A FRAMEWORK FOR FLEXIBLE DATA CONSISTENCY IN COLLABORATIVE EDITING

MOHD ADIB OMAR

Universiti Sains Malaysia

2009

A FRAMEWORK FOR FLEXIBLE DATA CONSISTENCY IN COLLABORATIVE EDITING

by

MOHD ADIB OMAR

Thesis submitted in fulfilment of the requirements
for the degree of
Doctor of Philosophy

June 2009

In The Name of Allah, The Most Beneficent, The Most Merciful

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude and appreciation to Professor Rosni Abdullah for her willingness to be my supervisor and Dr. Fazilah Haron for being my co-supervisor. I highly appreciate Professor Rosni for her guidance, patience, persistence and consistency for making this thesis possible. I would like to thank Dr. Fazilah for her visionary views, providing critical and resourceful advice. Also, my sincere thank to Associate Professor Abdullah Zawawi Haji Talib and Associate Professor Rahmat Budiarto for their sincere concern and support.

To my parents, Haji Omar Ismail and Hajjah Zakiah Haji Talib, I wish may Allah reward for your sacrifice and patience. As of for my in-laws, it is such a gratitude for having them for understanding my research undertakings and their patience waiting for my study completion. To my family, my wife, Mariana Saharin, and three children, Luqmanul Hakiem, Hanan Sakienah and Umar Muhsien, Allah has granted me patient family members. To my close relatives and friends, thanks for your concerns and prayers.

Special thanks go to my spiritual advisors, Ustaz Mohd Zamrus Mohd Ali, Dr. Ilias Said, Dr. Mohd Wira Mohd Shafiei and Dr Mohd Hasan, for their spiritual and intellectual advice. My thanks to all of my colleagues at the School of Computer Sciences particularly those who are at the Parallel and Distributed Computing Centre at level 4 and Postgraduate Laboratory at level 2.

Last and not least, my deep appreciation goes to Universiti Sains Malaysia for granting me a fellowship of the Academic Staff Training Scheme (ASTS). To Datin Fazia Ali and the team at Institut Pengajian Siswazah, USM, and Mr Ramli Osman and the team at the Personnel Division, Human Resource Department, USM, thanks.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CC	Collaborative Client
CS	Collaborative Server
CoFi	Consistency and Fidelity
CSCW	Computer Supported Cooperative Work
DAC	Data Awareness Controller
DAM	Data Awareness Module
DCC	Data Consistency Controller
DCM	Data Consistency Module
DDC	Data Distribution Controller
DDM	Data Distribution Module
DES	Data Encryption Standard
DISCIPLE	Distributed System for Collaboration Information Processing and Learning
dOPT	Distributed Operational Transformation
DSC	Data Security Controller
DSM	Data Security Module
ECC	Elliptic Curve Cryptography
FCE	Flexible Collaborative Editing
GOT	Generic Operational Transformation
GOTO	Generic Operational Transformation - Optimized
Grove	Group Outline and Viewing Editor
JAMM	Java Applet Made Multi-User
MCC	Computer Technology Corporation
OT	Operational Transformation
Reduce	Real-time Distributed Unconstrained Cooperative Editing
RSA	Rivest-Shamir-Adleman
PREP	Work in Preparation

KERANGKA KONSISTENSI DATA SECARA FLEKSIBEL DI DALAM PENYUNTINGAN KOLABORATIF

ABSTRAK

Persekitaran penulisan secara kolaboratif kebiasaannya dihadkan kepada aspek-aspek kolaborasi data yang tertentu. Untuk mengatasi masalah tersebut, kerangka fleksibel telah dicadangkan. Di dalam tesis ini, satu konsep kerangka fleksibel di dalam penyuntingan kolaboratif dibentangkan. Objektif pertama ialah untuk mengenal pasti kebolehlaksanaan penggunaan kaedah transformasi operasi di dalam penyuntingan kolaboratif segerak dan tidak segerak. Objektif kedua ialah untuk membuat explorasi tentang kaedah transformasi operasi di dalam kerangka untuk penyuntingan kolaboratif segerak dan tidak segerak. Objektif ketiga ialah mengintegrasikan elemen-elemen kolaborasi di dalam kerangka tersebut. Untuk mencapai objektif-objektif tersebut, langkah-langkah berikut telah diambil. Langkah pertama, satu konsep kerangka dibangunkan. Kerangka tersebut mengandungi empat aspek kolaborasi yang fleksibel. Aspek-aspek tersebut terdiri daripada pengagihan data, konsistensi data, kepedulian data dan keselamatan data. Langkah kedua, pendekatan konsistensi data secara flesibel dibentangkan dan ujikaji dijalankan untuk membuktikan konsep dijalankan. Ujikaji tersebut menunjukkan kebolehlaksanaan kaedah transformasi operasi di dalam penyuntingan kolaboratif segerak dan tidak segerak. Langkah ketiga, tesis ini membentangkan dua pendekatan pengurusan konsistensi data di dalam penyuntingan kolaboratif tidak segerak. Pendekatan pertama ialah untuk penyuntingan kolaboratif melalui pencegahan. Pendekatan kedua ialah untuk penyuntingan melalui pembaikkan. Kedua-dua pendekatan tersebut dinilai secara kualitatif melalui perbandingan dengan-dengan pendekatan-pendekatan yang sedia ada. Sumbangan pertama tesis ini ialah kebolehlaksanaan menggunakan pendekatan transformasi operasi di dalam penyuntingan kolaboratif segerak dan tidak segerak. Sumbangan kedua dan ketiga ialah teknik konsistensi data yang fleksibel

melalui penyuntingan kolaboratif melalui pencegahan dan penyuntingan melalui pembaikkan yang berdasarkan kaedah transformasi operasi di dalam penyuntingan kolaboratif tidak segerak. Sumbangan keempat ialah satu konsep kerangka fleksibel di dalam penyuntingan kolaboratif dibentangkan yang menumpukan kepada empat aspek kolaborasi.

A FRAMEWORK FOR FLEXIBLE DATA CONSISTENCY IN COLLABORATIVE EDITING

ABSTRACT

Collaborative editing environments are usually restrictive to specific data collaboration aspects. To overcome restrictiveness in data collaboration aspects, a flexible framework is proposed. In this thesis a conceptual collaborative editing framework is presented. The first objective is to investigate the feasibility of using operational transformation in both synchronous and asynchronous mode of collaboration in collaborative editing. The second objective is to explore a framework that uses operational transformation in both synchronous and asynchronous mode of collaboration. The third objective is to integrate elements of collaboration into the framework. In order to accomplish the objectives, the following steps are taken. First, the overall conceptual collaborative editing framework is presented. The framework consists of four flexible aspects of collaboration. The aspects are data distribution, data consistency, data awareness and data security. Second, flexible data consistency approach and its prove-of-concept experiment are presented. The experiments demonstrate that it is feasible to apply operational transformation technique in synchronous and asynchronous mode in collaborative editing. Third, preventive and corrective approach data consistency in asynchronous collaboration are presented. Both approaches are qualitatively evaluated with the existing respective preventive and corrective approaches. The first contribution of this thesis is to show that it is feasible to use operational transformation based technique in synchronous and asynchronous collaborative editing. The second and the third contributions are flexible data consistency technique in preventive and corrective asynchronous collaborative editing based on operational transformation. The fourth contribution is a conceptual framework consists of flexible collaborative editing framework that focuses on four aspects of collaboration

CHAPTER 1
INTRODUCTION

1.0 Introduction

Collaborative computing, also known as Groupware or Computer Supported Cooperative Work (CSCW) application is "*a computer-based system that supports groups of people engaged in a common task (or goal) and that provide an interface to a shared environment*" (Ellis, et al., 1991; "Collaborative," 2002). Collaborative computing enables computer-based collaboration in wide-area network, static and dynamic resource sharing and instant or delayed feedback among collaborators. Figure 1.1 shows different categories of systems within collaborative computing (Ter Hofte, 1998). Collaborative editing is a part of co-authoring systems in collaborative computing.

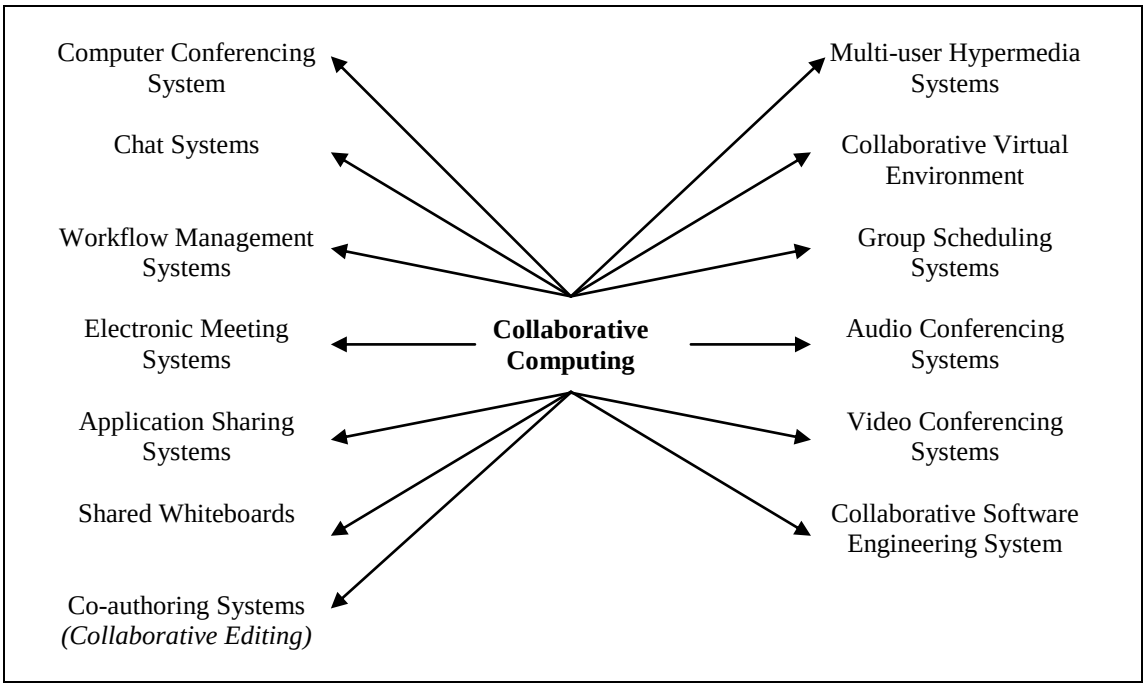

Figure 1.1: Categories of Collaborative Computing

Research issues in collaborative editing are almost identical to research issues in collaborative computing. Data consistency, concurrent management, user interface, access control, program integration and communication protocols are some related research issues in collaborative editing as well as in collaborative computing (Ellis et

al., 1991; Sun and Ellis, 1998). A common objective of collaborative editing is to allow coherent and consistent object sharing and manipulation by distributed users (Prakash and Shim, 1994; Strom et al., 1997).

Collaborative editing systems usually support synchronous mode of collaboration (Ellis and Gibbs, 1989; Ressel et al., 1996; Nichols et al., 1995; Sun et al., 1998) using variations of operational transformation technique (optimistic approach) to maintain data consistency among collaborators. Some of the collaborative editing systems employ asynchronous mode of collaboration (Fish et al., 1988; Decouchant et al., 1996; Neuwirth et al., 1994) using variations of locking-based mechanism (pessimistic approach) to maintain data consistency among collaborators.

Synchronous collaboration is analogous to a telephone conversation where the collaborating parties must be connected to each other at all time. Asynchronous collaboration does not require collaborators to be connected to each other at all time. It is analogous to electronic mail systems where the sender and receiver do not necessarily go online at the same time. Although there is no known technical reasons why these two modes of collaboration cannot co-exist in one collaborative editing system (Shen and Sun, 2002), only few collaborative editing systems provide these two collaboration modes in their systems (Begole, 1998; Molli et al., 2002; Pacull et al., 1994).

1.1 Motivation

Bentley and Dourish (1995) advocated flexible collaborative systems that support "*users with different working practices, level of expertise and personal preferences*." The outcome of supporting both optimistic and pessimistic data consistency management approach results flexible data consistency management. Since the choice of data consistency management approach (Greenberg and Marwood, 1994)

influences data notification (awareness), flexible data consistency approach leads to flexible data notification (awareness).

Synchronous (Ellis and Gibbs, 1989; Ressel et al., 1996; Sun et al., 1997b) and asynchronous (Fish et al., 1988; Neuwirth et al., 1990; Decouchant et al., 1996) collaborative editing systems are multi-user systems that allow users to view and edit shared documents at the same time (synchronous) and at different times (asynchronous) from geographically dispersed sites connected by communication networks. Data consistency management is one of the most significant challenges in the design and implementation of these systems particularly when the shared data are replicated among the participating collaborators (Sun and Ellis, 1998; Ionescu and Marsic, 2000).

Notification is an essential feature in collaborative systems, which determines when, what, and how updates made by one user are propagated, applied, and reflected on other users' interfaces. Notification plays an important role in determining a system's capability and flexibility in supporting different kinds of collaborative work. If a system has adopted a notification strategy that *frequently* propagates one user's actions to others, then this system is capable of supporting *real-time* (or *synchronous*) collaborative work, where multiple users can collaborate at *the same time* (Shen and Sun, 2002).

In contrast, if a system has adopted a notification strategy that *infrequently* propagates one user's actions to others, then this system is more suitable for supporting *non-real-time* (or *asynchronous*) collaborative work, where multiple users can collaborate at *different times*. Usually, one collaborative system uses only one notification strategy, and existing collaborative systems have been classified to be either real-time or non-real-time. However, there is no technical reason that a system cannot use multiple

notification strategies to support both real-time and non-real-time collaborative work (Shen and Sun, 2002).

Data consistency and data notification are inseparable with data distribution (Ellis et al., 1991). Data distribution sets the data communication topology of the collaborative editing. Data distribution can be centrally managed, fully replicated and partly replicated (Phillips, 1999). Since the data distribution aspect may fall from the range of centrally managed to fully replicated, flexible data distribution is a compromising idea to benefit both extremes of the data distribution design choice.

1.2 Problem Statement

Little research is done in notification in collaborative editing although it is identified as one of the important area to be explored (Sun and Ellis, 1998; Li et al., 2000). With few exceptions (Begole, 1998; Sun and Sosic, 1999), most of the collaborative editing systems tend to opt for either using locking-based (Knister and Prakash, 1990; Decouchant et al., 1996; Wang et al., 1999) or non-locking based (Ellis and Gibbs, 1989; Nichols et al., 1995; Ressel et al., 1996) in their consistency management approach. This scenario implies most of the collaborative editing systems opt for delayed or immediate notification.

Immediate notification is desirable only if the network latency is acceptable and continuous network connectivity. Delayed notification, however, is preferred if the network connectivity is unreliable. With the proliferation of collaborative work particularly in collaborative editing, having not only a flexible collaborative editing environment that can support both delayed and immediate notification but also adaptive to the user needs and system constraints will give better choice for the users to decide what kind of the notification is best for their working conditions, is a necessity (Bentley and Dourish, 1995; Litiu, 2001).

To address the above issues, we propose a collaborative editing system that is flexible to support both synchronous (real-time) and asynchronous collaborative editing (Omar et al., 2001). In term of consistency management approach, we propose to support both pessimistic and optimistic mechanisms, to be embedded in our work (Omar et al., 2004).

One feedback from our work in (Omar et al., 2004) is collaborative editing lacks of security measures in securing the shared data. As we check with the literature, most of the collaborative editing systems focus on several dimensions such as data consistency algorithms (Sun and Ellis, 1998), concurrent management (Ellis and Gibbs, 1989; Ionescu and Marsic, 2000), access control (Neuwirth et al., 1994), communications (Fish et al., 1988; Wang et al., 1999; de Lara et al., 2001), integration (Li and Li, 2002) and awareness related issues (Gutwin and Greenberg, 2002). Also, fault tolerance and error recovery are issues that need more attention in this area (Knister and Prakash, 1990; Pacull et al., 1994; Ionescu and Marsic, 2001; Qin and Sun, 2001; Li and Li, 2006).

In addition, based on our observation these dimensions will be better understood if they are unified in a single framework. A framework that supports these dimensions will provide a comprehensive view of collaborative editing because each dimension has its own important aspect. The understanding of these aspects cohesively will help future collaborative editing designers when they are designing collaborative editing systems.

As the research progresses, our work evolve into a framework that addresses flexible data consistency management, data distribution, data awareness (notification) and data security that attempt to include the dimensions mentioned above.

1.3 Research Objectives

The objectives of our research are:

- To investigate the feasibility of using operational transformation (data consistency management technique) in both synchronous and asynchronous mode of collaboration in collaborative editing environment, and to find a technique in achieving data consistency of shared data among collaborative users regardless of the collaboration mode selected by them. Usually, operational transformation is a technique of achieving data consistency without restricting collaborating users to make concurrent changes to the shared data in synchronous collaborative editing (Sun and Ellis, 1998; Ter Hofte, 1998).
- To explore a framework that uses operational transformation in both synchronous and asynchronous mode of collaboration. Recently, (Li and Li, 2006) reported that operational transformation technique can also be used in asynchronous collaboration but they did not elaborate further on how to use operational transformation in asynchronous collaborative editing. We would like to integrate the operational transformation technique into asynchronous collaborative editing.
- To integrate elements of collaboration into the framework. The first two objectives lead to a flexible data consistency management. In addition, since having flexible data consistency management involves other aspects of collaboration such as data distribution, data notification (awareness) and data security, we are also exploring the feasibility of making these aspects flexible.

1.4 Approach, Scope and Limitation

In order to accomplish the objectives, the following steps are taken. First, the overall conceptual collaborative editing framework is developed in Chapter 3. Second, flexible data consistency approach and its prove-of-concept experiment are presented in Chapter 4. Third, preventive and corrective approach data consistency in asynchronous collaboration are presented in Chapter 5.

A conceptual framework is developed to capture the flexible elements of data consistency, data distribution, data notification and data security. These elements are put together as cohesive unit in a flexible framework. The framework attempts to address what Bentley and Dourish (1995) have envisioned that a collaborative system should be a customizable medium.

Litiu and Prakash (2000) presents access control, concurrency control, coupling of views and extensible architecture as some of the many dimensions to enable flexibility and adaptability in the CSCW systems. Iqbal et al. (2002) proposes an integrated CSCW framework that supports the following dimensions: security model, transaction model, ontological model, coordination model and user interface model.

In our work, besides having flexible collaborative editing framework based on data consistency, data distribution, data awareness and data security, the work also include on applying operational transformation technique in asynchronous collaborative editing. The scope of this research is focused on the manipulation basic textual documents as an initial work that will eventually lead to formatted text and graphical based documents.

1.5 Contributions

The first contribution is the fulfillment of the first objective of this research. The first contribution of this thesis is to show that it is feasible to use operational transformation based technique in synchronous and asynchronous collaborative editing (Chapter 4).
The second and the third contribution are related to the exploration of operational transformation technique in asynchronous collaborative editing. The second contribution is a flexible data consistency technique in preventive asynchronous collaborative editing based on operational transformation (Chapter 5, Section 5.1). This technique is to verify that the operational transformation based technique is applicable in asynchronous collaborative editing as claimed by (Li and Li, 2006).

The third contribution is a flexible data consistency technique in corrective asynchronous collaborative editing based on operational transformation (Chapter 5, Section 5.2). This technique is to verify that the operational transformation based technique is applicable in asynchronous collaborative editing as claimed by (Li and Li, 2006).

The fourth contribution is a conceptual framework consists of flexible collaborative editing framework that focuses on four aspects of collaboration (Chapter 4). The aspects are:

- Data consistency (Ellis et al., 1991; ter Hofte, 1998; Sun and Ellis, 1998)
- Data distribution (Begole, 1998; Phillips, 1999; Bargh and Hofte, 2000)
- Data awareness (Data notification) (Greenberg and Marwood, 1994; Gutwin and Greenberg, 2002; Shen and Sun, 2002)
- Data security (Qin and Sun, 2001; Ionescu and Marsic 2001; Verissimo et al., 2003; Tolone et al., 2005)

1.6 Organizations

This thesis is organized as follows:

Chapter 2: The background and related work for data consistency management in collaborative editing are presented. The data inconsistency problems, data consistency model and data consistency mechanisms are elaborated. The overview of the data distribution mechanisms and data notification mechanisms are discussed. The related works of collaborative editing systems are evaluated based on the matrix of the collaboration aspects. The aspects are data distribution, data consistency and data notification.

Chapter 3: Collaborative editing framework is presented as a means of attaining flexible collaborative editing. The framework covers flexible data distribution, flexible data consistency, flexible data awareness and flexible data security.

Chapter 4: The approach of achieving flexible data consistency based on operational transformation is presented. The proof-of-concept designs on the feasibility of the operational transformation and experimental results on data consistency are presented

Chapter 5: The discussion of flexible preventive and corrective data consistency approach are presented. First, a flexible data consistency approach in preventive asynchronous collaborative editing based on operational transformation technique is presented. Second, a flexible data consistency approach in corrective asynchronous collaborative editing based on operational transformation technique is presented.

Chapter 6: The current work is concluded. Suggestions are proposed for future work.

CHAPTER 2
STATE OF THE ART

2.0 Introduction

Maintaining the consistency of shared data is one of the major issues in collaborative editing (Decouchant et al., 1996; Strom et al., 1997; Sun and Ellis, 1998; Sun, 2002; Gu et al, 2005; Orgun and Xue, 2006; Li and Li, 2006c; Imine, 2008). The background of some major concepts to be covered in this thesis and the related work are the main focus of this chapter. First, the data inconsistency problems in collaborative editing (Section 2.1) are presented. Second, the underlying properties ensuring data consistency in collaborative editing are elaborated (Section 2.2).

The next three sections focus on the aspects of consistency management in collaborative editing systems: data distribution (Section 2.3), data consistency mechanisms (Section 2.4) and data notification (Section 2.5). The related work is evaluated in Section 2.6. The chapter ends with a unified view, summary and conclusion in Section 2.7.

2.1 Data Inconsistency Problems in Collaborative Editing

Inconsistency problems have been identified as part of concurrency control measures in collaborative applications. Three inconsistency problems are divergence, causality violation and intention violation (Ellis and Gibbs, 1989; Sun et al., 1996b; Suleiman et al., 1997; Sun et al., 1998; Vidot et al., 2000; Sun, 2002). Each problem has its own unique properties. These problems are interrelated and yet are independent from each other.

Divergence - Due to concurrent operation generation and network latency, different sites might receive the remote operations in different sequence. Usually, the execution

of operations follows the reception order. Different sites might receive and execute incoming operations at different order. Therefore, each site may have inconsistent final results (Sun et al., 1998; Sun, 2002).

Causality Violation - Due to concurrent operation generation and network latency, the remote operations may be executed out of their natural cause-effect order because they may arrive out of their original sequence (Sun et al., 1998; Sun, 2002). As a result, some of the participating users in collaborative sessions will see the effect of certain action by other user and later see the cause of it. The correct way of observing is to see the cause of the action and later see the effect of it.

Intention Violation - Due to concurrent operation generation, the discrepancy of intended effect of the operation arises between the operation generation and the operation execution period (Sun et al., 1998; Sun, 2002). The intended effect of the operation generation and at the actual operation execution is different.

2.2 Data Consistency Model in Collaborative Editing

A consistency model provides a conceptual framework on how to overcome the inconsistency problems which arise due to concurrency of operations and non-deterministic network delays as mentioned in the previous section. A collaborative editing system should have the following properties in order to achieve data consistency: convergence, causality preservation and intention preservation (Sun et al., 1996b; Sun et al., 1998; Sun and Ellis, 1998; Sun, 2002).

Convergence - When all of the generated and executed operations are applied to all of the shared data on collaborating sites, all of the shared data are expected to be identical. The convergence property ensures that all of the collaborating sites reach to

the same editing state at the end of the collaborative editing process (Sun et al., 1996b; Sun et al., 1998; Sun and Ellis, 1998; Sun, 2002; Imine, 2008b).

Causality Preservation - Any pair of dependent operations must follow the same sequence of executions at the remote sites as well as at local site. The causality preservation property ensures that the orders of the dependent operations are adhered during the collaborative editing session (Sun et al., 1996b; Sun et al., 1998; Sun and Ellis, 1998; Sun, 2002).

Intention Preservation - The intention preservation property ensures two conditions. First, it ensures the effect of the executed operation at the remote site produces the identical effect when the same operation is executed at local site at the time of its generation. Second, it ensures no interference among effects of independent operations (Sun et al., 1996b; Sun et al., 1998; Sun and Ellis, 1998; Sun, 2002; Li and Li, 2007).

2.3 Data Distribution and Architectural View

A wide range of development approaches have been undertaken by the researchers in the area such as by using reference models, architectural styles and distribution architectures (Phillips, 1999; Litiu, 2001; Junuzovic and Dewan, 2006). Distribution architectural view describes the run time distribution of systems states and computation at different sites connected by a network (Begole, 1998; Phillips, 1999; Litiu, 2001; Junuzovic and Dewan, 2006). Centralized, replicated and sometimes hybrid architectures are common approaches chosen by the collaborative editing system developers since these systems are similar to distributed systems.

2.3.1 Centralized Architecture

In a centralized architecture, shared data are kept on a central location i.e. server (Greenberg and Marwood, 1994; Begole, 1998; Bargh and ter Hofte, 2000; Junuzovic et al., 2005). Any modification to the shared data must be performed at the central location. Appropriate consistency management mechanisms are applied to the centralized location so that the integrity of the shared data is preserved. Various consistency management mechanisms used in collaborative editing will be described in details in Section 2.4 of this chapter.

Consistency management in a centralized architecture is quite straight forward. The common approaches are through locking-based mechanisms like floor control and turn-taking protocol (Fish et al., 1988; Begole, 1998; Litiu and Prakash, 2000). Alternatively collaborative applications may use variations of locking-based approaches (Greenberg and Marwood, 1994; Sun, 2002). Managing consistency of objects using non-locking approaches in a centralized architecture is possible (Nichols et al., 1995). In fact, non-locking approach may allow better concurrency of operations (Bhola et al., 1998).

2.3.2 Replicated Architecture

On the other hand, on replicated architecture, each replica has shared artifacts (Greenberg and Marwood, 1994; Begole, 1998; Bargh and ter Hofte, 2000; Gu et al., 2005; Sun et al., 2006; Gu et al., 2007). The shared data are replicated on each collaborating replica and any modification to the shared data can occur at any replica. Appropriate distributed consistency mechanisms are needed to maintain consistency of shared artifacts on each replica. Various consistency management mechanisms used in collaborative editing will be described in detail in the next section of this chapter.

Since the shared data are replicated to all cooperating sites, consistency management in replicated architecture must ensure the integrity of the replicated shared data so that every time the data are retrieved for modification or execution, the changes must be propagated to all replicas ensuring identical data states at quiescence (Sun et al., 1996b; ter Hofte, 1998; Zafer, 2001; Gu et al., 2007). The locking and non-locking based approaches are still applicable in replicated architecture (Citro et al.,2007).

2.3.3 Hybrid Architecture

Besides centralized and replicated architectures, hybrid architecture is another way to distribute data. In hybrid architecture, some portions of the data are kept at a centralized server and some data are replicated to each of the participating site (Begole, 1998; Phillip, 1999; Junuzovic and Dewan, 2006). Appropriate distributed consistency mechanisms are needed to maintain consistency of shared artifacts on each replica and at the central server.

Various consistency management mechanisms used in collaborative editing will be described in detail in the next section of this chapter. An important issue in hybrid architecture is to decide which shared artifacts should be centralized and which are to be replicated (Bargh and ter Hofte, 2000; Mao et al., 2003; Bu et al., 2006).

2.3.4 Discussion

In centralized architecture (Fish et al., 1988; Junuzovic et al., 2005), data centralization is desirable if the shared data is too big and costly to replicate i.e. a national healthcare database. However, the parallelism of action is restricted as only one user can make changes on the server. Others have to wait until the user is done. The strength of the centralized architecture is simplicity because it avoids redundancy and replication. The

weak side of the centralized architecture is congestion. The congestion is expected if the updates are frequent and the updates are made by multiple users.

On the other hand, data replication has its own advantages (Begole, 1998; Junuzovic et al., 2005; Oster et al., 2006). High degree of parallelisms can be achieved as multiple users can manipulate the same replica. In addition, replication is desirable to increase concurrency of operations and optimize the computing resources by providing better interactive responsiveness and fault-tolerance (Strom et al., 1997). However, complex concurrency and consistency management are employed to prevent and manage any conflict that arises due to concurrent actions (Sun and Ellis, 1998).

Hybrid architecture designers try to bring both benefits from the centralized and replicated architecture but hybrid architecture poses even more challenging consistency management mechanisms because some of shared artifacts are located in a central location and some in each replica (Pacull et al., 1994; Nichols et al., 1995; Begole, 1998; Junuzovic and Dewan, 2006).

The approach taken in this thesis is to replicate the shared data in the central location as well as in each replica. Since the nature of collaborative editing involves frequent exchange of textual updates and the textual data is rather small in size, data replication is acceptable. To minimize the complexity of the replicated approach, a centralized server is introduced.

2.4 Data Consistency Mechanisms Overview

A part of data consistency management is concurrency control. Concurrency control mechanisms include the management of potentially interfering collaborative activities of human-computer interactions that operate in parallel (Greenberg and Marwood, 1994; ter Hofte, 1998; Dewan and Hedge, 2007; Imine, 2008).

The typical approaches of consistency management mechanism are classified as locking-based (pessimistic) and non-locking-based (optimistic) data management mechanisms (Ellis and Gibbs, 1989; Bhola et al., 1998; Phillips, 1999; Bargh and ter Hofte, 2000). Figure 2.1 shows data consistency management overview. A hybrid approach integrating the optimistic and pessimistic approach is also suggested by (Sun and Sosic, 1999b; Sun, 2002; Mao et al., 2003; Citro et al., 2007).

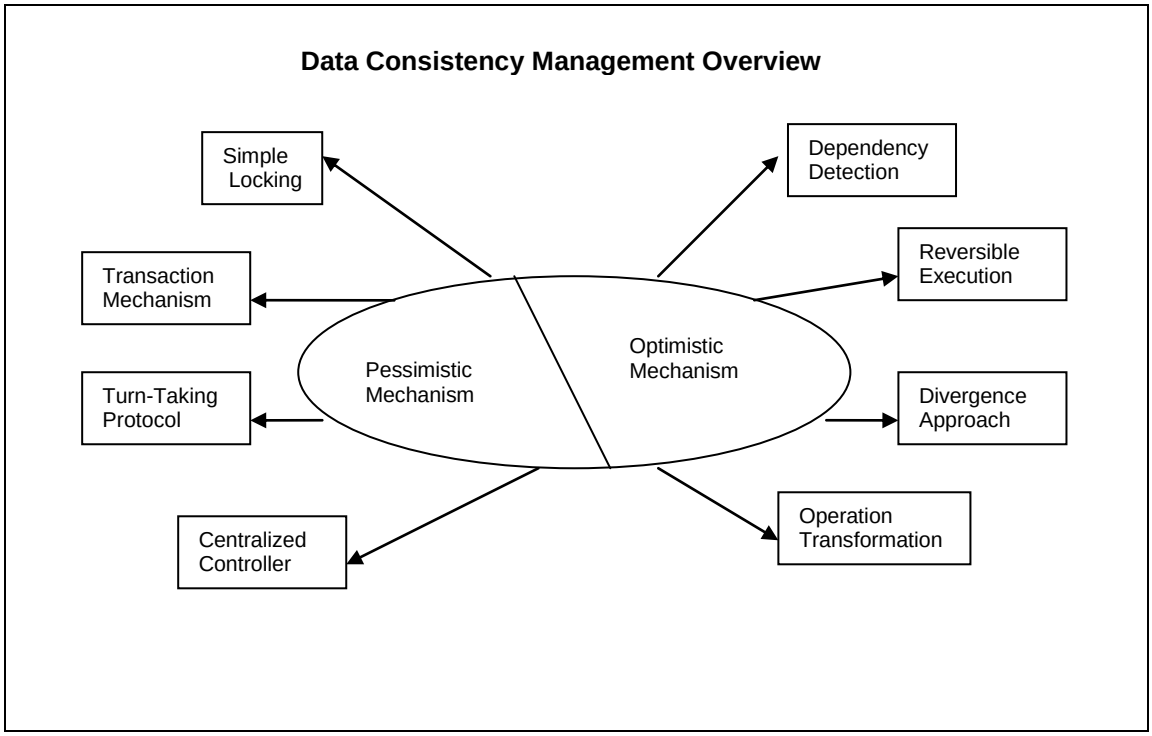

Figure 2.1: Data Consistency Management Overview

2.4.1 Locking-Based Mechanism

Locking-based mechanism ensures the consistency of shared data by preventing users from modifying the shared data concurrently. If one user is currently modifying the shared data, other users are prevented from making any modification until the current user finishes the editing. It is also known as pessimistic concurrency control. Pessimistic concurrency control can also be known as data inconsistency avoidance

(ter Hofte, 1998). Pessimistic concurrency control mechanisms include simple locking, transaction mechanism, turn-taking protocol and centralized controller.

In simple locking mechanism, before a shared artifact is modified, it has to be locked. The locking process begins with the request of the available lock from a locking controller. If the lock is available, the request is granted and given to the requested user. Otherwise, the requester has to wait for other user to release the key. Only when the key is granted, changes to shared data can be made. When the modifications are done, the lock is given back to the controller (Ellis and Gibbs, 1989; Ellis et al., 1991; Greenberg and Marwood, 1994; Preston and Prasad, 2006; Li et al., 2007).

Transaction mechanism is adopted from the database system. The transaction provides strict consistency control because a transaction can be either successful or failed. The transaction consists of a sequence of resource manipulation tasks coupled to one another, which either succeed or fail as a whole. If whole tasks or operations succeed, then they transform the shared resources from one consistent state to another. Otherwise, these operations will be aborted and all of the changes made on the shared resources will be undone (Ellis and Gibbs, 1989; Ellis et al., 1991; Bargh and ter Hofte, 2000).

Turn-taking protocol is similar to floor control mechanism; each user has to take turn to make desired modifications (Ellis and Gibbs, 1989; Ellis et al., 1991; Sun et al., 1998; Mao et al., 2003). The access to the right of the floor can be implemented using software-mediating turn taking internally or using external social protocol-mediating turn taking among collaborating users manually through user interface (Greenberg and Marwood, 1994). In centralized controller, each modification has to go through a centralized server. The centralized server will broadcast all of the changes to the other clients (Ellis and Gibbs, 1989; Ellis et al., 1991).

2.4.2 Non-Locking Based Mechanism

Non-locking based mechanism or optimistic concurrency control is data inconsistency detection and correction (Ellis et al., 1991; Prakash, A., 1999; Li and Li, 2006c; Imine, 2008). It allows inconsistencies to take place but later it detects the inconsistencies and corrects them. Optimistic concurrency controls include dependency-detection, reversible execution, operation transformation, divergence approach.

In dependency detection, timestamps are used as a method to identify conflicting actions in dependency-detection approach. The conflicting actions are resolved manually (Ellis and Gibbs, 1989; Ellis et al., 1991). Reversible execution is a mechanism in which some prior executions can be undone if there are other executions that should be performed first. Later, the undone operations will be re-executed in the correct order (Ellis and Gibbs, 1989; Ellis et al., 1991).

The operation transformation approach is similar to dependency detection but it supports automatic conflict resolutions (Ellis and Gibbs, 1989; Nichols et al., 1995; Ressel et al., 1996; Sun et al., 1997; Begole, 1998; Jung and Song, 2006; Imine, 2008). Operations can be generated and executed concurrently but they may be transformed before their execution so that the execution of the same set of properly transformed operations in different orders could produce identical document states (Ellis and Gibbs, 1989; Sun et al., 2004; Li and Li, 2006c).

Operation transformation involves adjusting the parameters of one operation according to the effects of other executed independent operations so that the execution of the transformed operation on the new document state can achieve the same effect as executing the original operation on the original document state (Sun et al., 1998; Sun and Sun, 2006; Imine, 2008).

Divergence approach allows data to be inconsistent and resolve the inconsistency through frequent synchronization. By applying continuous synchronization on a divergent stream of data will cause all of the data to converge eventually (Dourish, 1995).

2.4.3 Discussion

The strength of the pessimistic approaches is they prevent inconsistencies to occur. The order of execution is regulated so that each execution is sequentially ordered. The drawback of the pessimistic approaches is they sacrifice parallel executions of tasks. Consistency is maintained at the expense of lack of responsiveness due to the key management activities in simple locking and queuing delay in centralized controller (Greenberg and Marwood, 1994; Feng et al., 2008).

The strength of the optimistic approaches is they allow concurrent modifications and executions that utilize parallelism. However, some degree of complexity is introduced in dealing with consistency maintenance (Greenberg and Marwood, 1994; ter Hofte, 1998; Gu et al., 2005; Jung and Song, 2006; Ignat et al., 2007).

The approach taken in this thesis is to accommodate both pessimistic (simple locking) and optimistic (operation transformation) approaches into a framework. By having these approaches co-exist in a framework, it will make the framework more flexible in terms of the trade-offs of each approach offers. Sometimes pessimistic approach is more favourable than optimistic and vice versa depending on the needs of the users.

2.5 Collaboration Modes and Notification Mechanisms

Collaboration modes influence on how the shared data are updated. The update can be immediate in synchronous collaboration. The update is delayed until the request for

updating issued in asynchronous collaboration. Multi-synchronous collaboration allows the update to take place either immediately or at later time when there are changes in the shared data (Minor and Magnusson, 1993; Molli et al., 2002).

Notification becomes important when multiple users interact in collaborative editing since a user must know what changes that have been made by other users that may affect her current work (Ellis et al., 1991). In addition, the kind of notification supported will determine the capability and flexibility of the collaborative systems to support different kinds of collaborative work (Shen and Sun, 2002). Notification can be categorized into three: immediate, delayed or flexible.

2.5.1 Synchronous Collaboration and Immediate Notification

In a synchronous collaboration the shared data are updated immediately. Immediate operation propagation may be used to update the shared data. As for collaborative editing, multiple users should be able to perform collaborative editing activities at real time and the changes made should be immediately shown to each other (Ellis and Gibbs, 1989; Sun et al,. 1998; Gu et al., 2005; Li and Li, 2006c). Usually, if synchronous or real-time collaboration is employed, immediate notification is also implemented (Ellis et al., 1991). In real-time interaction, users expect to be notified immediately about changes made by others.

2.5.2 Asynchronous Collaboration and Delayed Notification

The shared data are updated when necessary. Delayed operation propagation is used to update the shared data. Asynchronous interaction employs delayed notification (Shen and Sun, 2002; Dewan and Hedge, 2007). The delayed notification can be implemented in two ways. First, the notification is done at periodic basis if all of the collaborators are on-line. Second, the users are only notified when users request for the recent updates.

2.5.3 Multi-synchronous Collaboration and Flexible Notification

The multi-synchronous systems support both immediate and delayed update of shared data. Flexible notification and operation propagation may be used to update the shared data (Li et al., 2000; Shen and Sun, 2002; Molli et al., 2002). Other form of multi-synchronous collaboration is to allow asynchronous collaboration in a synchronous system or vice versa (Ignat and Corrie, 2003; Dewan and Hedge, 2007). Additional feature of having alternative collaboration mode is added to the existing system to ease the collaboration needs.

2.5.4 Discussion

The advantage in synchronous interaction is the collaborating users will be immediately notified of any update made by any one of them (Shen and Sun, 2002). The disadvantage in synchronous interaction is each collaborating site must be constantly connected to each other so that updates can be propagated in real-time.

The advantage of asynchronous interaction is it does not require every collaborating member to be constantly connected to each other. Even if they are constantly connected but they may not update each other every time there are changes in their shared data. Therefore, the users will be notified upon their request or after certain definite predetermined time period (Shen and Sun, 2002). The disadvantage of asynchronous interaction is the delay of updates since the collaborating members are not constantly connected to each other.

Multi-synchronous interaction supports both synchronous and asynchronous interactions. The challenge of supporting both kinds of interactions is to ensure the consistency of shared data is maintained (Molli et al., 2002).

The approach taken in this thesis is to accommodate both synchronous and asynchronous interactions. These enable the framework to be more flexible in supporting both synchronous and asynchronous interactions.

2.6 Related Work

We have evaluated 13 existing systems that include synchronous (Grove, Joint Emacs, **Reduce**), asynchronous (Quilt, PREP, Alliance), multi-synchronous (DistEdit, Duplex) collaborative editing systems, collaborative infrastructures (Bayou, CoFi), framework (DISCIPLE) and toolkits (DistEdit, Jupiter, Flexible JAMM) that support collaborative editing. Figure 2.2 shows the matrix of collaboration aspects. The matrix shows how the reviewed systems are positioned based on their data distribution, data consistency management and their nature of interactions (synchronous or asynchronous collaboration). Based on the table and figure presented, several observations are made.

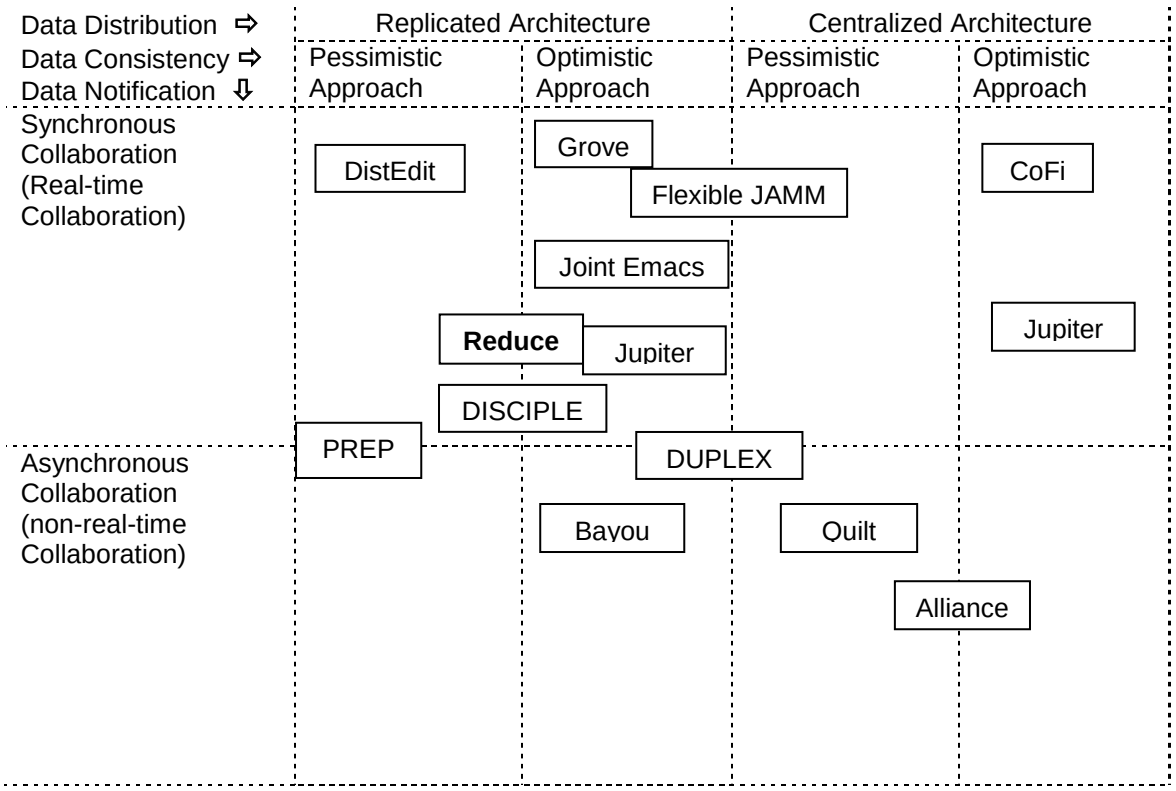

Figure 2.2: Related Work in the Matrix of Collaboration Aspects

2.6.1 Group Outline and Viewing Editor (Grove)

Group Outline and Viewing Editor (Grove) was developed at the Microelectronics and Computer Technology Corporation (MCC), around 1988 (Ellis and Gibbs, 1989; Ellis et al., 1991). The purpose of Grove prototype was both to explore implementation alternatives for a real-time multi-user tool and to collect informal observations on its use (ter Hofte, 1996). Collaborative editing issues addressed by Grove are concurrency control collaborative editing, consistency of shared data and performance-fast local response time.

Data Distribution. The shared data in Grove are fully replicated. Every collaborating site in Grove system keeps and maintains the exact copy of shared data.

Data Consistency. Grove uses optimistic data consistency management. Grove pioneered a non-locking consistency management technique called distributed operational transformation (dOPT) algorithm. It is an optimistic approach since it allows concurrent operation generations and executions. Concurrent operation generations and executions may result in different final document outcomes at the collaborating sites. Therefore, these operations are transformed before they are being executed on each site so that the execution of the same set of properly transformed operations in different orders would produce identical final document states.

Data Notification. Since Grove allows concurrent non-locking updates, the shared data are updated synchronously. It implements real-time notification or immediate notification. Changes made in one editor will be immediately propagated to all collaborating editors.

2.6.2 Joint Emacs

Joint Emacs was developed at the Institute for Computer Science, University of Stuggart around 1996 (Ressel et al., 1996). It is a prototypical group editor that follows Emacs-style text editor. Its main intention is to show the proposed concurrency control and group undo algorithm called aDOPTed. Its approach is a feasible alternative to dOPT algorithm, the pioneer of operational transformation algorithms (Ellis and Gibbs, 1989). The issues addressed by the Joint Emacs system include concurrency control collaborative editing, consistency of shared data and performance-fast local response time.

Data Distribution. The shared data in Joint Emacs are fully replicated. Every collaborating site in Joint Emacs system keeps and maintains the exact copy of shared data.

Data Consistency. Like Grove, Joint Emacs opts for optimistic consistency data management. (Ressel et al., 1996) has found that the distributed operational transformation (dOPT) algorithm implemented in Grove fails to produce identical final outcomes if the collaborative editing session has more than two participants.

(Ressel et al., 1996) proposed and implemented an improved version of dOPT algorithm called aDOPTed algorithm. It is still an optimistic approach since it allows concurrent operation generations and executions. Concurrent operation generations and executions may result different final document outcomes at the collaborating sites. Therefore, these operations are transformed before they are being executed on each site so that the execution of the same set of properly transformed operations in different orders would produce identical final document states.