ELSEVIER

Computer Communications 19 (1996) 590–596

computer communications

CCDES – a collaborative compound document editing system

Chien-Hung Lin, Shih-Chan Huang, Shyan-Ming Yuan*

Department of Computer and Information Science, National Chiao Tung University, Hsinchu, 30050 Taiwan, ROC

Received 29 June 1994; accepted 9 January 1995

Abstract

A collaborative editing systems allows co-authors at different locations to edit a shared view of a single document simultaneously. A compound document binds various types of information to create a single seamless presentation. A collaborative compound document editing system is developed to combine both the systems described above. It supports distributed editing with replicated compound documents, and integrates notification mechanisms into concurrence control modules. The real-time conversational facilities and the mechanisms for tolerating the process faults are provided. The collaborative compound document editing system is endured with the coordinating and data exchanging capabilities. This paper also discusses design issues such as multi-user interfaces and the presentation of compound documents, and proposes our approaches.

Keywords: Distributed computing; Compound document editing; Collaborative computing; Multi-user interfaces

1. Introduction

In recent years, because of the progress in computer networks and multimedia, the combination of these technologies makes distributed multimedia applications possible. Through a computer network, a distributed application can share each member's resources and allow users in different locations to coordinate their activities or to accomplish a common task at the same time. In addition, multimedia applications enhance the expression of electronic documents, and facilitate vivid presentation and communication among users.

The Collaborative Compound Documents Editing System (CCDES) is one such distributed multimedia applications. It enables co-authors to produce compound documents collaboratively, and to share the same contents simultaneously. To perform such work, there involves frequent interactions among co-authors. The system allows the members of an editing group to exchange ideas and information through networks in real-time. Just as a chemical compound binds different kinds of atomic elements, a compound document binds several types of information to create a single seamless presentation [1]. In other words, a compound document contains text, graphics, images, a spreadsheet and other new information types.

To implement a collaborative compound document editing system, there are three fundamental strategies: an all-in-one integrated system, a plug-and-play component system, or a hybrid system which combines the two approaches [1].

In an integrated system, the editing and communication tools are really just different modules within the same program. A single monolithic program provides all of the requisite functionalities. The chief advantage of such a system is that the individual tools can be tightly coupled for smooth operations. There is a unified user interface, and there won't be any data exchange problems between the built-in tools. The main problem with the system is that they must handle many things well instead of just one. Each of the editing tools included must provide similar functionalities and usage of single editing tools, or no-one will use the system to do 'real work'.

In a plug-and play component system, existing single user and single media systems are loosely coupled and are coordinated to present the illusion of a single 'virtual' document. Co-authors express ideas using a set of services provided by external software packages. Developers can reuse the work of others and avoid a duplication of effort. Users will benefit from the ability

* Corresponding author. Email: smyuan@cis.nctu.edu.tw

0140-3664/96/$15.00
PII S0140-3664(96)01048-1

to adapt a system's behaviour to their individual needs. However, there are a lot of difficulties with this strategy, such as the coordination and communication between different systems, the multi-user interface and media integration, and the license for each existing system.

The hybrid approach is to create a system kernel which provides integrated editing capabilities for fundamental document types, but otherwise serves as a coordination framework for a component system.

We have adopted the integrated model. The main reason is that we wish to have a prototype system as soon as possible, and we could not get the source programs and licenses of requisite stand-alone tools to build a component system. In our system, CCDES, there is a monolithic program which provides all of the editing and communication functionalities. To follow the user's customs, we emulate the manipulating convention of MacDraw.

The remainder of the paper is organized as follows. Section 2 shows some design issues and our approach. Section 3 briefly describes our collaborative compound document editing system. It supports distributed editing with replicated data, and provides a chairperson for consistency, as well as simple notification mechanisms. Section 4 describes related works in this area. Section 5 describes the implementation of the system. Section 6 concludes this paper, and gives ideas for future research.

2. Design issues and design policy

The design issues of the collaborative editing system could be classified according to the system modules of the whole editing system which includes multi-user interfaces, concurrency control and other issues about the distributed system module. In this section, the design issue and our design policy will be presented.

2.1. Multi-user interface

The multi-user interface is different from the single-user interface. It should describe the group activities to let all members in the system know what is happening.

2.1.1. View-sharing paradigms

WYSIWIS (What You See Is What I See) is the most popular abstraction of the view-sharing paradigm in collaborative editing systems. In a WYSIWIS system, each co-author can watch the entire sharing document through the multi-user interface module. Most of time, the authors need a private work space to finish an atomic operation. The content of the private work space is too trivial for the other co-authors to know. The authors may have a private work space which is a subset of the whole shared document.

In CCDES, each co-author in the system has a private work space. While the co-author constructs an object in the private work space, the view of the other co-authors will not be disturbed. When the object is constructed completely, it will be bound to the compound document. Each member in the group will not discover an object of the compound document until it is constructed completely.

2.1.2. Group focus and distraction

In a single-user interface, the author usually has a mental note of the editing document when he or she is working on the document. Because each change of the document content is made by the same author, it is acceptable that there is no contextual clue about the processing object.

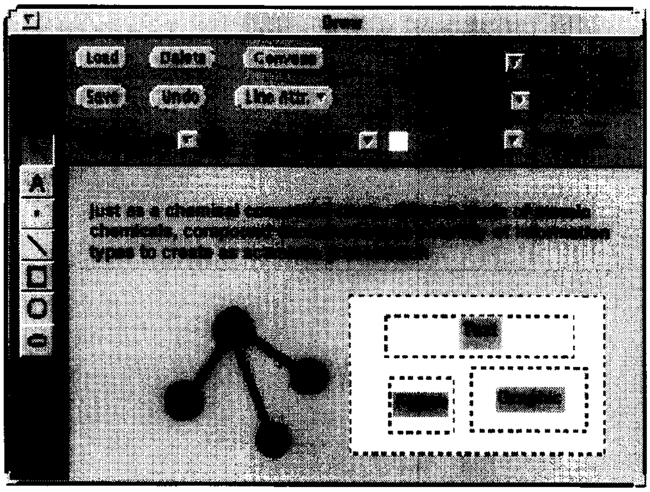
On the other hand, it is impossible for each co-author in a collaborative editing system to keep a mental note of the shared document. The contextual clues about which object is processing are needed. In CCDES, the object in a document should be locked before it can be processed. On the display, each locked object is dimmed. The co-authors will know which objects are processed if they are dim.

2.2. Concurrence control

Collision avoidance is the most common method for concurrency control in all distributed systems. In CCDES, the central coordinator control is adopted to avoid collisions. The coordinator who is the chairperson of the group receives the locking requests from group members. Each object in the document can be accessed by an author after the author gets the lock of the object from the coordinator.

2.3. Revisability

A compound shared document should be stored in a



Just as a chemical [illegible] chemicals, compound [illegible] of information types to create an [illegible]

Fig. 1. Hierarchical documents.

revisable form. A PostScript document is non-revisable. All information which includes text, graphics and image is mixed together and cast into a static file. It is difficult to modify information in a PostScript file in a collaborative way. There may be a better document format to store information in storage than the format of the PostScript file. In a compound shared document, all information is broken into a hierarchical structure (Fig. 1). Users can edit any subdocument of the whole document independently. Each subdocument can be stored in a different host where the document root is. In CCDES, the smallest unit of information is an object. Several objects can be composed of a hierarchical structure. The compound shared document can keep its visual integrity even if the subdocuments are manipulated by unrelated processes or stored in different files.

2.4. Document sharing

To collaborate on the production of compound documents, it is necessary to provide a mechanism through which the co-authors can access one copy of a shared document, instead of the traditional mechanism which is to mail different versions of the documents back and forth through all collaborative processes. In CCDES, the data access mechanism is via a central coordinator. This mechanism allows all co-authors to share the same document simultaneously. The information of the shared document will be replicated to all members of the editing group. With a central coordinator control and data replication, the information in the shared document can be guaranteed for data consistency.

3. System overview

The CCDES is implemented to support multi-users in editing shared compound documents collaboratively in an X-windows environment. The hardware configuration and software architecture are described as follows.

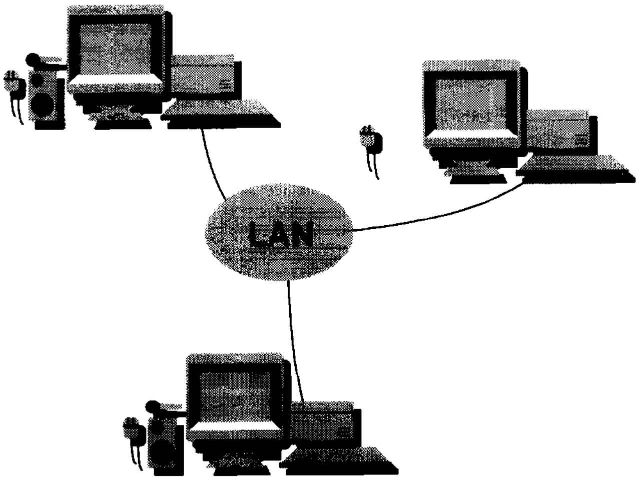


Fig. 2. Hardware configuration of CCDES.

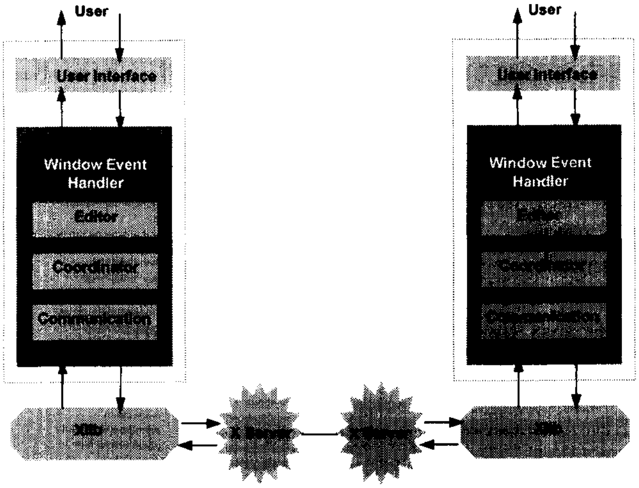


Fig. 3. Software architecture of CCDES.

3.1. Hardware configuration

The CCDES prototype is implemented on Sun SPARC IPC workstations. The network environment is the NCTU campus network, which is a subnet of the Internet. In the CCDES system, each workstation is equipped with a display, mouse, keyboard, microphone and speaker (the microphone and speaker are optional). The hardware architecture of the CCDES is illustrated in Fig. 2.

3.2. Software architecture

The CCDES prototype is controlled by a monolithic program which consists of the user interface and event handler modules. The software architecture of CCDES is illustrated in Fig. 3.

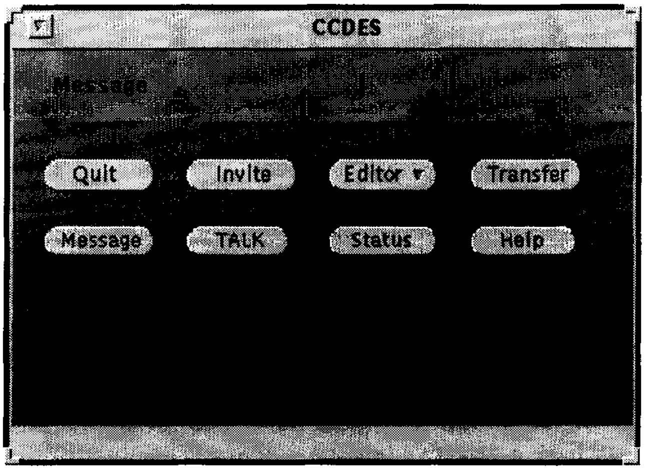


Fig. 4. Main menu interface of CCDES.

3.2.1. User interfaces

All user actions are interpreted by the user interface modules, which make appropriate calls to other modules. The attributes of the user interface are inherited from the X-window event handler. All functionalities are displayed on the window panel (Fig. 4) in the form of menus or buttons. In addition to the window panel, CCDES also provides a system status report window, a communication window and real-time audio communication.

3.2.2. Event handler

The event handler is the core of CCDES. When an event occurs in the X-window system, the X-window system passes the event to the event handler. According to the different event type, the event handler dispatches the event to different subsystems, such as the compound document editor, coordinator or communication modules.

Compound document editor

The compound document editor consists of a shared canvas, a drawing mode panel and an editing command panel. The shared canvas is the workspace of the collaborative editing. All actions on the shared canvas will be shown on other co-authors' canvases. Thus, every member has the same shared canvas content. The drawing mode panel provides several drawing mode selections, such as text, hand-written, rectangle, circle and ellipse from which users can select. The editing command panel supports several editing mode selections, such as line attributes, text fonts, background and foreground colours.

The content of the compound document is composed by different types of objects. The co-authors can move, resize or delete the objects of the document freely. All objects in the document are persistent. The chairperson can keep the objects of the document for future use. The compound editor is shown in Fig. 5.

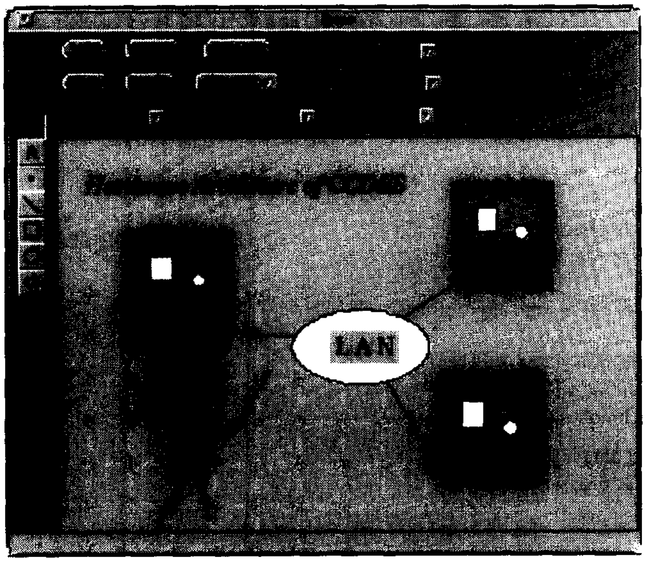


Fig. 5. Compound document editor of CCDES.

Coordinator

The coordinator maintains the order of all requests which come from all of the co-authors, so that all shared data accesses which co-authors request will not conflict (Fig. 6). Three features of the coordinator are described as follows:

1. *Mutual exclusion*: an object in the shared document can be accessed by one user at one time.
2. *First Come First Served*: the operations of the CCDES system are accomplished in the order of the requests received by the coordinator.
3. *Fairness*: no co-author process has a higher priority than others.

Communications

To facilitate group activities, CCDES supports several forms of communication facilities among the group members, such as media file transfer, message transfer, multi-way text-based talk and a real-time spoken subsystem:

1. *Media file transfer*: the media file transfer facility supports users in transferring their files that are different types of format, including text, image, voice and graphics. On the receiver side, the facility can recognize the format of the file and represent it on the receiver's display or audio facility.
2. *Message transfer*: the message transfer facility is an on-line textual message broadcasting system in which the user can specify the receivers from among all the members. The primary purpose of the facility is to let participants exchange messages privately.
3. *Multiway text-based talk*: the multiway text-based talk facility is a real-time interactive talk facility (Fig. 7). The content of the discussion can be saved.
4. *Real-time spoken subsystem*: the real-time spoken subsystem allows users to discuss verbally in real-time using the Sun workstation audio device and

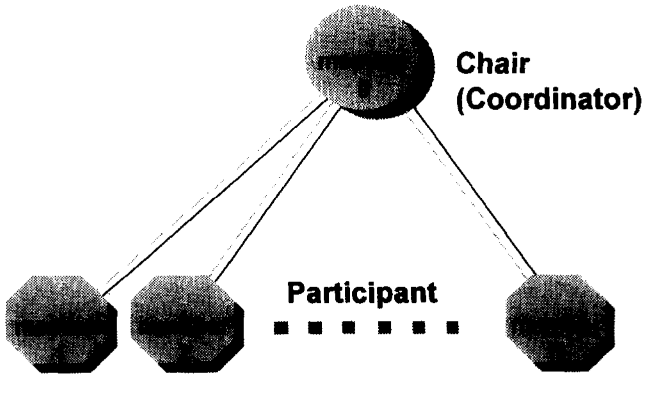
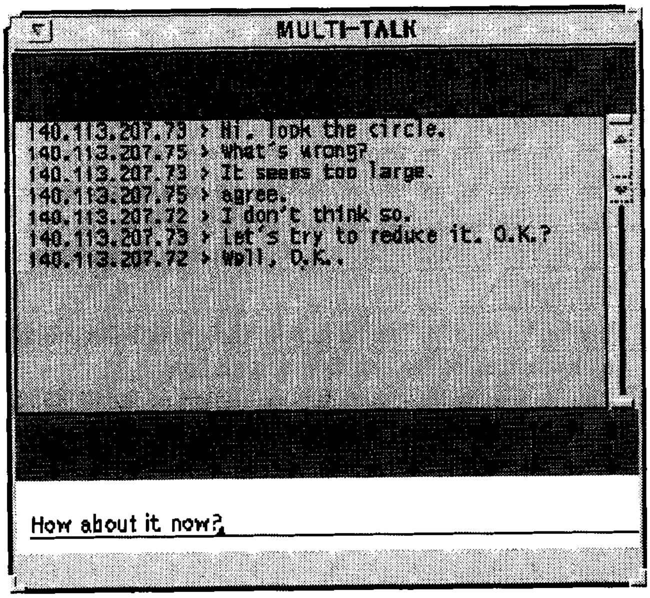


Fig. 6. Central controlled coordination structure. . . . . . Detecting survival; ——Request/response.



```
140.113.207.73 > Hi, look the circle.
140.113.207.75 > what's wrong?
140.113.207.73 > It seems too large.
140.113.207.75 > agree.
140.113.207.72 > I don't think so.
140.113.207.73 > Let's try to reduce it. O.K.?
140.113.207.72 > Well, O.K..
```

How about it now?

Fig. 7. Multiway text-based talk of CCDES.

microphone. The users' voices are recorded and broadcast each second. This facility increases realistic, convenient and efficient group work.

4. Related work

Different approaches to the problems of collaborative multimedia editing have been taken, and several prototypes have been built. A short description of some representative examples are given in Table 1.

5. Implementation

We have implemented a prototype of CCDES which is a central controlled and data replicated collaborative editing model. CCDES supports compound documents consisting of text, static graphics and hand-written images. CCDES also provides fundamental real-time

Table 1
Approaches to collaborative multimedia editing

	User interface	Data access method	
CCDES	multi-user editor text, graphical handwriting interface real-time interactive conversation facilities social role types (chair, author)	central coordinator replicated files	WYSIWIS and private work space
SgrEdit [2]	multi-user text editor	insert position lock mechanisms	
Quilt [3,4]	revise suggestion message facility, notification facility social roles type (co-author commenter & editor)	cooperation types (shared, executive & editor)	
PREP [5]	social role types	loosely-coupled collaboration related columns of chunks	
DistEdit [6]	social role types (master, observer)	tool kit for single user editor	
Aspects [7]	text, graphics and pixel images WYSIWIS social role types (master participant)	cooperation type (free for all, medium, full mediation)	
MMM [8]	text-string, graphical object	fine grained sharing	
MACE [9]		granularity	
CoMEdiA [10]	multi-user interface multimedia editing social role types (chair, author, secretary, commenter, reader) WYSIWIS	non-sequential organization of files communication styles (synchronous, asynchronous, off-document) access techniques (chunk, position)	
IRIS [11]	collaborative editor	replicated data dynamic voting for consistency	
MGEAT [4]	real-time interactive conversation facilities text, graphical, handwriting interface		

communication in text or audio among group members. Users of the collaborative group can transfer several kinds of media files or exchange messages through CCDES.

5.1. System development environment

CCDES is developed in the X-Window system and is run on Sun SPARC workstations. To build the user interface, XView and Open Windows Developer's are used as two development tools.

5.2. Data structure for compound documents

To handle the different data types of compound documents, there must be a complicated data structure. To have an available prototype as soon as possible, we choose text, simple graphics and hand-written images as elements of our compound document. In addition, the organization type of our system is Flat according to the CHM model [12], that is, no hyper-organization is possible (as in traditional documents). Each component on our compound document is regarded as an object. Each object has its ObjectType. According to different ObjectTypes, there are different data structures attached to the Object field, and different procedures and functions for manipulating or rendering this data. All the objects on the document are linked up with a double link list. Every object has a unique ObjectID according to the order of its creation. The co-author who is locking or editing the object is recorded to avoid conflicts of data access. Of course, there are more details about object attributes such as foreground/background colours, line attributes, text fonts, etc.

5.3. Mechanism for data transfer

CCDES is implemented in a replicated data model. All information about the new object and every authorized action on the object must be transferred to all members in the session. We apply the interclient communication delivery mechanism in X to accomplish data transfer among all members. To adapt both the data structure in our document and the format of the delivery mechanism of X to each other, we implements the translating functions. The ObjToStr function translates the data structure of an object to the specific form of a string. On the remote side, the StrToObj function operates the opposite action.

5.4. Management for coordination

Each action on the object must be authorized by the coordinator, usually the chairperson in the session, so then it can be broadcast to other members and be performed on all sites. The coordinator always keep information about all members and all objects on the document, such as the presence of members, the current locker of the specified object, etc. According to this information, the coordinator can make a decision on whether to admit the requests of members or not. For example, member M_i wishes to lock the object O_j. When the request for locking object O_j is received, coordinator C_k should check two points:

1. Is member M_i locking or editing another object?
2. Is object O_j being locked or edited by another member?

If both answers are 'No', coordinator C_k allows M_i to lock O_j. Otherwise, C_k notifies M_i that the request is not acceptable.

5.5. Fault tolerance

To avoid a breakdown of one session as a result of the crash of one site, there must be prevention and recovery mechanisms, or at least the session should be kept going. For simplicity, our approach to achieve fault tolerance is 'watch and notify'. In our system there are three roles in the session: chairperson, vice-chairperson and members. Usually, the chairperson takes responsibility for coordination and notification. Once a new member comes in, a 'detector' process is created to alternately examine the existence of the chairperson and the new member. If the new member fails, the 'detector' will notify the chairperson to unregister the member. If the chairperson is examined to be in failure, the vice-chairperson is notified to inherit the position of chairperson, and the new chairperson should choose a vice-chairperson immediately, and notify all members that the new chairperson and vice-chairperson have taken over.

6. Conclusion

In this paper, we have introduced the concepts of our collaborative editing system and the compound document. Furthermore, the three models to implement such a system are described. Next, we surveyed existing collaborative systems. The design issues such as the multi-user interface, concurrency control and the compound document were discussed, and our approaches were proposed. Moreover, we described the hardware and software architecture of our system. The implementations of compound document, data transfer, coordination and fault tolerance were outlined.

In CCDES, we try to combine collaborative editing and compound document processing. CCDES can also tolerate the failure of member processes. The fail-detect mechanism is constructed between the chairperson and members (including a vice-chairperson). It strictly monitors the existence of the chairperson. However, such a mechanism lacks the ability to determine whether the failure is caused by a site crash or a broken connection. Moreover, it should make a double-check when some failures are reported.

Acknowledgements

This work was partially supported by National Science Council of Republic of China under grant No. NSC83-0408-E009-032.

References

[1] M.E. Hodges and R.M. Sasnett, Multimedia Computing: Case studies from MIT Project Athena, Addison-Wesley, 1993.

[2] H.M. Vin et al., Multimedia conferencing in the Etherphone environment, IEEE Computer, 24 (10) (October 1991).

[3] I. Greif, R. Seliger and W. Weihl, A case study of CES: a distributed collaborative editing system implemented in Argus, IEEE Trans. Software Engineer., 18 (9) (September 1992) 827–839.

[4] W.Y. Leu and S.M. Yuan, X Window based group collaborative editing system, Information and Software Technology, 35 (11/12) (November/December 1993) 639–646.

[5] T. Ohmori et al., Distributed cooperative control for sharing applications based on multiparty and multimedia desktop conferencing system: MERMAID, IEEE, pp. 538–546, 1992.

[6] M.J. Knister and A. Prakash, DistEdit: a distributed toolkit for supporting multiple group editors, ACM CSCW 90 Proceedings, pp. 343–355, October 1990.

[7] V. Biel, Groupware groups up, MacUser (June 1991).

[8] E.A. Bier, S. Freeman and K. Pier, MMM: The multi-device multi-user, multi-editor: interface architecture for shared editors on a single screen, in Proc. ACM CHI'92 Conf. on Human Factors in Computing Systems, Monterey, CA, May 1991, SIGCHI ACM, New York, pp. 645–646.

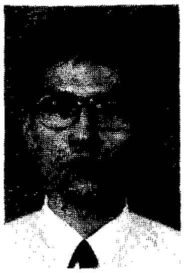
[9] C. Neuwirth et al., Issues in the design of computer support for coauthoring and commenting, in Proc. CSCW '90, 1990.

[10] ShrEdit, ShrEdit 1.1: A Shared Editor for the Apple Macintosh. User's Guide and Technical Description, Cognitive Science and Machine Intelligence Laboratory, University of Michigan, November 1990.

[11] U.M. Borghoff and G. Teege, Application of collaborative editing to software-engineering projects, Software Engineering Notes, 18 (3) (July 1993) A56–A64.

[12] C. Hornung and A. Santos, Proposal for a reference model for cooperative hypermedia systems, in 1st Eurographics Multimedia Workshop, Stockholm, Sweden, 1991.

Chien-Hung Lin received a BE degree in computer science from Tamkang University, Taiwan in 1987, and an MS degree in computer science from the National Chiao Tung University, Hsinchu, Taiwan in 1994. Since 1994 he has worked as an R&D engineer for NEC Taiwan. His research interests include computer supported cooperative work and group communication.



Shih-Chan Huang received a BS degree in computer science from the National Chiao Tung University, Hsinchu, Taiwan in 1993. He is currently working toward the MS degree in computer science at the National Chiao Tung University, Hsinchu, Taiwan. He also works as a research associate at the Advance Technical Center of the Industrial Technical Research Institute, Taiwan. His research interests include network management and computer supported cooperative work.

Shyan-Ming Yuan received a BSEE degree from the National Taiwan University in 1981, an MS degree in computer science from the University of Maryland, Baltimore County, in 1985, and a PhD degree in computer science from the University of Maryland, College Park, in 1989. Since September 1990 he has been an Associate Professor at the Department of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan. His current research interests include distributed system design, fault-tolerant computing and computer supported cooperative work.