

Semantic Preservation in Real-time Collaborative Graphics Designing Systems

Xueyi Wang, Jiajun Bu, and Chun Chen

College of Computer Science, Zhejiang University

Hangzhou, Zhejiang, 310013 China

xueyiwang@yahoo.com, bjj@cs.zju.edu.cn, and cchen@cs.zju.edu.cn

ABSTRACT

While most of researches focus on resolving the syntactic consistency problem in real-time collaborative graphics designing system, the semantic preservation problem should not be neglected. Graphics are created with user's meanings. These meanings should be notified to other users and should not be arbitrarily modified by others. In this paper, we resolve the semantic preservation problem by first classifying it into two categories: static semantic preservation and dynamic semantic preservation. Then a semantic preservation model is proposed. In this model, first we design semantic expressions to express user's meanings, and then we give the semantic preservation and conflict resolution approaches based on semantic expressions in detail. This model has been tested in the CoDesign system.

Keywords

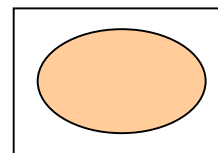
CSCW, real-time collaborative graphics design, semantic preservation, semantic expression

INTRODUCTION

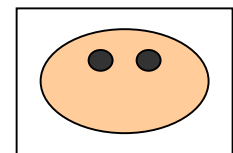
Real-time collaborative graphics designing system is a subclass of Computer-Supported Collaborative Work (CSCW) system, which allows users to view and design same graphic document simultaneously from geographically dispersed sites connected by networks. It requires that local site's operations should be executed immediately and the latency for reflecting other sites' operations should be low. To achieve high responsiveness in these systems, a replicated architecture is often adopted, and operations are replicated to all sites.

While the syntactic preservation problem caused by the replicated architecture has been well resolved by many works [3, 4, 6, 9], the semantic preservation problem also should be considered. The collaborative work may be less efficient if user's meanings cannot be clearly understood by other users. Considered the following situation, as shown in Figure 1. Initially, there is a yellow circle on the drawing area (Figure 1a). User A regards the circle as a blank face,

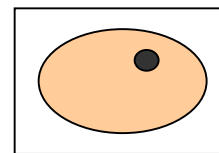
and draws two small black circles on it, which are regarded as eyes (Figure 1b). However, if user B does not catch on this meaning and erases one black circle, then user A's meaning is violated (Figure 1c). Of course, this situation can be resolved through the following steps: (1) user A undoes user B's operation, (2) user A tells his meaning to user B through other channels such as chat. But what if a latecomer C joins the collaborative group, or user B just ignores user A's chat content.



(a) Initial Drawing



(b) User A adds two black circles as eyes.



(c) User B erases a circle, and violates user A's semantic intention.

Figure 1. Example of semantic violation

This paper analyzes semantic preservation problem and proposes a semantic model to resolve this problem. The semantic model includes definition of semantic expressions, usage of semantic expressions, and semantic conflict resolution approach. Our goal is to integrate this model into the collaborative graphics designing system and enhance the efficiency of collaborative work. We have tested the model in the CoDesign prototype system. The CoDesign prototype system is a multi-level collaborative graphics designing system [1, 2], which supports both object-based and bitmap-based collaborative design. This paper discusses the object-based collaborative design only. Graphics are composed by objects, such as rectangle, polyline, etc. Objects are created with attributes, such as position, color, angle, etc., and are modified by changing one or more attributes.

The rest of this paper is organized as follows. First, the semantic preservation problem is discussed and a

LEAVE BLANK THE LAST 2.5 cm (1") OF THE LEFT COLUMN ON THE FIRST PAGE FOR THE COPYRIGHT NOTICE.

classification of the problem is represented. Then, a semantic preservation model are given, includes semantic expression definition, semantic preservation, and semantic conflict resolution. Our work is compared to related works in Comparison Section. Finally, major achievements of our work and remaining challenges are discussed.

SEMANTIC PRESERVATION PROBLEM

Syntactic preservation is wildly used in real-time collaborative graphics designing system. It aims at promising the same operation execution order and the same result of all users' operations at all sites. But user's meanings may not be clearly understood by other users only through viewing the execution of operations. Things may even be worse when users' operations are interleaved with each other and difficult to distinguish. We call this problem as **semantic preservation problem**. Because misunderstanding of user's meanings may cause confusion among users and low efficiency in collaborative work, additional measures should be adopted to share these meanings in the group.

In collaborative text editing systems, the problem is not serious. User's meanings can easily be understood with each other if all users comply with certain language's syntax. That is to say, semantic preservation can be achieved through complying language syntax in collaborative text editing systems. But things are not the same in collaborative graphics designing systems. It is almost impossible to devise completed designing rules and ask users to comply them.

Now we analyze the semantic preservation problem in detail. We classify this problem into two categories: static semantic preservation and dynamic semantic preservation, and discuss them as follows:

Static semantic preservation

Objects are created and modified with user's meanings. For example, if a user wants to draw a face, he may draw a big circle as face, two small circles as eyes, a curve as mouth and etc., thus all these objects are created with user's meanings (i.e. a facial feature). When these meanings are preserved, we call it static semantic preservation. Since these meanings can be related with one or more objects, we further classify the static semantic preservation as single object semantic preservation and object-object semantic preservation:

Single object semantic preservation

Single object, such as line, circle, contains user's meanings. These meanings can be expressed as object attributes, such as color, size, etc. For example, as shown in Figure 1b, user A draws two black circles as eyes, and color attribute of the circles contains user A's meanings (black eyes). If user B modifies the color of that circle, user A's meaning is violated (Figure 1c). We call this type of semantic preservation situations as single object semantic preservation. If user does not want other users to modify

these attributes (i.e. meanings) arbitrarily, measure should be adopted to protect them, and other users can operate the object with certain considerations.

Unlike the lock of the object, when promising the single object semantic preservation, users need only restrict operating some attributes of the object but not prohibit operating the whole object. For example, if user A fixes the small circles' color (black color denotes eyes) in Figure 1b, other users can still change position attribute of circles without violating user A's meaning. Furthermore, the restrictions on object are not imperative, and other users can still modify these restrictions through certain mechanisms. We will discuss it in Section 3.

Object-object semantic preservation

There also contains user's meanings among objects. These meanings can be expressed as object relations, such as relative position, overlap order, etc. For example, also shown in Figure 1b, user A regards the big yellow circle as a blank face, and two small black circles as eyes. Then the relative position of the three circles contains user A's meaning (i.e. a face with only eyes on it). We call this type of semantic preservation situations as object-object semantic preservation. These meanings can also be preserved and notified to other users if user A wants to, so other users cannot modify the related objects arbitrarily.

Note that object-object semantic preservation is unlike group operation. Objects are grouped and changed the same attributes in the group operation. After the operation is finished, these objects are ungrouped. In object-object semantic preservation, the restrictions on the objects are always available, and the unrestricted attributes of these objects can be modified freely. For example, if user A fixes the position and layer attributes of the three circles in Fig. 1b, users can change the color of any circle without violating the meanings of user A.

Dynamic semantic preservation

Besides the static semantic preservation, there also exists dynamic semantic preservation problem. For example, as shown in Figure 1a, user A regards the big yellow circle as a blank face, and intends to draw facial features on it. If he does not want other users to break it, his meanings on the object should be preserved and notified to other users. Another example, if user A wants to draw a flower picture and does not want to use black or gray color, these meanings should also be preserved and notified to other users. We call this type of semantic preservation situations as dynamic semantic preservation. It means users plan to do something but not actually do. Measure should also be adopted to broadcast these meanings to other users.

In above semantic preservation problems, user's meanings are preserved only when he intends to preserve them. User can also choose not to preserve them. The static semantic preservation focuses on preserving the existed meanings contained in the objects, and the dynamic semantic

preservation focuses on preserving user’s purpose. Both of them may be useful when we want to achieve reciprocal understanding of all users, reduce misunderstanding, and enhance efficiency of collaborative work.

SEMANTIC PRESERVATION MODEL

We propose the semantic preservation model in this section. First we propose two semantic expressions to express the static and dynamic semantic preservations. Then how to use the semantic expressions to achieve semantic preservation is discussed. Finally the semantic conflict problem is discussed and resolved.

Semantic expression definition

As we have shown above, the static semantic preservations acts on the attributes of objects, so we can use the combination of object attributes to express them. Here is the Static Semantic Expression (SSE):

Definition 1: Static Semantic Expression

The Static Semantic Expression (SSE) is expressed as follows:

$$SSE = (SV, (ObjId1, Comment, (Attrib1, Value), [(Attrib2, Value)], \dots), [(ObjId2, Comment, (Attrib1, value), \dots)], \dots, SE1, SE2, \dots). \quad (1)$$

SV denotes the state vector [5], which is used to record the uniqueness of the SSE and the causal relations with other SSEs and operations. ObjId denotes the unique identification of a certain object in the document. Comment denotes the description user want to give for the SSE. Attrib denotes the attribute of the object, such as COLOR, POSITION, etc. Value denotes the certain attribute value. For example, if Attrib is COLOR, then Value can be RED, BLUE, etc. SE denotes the semantic expression defined previously, includes both Static Semantic Expression (SSE) and Dynamic Semantic Expression (DSE). It means new SE can define on other SEs. The square brackets mean the elements in it are optional.

When expressing user’s meanings of a single object, the expression is reduced as follows:

$$SSE = (SV, (ObjId1, Comment, (Attrib1, Value), [(Attrib2, Value)], \dots), \dots, SE1, SE2, \dots). \quad (2)$$

For example, user A’s meanings on the two small circles (eyes) in Figure 1b can be expressed as follows:

$$SSE1 = (SV, (SmallCircleId1, “left eye”, (COLOR, BLACK))).$$

$$SSE2 = (SV, (SmallCircleId2, “right eye”, (COLOR, BLACK))).$$

Now see an example of object-object semantic preservation, in Figure 1b, user A’s meaning on the three circles (face) can be expressed as follows (Note that all the three circles have the position and layer restriction):

$$SSE3 = (SV, (BigCircleId, “face”, (POSITION, Value), (LAYER, Value)), (SmallCircleId1, “left eye”, (POSITION, Value), (LAYER, Value)), (SmallCircleId2, “right eye”, (POSITION, Value), (LAYER, Value)))$$

The dynamic semantic preservation acts on the attributes of objects as well as graphic document itself and attributes are not fixed on certain values, so the expression is more complex than SSE. Here we give the Dynamic Semantic Expression (DSE).

Definition 2: Dynamic Semantic Expression

The Dynamic Semantic Expression is expressed as follows:

$$DSE = (SV, (ObjId1 | NULL, Comment, [(Attrib1, Value Range)], [(Attrib2, Value Range)], \dots), [(ObjId2, Comment, (Attrib1, Value Range)], \dots, SE1, SE2, \dots). \quad (3)$$

We use Value Range in DSE instead of using Value as in SSE, because these meanings cannot be expressed clearly by a certain attribute value. The Value Range contains many data formats, like Boolean expression, enumeration expression, etc. For example, if user A plans to draw a flower with other users and does not want to use black or gray color in the graphic document, the expression can be expressed as follows (Note that the semantic expression is only related with graphic document itself, so there is no ObjId in the expression):

$$DSE1 = (SV, (NULL, “draw a flower”, (COLOR, NOT BLACK AND NOT GRAY))).$$

Another example, if user A wants to draw facial features on the big yellow circle, as shown in Figure 1a. The DSE is as follows:

$$DSE2 = (SV, (BigCircleId, “I plan to draw a face on it”, (COLOR, YELLOW))).$$

It should be pointed out that the semantic expressions are progressive. For example, as shown in Figure 1, if user A want to define the face, first he can define three SSEs (single object) on each of the three circles: the big yellow circle as blank face, the two small black circle as eyes, then he can further define a SSE (object-object) on the three circles as a whole (i.e. a face). The four expressions are shown as follows:

$$SSE4 = (SV, (BigCircleId, “blank face”, (COLOR, YELLOW))).$$

$$SSE5 = (SV, (SmallCircleId1, “left eye”, (COLOR, BLACK))).$$

$$SSE6 = (SV, (SmallCircleId2, “right eye”, (COLOR, BLACK))).$$

$$SSE7 = (SV, (BigCircleId, “face”, (POSITION, Value), (LAYER, Value)), (SmallCircleId1, “left eye”, (POSITION, Value), (LAYER, Value)), (SmallCircleId2, “right eye”,$$

(POSITION, Value), (LAYER, Value)), SSE4, SSE5, SSE6).

After a semantic expression has been defined, it should be transferred to other sites as an operation (with state vector). We called this type of operations as **semantic expression operation** (SO), which is separated from those original operations, called **normal operation** (NO). The difference between SOs and NOs is: SOs do not change any attributes of any objects, it just defines some non-imperative restrictions on objects to express user's meanings.

Note that the semantic expressions discussed above are given for description purpose. Users need not write these expressions; instead they can just choose the semantic expression type, select the required objects, and set the required attributes, then the semantic expression is automatically generated by the system. For example, if user wants to preserve his meanings on the three circles (face) in Figure 1b, he just chooses semantic expression type (SSE), the three circles, and the POSITION and LAYER attributes of three circles, then the proper semantic expression is automatically generated.

Semantic preservation

The common goal of collaborative work may easily be departed during the collaborative process, so when the semantic expressions have been transmitted to other sites, they should be active during the following collaborative process to coordinate collaborative works. We use the following rules to determine when to use these expressions:

- **Rule 1:** System displays the semantic expressions in the window when receiving them from other sites.
- **Rule 2:** When user selects or checks the objects, the semantic expressions related with these objects are shown in the window.
- **Rule 3:** When user violates the semantic expressions of certain objects, warning messages will display, and then user should decide whether to violate these meanings.

The above three rules are independent. When generating semantic expressions, they are immediately transferred to other sites. In order to keep the collaborative process smoothly, the received semantic expressions should not interrupt user's normal designing works, so we use a semantic display window to show these received expressions (rule 1). Also we link all semantic expressions with objects. If user selects or checks objects, semantic expressions related with them will be displayed (rule 2). When user operates on the objects, the system checks whether there are related semantic expressions for these objects, if does, then the system checks whether these expressions are violated and gives the warning messages (rule 3).

It should be noted that the semantic expressions could be violated. If user intends to violate it, the violated semantic

expressions will be deleted, and a delete operation will be transferred to other users. We take the semantic expressions as a support mechanism to enhance reciprocal comprehensions in collaborative group, but not as a coercive mechanism.

Semantic conflict resolution

Similar to NOs, SOs can also cause conflicting problem. When two or more concurrent operations including SOs act on the same object, conflicting situation may occurs. For example, in Figure 2, user A defines a static semantic expression (SSE) to group the three circles as a face, while user B moves a black circle at the same time. If both sites execute the operation immediately after receiving it, the graphics is not consistent (Figure 2a).

To resolve the conflict situation, we first separate the relations of NOs and SOs as conflicting and compatible, defined as follows:

Definition 3: Conflicting relation

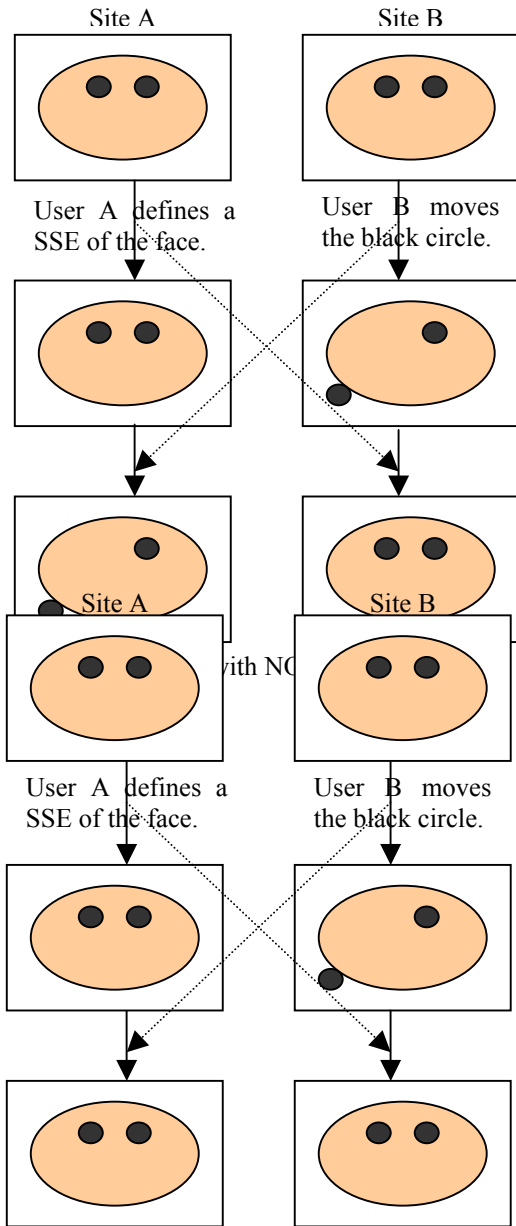
Given two operations O_1 and O_2 including at least one SO, O_1 and O_2 are called conflicting, denoted as $O_1 \otimes O_2$, if and only if: (1) O_1 and O_2 are concurrent, (2) O_1 and O_2 act on the same attribute of the same object with different values.

Definition 4: Compatible relation

Given two operations O_1 and O_2 including at least one SO, O_1 and O_2 are called compatible, denote as $O_1 \odot O_2$, if and only if O_1 and O_2 are not conflicting.

Note that semantic conflict problem is caused at attribute level, but not at object level. Users can modify other attributes of the object without causing conflict. For example, in Figure 2, if the expression defined by user A does not include color attributes, then the user B can change the color attributes of those circles without causing conflict.

We resolve this conflict problem by adopting semantic-precedence approach. When a SO conflicts with a NO, we simply preserve the SO and remove the NO. SO is always related with some NOs and SOs executed before, and SO can be regarded as a confirmation for previous related operations, so it is better to keep the SO and remove the NO. The correct result of above example is shown in Figure 2b. Another case, when two or more SOs conflict, we preserve the first SO and remove other SOs according to the SV. There also exist more complicated conflict situations. For example, O_1 and O_2 , O_2 and O_3 are conflicting, but O_1 and O_3 are compatible. This situation can occur because SOs may be related with many objects and attributes. Here we give an algorithm to insert both SO and NO operations:



(b) Conflict resolved by semantic-precedence.

Figure 2. Example of semantic conflict

Algorithm 1: Assume O_i will be inserted and we maintain two operation sequences (operations in it are ordered according to the SV): Inserted Operation Sequence (IOS) and Discarded Operation Sequence (DOS).

(1) If $O_i \in \text{NO}$, and there exists $O_j \in \text{SO}$ and $O_j \in \text{IOS}$ that satisfies $O_i \otimes O_j$, we put the O_i into DOS.

(2) If $O_i \in \text{SO}$:

(a) If there exists $O_j \in \text{NO}$ and $O_j \in \text{IOS}$ that satisfies $O_i \otimes O_j$, we put those NOs into DOS;

(b) If there exists $O_j \in \text{SO}$ and $O_j \in \text{IOS}$ that satisfies $O_i \otimes O_j$ and $SV_i < SV_j$, we put O_j into DOS, then retrieve the

operations that exist in DOS, conflict with O_j and compatible with O_i . If $SV_i > SV_j$, we put O_i into DOS.

For example, assume O_1 and O_3 are NOs, O_2 and O_4 are SOs, and $SV_4 < SV_2$, the relations among these operations shown as Table. 1:

| Relation | O_1 | O_2 | O_3 | O_4 |
|----------|-----------|-----------|-----------|-----------|
| O_1 | | \otimes | \odot | \otimes |
| O_2 | \otimes | | \otimes | \otimes |
| O_3 | \odot | \otimes | | \odot |
| O_4 | \otimes | \otimes | \odot | |

If these operations are inserted as O_1, O_2, O_3, O_4 , then the IOS and DOS will be:

After inserting O_1 : IOS = $\{O_1\}$, DOS = NULL.

After inserting O_2 : IOS = $\{O_2\}$, DOS = $\{O_1\}$.

After inserting O_3 : IOS = $\{O_2\}$, DOS = $\{O_1, O_3\}$.

After inserting O_4 : IOS = $\{O_3, O_4\}$, DOS = $\{O_1, O_2\}$.

Operations in DOS can be removed if they no longer cause conflict situation according to the SV [11], so we only keep those operations that may still cause conflicting.

COMPARISON

In order to achieving mutual awareness in collaborative works, additional video channels are implemented in some systems, such as VideoDraw [7], TeamWorkStation [10], and VideoWhiteboard [12]. In these systems, a video recorder is installed on each user's place, and user's work is transmitted to others through video. Therefore, Other users' works can be viewed simultaneously. But this type of systems can only support 2 ~ 3 people working together, and user's meanings may not be clearly understood by others only through observing the video. The cost of hardware is also a problem for these systems.

Access rights have been used in Intermezzo [8], COCA [13] to prohibit the unwanted operations on the objects. Intermezzo uses policies combined by a set of access rights to describe the collaborative context. User roles are used to denote users' categories under certain policies, and user's role can be evaluated dynamically at runtime. COCA proposes integrity constraints on objects. Constraints are defined on a set of objects and transactions are used to prevent the interrupts from others. But the problem is: access rights restrict the number of qualified users to operate certain objects, and user's meanings cannot be clearly expressed only through access rights.

Active rules are also used in COCA. An operation may cause the execution of another operation through recursively triggering rules. Active rules have been wildly used in database systems [14], but there are defections when adopting them in collaborative graphics designing system. Defining rules is difficult because of the dynamic characteristic of the collaborative work, only a few

activities can be explicitly expressed by active rules, and users may not easily understand these rules (the rules may also contain user's meanings). Furthermore, the descriptions of user's meanings cannot be expressed by active rules.

Comparing to related works, the semantic preservation model proposed in this paper has the following characteristics:

- *User's meanings can be expressed fully.* User's meanings are always related with attributes of objects and can be expressed by semantic expressions. We also provide the Description field to express those meanings unrelated with attributes.
- *User's meanings are expressed easily.* When user chooses the objects, semantic expression type, and attributes, and writes the description, the semantic expression is done.
- *User's meanings can be understood clearly.* User can select objects and check the semantic expressions related with them. Also user can be notified when semantic violation situation occurs.
- *Semantic expressions are used as a measure to express the user's meanings, but not compulsory measure.* Each user has his comprehension on the collaborative drawing, and collaborative process is also a communion process with each other. Constraint rules cannot obtain this goal.
- *Collaborative works can be more efficiency.* Obviously, user's meanings can be understood by others through semantic expressions, so the efficiency of collaborative work will be enhanced.

To our knowledge, the CoDesign system is the only one that achieves the semantic preservation, and the semantic preservation model proposed in this paper has never been addressed by other works.

CONCLUSION AND FUTURE WORK

In this paper, we have discussed the semantic preservation problem and proposed a semantic preservation model including semantic expression definition, semantic preservation and semantic conflict resolution to resolve this problem in real-time collaborative graphics designing system. Main contributions of this paper include the partition of the semantic preservation situations (static semantic preservation and dynamic semantic preservation), the semantic expressions (SSE and DSE), the approach of implementing semantic preservation, and conflict resolutions. It is the first time to propose the semantic preservation in real-time collaborative graphics designing system.

The semantic preservation model proposed in this paper has been implemented in the CoDesign prototype system. The CoDesign prototype system is used to test the

feasibility of our model and research for other issues associated with the collaborative graphics design.

Our goal is to make the collaborative graphics design more smoothly and more efficiently. Now we are looking for further steps beyond this model, such as understanding user's meanings automatically, forecasting user behavior and so on.

REFERENCES

1. Jiang Bo, Chen Chun, Bu Jiajun. CoDesign - A collaborative pattern design system based on agent. Proceedings of the 6th International Conference on CSCW in Design, Canada, 2001:319~323
2. Zeng Yi, Xu Duanqing, Chen Chun. Research on Internet based Collaborative Pattern Design System. Proceeding of 1999 National Workshop on Computer-Aided Industrial Design and Conceptual Design, China, 1999:213~318.
3. D. Chen and C. Sun. A distributed algorithm for graphic object replication in real-time group editors. Proceedings of ACM Conference on Supporting Group Work, Phoenix, Arizona, USA, Nov. 1999. pp. 121-130.
4. M. O. Pendergast. GroupGraphics: prototype to product. In S. Greenberg, S. Hayne, and R. Rada, editors, Groupware for Real-time Drawing: A Designer's guide, pages 209-227. McGraw-Hill, 1995.
5. Ellis, C. A. and Gibbs, S. J. Concurrency control in groupware systems. In proceedings of the ACM SIGMAD Conference on Management of Data, 1989:399~407.
6. R. Kanawati. LICRA: A replicated-data management algorithm for distributed synchronous groupware application. Parallel computing, 22:1733--1746, 1997.
7. Tang, J.C. & Minneman, S.L. VideoDraw: a video interface for collaborative drawing, in Proc. CHZ '90 Human Factors in Computing Systems (Seattle, WA., April 1-5). ACM, New York, 1990, pp. 313- 320.
8. W. Keith Edwards. Policies and Roles in Collaborative Applications. Proceedings of ACM CSCW'96.
9. T. P. Moran, et al. Some design principles for sharing in Tivoli, a whiteboard meeting-support tool. Groupware for Real-time Drawing: A Designer's guide, pages 24-36. McGraw-Hill, 1995.
10. Ishii, H. TeamWorkStation: towards a seamless shared workspace, in Proc. CSCW '90 Conference on Computer-Supported Cooperative Work (Los Angeles, CA., October 7-10). ACM, New York, 1990, pp. 13-26.
11. C. Sun, X. Jia, Y. Zhang, Y. Yang, D. Chen. Achieving convergence, causality-preservation, and intention-preservation in real-time cooperative editing systems. ACM Transactions on Computer-

Human Interaction, Vol.5, No.1, March, 1998, pp.63-108.

12. Tang, J.C. & Minneman, S.L. VideoWhiteboard video shadows to support remote collaboration, in Proc. CHI '91 Human Factors in Computing Systems (New Orleans, LA., April 27- May 2), ACM, New York, 1991, pp. 315-322.
13. D. Li, L. Zhou, and R. R. Muntz. A new paradigm of user intention preservation in realtime collaborative editing systems. In Proceedings of the Seventh International Conference on Parallel and Distributed Systems, Iwate, Japan, July, 2000.
14. J. Widom and S. Ceri, editors. Active Database Systems: Triggers and Rules for Advanced Database Processing. Morgan Kaufmann Publishers, Inc., 1996.