforms, operating systems, window systems, and graphical user interfaces. The system should support varying audio and video formats equally. In addition, all participants shouldn't have to have identical output devices. If video equipment is not available to one or more participants, all members should be able to communicate by voice alone.

It is important that the indefinite area in which coworkers are distributed be covered. If LANs and WANs with different architectures running different protocols are supported, geographical distribution is flexible. User shouldn't be able to detect whether coworkers are next door or on the other side of the globe.

Collaboration-transparent applications

One approach to establishing a CSCW system is to operate unmodified single-user applications within a group of users. Using telepointers, multiple users can jointly view the output of such an application and point to application windows. To jointly operate this application, it must be possible to shift the right to provide input to the application from one user to another.

This approach is independent of any application domain. Figure 2 shows the framework of supporting tools. Appropriate single-user applications are used to adopt domains; for example, a shared

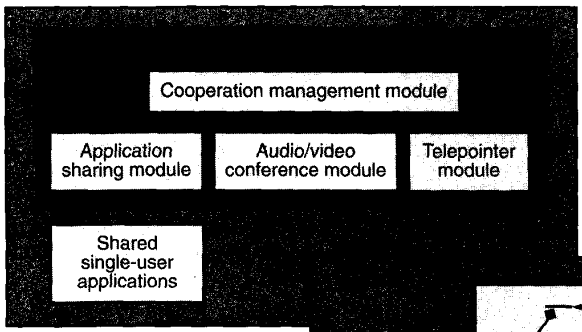

Cooperation management module
Application sharing module
Audio/video conference module
Telepointer module
Shared single-user applications

Figure 2. System overview.

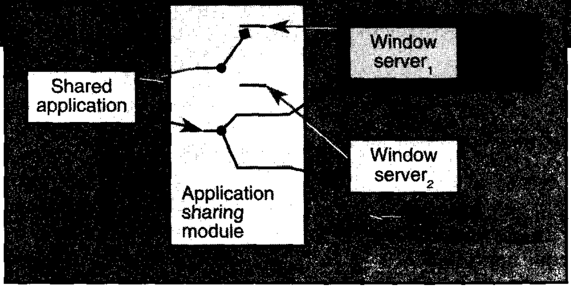


Figure 3. Application sharing by distributing the application's user interface.

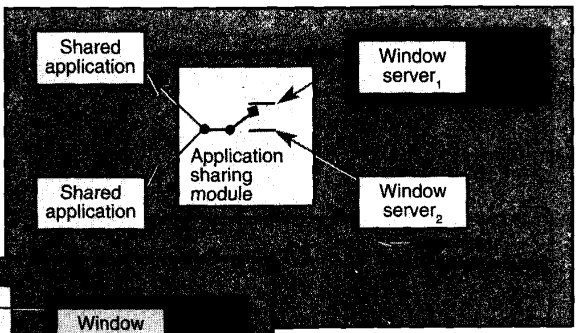


Figure 4. Application sharing by distributing the user's input.

desktop-publishing application might be used for collaborative editing. However, the shared application can only be used in a pure WYSIWIS fashion. For instance, while showing all users the same page simultaneously, the desktop-publishing application will limit editing to that particular page; users won't be able to edit different pages at the same time.

This approach presents considerable advantages, however. For instance, users don't have to learn new applications for cooperative work, and the applications can be used without modification.

Cooperation management module. The CMM is used to administer a dynamic set of participants, who can enter and exit during telecooperation sessions. The session chair, the person initiating the sessions, sets session policies and invites potential collaborators to participate. There are three kinds of sessions:

(1) *open*, where no permission is required;
(2) *joinable*, where the chair's permission must be obtained; and
(3) *closed*, where only those who are invited can participate.

Floor control. A floor-control assignment policy must be set down so that only one participant has the floor at a time during a session. The floor user has the right to provide input to the application being shared.

There is a distinction between floor policies and floor mechanisms.[4] Floor policies describe how participants request the floor and how it is assigned and released. Floor mechanisms are low-level means used to implement the floor policies.

There are three floor-control policies:

(1) The session chair assigns the floor to a session contributor, who returns it to the chair after use.
(2) The floor holder passes the floor to another session contributor who has requested it.
(3) The floor is assigned to the participant heading a request queue as soon as the current floor holder releases it.

Four primitives are required as mechanisms for implementing floor policies: (1) request, (2) release, (3) assign, and (4) revoke. Information about the participant holding the floor is relayed from the CMM to underlying services.

Application sharing module. Application sharing provides output to all session participants simultaneously while limiting the right to input to one participant at a time. The idea is to intercept window protocol traffic between the application and its terminal server. Two major variations of an application sharing module (ASM) are possible[5,6]:

(1) The application can execute on one machine, and its user interface can be displayed on several machines (for example, requests from the application to the window server can be multiplexed and sent to the remote window servers as well). The remote window servers can then produce the same graphical output on their terminals (see Figure 3). Only input from the floor holder is passed to the application; other input is filtered out.
(2) Copies of the application execute on all machines simultaneously. The floor holder's input is multiplexed and synchronously sent to all other replicated applications. Hence, all replicas are provided with the same input and present the same output to the users (see Figure 4).

In both variations, the described floor-control mechanisms have to be implemented in the ASM. In the first solution, neither the application nor its data is replicated; only the graphical user interface is distributed. The ASM has to convert the different hardware capabilities of the involved window servers to adapt the protocol streams to local characteristics of the destination sites (for example, different color setups and screen sizes, byte ordering, and so forth). Conveying the output of an application might require considerable network bandwidth.

Less network traffic is involved in the second alternative, since user input has a much lower bandwidth than application output. The conversion between window servers is also easier, since input is always directed to the local application. However, not only is replication of the application and data required, but the workstation environments (data hierarchy, system configurations, and so forth) of the partners must also be identical. The environments must be modified consistently during a session. Furthermore, the application must run on all participating machines; this fact could cause licensing problems or make applications unavail-

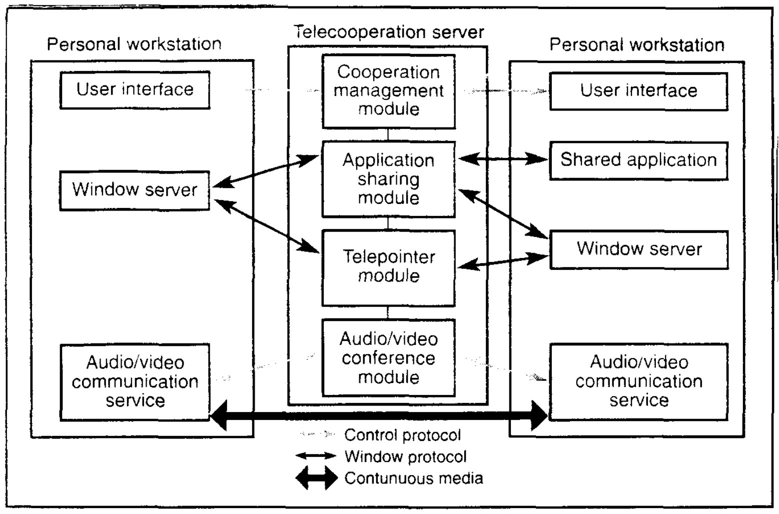


Figure 5. Centralized implementation approach.

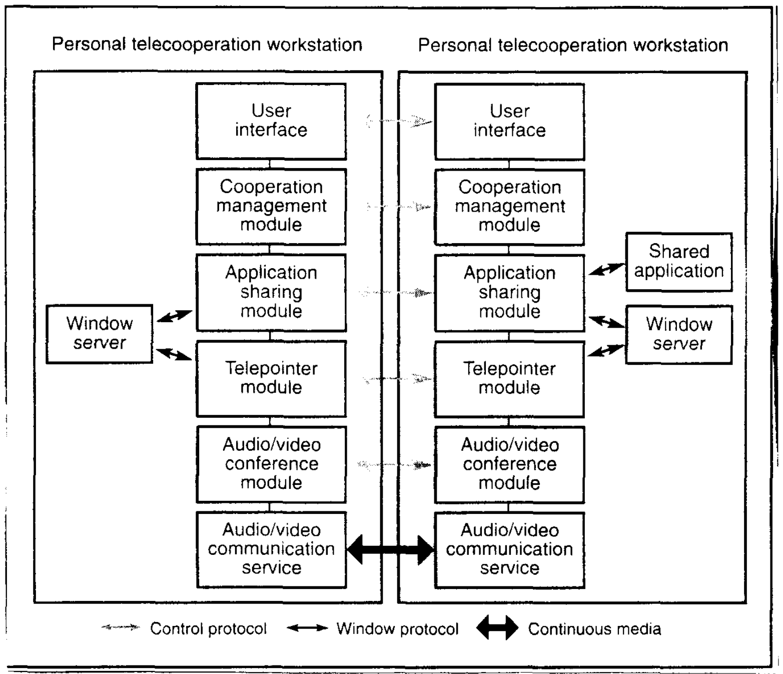

Personal telecooperation workstation
Personal telecooperation workstation
User interface
Cooperation management module
Application sharing module
Telepointer module
Audio/video conference module
Audio/video communication service
Window server
User interface
Cooperation management module
Application sharing module
Telepointer module
Audio/video conference module
Audio/video communication service
Shared application
Window server
Control protocol
Window protocol
Continuous media

Figure 6. Distributed implementation approach.

able to certain machines. Therefore, we favor the first solution.

Telepointing module. The telepointing module (TPM) provides a user with a set of globally visible pointers. When a telepointer is moved into a shared window, it becomes visible to all participants. Therefore, the TPM tracks the movements of locally used telepointers and distributes these movements to the remote sites.

Coordinate transformations have to be performed to let remote telepointers appear in the same position in the shared window as on the source site. An adjustable update frequency on remote screens keeps telepointers from turning into rubbers when being moved. (If the telepointed position is updated more often than the underlying windows are refreshed, there is a track of the pointer movements.)

Telepointers also have to be synchronized with related audio streams coming from the audio/video conference module (AVM) (see below). This can be achieved by delaying the presentation time of telepointers until the corresponding audio is received or by dropping intermediate positions of the telepointer when audio is faster.

Audio/video conference module. Cooperative work requires support by audiovisual communication, and the AVM enhances the system. The AVM is based on an A/V communication service that allows it to multiplex and distribute audio and video sources to remote participants. It also includes audio and video processing capabilities (such as compressing and decompressing A/V data and mixing audio channels). For audio and video, data isochronous transport services are required between participants. Thus, with such a transport system, suitable quality-of-service parameters must be negotiated.

Implementation issues. When structuring a telecooperation system like the one described above, a major concern is the choice between centralized and distributed implementation.

Centralized approach. Here, the implementation follows the client-server model. Each of the modules runs as a server process on a dedicated server machine. Local machines can only let the user communicate with the central components. Thus, user agents are the clients who request certain functionality from the server modules (see Figure 5). Specific protocols are needed to communicate with the server modules. Even tasks that could be performed locally have to be requested from the server (for example, telepointer positions are sent to the central telepointer module, which in turn sends this information back to the local window server to display the telepointer). To avoid such indirect routing for A/V data, each local machine runs its own A/V communication service.

The advantage of the centralized approach is that all information is inherently kept in a consistent state. Network traffic might be high, since all data arising locally has to be propagated to the central server — even when this data is only of local interest. The centralized approach is viable for closed, relatively small user groups and works well in LAN environments. More widely dispersed users need

Table 1. Flexibility of a CSCW system sharing collaboration-transparent applications.

Area	Flexibility	Description
Application field	Yes	Can use single-user applications for any domain. The only domains not covered are those where explicit group knowledge is required.
Functional field		
Interaction	Restricted	Only synchronous, no asynchronous, interaction is possible. Interaction can be implicit (using the shared-application facility) or explicit (using the audio/videoconference facility).
Coordination	Yes	Supports dynamic group sizes. Coordination is based on a floor token mechanism.
Distribution	Yes	The participants can be dispersed worldwide.
User-specific reactions	No	All users receive the same reaction from the shared application. Only cooperation management produces different reactions with respect to user roles.
Visualization	No	Windows can only be moved; no individual views are possible.
Data hiding	Restrained	Data is located at the site executing the shared application.
Technical field	Yes	Supporting different hardware systems and coding formats is possible. Converters required to mediate.

high-speed networks not only for continuous media transfer but also for other data transmission (for example, shared applications, control information, and so forth). The German Berkom II project[7,8] uses a centralized MMC system approach similar to the one we describe.

Distributed approach. Here, the implementation follows a peer-to-peer model.[9] Each module runs a local agent on each participant's machine, and each agent has the same potential functionality (see Figure 6). Thus, the telecooperation system can run even if only two machines are connected. In this case, it's easier to incorporate fault tolerance against machine failures.

The agent communication hides the hardware and the agents' implementation-specific interior and leads to a common protocol interface of agents belonging to the same type. Communication between agents for different modules is a workstation-local matter. Network communication between agents can be aimed directly to the recipient. Only data updates have to be propagated, thus reducing the network load considerably. Still, distributed data requires consistency.

The distributed approach offers higher telecooperation service scalability for large open-user groups. It also facilitates incorporation of heterogeneous workstations. The Joint Viewing and Teleoperation Service system in the Research on Advanced Communications in Europe's Coordination, Implementation, and Operation project[9,10] follows this approach.

Flexibility. Applying the previous taxonomy, Table 1 evaluates the flexibility of the CSCW systems we described. Many requirements are fulfilled, but when group-specific reactions are needed or where group-specific application data or even group-specific applications are concerned, the Sharing Collaboration-Transparent Applications approach lacks almost all support and functionality.

Group cooperative networking

We have found that several requirements are not satisfied by a CSCW system such as the one described above. Nevertheless, it does provide a framework so that you don't have to start from scratch. We plan to integrate the modules we described above by reusing them within a more powerful framework called CoNus (Cooperative Networking for Groups).

The CoNus architecture provides subsystems for

- audio and video communication (reuse of the AVM module),
- the shared-window system environment (reuse of the ASM and TPM), and
- the collaboration-aware CoNus framework.

Additional subsystems can be integrated later (for example, with an annotation tool).

Figure 7 shows the overall architecture that satisfies the requirement of separating explicit interaction transfer by A/V from other collaborative applications that represent mainly implicit interaction. The shared-window subsystem is important in the distribution of standard single-user applications. The CoNus subsystem provides a framework that controls the A/V subsystem, the shared-window subsystem, and an overall user interface. We are developing this framework so that it can easily be extended by further problem-specific group applications.

The CoNus subsystem. This subsystem provides users with collaboration-aware

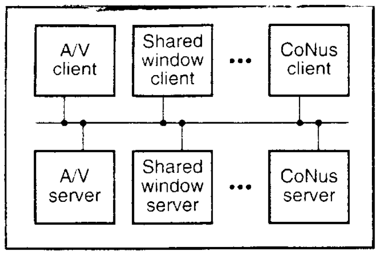


Figure 7. CoNus architecture.

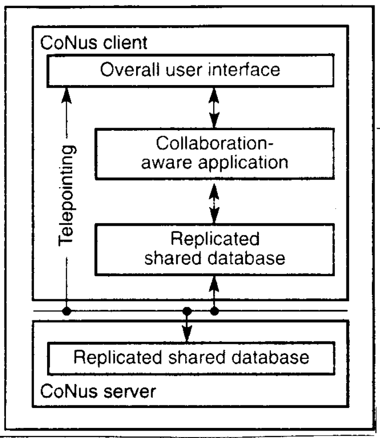

CoNus client
Overall user interface
Collaboration-aware application
Replicated shared database
Telepointing
Replicated shared database
CoNus server

Figure 8. CoNus subsystem architecture.

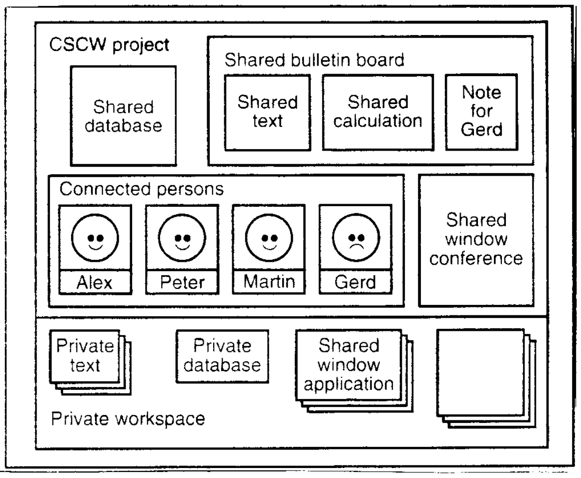


Figure 9. Appearance of common workspace.

applications for management tasks and offers a framework that includes access to the A/V and shared-window subsystems. It provides a user interface that is easy to handle, a set of basic applications that complete those for the shared-window subsystem, and sharable replicated data for managing collaboration, as indicated in Figure 8. CoNus data is organized in a database that includes references representing real-world associations.

CoNus is highly scalable with respect to technical issues and will run on various machines with different operating and window systems. Integration of audio and video connection requires special I/O devices that are handled flexibly should they lack video boards or cameras. As Figure 8 shows, data and applications are managed in a replicated architecture.

Interfaces for human-human interaction. User interfaces for CSCW systems and applications include techniques for human-computer interaction and human-human interaction. These two modes cannot be strictly separated since, for example, setting up a conference requires action with a conference tool and includes interaction with conference participants.

We decided to implement a common workspace area where the interactions take place; outside this area, applications and data are private. Opening the common workspace indicates the user's readiness for interaction with others. An important parameter for initializing the workspace defines the group or project whose members are allowed to access each other's data.

The common workspace is flexible regarding group size: For someone working alone, it runs like a single-user workspace. However, as soon as two or more people open their workspace applications for the same group or project, the users are notified about the new cooperative state, and A/V connections start automatically. All users can interact via audio and, where possible, video communication. They can also see which other projects or groups are running in their workspaces.

Users who don't have permission to enter a certain workspace can request temporary membership; this can be granted by the group members currently in the workspace. Users with limited permission acquire the same rights as original group members. This limited permission can be withdrawn any time and terminates when all regular members or the users who have it leave the common workspace. Just as people get the right to access data and applications when entering a common workspace, data moved into the common workspace becomes accessible to all connected users.

A WYSIWIS mode is in effect when multiple users are interacting in the common workspace; that is, all members see the same workspace layout. However, individual workspace sizes and locations can vary. To provide transparency among the members, each mouse pointer is remotely displayed. This technique shows which user is accessing which data or application at a given moment, so long as the group of cooperating users is not too large (that is, up to 40 persons).

As Figure 9 shows, the common workspace includes several icons or subareas that represent shared data or applications. A specific application icon starts a shared-window conference. Dropping a single-user-application icon onto the shared-window-system area starts joint work with this application.

One workspace area serves as a shared bulletin board. Users can drop data icons onto the bulletin board and open them to start working there. This provides synchronous loosely coupled collaboration. WYSIWIS is relaxed when all users see who else is accessing the shared data; however, they won't see exactly what each other user is doing. Furthermore, the shared bulletin board can be used for asynchronous work if data is not removed and others can read or write it later.

If shared data also contains private sections, these sections won't be visible to other group members. Data left open on the bulletin board can be filed in a shared database system that supports project-oriented data management.

Users can also work autonomously. A group member wanting to leave the work place and work on the network can move a copy of all the shared data with access rights to a portable machine and continue working there. Although work done autonomously can cause inconsistencies with work done jointly, the inconsistencies will be detected when the user reconnects to the system. The reconnecting user will be notified about specific inconsistent data and the coworker(s) he or she conflicts with. Resolving conflicts is initially left to the individuals, but suggested solutions can be automatically generated later. The autonomous work mode is supported by the replicated architecture with respect to data and applications.

Sharable collaborative applications. We decided to focus on the management workfield in the first approach. Two basic applications are important for managing cooperative work: (1) a calendar tool and (2) a person and address management tool.

These applications should depend on each other, and it should be easy to switch between them. Take, for example, a calendar tool that shows a meeting appointment. Selecting and opening the meeting object displays the names of the meeting participants. Selecting and opening the participant data activates the person and address management tool, which shows the public data associated with the person. This is implemented using the database approach and references between the applications.

There are special files for each kind of application, and each application is associated with a data file that can be visualized and accessed by the application. Figure 10 shows how application connection is achieved. Making required applications switches is easy.

The application connection requires a certain amount of effort if existing tools need to be modified. The CoNus prototype contains applications that demonstrate the connection capability described above (rather than highly sophisticated application functionalities). Figure 11 shows how the calendar application visualizes corresponding data.

A management tool is planned that helps users negotiate times for meetings and events. This same tool will help with project plans relative to tasks and milestones. It can contain both private and public data; private data is not visible when the calendar tool is shared. Another important feature automatically indicates free time slots for group meetings.

The person and address management tool integrates users into organizational units and projects. With the group and project membership, users gain access rights to associated common workspaces that provide security for group and project

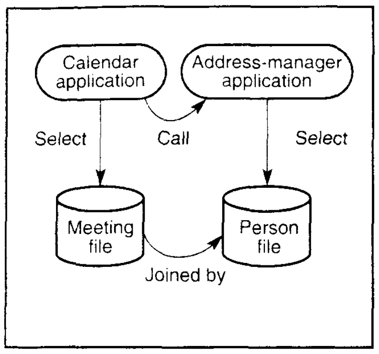


Figure 10. Application connections.

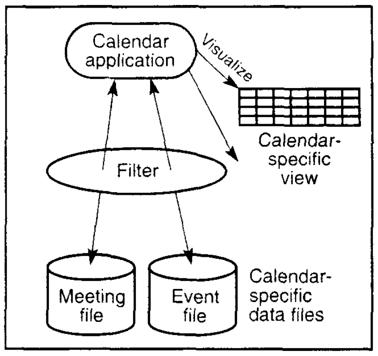


Figure 11. The application as a controller of data and view.

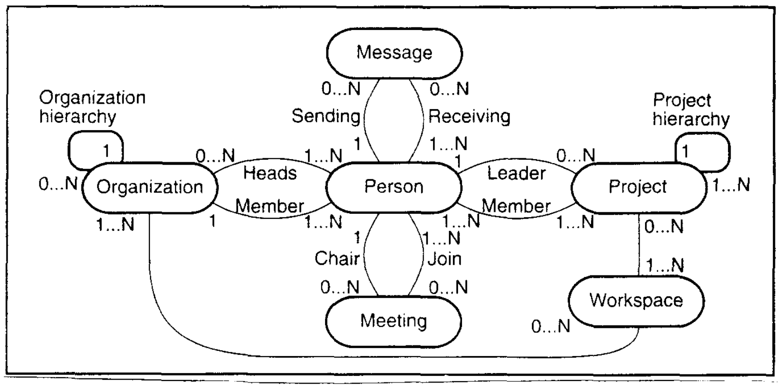


Figure 12. Entity relations for a person.

data. User-related data such as addresses and birthdays are easy to work with and share.

Data security must be heeded. If someone gives read access of personal data to someone else, the recipient must not be allowed to copy or use the specific data.

As described above, applications have one thing in common: They offer basic database functions. They serve as filters, showing only data the user has at least read access to. From assorted accessible data, the user can select specific information of interest and also view isolated data. Visualized data can be sorted according to any associated attribute. Since the data is organized with relationships in a database and their applications are connected, no further association function is necessary.

Databases. Organizing shared data in a database is an appropriate way to represent all the relationships among the entities involved in cooperative work. The person should be the center of interest in a CSCW system, and an entity relationship graph can be constructed around the person entity.

People belong to or head organization units, chair or participate in meetings, and send and receive messages and mail. Organizational units and projects are usually embedded in a hierarchy. Figure 12 shows a small portion of a person's relationships. People can execute tasks, schedule meetings, and own and share documents. The list in Figure 12 is admittedly short and could be extended quite a bit.

Another approach to define an entity relationship graph can start with the document entity (*anything*), the event entity (*anytime*), or the common workspace (*anywhere*). While the first step defines the related data for one central entity, the second step introduces new entities to extend the definition of relations. For example, a meeting is joined and chaired by people, takes place in a common workspace, has an agenda (a sequence of tasks), and needs a set of sharable documents.

The database can be replicated on the server and the client machines to facili-

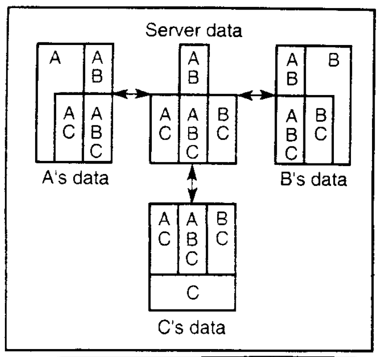


Figure 13. Shared-data replication among the server and three clients.

tate switching into an autonomous work mode. The applications will also be replicated; this should cause no problem, as long as their functionality is not changed. However, data consistency can be a problem, especially when one, several, or all users work autonomously. This is why the CoNus server should keep a copy of the replicated database, as described in Figure 13 for three users.

Our computer-supported cooperative work groupware lab is involved in several Siemens projects as well as other projects funded by German Telekom and the European Commission. Application domains include maintenance, remote troubleshooting, computer-based training with remote tutors, and multimedia telecooperation. We are making prototypes available this year. ■

Acknowledgments

We thank all our colleagues within and outside Siemens who are involved in the Berkom II, RACE CIO, Aramis, Ecole, Malibu, CoNus, and WiTek projects for their valuable contributions and discussions.

References

1. G. Völksen, "Approach Strategies to Groupware," *Proc. Groupware 93 Europe Confs.*, 1993, pp. 483-497.

2. A. Jarczyk, P. Loeffler, and G. Völksen, "Computer-Supported Cooperative Work: State of the Art," Version 1.0, Siemens AG tech. report, Munich, 1992.

3. K. Watabe et al., "Distributed Multiparty Desktop Conferencing System: MERMAID," *Proc. CSCW 90*, ACM Press, New York, 1990, pp. 27-38.

4. T. Crowley et al., "MMConf: An Infrastructure for Building Shared Multimedia Applications," *Proc. CSCW 90*, ACM Press, New York, 1990, pp. 329-342.

5. W. Minenko, "Transparent Application Sharing Under X Window," German Research Center for Artificial Intelligence (DFKI) tech. report, Siemens AG, 1993.

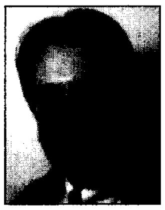
6. J.E. Baldeschwieler, T. Gutekunst, and B. Plattner, "A Survey of X Protocol Multiplexors," *ACM Computer Comm. Rev.*, Vol. 23, No. 1, Apr. 1993, pp. 16-24.

7. S. Cronjäger, W. Reinhard, and J. Schweitzer, "Functional Components for Multimedia Services," *Proc. Int'l Conf. Comm.*, 1993, IEEE Press, New York, pp. 1,563-1,568.

8. R. Herrtwich et al., "The Berkom MMC Service," *Proc. ACM Multimedia 93 Conf.*, ACM Press, New York, 1993, pp. 457-463.

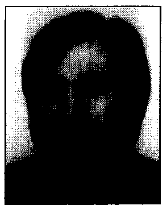
9. G. Dermler et al., "Constructing a Distributed Multimedia Joint Viewing and Teleoperation Service for Heterogeneous Workstation Environments," *Proc. Fourth IEEE Workshop Future Trends of Distributed Computing Systems in the 1990s*, IEEE CS Press, Los Alamitos, Calif., Order No. 4430, 1993, pp. 8-15.

10. W. Bauerfeld, "RACE Project CIO (R2060): Coordination, Implementation, and Operation of Multimedia Tele-Services on Top of a Common Communication Platform," *Proc. Int'l Workshop Advanced Comm. and Applications for High-Speed Networks*, Siemens (ZDM SV Log3), Munich, Germany, 1992, pp. 401-405.



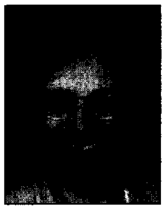
Walter Reinhard joined Siemens Corporate Research and Development in 1987 and has been a member of the CSCW and telecooperation research group at the German Research Center for Artificial Intelligence (DFKI) since 1990. His main research interests include the design, implementation, and security of telecooperation systems.

Reinhard received a diploma in electrical engineering from the University of Saarland in 1987.



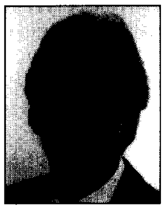
Jean Schweitzer, who joined Siemens Corporate Research and Development in 1986, is head of the telecooperation research group. His interests include the design and realization of multimedia CSCW systems, and their validation in field trials within representative closed user groups.

Schweitzer received a diploma in electrical engineering from the University of Karlsruhe in 1978 and a PhD in electrical engineering from the University of Saarland in 1987. He is a member of the IEEE Computer Society.



Gerd Völksen joined Siemens Corporate Research and Development in 1989 and has been a member of the CSCW and telecooperation research group since 1992. His research interests include CSCW system design, cooperative user interfaces, 3D collaborative modeling, and groupware for nonstandard applications.

Völksen received a diploma in computer science and physics from the University of Dortmund and a PhD in computer science and mathematics from the University of Marburg.



Michael Weber is a member of the telecooperation group at the German Research Center for Artificial Intelligence in Saarbrücken and will join the Distributed Systems Department at the University of Ulm this summer as a professor. His research interests include the design of telecooperation and CSCW systems, and distributed systems and architectures.

Weber received a diploma and a PhD in computer science from the University of Kaiserslautern in 1985 and 1990, respectively. He is a member of the IEEE, the IEEE Computer Society, and Gesellschaft für Informatik e.V.

Readers can contact Reinhard, Schweitzer, and Weber at the Siemens Project Office, German Research Center for Artificial Intelligence, D-66123 Saarbrücken, Germany; e-mail {reinhard, schweitzer, mweber}@dfki.uni-sb.de; and Völksen at Siemens Corporate Research and Development, Otto-Hahn-Ring 6, D-81739 Munich, Germany; e-mail gerd.voelksen@zfe.siemens.de.