

# A New Consistency Model in Collaborative Editing Systems

Xueyi Wang, Jiajun Bu, and Chun Chen

College of Computer Science, Zhejiang University

Hangzhou, Zhejiang, 310013 P.R.China

[xueyiwang@yahoo.com](mailto:xueyiwang@yahoo.com), [bjj@cs.zju.edu.cn](mailto:bjj@cs.zju.edu.cn), and [cchun@cs.zju.edu.cn](mailto:cchun@cs.zju.edu.cn)

## ABSTRACT

Consistency maintenance is a basic issue in computer-supported cooperative editing systems. Aimed at providing a clear, appropriate and extended description on this issue, a new three-level consistency model is proposed: (1) Operation Consistency, which promises all operations are executed at the same order; (2) Content (Syntactic or Intention) Consistency, which promises the effects of all operations are the same at all sites; and (3) Semantic Consistency, which promises the user meanings are the same at all sites. Finally, evaluations on existing collaborative editing systems are discussed.

## Keywords

CSCW, collaborative editing, consistency maintenance, operation consistency, content consistency, semantic consistency.

## INTRODUCTION

Collaborative editing systems are a special class of CSCW (Computer-Supported Cooperative Work) systems. They have many applications such as textile pattern design, electronic conference and collaborative documentation edit in order to shortening the time, reducing the cost, and improving the quality. There are two basic characteristics in these systems [1,6,13,14,15]: (1) *real-time*: the response for local operations is quick and the latency for remote operations is low; (2) *distributed*: users can use any machine connected by networks.

To achieve good responsiveness, a replicated architecture should be adopted [15]: shared documents are replicated at the local storage of all collaborative sites. How to maintain the consistency of the shared documents is one of the most significant issues in these systems with a replicated architecture. In this architecture, operations generated at one site should be transferred to other sites to ensure that all sites have the same set of operations. There are two cases to cause inconsistency: (1) *unordered operations*: operations may be arrived at other sites unordered due to network's unreliability, and (2) *current operations*: multiple operations seem to be generated at the same time due to the network's latency. These cases should be well resolved, otherwise the shared documents will not be consistent, and the collaborative work cannot continue.

We propose a new consistency model in this paper. this model has three levels: (1) *Operation Consistency*, which

promises all operations are executed at the same order; (2) *Content (Syntactic or Intention) Consistency*, which promises the effects of all operations are the same at all sites; and (3) *Semantic Consistency*, which promises the user meanings are the same at all sites. Our goal is to make the consistency issue be represented more clear, suitable and complete.

The rest of this paper is organized as follows. In the following section, the previous consistency model is analyzed and its deficiencies are discussed. Then in the next section, our new consistency model is given in detail. Then comparison with previous consistency model is proposed. Finally, our model is summarized in last section

## MOTIVATION

Existing consistency model [14] separate the consistency issue into three parts: convergence, causality preservation and intention preservation, which is given as follows:

**A Consistency Model.** A collaborative editing system is said to be consistency if it always maintains the following properties:

- (1) *Convergence*. When the same set of operations has been executed at all sites, all copies of the shared document are identical.
- (2) *Causality preservation*. For any pair of operations  $O_1$  and  $O_2$ , if  $O_1 \rightarrow O_2$ , then  $O_1$  is executed before  $O_2$  at all sites.
- (3) *Intention preservation*. For any operation  $O$ , the effects of executing  $O$  at all sites are the same as the intention of  $O$ , and the effect of executing  $O$  does not change the effects of independent operations.

Here we first discuss the three parts of the consistency model:

**Convergence.** In collaborative editing systems, operations may arrive and be executed at different site in different order, resulting in different final result [15]. This problem is called divergence. To resolve the problem, serialization protocols are adopted to promise the same final result [1,2,10] by defining a total ordering relation of operations.

**Causality Preservation.** A new operation is always generated depending on the previous executed operations, so the new one should be executed after all the previous ones have been executed. The relation can be called causal

order. Due to unreliability of networks, operations may be received and executed out of causal order, so mechanism should be adopted to delay some operations to promise the right order [2,7,11,15].

**Intention Preservation.** Each operation has its own intention, which should not be violated. Since the same execution order of operations should be first promised, the concurrency operations may cause the intention of some operations to be violated [4,6,8,9,13,15].

From our point of view, this model is quite good to present the consistency issue in collaborative editing systems, but we also think it is not an ideal one and has the following deficiencies and ambiguities:

(1) *What does the convergence exactly mean?*

Can it be expressed as this: Whenever same set of operations has been executed at all sites, all the shared documents are identical? The consistency of final result cannot promise the consistency in the design procedure. We can easily design such case: the execution orders of a certain operation sequence are different at different sites, but the final result is the same.

(2) *Can convergence include causality preservation?*

According to real-time characteristic, operations are executed immediately at local site then transferred to other sites, so the local site must promise the causality preservation. In order to promising convergence, the execution order at other sites must be the same as at local site, and the causality preservation is promised naturally.

(3) *Is intention preservation independent of convergence and causality preservation?*

Convergence and causality preservation aim at the consistency of operations (providing the same execution order of operations), while intention preservation aims at the consistency of the content of operations (promising the user intention contained in the operations).

(4) *Does this model include all the ranges of consistency issue in collaborative editing systems?*

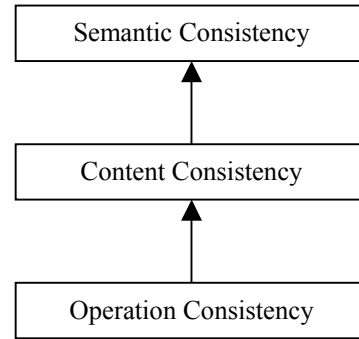
There is semantic consistency issue in collaborative editing systems, which may also be included in the consistency model. In text-editor, this issue is always related with syntax of certain language [3,15]. But it should be noted that: semantic consistency means some rules are defined on the content of collaborative work, and the languages' syntax naturally become the rules in the text-editor. We can design rules other than languages' syntax.

Upon the deficiencies and ambiguities proposed above, we give our new consistency model to resolve it in next section.

**THE CONSISTENCY MODEL**

As we have discussed above, it seems there are three consistency issues: operation consistency, content consistency, and semantic consistency. Operation

consistency is the basis of content consistency and content consistency is the basis of semantic consistency. In order to fitting the three issues, we propose a three-level consistency model, as shown in Fig. 1.



**Fig.1** Three-level consistency model

This model is proposed as follows:

**Two Basic Definitions**

To propose the consistency model, here we first give two basic definitions:

**Definition 1.** Causal relation

Given two operations  $O_1$  and  $O_2$  generated at site  $i$  and  $j$  respectively.  $O_1$  is said to causally precede  $O_2$ , denoted as  $O_1 \rightarrow O_2$ , if and only if (1)  $i = j$  and  $O_2$  is generated after  $O_1$ , (2)  $i \neq j$ , and  $O_2$  is generated after  $O_1$  is executed at site  $j$ .

**Definition 2.** Concurrent relation

Given two operations  $O_1$  and  $O_2$ , they are said to be concurrent or independent, denoted as  $O_1 \parallel O_2$ , if and only if neither  $O_1 \rightarrow O_2$ , nor  $O_2 \rightarrow O_1$ .

For any two operations  $O_1$  and  $O_2$ , if the execution order is  $O_1 \rightarrow O_2$ , we called it *causal order*.

For any two operation  $O_1$  and  $O_2$  that satisfy  $O_1 \parallel O_2$ , we can define an execution order on them. That is to say, we can define either  $O_1$  executes before  $O_2$  or  $O_2$  executes before  $O_1$  to keep consistency, as long as all sites promise the same execution order [11]. We call it *concurrent order*.

**Operation Consistency**

The first level of the consistency model is *Operation Consistency*, defined as follows:

**Definition 3.** Operation Consistency

When the same set of operations are executed according to causal order and concurrent order at all sites, all copies of the shared document are identical.

Note that Operation Consistency is the same as the Convergence in previous consistency model, where the convergence is achieved by total order relations. When all the operations are executed at causal order and concurrent order, a total order of operations can be achieved. Here we have a theorem.

**Theorem 1.** A total order can be achieved by achieving both causal order and concurrent order.

Prove:

1. When there are only two operations O1 and O2 in operation sequence, if O1 and O2 are causal relation, then the causal order is the total order; if O1 and O2 are concurrent relation, then the concurrent order is the total order.

2. Assume operation sequence O1, O2, ..., On are ordered by a total order relation. To a new operation Oi, we can find operations Oj1 that right before Oi and Ojm that right after Oi according to causal order (causal order is transitive, so there must be a first one and a last one in the causal order). So Oj2, ..., Ojm-1 must be concurrent with Oi. Since we have defined a concurrent order on these concurrent operations, so Oi must have a determined position in the operation sequence, and the new operation sequence also keeps a total order.

End prove.

### **Content Consistency**

**Definition 4.** Content Consistency

For any operation O, the effects of executing O at all sites are the same as the intention of O.

Note that Operation consistency is the basis of content consistency. Operation consistency promises the same execution order and result, and content consistency promises the result is what users expect.

Here we give an example: In a text editor system, the origin text is "ABCDE", operation O1 Ins["H",2] and operation O2 Ins["P",4] are concurrent operations generated at different sites. If the execution order is O1 precedes O2, then the final result is "ABHCPDE" under operation consistency, and "ABHCDPE" under content consistency.

### **Semantic Consistency**

**Definition 5.** Semantic Consistency

For any operation O, the effects of execution O at all sites abide by the rules defined before or during the collaborative work.

Based on the operation consistency and content consistency, there exists semantic consistency issue. For example, in collaborative text editing systems, although the final result can be syntactically correct, but may be semantically incorrect because of grammatical error [3,15].

From our point of view, the occurrence of semantic inconsistency is that the collaborative work contains some rules. These rules can be grammar in text editor, restriction on color in graphics editor, user's meanings contained in the operations [16], and etc. It should be noted that grammar is one kind of rules can be used on text editor; users can define new rules before or during the collaborative work.

### **COMPARISON**

Previous consistency model divides the consistency issue into three parts: Convergence, Causal Preservation, and Intention Preservation, while our consistency model divides this model into three-level: Operation Consistency, Content Consistency and Semantic Consistency.

As we have proved in Theorem 1, causal order and concurrent order can be combined to achieve a total order (Operation Consistency). So if the convergence means that: whenever the same set of operations has been executed at all sites, all copies of the shared document are identical, then the operation consistency equals to the convergence in previous model, and the causal preservation can be included in convergence.

The intention preservation in previous model means the user's intention should be kept, and equals to the content consistency in our model.

There exist rules in collaborative work, such as grammar in text editor. Violating these rules may not cause any operation or content consistency problem, but the collaborative work can be low efficiency. We conclude it as semantic consistency in our model.

### **CONCLUSION**

In this paper, we try to give a three-level consistency model to replace the previous consistency model proposed by [15]. As we have discussed above, the previous consistency model shows ambiguities and deficiencies, which may not be the best suitable one for describing the consistency issue in collaborative editing systems. Aimed at giving a more clear, suitable and complete consistency model, we divide the consistency issue into three parts: operation consistency, content consistency and semantic consistency. Operation consistency promises the same execution order and result at all sites, content consistency promises the intentions of operations are kept, and semantic consistency promises the rules defined before or during the collaborative work are abided. The former consistency issue is the basis of the later one. This consistency model is first to be proposed.

The consistency model given in this paper is far from maturity, we hope there can be further researches done on it to form a mature one.

### **REFERENCES**

1. Bernstein, P., Goodman, N., and Hanzilacos, V. *Concurrency Control and Recovery in Database Systems*. Addison-Wesley, Reading, Mass. 1987.
2. Birman, K., Schiper, A., and Stephenson, P. Lightweight causal and atomic group multicast. *ACM Transaction on Computing System*. 9, 3, Aug. 1991, 292-314.
3. Dourish, P. The parting of the ways: Divergence, data management and collaborative work. In *Proceedings of the 4th European Conference on Computer-Supported Cooperative Work*, September 1995, 215-230.

4. Dourish, P. Consistency guarantees: Exploiting application semantics for consistency management in a collaboration toolkit. In *Proceedings of the ACM conference on Computer-Supported Cooperative Work*, November 1996, 268-277.
5. Ellis, C. A., Gibbs, S. J., and Rein, G. L. Groupware: Some issues and experiences. *Communication of the ACM* 34, 1, Jan. 1991, 39-58.
6. Ellis, C. A., and Gibbs, S. J. Concurrency control in groupware systems. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, Seattle, WA, USA, May 1989. 399-407.
7. Fidge, C. Timestamps in message-passing systems that preserve the partial ordering. In *Proceedings of the 11th Australian Computer Science Conference*, 1988, 56-66.
8. Greenberg, S. and Marwood, D. Real time groupware as a distributed system: Concurrency control and its effect on the interface. In *Proceedings of the ACM conference on Computer-Supported Cooperative System*, November, 1994, 207-217.
9. Karsenty, A. and Beaudouin-Lafon, M. An algorithm for distributed groupware applications. In *Proceedings of the 13th International Conference on Distributed Computing Systems*, May 1993, 195-202
10. Lamport, L. Time, Clocks, and the Ordering of Events in a Distributed System. *Communication of the ACM*. 21, 7, July, 1978.
11. Li, D., Zhou, L., and Muntz, R. R. 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.
12. Raynal, M. and Singhal, M. Logical time: capturing causality in distributed systems. *IEEE Computer Magazine* 29, 2, February 1996, 49-56.
13. Ressel, M., Nitsche-ruhland, D., and Gunzenhauser, R. An integrating, transformation-directed approach to concurrency control and undo in group editors. In *Proceedings of the ACM Conference on Computer-Supported Cooperative Work*, New York, USA, November 1996, 288-297.
14. Sun, C., and Ellis, C. A. Operational Transformation in Real-Time Group Editors: Issues, Algorithms, and Achievements. In *Proceedings of the ACM conference on Computer-Supported Cooperative Work*, Seattle, Washington, USA, 1998, 59-68.
15. Sun, C., Jia, X., Zhang, Y., Yang, Y., and Chen, D. Achieving convergence, causality-preservation, and intention-preservation in real-time cooperative editing systems. *ACM Transaction on Computer-Human Interaction*. Vol.5, No.1, 1998. 63-108.
16. Wang, X., Bu, J., and Chen, C. Semantic preservation in Real-time Collaborative Graphics Designing Systems. Submitted to the 4th Workshop on Collaborative Editing.