Evaluating and Implementing a Collaborative Office Document System

Andy Adler[a], John C. Nash[b] and Sylvie Noël[c*]

[a] *School of Information Technology and Engineering, University of Ottawa, Ottawa, ON K1N 9B5, Canada*
[b] *School of Management, University of Ottawa, ON K1N 9B5, Canada*
[c] *Communications Research Centre, Industry Canada, PO Box 11490, Station H, Ottawa ON K2H 8S2, Canada*

Abstract

Collaborative work with office suite documents such as word-processing, spreadsheet and presentation files usually demands special tools and methods. For this application, we have developed TellTable, a relatively simple web-based framework built largely from available software and infrastructure. TellTable allows the use of existing office-suite software in a collaborative manner that is controlled but is familiar to users of common single user software. From the literature and our research, we identify twelve challenges to collaborative editing software that we use in an evaluation checklist: time and space, awareness, communication, private and shared work spaces, intellectual property, simultaneity and locking, protection, workflow, security, file format, platform independence, and user benefit. We then use this checklist to characterize TellTable in comparison to some other collaborative office tools.

[*] Corresponding author.
Email addresses: adler@site.uottawa.ca (A. Adler), nashjc@uottawa.ca (J. Nash) sylvie.noel@crc.ca (S. Noël)

(Space for publisher information)

1. Introduction

Many work environments require collaborative editing of documents, drawings, presentations and spreadsheets; a good example is a scientific paper, such as this one, in which researchers need to jointly develop and refine a document. Technology has simplified such collaboration, whether the participants are co-located or in different geographic locations. Today, documents are quickly and efficiently distributed through email, while the telephone, chat tools and Web-based conferencing tools let people "meet" without leaving their offices. In some organizations, specialized tools have been set up to allow for better control and management of this collaboration. Perhaps the best-known example of a system to assist the management of documents handled by many users is the Lotus Notes / Domino system (IBM Corporation, 2003; Kenneth Moore, 1995) which applies to different uses than the collaborative editing focus of the present article.

For most people, apart from voice telephone, the only reliable bi-directional electronic communications protocol is email, and it is thus used in a crude asynchronous manner for jointly editing office suite documents. Multiple copies of the same document can lead to confusion, as group members make conflicting modifications to their copy, which must then be integrated into a coherent document. If authors accidentally end up working on different versions of a document, e.g., following delivery delays, the problem of integration becomes even more complex.

To address the requirements of collaborative editing, many different software systems have been developed, some commercial and some academic. Since an exhaustive review of such systems is beyond the scope of this paper, we refer the reader to review articles (e.g., Newman and Newman, 1992; Noël and Robert, 2003). The requirements of collaborative teams can be much more difficult than those for standalone software. In this paper, we compile a checklist of challenges in meeting the needs of collaborating authors. Subsequently, we use this to review a sample selection of approaches to collaborative editing software to provide a comparison with TellTable, a tool we have been developing to meet some of our own needs (Adler and Nash, 2004; Adler, Nash and Noël, 2004; Nash, Adler and Smith, 2003; Nash, Adler and Smith, 2004). TellTable was developed initially to provide audit trails for spreadsheets, and its application to collaborative editing for word processing documents and presentations was a serendipitous side-benefit. The goal of this paper is twofold: to build a checklist based on the principal social and technical issues, and to use it to assess and possibly redirect our own development efforts.

We also note that the TellTable system is an open infrastructure. Our code, which can be downloaded from SourceForge.net, links together and builds on well-known open-source tools. It also runs on commonly available hardware. Indeed, our first server was a discarded Pentium II 233 PC. While TellTable is a lightweight, inexpensive infrastructure, in this paper we ignore cost-of-installation differences when comparing TellTable to other collaborative systems.

We have deliberately limited the scope of this paper to office suite applications: collaborative word processing, spreadsheets, and presentations. There are many other forms of collaborative editing, for instance of software, designs, artwork, animations, or of audio or video presentations. TellTable is, in fact, quite capable of running such applications, but we focus on the needs of users of traditional office productivity applications in this paper. To clarify our terminology, we refer to "documents" as the end product that collaborative editing groups are trying to create; documents can refer to a text, a spreadsheet file, or a slide presentation. We use "artifacts" to refer to the individual sections of a document. For example, a paragraph, a graph, or a single slide, constitute artifacts.

2. Software Challenges in Collaborative Authoring

We have developed the following list of software capabilities that are required by most collaborating teams. This list is based on reviews of existing software capabilities (Noël and Robert, 2003), interviews with collaborating editing teams (Noël and Robert, 2004), our experience in designing and using such software (Adler and Nash, 2004; Nash, Adler and Smith, 2003) as well as a review of the existing literature on collaborative work, e.g., Cerratto and Rodriguez (2002). While this list is large, and such features are not required by all such groups, our experience is that when such features are made available, users are quick to adopt them and soon consider them essential.

2.1. Management of Time and space

Collaborative projects can be categorized according to space and time (Johansen *et al.,* 1991). The work can take place in the same room (proximal space) or in several different rooms, buildings, or even countries (distal space). People can work together on the project at the same time (synchronous) or at different times (asynchronous). Table 1 shows examples of different collaborative applications for each of these situations. Time and space issues directly impact software design choices. For example, strictly asynchronous applications like email do not require attention to issues like simultaneity, while applications meant for synchronous, proximal work need to take into account the physical distribution of the potential users.

		Space	
		Proximal	Distal
Time	Synchronous	Electronic meeting software	Videoconferencing
	Asynchronous	Shared bulletin board	Email

Table 1. Examples of collaborative applications according to time and space.

2.2. Awareness

Awareness is an “understanding of the activities of others, which provides a context for our own activity” (Dourish and Bellotti, 1992). It is central to a successful collaboration; lack of awareness results in duplicate or neglected tasks. Awareness applies to either the task or the group. Task awareness can be promoted in part by letting group members know the current status of the document, by giving them access to recent modifications to the document (when and by whom), or by informing them who is supposed to do what (coordination). Group awareness can be promoted by clarifying the status of group members and their recent activities. Various technical methods of promoting awareness have been developed, such as using different colors for each member's input (task and group awareness), sending a notification whenever a document is modified (task awareness), showing each person's cursor in a synchronous system (group and task awareness) (Mitchell, 1996) or showing how busy the other members are to see if they can be interrupted (group awareness) (Dabbish and Kraut, 2004).

2.3. Communication

Communication is essential to collaboration, and can happen at any time during the collaborative process. Beck (2003), in a study on collaborative writing, found that groups tend to discuss the content and the structure of the final document while they are writing, while discussions concerning the work organization are done before, during, and after the process of writing. But communication tools can be problematic, since the chosen tool can affect the structure of the message and its reception (Posner and Baecker, 1993). There are many ways that people can communicate; while face-to-face meetings ensure a rich exchange, email and the phone are very popular ways of exchanging information (Noël and Robert, 2004). Other, less popular ways of communicating include chat, fax and videoconferencing tools.

Designers of collaborative software can choose to let the group communicate using external tools or incorporate communication tools in the collaborative application. The advantage of integrated communication tools is that they place the conversation in context (e.g., comments attached to specific paragraphs in word processors), thus increasing awareness. This decreases the possibility that the message becomes lost in the volume of communication that each group member can receive (e.g. email overload). Integrated communication has its disadvantages, the most important being that the user must open the application in order to access these messages. Because of this, an urgent message might not be seen in a timely manner. When comments are allowed, the software designer must decide whether they are separate from the artifact, thus isolating them from their context, or are integrated with or alongside the section to which they refer. In the latter case, the comments need to be easily distinguishable from the artifact itself, especially for a written document. Other potential problems (Cadiz *et al.*, 2000) with comments include orphaning, when the artifact to which a comment refers is removed, and irrelevance, when the section is so modified that the comment no longer makes sense.

The tendency of group meetings to use drawings and discussions stands in contrast to the communications capabilities offered by most collaborative software, which are usually limited to text. Other media, such as voice or free-hand drawings, could be used to facilitate and clarify users' communications. While technological limits explain past failures at integrating these other media into collaborative software, current computing power and network capabilities should make this possible.

2.4. Private and shared work spaces

In a synchronous system, users can see each other's contributions immediately. This raises the issue of privacy. People may wish to keep part of their work concealed, for example when taking private notes or when preparing a response. If a system does not offer a private workspace, those users who require space for private notes will leave such information off the system, making links between private notes and public contributions difficult to maintain. More importantly, once a user begins to use private notes, their commitment to using the collaborative system is reduced.

2.5. Intellectual property

One important social and legal issue that seems to have had little consideration in the context of collaborative software is intellectual property, especially when a collaborative application is used to share existing artifacts, such as slides, pictures or drawings. Some users may be willing to share an artifact they created without reserve, while others may want to retain intellectual property of the artifact and make it clear that it can only be used under certain conditions. We assume that it is beyond the capability of a software system to prevent copying entirely (in the most extreme case by photographing the screen). However, it would be useful to label and track such artifacts, particularly as an aid to editors who must obtain or verify permissions. We are not aware of any system that integrates these capabilities.

2.6. Simultaneity and locking

Simultaneity occurs when more than one person tries to work on the same copy of a document at the same time (Palmer and Cormack, 1998a, 1998b). Mitchell (1996) lists several ways of avoiding the possible conflicts that may arise from simultaneous work. With social control, the application has no internal mechanism to avoid these conflicts, leaving the group members to negotiate among themselves how they will work on the document. With mutual exclusion, the application lets only one person work on the document at any one time. Locking lets several people work on the document at the same time by making the part of the document that one person is working on unavailable to others. Transactional operations let several people work on the same document at the same time by recording the individual operations made by each user and then transmitting these operations to the other users and ensuring that the order of operations is kept intact (Sun, Sun and Chen, 2004). Other ways of handling conflicts (Ignat and Norrie, 2003) include having the system reject all conflicting changes, having the system select arbitrarily one user's commands and reject all the others, or having the system generate different versions of the document in order to respect the intentions of all the users.

Locking can be non-optimistic or optimistic (Greenberg and Marwood, 1994). The non-optimistic approach requires that the user wait until the lock request has been granted before being allowed to edit the document. The optimistic approach assumes that most lock requests will be granted, and so it gives a tentative approval, letting users immediately edit the document. If a lock is denied, the changes must be deleted and the artifact restored to its original state.

The size, or the granularity, of locked sections can have a direct impact on the type of collaboration possible. When the granularity is large (e.g., several pages or a complete chapter), group members can only work on the same section in turns. When the granularity is smaller (e.g., a single sentence or a single paragraph), group members can work on the same section at the same time, just not at the exact same site. Grain size can be pre-determined by the designer, or it can be left to the group members to determine. Locked sections can be shared (only one person can work on the section but others can still read it) or exclusive (only the person working on the section can access it).

Although simultaneity and locking techniques offer a technological approach to preventing conflicting changes in a collection of documents, they cannot prevent all such conflicts. Consider a set of documents, A and B, which use inconsistent terminology. Two users independently discover this; the first edits A to match B, while the second edits B to match A. These changes

are clearly in conflict, and yet will be permitted by all conflict management techniques presented in this section.

2.7. Protection

The very nature of collaborative work requires that an application offer some sort of protection of the work that has been accomplished. If a person's contributions are erased by another, accidentally or deliberately, the project may be delayed and conflict generated. There are several methods to protect work, such as undo, change tracking, and version control.

"Undo" lets users recuperate from errors. With collaborative software, the designer must decide whether each person may undo only their own modifications, or others' as well. This last possibility corresponds, in effect, to rejecting changes in a change tracking system.

Change Tracking promotes task awareness by allowing users to see changes and who made them. However, displaying every change inside the document itself can increase the cognitive load of understanding the document. Offering different ways to display these changes such as making it possible to turn off tracking are good ways to help reduce this cognitive load (Kim and Severinson Eklundh, 2000). Changes can also be displayed separately. Although isolating the changes from the document can make understanding more difficult, it is a good way to display the document's evolution over time.

Version control tracks the evolution of a document, while archiving keeps older copies of the document. Users may be less reluctant to delete unused artifacts in a document knowing that these may be recovered from the version archives (Kim and Severinson Eklundh, 1998). Archiving can be automatic or manual. The latter has the advantage that versions can be labeled in ways meaningful to a project, but relies on the group members remembering to do it.

When several active versions of the document exist, users can become confused and make conflicting changes. Having a single version on a central server eliminates this potential problem. Another approach is to use social control, for example by requiring that the person wanting to work on a document declare their intentions to the others. Social control can be augmented by using a token to indicate which user has the right to edit. For example, the Perl software authors use the concept of "patch pumpking" named after a toy pumpkin (FAQs.org, c. 2004). A conflict management technique for collaborative software development is the merge function provided by packages such as CVS (Cederquist, 2002). In such a system, users may arbitrarily modify documents. If multiple users edit the same file, CVS attempts to merge the edits into the file of the second user wanting to commit changes. The merge operation is signaled to the user, who should then check its correctness. In software development practice anecdotal evidence suggests that "merge" typically works well. Unfortunately, merge is a much more difficult feature to provide at the application level for office applications, because it must be aware of the application file structure.

2.8. Workflow

Collaborative development of documents is an important managerial activity. For example, "visible management" techniques (Nash and Nash, 1997, 1998) assist organizations to correctly control workflow processes, including document creation and approvals. With inter-networked PCs and email communications, many organizations have been struggling to develop workflow

processes to ensure that appropriate verifications and approvals are performed and archived copies are maintained. For these applications, a collaborative document system must integrate workflow processing. For example, a draft presentation may need to have comments from users A and B before approval by C. To maintain awareness, the workflow status of a document must be clear to system users. Ideally, this information will be maintained with the document.

2.9. Security

Collaborative work introduces many security concerns that do not exist for the individual author. We categorize these as server vulnerabilities, client computer vulnerabilities, access control and access level control.

Server vulnerabilities: collaborative software typically requires a network-based server that, as a minimum, maintains a repository of file versions. Such a centralization of files and applications presents a concentrated point of vulnerability. Network servers are potentially vulnerable to cracking through the applications running on the server. In more complex applications, the server is required to perform more complex operations, exposing it to a larger "pool" of possible attacks, as well as "combination" attacks using vulnerabilities in multiple services.

Client vulnerabilities: collaborative editing is not without risk to a client PC, even when exchanging draft versions via email. Most sophisticated office suites support macro languages, which are a popular vehicle for computer virus transmission. This is especially true of the Microsoft Office application suite, largely because of its ubiquity (Ferguson and Schneier, 2003). Any collaborative application that runs software on the client machine could potentially be a vehicle for such virus transmission

Access control: typically collaborative endeavors have a well defined group of participants, with non-members excluded, although this is not true of some applications, such as wikis (wiki.org, c. 2004). To ensure legitimate access, most current systems require a login with a username and password. There are well known problems with password-based systems, largely from users forgetting passwords or inadvertently revealing them (Dhamija and Perrig, 2000). This is a significant concern in a web-based system, where a user logging into the system from an untrusted PC may inadvertently reveal access codes (Ferguson and Schneier, 2003).

Access level control: group members often have different access privileges. For example, documents may be available to certain people on a "view-only" basis. As in many endeavors, it is a good idea to have at least one person in charge of the project (Noël and Robert, 2004). The following system-access capabilities may be restricted: Editing, Document viewing, Access to previous versions, Access to change records and auditing software, Workflow and approvals, Adding new documents, Adding new group members, Changing member privileges, Forcible unlocking of files, Merging of conflicting versions, and Downloading of files from the system.

In terms of security, it is often far more difficult to protect against elevation of privilege than to exclude those with no access. This effect is made more difficult in a complex system, where there is a tendency to provide complex sets of configuration options, which make the system administrator's task more difficult. Even without operator error, the combinatorics of many configuration options makes system security difficult to test (Ferguson and Schneier, 2003).

2.10. File Formats

Most modern office productivity software stores documents in rigidly defined file formats. In

many cases these formats are proprietary, although there has recently been a shift to standardization of some formats such as the OpenOffice suite files (Oasis Open 2004). File format issues can introduce significant difficulties for collaborative applications, which must either store information in a binary format, or convert contents to and from these formats.

There is much useful information for the editing group that should be attached to a file (e.g., comments, version numbers, workflow status). Unfortunately, office document formats do not flexibly incorporate such information. Applications that seek to do this must store the information in a structured form elsewhere, or in unstructured comment fields in the document. Unfortunately, such comments may be moved, changed or completely removed by the application.

Another problem is the rapid pace of development of office suite applications. Files created with older versions of the software or different software often do not display or print with correct formatting, if they can be loaded at all.

File format issues introduce further difficulties to a user. For example, Microsoft Word stores viewing and printer settings in the document. Thus the simple action of opening a document on a machine with a printer default different from that in the document will change the file. If the user agrees with the automatic request to save the document, the file will have a different length and apparent content, even though no genuine changes were made.

2.11. Platform independence

Collaborative tools based on proprietary software on the client machine are limited to the computer platforms supported by these packages. Network-based collaborative tools let users work on different computing platforms. Web-based systems use a “standard” web browser as the client, although browser capability varies significantly. For example, in our TellTable system, we need a web browser with Java capability, but there are popular browsers that do not provide Java by default (e.g., Mozilla, recent versions of Internet Explorer).

2.12. User Benefit

Successful collaborative tools must provide their users with an overall benefit. Here we define 'user benefit' as encompassing ease of use, flexibility or other gain for each of the users. Inflexible software encourages users to keep “black book” documents of the “real” information, and only reluctantly enter data into the collaborative application. In some cases the collaborative software transfers the workload from one group to another, discouraging participation by the latter group (Grudin, 1992, 1994). To provide user benefit, a collaborative system must be sufficiently flexible and offer ease of use.

Flexible systems often introduce concerns about security, especially in collaborative financial spreadsheets applications. In our opinion, such concerns are best dealt with by audit software, which is able to detect patterns of suspicious and erroneous activity (Adler and Nash, 2004; Nash *et al.*, 2003). Such audit capability is facilitated by the existence of a version history of collaborative documents. A system that accidentally encourages users to use black book spreadsheets is extremely difficult to audit.

In the following section, we have excluded considerations on user benefit and security. Assessing these elements in a fair and impartial manner would require having full access to the systems, which unfortunately was not always possible.

3. Existing Technology Examples

Because there are so many offerings of collaborative tools and groupware which might be considered in a paper such as this, the following is a selective sample of approaches that allow us to render concrete the criteria above.

3.1. Word Processing

Many applications have been developed to support collaborative writing (for lists of existing and defunct systems, see Dourish and Bellotti, 1992; Mitchell, 1996; Newman and Newman, 1992; Noël, 2001; Noël and Robert, 2003; Posner and Baecker, 1993). In some cases, software is described in general terms in papers or used as a tool for attempting to assess user reactions, but details of the architecture and capability are difficult to find. An example is GroupWriter (CMI, 2004) from the University of Arizona Center for the Management of Information whose numerous papers and web page concern the collaborative experience rather than technical detail.

The importance of collaborative writing can be seen by the evolution of word processors, where early products were restricted to opening and editing only their own files. Today, people expect to be able to port documents from one application to another. However, this portability remains imperfect, particularly in respect to some of the formatting. The major word processors for single personal computers have also added features that are useful for collaborative writing, such as change tracking, commenting, and document comparison. Some collaborative word processing systems include:

BSCW (Basic Support for Co-operative Work) (Bentley *et al.*, 1997a, 1997b) aims to support all sorts of collaborative work, not just collaborative writing. Each group has a shared workspace on a central server, and accessing the shared workspace requires a login. Group members can upload or download documents to this workspace. Editing can be done on each member's personal computer or through the web browser (if members are using MS Office XP or 2003 on Win2000, XP or 2003, and IE 5.5+ as their browser). Version control is available, so that users do not accidentally destroy previous versions when uploading a new version of the document. - BSCW uses a Java applet to create an interface to the shared workspace. BSCW offers some basic workflow control through the use of project folders, which contain Gantt charts, and circulation documents, which contain a task list showing who is in charge of what. Users can communicate through notes that appear in discussion threads or by using the available email client.

Collaboration Manager for Word (CMW) by Chasseral (2004) is a collaboration system that can be used either in a peer-to-peer model, with each person installing the software on their computer, or in a client-server model, with the software installed on a central server. In either case, collaborators write with their copy of Microsoft Word 2000. There are concurrent access, access permissions, and process management functions such as audit trails.

CoWord (CoWord, 2004; Sun, Sun and Chen, 2004) is a real-time collaborative word processor that links Microsoft Word 2000 users together. It offers an unrestrained approach to collaborative editing, so there is no assurance that individual contributions are included, there is no validation, approval, workflow or auditing. Each user's contribution is color-coded. Simultaneity issues are resolved through transactional operations. A Collaborative Document

Repository Manager (CDRM) is installed on a computer to give users access to the shared documents. Document access is password protected.

EquiText (Equipe in Portuguese and Text in English) is a Brazilian project aimed at supporting distance learning (Rizzi *et al.*, 2000a, 2000b). Users write, edit and comment on an on-line document. Locking is done at the paragraph level. A document's history is available at all times and it is possible to see who contributed what to which paragraph. Deleted paragraphs can be retrieved but there is no validation, approval or workflow support.

MoonEdit (Dobrowolski , 2005) supports collaborative editing over the Internet through its own application, which runs on Windows, Unix and Linux. Several users can type at the same time and their contribution is color-coded. Documents can be saved either in the proprietary MoonEdit format, or as simple text or exported as HTML. A history slider makes it possible to view the complete evolution of the document, as well as who contributed what.

SubEthaEdit (2004) lets a group of Macintosh users edit a document in real time. Originally developed to support software developers, it is now used in other situations. One reported application is that of bloggers at conferences using it to share notes about presentations. When someone wishes to work on a document, they must get permission from its creator. Concurrency is handled through transactional operations, as there is no locking. To see what someone else is doing, one must explicitly click a person's name.

In an initial draft of this paper, we considered including blogs and wikis as collaborative authoring tools for text documents, which in a strict sense they are. On consideration of the many variations on these themes, however, along with the weak security and administrative control, we felt that they were evolving away from the tasks of authoring office documents. We thus do not consider blogs and wikis here.

3.2 Spreadsheets

We began our TellTable work seeking an audit trail of spreadsheet changes (Adler and Nash, 2004; Adler, Nash and Smith, 2003). Section 4 gives more details about TellTable as do Adler and Nash (2004), Nash *et al.* (2003), Adler,Nash and Smith (2004) and Adler, Nash and Noël (2004).

Applied Analytix (2004) offers a product that claims to turn a Microsoft Excel workbook into a secure multi-user application that includes an audit trail and security. This integrates Excel with an OLAP database, TM1, on which information is kept. As we do not have access to this product, we were unable to determine how it handles multiple users or simultaneity issues.

Microsoft Excel lets users share a spreadsheet over a LAN. The software allows users to see who is working on the document. Excel lets changes be tracked and accepted by the file owner, thereby resolving conflicts. The document is updated when someone attempts a “save”, or automatically at a preset interval. However, Excel does not provide protection of data entry fields from being changed (Coombes and Trim, 2004). We have yet to find a reliable and trustworthy method to ensure that the Excel change record is complete.

Myspreadsheet (2004), like TellTable, also uses the idea of a single spreadsheet on a central server accessible via a browser. Several users can synchronously manipulate the same spreadsheet, although it is unclear how simultaneity problems are handled. Importing of Lotus or Excel files is allowed. Myspreadsheet is written in Perl and requires a Unix or Linux server. Myspreadsheet does not offer an audit trail of changes, and even failed a simple compound interest calculation.

3.3. Presentations

CoPowerPoint (Collaborative Internet Computing Research Group, 2004) is a collaborative slide management software created by the same team that developed CoWord. CoPowerPoint migrates CoWord features to Microsoft PowerPoint. We do not know of any other such tools apart from our own TellTable (executing OpenOffice.org *impress*).

3.4. Characterization of systems

Table 2 shows a table of the collaborative software packages discussed in section III, in terms of the criteria for challenges developed in section II. Each system is categorized according to whether it offers full, partial or no support for each criterion, although such evaluation necessarily is somewhat subjective. The checklist categories "security","intellectual property" and "user benefit" are not used because we do not have adequate information on systems other than TellTable to assess these items. Although we were able to try out at least a test version for the majority of these software packages to create this table, we were not able to test CMW and Applied Analytix. .

Software	**Time**	**Space**	**Workspaces**	**File format**
Applied Analytix	Async	Distal	Shared	Proprietary
BSCW	Async	Distal	Private & Shared	None
CMW	Sync & Async	Distal	Shared	Proprietary
C WordPerfect	Async	Distal	Shared	Proprietary
CoWord	Sync	Distal	Shared	Proprietary
CoPowerPoint	Sync	Distal	Shared	Proprietary
EquiText	Sync & Async	Proximal	Shared	HTML
MS Word	Async	Distal	Shared	Proprietary
MS Excel	Sync & Async	Distal	Private & Shared	Proprietary
MoonEdit	Sync	Distal	Shared	Proprietary, simple text, HTML
MySpreadsheet	Sync	Distal	Shared	Proprietary
SubEthaEdit	Sync & Async	Distal	Shared	Proprietary
TellTable	Async & Sync	Distal	Shared	XML

Table 2A. A characterization of some collaborative authoring tools. *Async*: Asynchronous; *Sync*: Synchronous

Software	Platform independence	Awareness	Communication	Simultaneity and Locking
Applied Analytix	1	(unknown)	(unknown)	(unknown)
BSCW	Yes(*)	TL, TC, R	E, D	ME (when files are edited online)
CMW	2	TC	0	LP
C WordPerfect	W	TC	Cit	ME
CoWord	2a or 2b and 4	UL, CC	0	SA, TO
CoPowerPoint	3 and 4	UL, CC	0	SA, TO
EquiText	Yes	TC	Cst	LP
MS Word	W or M	TC	Cit	ME
MS Excel	W or M	TC, UL	WD	ME
MoonEdit	W, L, U	CC, UL	0	SA
MySpreadsheet	Yes	(unknown)	0	SA
SubEthaEdit	M	CC, UL	0	SA
TellTable	Yes	TC	0	ME or SA

Table 2B. A characterization of some collaborative authoring tools. *0*=None; *1*=Requires MS Excel; *2a*=Requires MS Word 2000; *2b*=Requires Word XP; *3*=Requires MS Powerpoint 2000; *4*=Requires Internet Explorer 5.0+; *W*=Windows OS; *M*=Mac OS; *L*=Linux OS; *U*=Unix OS; (*)Unless editing online; *TL*=Task list; *TC*=Track changes; *R*=Roles; *UL*=User list; *CC*=Colored cursor; *E*=Email; *D*=Discussion tool; *Cit*=Comments in text; *Cst*=Comments separate from text; *WD*=Web-based discussion tool; *ME*=Mutual exclusion; *LP*=Locking at paragraph level; *SA*=Simultaneous access; *TO*=Transactional operations.

Software	Protection	Workflow
Applied Analytix	(unknown)	Some audit trail
BSCW	V, AC	Project folders and circulation folders
CMW	AC	0
C WordPerfect	V	0
CoWord	H	0
CoPowerPoint	H	0
EquiText	V, H	0
MS Word	V	0
MS Excel	0	Basic auditing tool
MoonEdit	H	0
MySpreadsheet	(unknown)	0
SubEthaEdit	AC	0
TellTable	V, H	Audit trail possible for spreadsheet

Table 2C. A characterization of some collaborative authoring tools. *0*=None; *V*=Versioning; *AC*=Access control; *H*=Complete history of all transactions.

4. Implementation and use of TellTable

TellTable allows a single user application to be used collaboratively by running the application on a server and allowing users to access it from a java-enabled web browser (Adler, Nash and Noël, 2004; Nash, Adler and Smith, 2003, 2004). Users log into the application with a

username and password, and view a dynamically generated HTML page with a list of files to edit. When the user clicks to edit an office document file, it is opened with the appropriate application on the server and the display exported to the browser via the VNC (Virtual Network Computing, Richardson, 1998) protocol. The user's browser loads a java VNC viewer, and all keyboard and mouse activity in the applet are sent to the server. Because of the network efficiency of the VNC protocol, this approach is useable, even on low bandwidth connections, such as dial-up. The TellTable server framework can be used to execute many software packages that run on single-user PCs, though the main usage to date has been to run the OpenOffice.org software. We have, however, successfully tested Microsoft Office components run under the CodeWeavers Crossover Office environment.

TellTable uses the CVS versioning system (Cederqvist, 2002) to maintain a full history of the development of a document, including the time and authorship of versions. Note that other, functionally similar, version control tools could replace CVS as the back-end of the TellTable infrastructure. The CVS interface is maintained separately from the audit trail that is part of the OpenOffice *calc* spreadsheet files. The latter functionality is quite simply implemented by turning on change recording and display in a spreadsheet file, and then removing the menu commands that allow the recording and display to be disabled. We can also save an audit trail for OpenOffice word processing documents, presentations or drawings, but have as yet developed neither the semantics nor the software for analysis of such audit trails. By contrast, significant work has been done on TellTable Analyse, a companion program to TellTable, which provides an extensive set of tools to filter and display the change records of a *calc* spreadsheet.

The history and evolution of TellTable are documented elsewhere (Nash, Adler and Smith, 2003, 2004); however it is instructive to review here its impact on collaborative activities. We have used TellTable for several quite large, multi-section (and in one case dual language) university courses that had multiple professors and teaching assistants all recording marks. We have also used it in writing this paper, as well as several other academic articles. Despite the fact that the initial implementation of TellTable was on a rather limited-capacity server and the software was undergoing quite rapid revision, the early users managed to work surprisingly well with it. Our biggest problems arose because we imposed quite severe restrictions on the time of sessions, whereas some of our markers assumed they could record as they marked. Resulting timeouts proved a nuisance. We regard this as a configuration and setup issue.

How does TellTable fare when our checklist is applied to it?

1) Time and Space: TellTable offers both asynchronous (by locking the document) and synchronous work models. TellTable is meant for distal work although it is also possible to use it in proximity.

2) Awareness: TellTable supports both task and group awareness. Awareness of the task is increased by displaying the current version number and giving access to past versions. Group awareness is increased through change tracking (available within OpenOffice.org) as well as by showing who worked on each version. One simple way of increasing awareness could be to add an RSS (Real Simple Syndication, see RSS Advisory Board, c. 2004) feed that would alert users whenever a document they were tracking was modified.

3) Communication: There is no communication tool integrated within TellTable. Although in general, users communicate through email and phone, we have also observed that users leave each other messages embedded within the document. This suggests that people do require some type of embedded communication tool.

4) Private and shared work spaces: At present, there is no private work space within TellTable, although our experience with the system suggests that private notes are very useful.

5) Intellectual property: At present, there is no way to indicate intellectual property within TellTable. One approach that could be used for inspiration is that of the Creative Commons (c. 2004).

6) Simultaneity and locking: TellTable now offers two different editing styles. The document can be locked down completely (mutual exclusion) so that only one person can work on it at a time, or a writer can share the document with others. In the latter case, the group members have access to a single cursor, requiring some form of social control to decide who is in control of the cursor at any one time.

7) Protection: OpenOffice.org includes both change tracking and undo, while CVS furnishes version control for TellTable.

8) Workflow: At present, there is no workflow capability within TellTable. However, the access interface is set up to add such capability.

9) Security: Since TellTable runs applications and keeps its files on a central server, it is potentially vulnerable to numerous attacks, ranging from network attacks to those from users trying to escalate their privilege level. Protection is implemented by careful configuration of the TellTable server, including minimization of the actions allowed to psuids. A detailed description of the software design of TellTable is given by Adler, Nash and Noël (2004). Users have to log in with a password. Access level is controlled through the specification of user privileges by the system administrator. While working within TellTable, users have limited access to the server.

User behavior can impact the security of web-based collaborative solutions. One user was generally unable to use the software. Upon observing him, we discovered he was clicking on a web link to the server from within a “hotmail” account. Hotmail presents links within a frame in its window, and edits applets and javascript to prevent applications from “breaking out" of the hotmail window. If this interface had provided the full functionality of the web (including cookies and applets) and did this fast enough, we would not have detected the vulnerability. However, the result was that access to our server was usually impossible. The solution to both the access and security issues was to avoid use from within hotmail. However, the scenario illustrates that it would be fairly easy to mount a “man in the middle” attack on a web based collaborative editing system. Such an attack could capture keyboard events, including passwords, as well as gaining access to sensitive documents.

10) File formats: By placing the application on a central server, TellTable ensures that all users use the same version of said application.

11) Platform independence: By using open source tools and applications, TellTable can be used on various operating systems and on various browsers.

12) User benefit: The philosophy behind TellTable is to enable collaborative use of single-user software. This differs from many other collaborative systems, which are designed and built with collaborative use in mind. The advantage of the TellTable approach is the flexibility of using many different software packages, and the ability to leverage the considerable work that has been put into such single-user software, and user familiarity with it. Our experience (from personal use and pilot tests, Adler and Nash, 2003) is that TellTable is flexible and easy to use.

We have implemented a tool to allow copy-and-paste from external sources of material for documents. The usual copy/paste methods are blocked as a consequence of security provisions in the Java-VNC viewer that runs in the Web-browser. Our current mechanism lets the user load text or a file into a special clipboard that becomes a second OpenOffice window within the VNC environment, which then lets regular copy and paste be used.

The VNC protocol allows two or more users to share the “screen”. Controls to allow such synchronous editing for this have been added to the interface. Clearly, such a feature presents an

entirely new set of usability and human interaction issues, but offers many new possibilities for the TellTable infrastructure.

OpenOffice is not the only tool we can run for handling office documents. We have experimentally verified that we can run Microsoft Office components, though practical use will require legal advice on whether such operation from a server is permitted under the Microsoft EULA. We have also tested the Gnumeric spreadsheet which is somewhat smaller, and thus loads faster, than OpenOffice *calc*. We believe most single-machine programs can be shared collaboratively through TellTable.

Since we have built TellTable from many available software components, any deficiencies in our building blocks are carried forward. At the time of writing we are conducting a review of the TellTable server software to improve its documentation, security and clarity, and the VNC server and client code to correct what we believe is a packet handling error under high network load conditions.

The web-interface to TellTable allows for the addition of many customized workflow features via coding that is little more than web-page creation and editing. Similarly, we are expanding, documenting and streamlining the administrative "pages" for TellTable itself.

Because files are centrally stored on a server, there is the possibility that a server castastrophe could result in loss of information. With TellTable, this is more a perceived issue than a reality, since we have included automatic backup to an external repository. However, it will be useful to inform users, possibly by providing information within the main file-access interface page.

5. Conclusions

We set out to evaluate our collaborative editing system, TellTable, and developed a checklist based on the following challenges to collaborative editing software: 1) time and space, 2) awareness, 3) communication, 4) private and shared work spaces, 5) intellectual property, 6) simultaneity and locking, 7) protection, 8) workflow, 9) security, 10) file format, 11) platform independence, and 12) user benefit. This has helped us to better identify the strengths and weaknesses of TellTable and should help guide us with future development of this tool. We believe that no current system, including our own, addresses all of these issues, and clearly, many applications do not need to do so. On the other hand, our experience, even for small groups with fairly simple requirements, is that the availability of each of these features would provide significant benefit (Adler and Nash, 2004).

We also note that TellTable's collaborative capability came about as a result of the application of a web-based interface to a more or less traditional office-suite package, along with a little "glue". The keys to the benefits we have realized are, in our view,

1) the storage of files on server;

2) the re-use of existing software components (CVS, OpenOffice, Apache-SSL); and

3) the use of web-based tools to provide the interface.

Currently, people who use the Internet to accomplish common collaborative tasks, such as editing documents, are often frustrated by the limits of the software tools available – which are typically not designed with such collaboration in mind. Given the importance of collaborative

editing, it seems essential that good tools be developed to allow efficient and structured work. In this paper, we have attempted to clarify issues in this development by developing a list of challenges of collaborative editing systems, and by describing and characterizing our TellTable system in that context. .

Acknowledgments

This project was supported by funding from NSERC Canada.

References

A. Adler and J.C. Nash (2004) "Knowing what was done: uses of a spreadsheet log", *Spreadsheets in Education (eJSiE)*, Vol. 1, pp. 118-130.

A. Adler, J.C. Nash and S.Noël (2004) TellTable: a Server for Collaborative Office Applications , Proc. CSCW 2004, Chicago, November 6-10 2004, http://dsonline.computer.org/collaborative/events/iwces-6/.

Applied Analytics, (2004, Mar.) [Online], http://www.appliedanalytics.com

Beck, E. (1993) "A survey of experiences of collaborative writing", in *Computer Supported Collaborative Writing* (M. Sharples, Ed.). London: Springer-Verlag, 1993, pp. 87-112.

R. Bentley, T. Horstmann, and J. Trevor (1997a) "The World Wide Web as enabling technology for CSCW: The case of BSCW". *Computer Supported Cooperative Work: The journal of Collaborative Computing*, Vol. 6 , pp. 111-134.

R. Bentley, W. Appelt, U. Busbach, E. Hinrichs, D. Kerr, K. Sikkel, J. Trevor and G. Woetzel (1997b) "Basic support for cooperative work on the World Wide Web", *International Journal of Human Computer Studies: Special Issue on Novel Applications of the WWW*, Vol. 46, pp. 827-846.

J.J. Cadiz., A. Gupta and J. Grudin (2000) "Using Web annotations for asynchronous collaboration around documents", *Proc. ACM Conf. Computer Supported Cooperative Work*, pp. 309-318.

P. Cederqvist (2002) *Version management with CVS*, Bristol UK: Network Theory, 2002.

Center for the Management of Information (2004) http://www.cmi.arizona.edu/collaboration/Group%20Writer.html (accessed 2004-10-24).

T. Cerratto and H. Rodriguez (2002) "Studies of computer supported collaborative writing: Implications for system design". *Proc. COOP'2002, Fifth Int. Conf. Design of Cooperative Systems*, 2002.

Chasseral, Collaboration Manager for Word, (2004, Mar.) [Online], http://www.chasseral.com/products/index.shtml (accessed 2005-03-09).

Collaborative Internet Computing Research Group (2004) "CoPowerPoint Demo Centre", School of Computing and Information Technology, Griffith University, Brisbane, Queensland, Australia http://reduce.qpsf.edu.au/copowerpoint/ (accessed 2005-03-09).

P. Coombes and A. Trim, (2004, Mar.) *Authentication and protection of validated spreadsheets*. Web document, [Online]. http://www.wgivalidation.com/auth.pdf

CoWord, (2004, Mar.) [Online]. http://reduce.qpsf.edu.au/coword/ (accessed 2005-03-09).

Creative Commons (c. 2004) "Learn More about Creative Commons", http://creativecommons.org/learnmore

L. Dabbish and R.E. Kraut (2004) "Controlling interruptions: Awareness displays and social motivation for coordination", *Proc. CSCW2004*, pp.182-191, 2004.

R. Dhamija and A. Perrig (2000) "Deja Vu: A user study using images for authentication", *Proc. 9th Usenix Security Symposium*.

T. Dobrowolski (2005) MoonEdit [Online] http://me.sphere.pl/indexen.htm (accessed 2005-06-15)

P. Dourish and V. Bellotti (1992) "Awareness and coordination in shared workspaces", *Proc. CSCW'92*, pp.197-214.

FAQs.org (c. 2004) http://faqs.org/docs/perl5int/x122.html (accessed 2005-03-02).

N. Ferguson and B. Schneier (2003) Practical Cryptography, New York: John Wiley & Sons.

S. Greenberg and D. Marwood (1994) "Real time groupware as a distributed system: Concurrency control and its effect on the interface", *Proc. Conf. Computer Supported Cooperative Work*.

J. Grudin, "Why CSCW applications fail: Problems in the design and evaluation of organizational interfaces" (1992) in *Groupware: Software for Computer-Supported Cooperative Work* (D. Marca and G. Bock, Eds.), pp. 552-560. Los Alamitos, CA: IEEE Computer Society Press.

J. Grudin (1994) "Groupware and social dynamics: Eight challenges for developers". *Communications of the ACM*, vol. 37, pp. 92-105.

IBM Corporation (2003) The History of Notes and Domino, LDD Today, 29-Sep-2003, http://www-10.lotus.com/ldd/whatisnotes, (accessed 2004-10-02).

C.-L. Ignat and M.C. Norrie (2003) "Grouping/ungrouping in graphical collaborative editing systems", Proc. 5th Int. Workshop on Collaborative Editing Systems, Helsinki, Finland.

R. Johansen, D. Sibbet, S. Benson, A. Martin, R. Mittman and P. Saffo (1991) *Leading Business Teams: How Teams can use Technology and Group Process Tools to Enhance Performance*. Reading, Mass.: Addison-Wesley.

E. Kim and K. Severinson Eklundh (1998) "How Academics Co-ordinate their Documentation Work and Communicate about Reviewing in Collaborative Writing", Report #TRITA-NA-P9815, IPLab-151, Royal Institute of Technology (KTH), Sweden.

E. Kim and K. Severinson Eklundh (2000) "Change Representation in Collaborative Writing". Report #TRITA-NA-P0005, Royal Institute of Technology (KTH), Sweden.

A. Mitchell (1996) *Communication and Shared Understanding in Collaborative Writing*. Unpublished Master's Thesis. Computer Science Department, University of Toronto.

K. Moore (1995) "The Lotus notes storage system"**,** *Proceedings of the 1995 ACM SIGMOD international conference on Management of data,* New York: ACM SIGMOD, pages 427-428.

Myspreadsheet, (2004, Mar.) [Online] http://www.myspreadsheet.com accessed 2005-03-09

J. C. Nash and M. Nash (1997) "The Visible Management system: management ideas with library principles", in *Communication and Information in Context: Society, Technology and the Professions*, (Bernd Frohmann, ed.), Proc. 25th Conf., Canadian Assoc. Information Science: Toronto, pp. 124-130, 1997.

J. C. Nash and M. Nash (1998) "Visible management for design, programming and other creative processes", *Proc. CSME Forum 1998*, Vol. 3, (Editors M A Rosen, D Naylor, and J C Keewall), Ryerson Polytechnic University, pp. 224-230, 1998.

J. C. Nash, A. Adler and N. Smith (2003) "Audit and change analysis of spreadsheets", *Proc. 2003 Conf. European Spreadsheet Risks Interest Group*, eds. David Chadwick and David Ward, Dublin, London: EuSpRIG, pp. 81-90.

J. C. Nash, A. Adler, and N. Smith (2004) "TellTable Spreadsheet Audit: from technical possibility to operating prototype", *Proc. 2004 Conf. European Spreadsheet Interest Group*, eds. Patrick Cleary and David Ward, Klagenfurt, Austria, July 14-16, pp 45-56.

J. Newman and R. Newman (1992) "Three modes of collaborative authoring", in *Computers and Writing: State of the Art* (P.O. Holt and N. Williams, Eds.), Oxford: Intellect Books, pp. 20-28.

S. Noël (2001) "Comment assister l'écriture collective sur le Web", Unpublished Master's Thesis, Ecole polytechnique de Montréal.

S. Noël and J.-M. Robert (2003) "How the Web is used to support collaborative writing". *Behaviour & Information Technology*, Vol. 22, pp. 245-262.

S. Noël and J.-M. Robert, (2004) "Empirical Study on Collaborative Writing: What do co-authors do, use, and like?", *Computer Supported Cooperative Work: The Journal of Collaborative Computing*, Vol. 13, pp. 63-89.

Oasis-Open.org, (2004, Mar.) http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=office (accessed 2005-03-09).

C.R. Palmer and G.V. Cormack (1998a) "Lock-free distributed objects: A shared spreadsheet". Department of Computer Science, University of Waterloo. Technical Report CS-98-04.

C.R. Palmer and G.V. Cormack (1998b) "Operation transforms for a distributed shared spreadsheet", *Proc. Computer Supported Cooperative Work (CSCW'98)*, Seattle, WA.

I.R. Posner and R.M. Baecker (1993) "How people write together", in *Readings in Groupware and Computer-Supported Cooperative Work: Assisting Human-Human Collaboration* (R.M. Baecker, Ed.), pp. 239-250.

T. Richardson, Q. Stafford-Fraser and K.R. Wood, K.R. (1998) "Virtual Network Computing", *IEEE Internet Computing,* Vol. 2, pp. 33-38.

C.B. Rizzi, C.M.M.C. Alonso, L.M.J. De Seixas, J.S. Costa, F.R. Tamusiunas and A. Da Rosa Martins (2000) "Collaborative writing via Web – EquiText", *7th Cong. Int. Informatica en Educacion.*

C.B. Rizzi, C.M.M.C. Alonso, E.B. Hassan, L.M.R. Tarouco and L.M.J. De Seixas (2000) "EquiText: a helping tool in the elaboration of collaborative texts", *Proc. 11th Int. Conf. SITE'2000*, 2000.

RSS Advisory Board (c. 2004) "RSS at Harvard Law", http://blogs.law.harvard.edu/tech/currentBoard accessed 2005-3-3

SubEthaEdit, (2004, Mar.) [Online], http://www.codingmonkeys.de/subethaedit/ (accessed 2005-03-09).
D. Sun, S. Xia, C. Sun and D. Chen (2004) "Operational transformation for collaborative word processing", *Proc. CSCW2004*, pp. 437-446.
Wiki.org (c. 2004) http://wiki.org/wiki.cgi?WhatIsWiki (accessed 2005-03-08).