Challenges in Collaborative Authoring Software

Andy Adler
School of Information Technology and Engineering
University of Ottawa, Ontario, Canada

John C. Nash
School of Management
University of Ottawa, Ontario, Canada

Sylvie Noël
Communications Research Centre
Industry Canada, Ottawa, Ontario, Canada

Abstract

Collaborative work with office suite documents demands new tools and methods for their control and ease-of-use. We identify twelve challenges to such collaborative software: time and space, awareness, communication, private and shared work spaces, intellectual property, simultaneity and locking, protection, workflow, security, file format, platform independence, and benefit. We use these challenges to measure the success of TellTable, a web-based framework for collaborative use of office-suite tools.

I Introduction

Many work environments require collaborative writing and editing of documents, drawings, presentations and spreadsheets; a good example is a scientific paper, 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 having to leave their office.

However, the use of these new technologies can also cause problems. Multiple copies of the same document can lead to confusion, as group members make conflicting modifications to the document. "Electronic" meetings may cause or inflame conflicts in the group as these tools are weak in conveying emotional context. Unfortunately, since the de-facto way to jointly edit a document is to exchange draft versions between authors via email, there is typically no authoritative document or master copy. All collaborative authors are familiar with the version control "nightmare" that can result. One example: "Why did the document that you sent to the client omit the changes I made? Yes, you included the changes that I sent to Bob, but those were only for section 'A'; I'm sure I emailed these to you on Tuesday!" For most people, the only reliable bidirectional electronic communications protocol is email, and it is thus used for office suite applications, such as collaborative editing, spreadsheet modelling or analysis, and slide presentation creation, for which it was never designed.

In order to address the requirements of collaborative editing, many different software systems have been developed, by large and small companies, and academic teams (including ours). Unfortunately, the requirements of collaborative teams can be much more difficult than those for standalone software. In this paper, we compile a list of challenges in meeting the needs of collaborating authors. Subsequently, we review a selection of notable approaches to software design, and analyse them in terms of the challenges. Additionally, since collaborative editing is a new and rapidly changing field, we make a series of predictions of which issues and challenges will become important in this field. The goal of this paper is threefold: to discuss the principal social and technical issues, to review existing collaboration tools, and to explore future trends.

In this paper, we will refer to documents as the end product that collaborative editing groups are trying to create; documents can refer to a text, a drawing, a spreadsheet file, or a presentation. We use "artifact" to refer to the individual sections of a document. For example, a paragraph, a graph, or a single slide constitute artifacts.

I Collaborative Software Challenges

We have developed the following list of software capabilities that are required by most medium to large collaborating teams. This list is based on reviews of existing software capabilities [36], interviews with collaborating authoring teams [37], as well as our experience in designing and using such software [1, 32]. While this list is large, and such features are not required by all such groups, our experience is that as such features are made available, users are quick to adopt them and soon consider them essential.

1 Management of Time and space

Collaborative projects can be categorised according to space and time [23]. The work can take place in the same room (proximal space) or in several different rooms, buildings, and countries (distal space). People can work together on the project at the same time (synchronous) or at different times (asynchronous). Figure 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 work on 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

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

2 Awareness

Awareness is ‘an understanding of the activities of others, which provides a context for your own activity’ [13]. 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 colorsfor each member's input, sending a notification whenever a document is modified, or showing each person's cursor in a synchronous system.

3 Communication

Communication is essential to collaboration, and can happen at any time during the collaborative process. Beck [3], 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, as the chosen tool can affect the structure of the message and its reception [42]. There are many ways that people can communicate. Although face-to-face meetings ensure a rich exchange, email and the phone are also very popular ways of exchanging information [37]. Other, less popular ways of communicating include chat tools, fax machines, 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 to integrated communication tools is that they place the conversation in context (e.g. comments attached to specific paragraphs in word processors), thus increasing awareness. It also 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 must decide whether they are separate from the artifact (thus isolating them from their context) or integrating them with or alongside the section they refer to. In the latter case, the comments need to be easily distinguishable from the artifact itself, especially when this is a written document. Other potential problems [6] with comments include orphaning, when the section a comment refers to 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 discussion stands in contrast to the communications capabilities offered by most collaborative software. The communication medium offered is largely text, while other media could be used, such as voice or free-hand drawings, which may facilitate and clarify users' exchanges. While technological limits explain past failures

at integrating these other media into collaborative software, current computing power and network capabilities should make this possible.

4 Private and shared work spaces

In a synchronous system, users can see (almost) immediately each other's contributions. This raises the issue of privacy. It may be that people wish to keep part of their work concealed, for example when taking private notes or when preparing a response. Currently, very few collaborative applications appear to offer this capability. This implies that users who require space for private notes will leave this information off the system. In that case, links between private notes and contributions become difficult to maintain. More importantly, once a user begins to use private notes, their commitment to using the collaborative system is reduced.

5 Intellectual property

One important social 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 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 (after all, what software can prevent someone from photographing the screen?). However, a useful system in this regard would label and track such artifacts. We are not aware of any system with these capabilities.

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. Mitchell [28] 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. Other ways of handling conflicts [22] 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 [17]. 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.

With locking, the size, or the granularity, of locked sections can have a direct impact on the type of collaboration possible. When the granularity is large (for example, several pages or a complete chapter), group members can only work on the same section in turns. When the granularity is smaller (for example, 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 the size of the section they want to lock down. 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).

While simultaneity and locking techniques attempt to prevent conflicting changes in a collection of documents, they cannot prevent such conflicts. Consider a set of documents, A and B, which use inconsistent terminology. Users C and D independently discover this; C edits A to match B, and D 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.

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 someone else, accidentally or deliberately, this may put the project behind schedule and lead to conflict. There are several methods that can be used to protect work, such as *undo*, *change tracking*, and *version control*.

Undo lets users recuperate from errors. However, how many steps can the person undo? Should each person only be allowed to undo their own modifications or should they be able to undo other people's modifications? 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 of displaying these changes or making it possible to turn off tracking are good ways to help reduce cognitive load [26]. Changes can also be displayed separately. Although isolating the changes from the document can make understanding more difficult, this method 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 [25]. Archiving can be automatic or manual. The latter has the advantage that versions can be labelled in ways meaningful to a project, but relies on people remembering to archive the document.

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 declares their intentions to the others. Social control can be augmented by using a token, such as a small toy or button, 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). A conflict management technique for collaborative software development is the *merge* function provided by packages such as CVS [7]. 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 *merge* typically works well. Unfortunately, *merge* is a much more difficult feature to provide for office productivity applications, because it must be aware of the application file structure.

8 Workflow

Collaborative development of documents is an important managerial activity. For example, "visible management" techniques [30, 31] assist organizations to correctly control workflow processes, including document creation and approvals. With internetworked 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. In order to maintain *awareness*, the workflow status of a document must be clear to system users. Ideally, this information will be maintained with the document.

9 Security

Collaborative work introduces many security concerns that do not exist for the individual author. We categorize these as 1) server vulnerabilities, 2) client computer vulnerabilities, 3) access control and 4) access level control.

Server vulnerabilities: collaborative software typically requires a network based server which, as a minimum, maintains a repository of file versions. Such 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 writing is not without risk to a client PC, even if 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 [15]. Any collaborative application which 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, while others are excluded (although this is not true of some applications, such as wikis). To ensure

legitimate access, most current systems require a login with a username and password. There are well known problems with password based systems, from users forgetting passwords, or inadvertently revealing them. This is a significant concern in a web-based system. A user logging into the system from an untrusted PC may inadvertently reveal access codes.

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 [37]. The following system-access capabilities may be restricted: 1) Editing, 2) Document viewing, 3) Access to previous versions, 4) Access to change records and auditing software, 5) Workflow and approvals, 6) Adding new documents, 7) Adding new group members, 8) Changing member privileges, 9) Forcible unlocking of files, 10) Merging of conflicting versions, and 11) Download 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 [15].

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 [38]. File format issues can produce significant difficulties for collaborative applications. Such applications can either store information in such a binary format, or will be required to convert contents to and from these formats.

The requirements of *communication* mean that information should be attached to a file (comments, version numbers, workflow status, etc.). Unfortunately, office document formats do not flexibly incorporate such information. Applications which 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 developed 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 also introduce further irritants 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.

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. In our TellTable system, we need a web browser with Java capability, but there are popular browsers that do not provide Java by default (Mozilla, versions of Internet Explorer with Win XP).

12 Benefit

Successful collaborative tools must provide their users with an overall benefit. Inflexible software encourages users to keep “blackbook” documents of the “real” information, and 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 [20, 21]. In order to provide *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 [1, 32]. Such audit capability is facilitated by the existence of a version history of collaborative documents. A system that results in users maintaining blackbook spreadsheets is impossible to audit.

II Existing Technology

1 Wikis and Blogs

Wikis [27] are Web sites that aim to simplify Web page creation and maintenance for shared information. Wikis operate under an open and free access philosophy and do not impose any kind of workflow, procedure, or approval system. Weblogs (blogs), are personal publishing web pages. Archiving is automatic for both tools. Wikis have been described as “converging” since the original point and the various authors' identities are lost or blurred over time, while blogs are “diverging”, since they keep the content from the various authors (blogger and commentators) separate [14].

2 Word processing

Many applications have been developed to support collaborative writing (for lists of existing and defunct systems, see [13, 28, 33, 35, 36, 42]).

Indeed, 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. We now 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 have also added features that are particularly useful for collaborative writing, such as change tracking, commenting, and document comparison. Some collaborative word processing options include:

BSCW (Basic Support for Co-operative Work) [4, 5] 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, but work is done on each member's personal computer. There is no control over concurrency, so a newly uploaded version destroys any previous version. BSCW uses a Java applet to create an interface to the shared workspace. Because BSCW requires users to work with local software, there is no enforced change management and workflow.

Collaboration Manager for Word by Chasseral [9] 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 [12] 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 [45, 46]. 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. There does not appear to be any kind of validation, approval or workflow support.

SubEthaEdit [47] 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 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 appears to be handled through transactional operations, as there is no locking. To see what someone else is doing, the person must deliberately choose that person and click on their name to see their cursor and their work.

3 Spreadsheets

We began our work seeking an audit trail of spreadsheet changes [1, 32]. Section IV gives more details.

Applied Analytix [2] offers a solution that claims to turn an Excel workbook into a secure multi-user application that includes an audit trail and security. Their solution integrates Excel with an OLAP database, TM1. Information is kept on a central database. It is unclear how this solution 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, one issue with Excel files [11] is that it is impossible to protect the data entry fields from being changed. These authors did not find a trustworthy e-signature solution for Excel which could have provided approval and protection security.

mySpreadsheet [29] 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. Importation of Lotus or Excel files is allowed. mySpreadsheet is written in Perl and requires a Unix or Linux server. It does not appear to offer an audit.

WARP (Wide Area Resource Programming), a research project at the University of St Andrews, explored supporting users via networked computers. The researchers designed and implemented a prototype of a shared spreadsheet on Unix [50]. Another academic example is that of Palmer and Cormack [40, 41], who report on a prototype shared spreadsheet based on *sc*, Gosling's Unix spreadsheet. Their main interest was in developing and testing a concurrency algorithm.

4 Drawings and Presentations

CoPowerPoint is the first collaborative slide management software that we have encountered. Created by the same people who developed CoWord, CoPowerPoint migrates CoWord features to Microsoft PowerPoint.

Early examples of collaborative drawing tools include VideoDraw [49], Aspects [22, 28], Ensemble [34], GroupDesign [24], GroupDraw and GroupSketch [18] and CoDraw [19]. More recently, there have been new attempts at developing collaborative drawing tools.

Cliki [16] is a collaborative drawing tool based on the same principle as the Wiki Wiki Web: anyone can access a Cliki-based drawing and edit it inside their Web browser.

GRACE (GRAphics Collaborative Editing) is a project aimed at supporting real-time, synchronous collaborative drawing over the Internet [10, 48]. GRACE is an object-based, rather than bitmap-based, graphics system. Their approach is to replicate the shared document at each user's site so that editing is done on the local copy, and then propagated to the remote copies.

Moksha [43] is a prototype collaborative system that includes, among other things, a vector-based drawing tool. It incorporates sound to help promote awareness of actions that may be happening outside of the user's visual focus.

The Global Information Systems Group at the University of Zurich is developing the Universal Information Platform, a distributed information management system. One of the applications being built for the UIP is a collaborative graphical editor [22]. In addition to the basic operations offered by other collaborative drawing tools (create, delete, move, change color, change position), their editor also offers grouping and ungrouping operations.

OPEN_Studio [39] is a Web-based drawing environment that allows groups to work on and share drawings. It uses a Java applet to let users define a color (by controlling the amount of red, green, and blue), select a texture and brush size.

Our TellTable server framework can be used to execute OpenOffice.org *draw* or *impress* software on the server, and export a view to this application to a user's browser via the VNC java applet. At the time of writing, we have not conducted any pilot studies of this application, but use this capability internally to manage presentations.

Figure 2 shows a table of the collaborative software packages discussed in section III, in terms of the criterion for challenges developed in section II. Each system is analyzed according to whether it offers full, partial or no support for each criterion. Clearly, in many cases, this table represents our subjective assessments. Situations in which we were not able to determine the capabilities are indicted with a “?”.Some of the drawing systems are not included because they are still in development and it is therefore impossible to tell what features they will eventually include.

Software	Time	Space	Work spaces	File Format	Platform independence	Awareness	Communication	Simultaneity and Locking	Protection	Workflow
Wiki	SA	D	S	HTML	Yes	P	0	0	F	0
Blog	A	D	S	HTML	Yes	F	P	0	P	0
MS Word	A	D	S	Proprietary	No	P	P	P	F	0
C WordPerfect	A	D	S	Proprietary	No	P	P	P	F	0
BSCW	A	D	S	None	Yes	P	P	0	0	0
EquiText	SA	P	S	HTML	Yes	F	P	P	F	0
SubEthaEdit	SA	D	S	Proprietary	No	P	0	P	P	0
CoWord	S	D	S	Proprietary	No	F	0	F	F	0
CMW	S	D	S	Proprietary	No	?	?	?	?	0
MySpreadsheet	S	D	S	Proprietary	No	?	?	?	?	0
MS Excel	SA	D	P/S	Proprietary	No	P	P	P	F	0
Applied Analytix	A	D	S	Proprietary	No	?	?	?	?	P
CoPowerPoint	S	D	S	Proprietary	No	F	P	F	F	0
TellTable	A	D	S	XML	Yes	F	P	P	F	P

S=Synchronous
A=Asynchronous

S=Shared
P=Private

P=Proximal
D=Distal

0= no capability
P= partial capability
F= full capability

Figure 2. Comparison of representative collaborative software for office files.

III TellTable Tales

This document has been collaboratively written using the TellTable application server framework [1]. This software is free (under the LGPL license) and available at http://telltable-s.sf.net. Users access TellTable with a standard internet browser. The user logs into a web application and is presented with a list of files, and has the option to edit, view or download each file. Upon selecting a file to edit, a VNC [44] java applet is opened in the browser showing a view of the document to be edited using one of the OpenOffice.org office suite tools; all mouse and cursor activities are sent to the server to be processed, and the results returned to the user's browser window. Versions of files are maintained in a CVS repository. To date, we have chosen a mutual exclusion lock that lets just one user at a time edit a file, but all authorized users can download a copy of the file. So far, we have only developed audit tools for spreadsheets, since audit displays must match the particular office applications.

TellTable-s runs on a Linux platform using Perl scripts in an apache web server. A pool of userids with limited permissions running the vncserver software is maintained, and one is allocated to a

user editing a file. On the server, the allocated userid is given the file to be edited, and opens it with the OpenOffice.org application. Using the VNC client in the browser, the user is able to view the file and edit it on the server. Fortunately, the VNC protocol is extremely efficient [44], and this framework is responsive, even over a dialup connection. Security is maintained by carefully restricting the permissions of the allocated userid. A one-time password is created for each edit session, and is exchanged via the browser to the vncserver. Figure 3 shows a screenshot of this document being edited using TellTable.

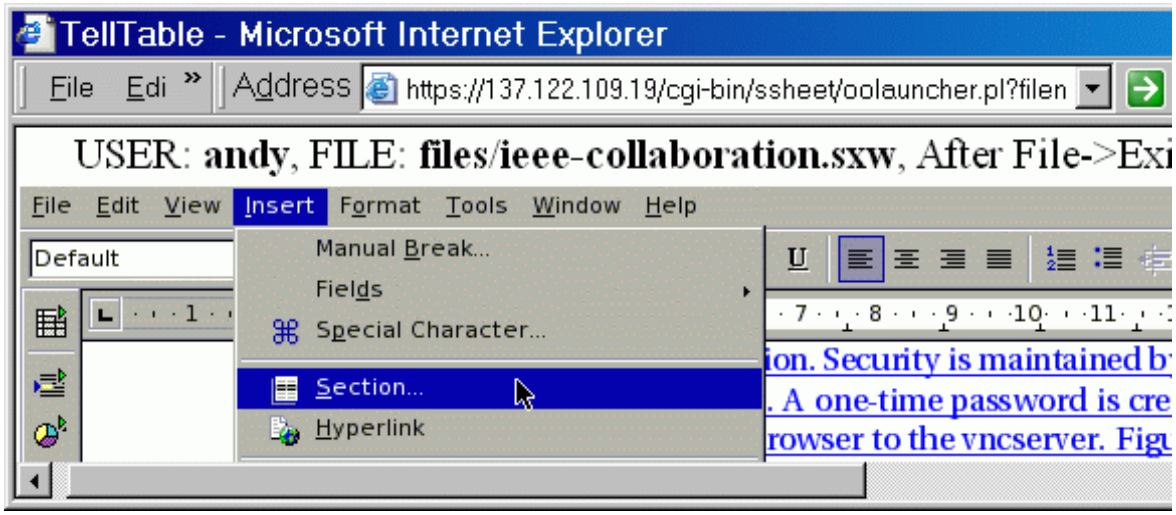

Figure 3: TellTable collaborative software editing this paper.

All of the authors have worked with traditional "email exchange" techniques to collaboratively author papers, and overall preferred this collaborative process, especially the communication and version control. The software provided immediate access to either the most recent version, or a notice that another author was editing the document. In contrast "email" processes demand considerable time and energy to determine if other authors have made modifications. Version control made it easy to delete unused material, knowing that it remained available in previous versions. We were able to work from anywhere using a browser – a convenient feature for mobile authors.

We discovered some difficulties with this approach. As always, there is a "learning curve", primarily associated with the teamwork aspects. Interestingly, some people are very reluctant to use collaborative tools and insist on keeping a personal repository of all documents. Since, the TellTable software (as yet) does not separate communication from content, we discovered that it was necessary to leave bold, italicized notes in the text, and make phone calls, to resolve "strategic" issues such as choice of terminology. The nature of collaboration can change with the phase of the project; a deadline can mean that a team will want to switch the collaborative process to be synchronous, and possibly face-to-face. Because TellTable does not use custom file formats, such transitions were accomplished smoothly.

User behavior can impact security of web-based collaborative solutions. One user was consistently 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” from within the hotmail window. In this case, the result was that access to our server was impossible, and the solution was to avoid use from within hotmail. However, this scenario illustrated 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.

IV Conclusions and Predictions

There are many current options available for authoring collaborative documents, spreadsheets, drawings and presentations. The possibilities range from simple systems such as wikis and blogs, to full application suites. In order to compare such systems, it is useful to clarify the requirements of collaborative editing software. Based on the literature, we have identified the following 12 *challenges* of collaborative 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) benefit. No current system addresses all of these issues. Clearly, many applications do not need to address all challenges. 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 [1].

The need for collaborative software is increasing. As organizations move toward paperless processes, there is an important void in document management. Whereas, in the past, businesses carefully tracked and archived all written correspondence, today's fast paced, networked, approach to business means that much important communication is only stored in the participant's email “Sent” folders, if at all. Properly managed, collaborative software facilitates this fast paced work, while maintaining workflow. We believe the following issues will become important in this field.

1. Although most communication solutions still use text, it will become important for other *media* to be used to convey information, such as voice or free-hand drawings, in order to facilitate and clarify users' communications. One of us uses a Wacom tablet to mark Acrobat versions of student assignments.
2. Within collaborative software, there is a need for *communication* at three levels: the work itself, the organization of the work and content, and about use of the collaborative tool. Cerratto and Rodriguez [8] showed that provision of a channel for each type of communication clarifies the work.
3. Without *benefit* to users, collaborative systems will not work well; users will tend to maintain a private repository of their work. Since there are people who seem not to like using collaborative software, it will be necessary to discover their reasons.
4. The *security* of collaborative applications is very important. Systems, such as Wikis, which do not provide security seem destined to become targets of spammers. Also, as collaborative software becomes more widespread, scripted attacks into such systems will become common. Especially vulnerable are web-based systems (such as our TellTable system) in which login

information can be recorded as users work with unsecured PCs. Furthermore, the login/security process must be quicker, since delays inhibit use.

5. Recording of *intellectual property* information will become important. As parts of documents are "cut and pasted" into others, it is vital to keep track of which artifacts may be redistributed and which may not.
6. The current rapid pace of software development means that software versions and *file formats* change regularly; documents created with older versions will display incorrectly or not at all. This is an increasing concern as paper records are no longer systematically kept.

V References

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

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

[3] E. Beck, "A survey of experiences of collaborative writing", in Computer Supported Collaborative Writing (M. Sharples, Ed.). London: Springer-Verlag, 2003, pp. 87-112.

[4] R. Bentley, T. Horstmann, and J. Trevor, "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, 1997.

[5] R. Bentley, W. Appelt, U. Busbach, E. Hinrichs, D. Kerr, K. Sikkel, J. Trevor and G. Woetzel, "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, 1997.

[6] J.J. Cadiz., A. Gupta and J. Grudin "Using Web annotations for asynchronous collaboration around documents", Proceedings of the ACM Conference on Computer Supported Cooperative Work, pp. 309-318, 2000.

[7] P. Cederqvist, Version management with CVS, Bristol UK: Network Theory, 2002.

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

[9] Chasseral, Collaboration Manager for Word, (2004, Mar.) [Online], http://www.chasseral.com/products/index.shtml

[10] D. Chen, Consistency maintenance in collaborative graphics editing systems. Unpublished doctoral thesis, Griffith University, 2001.

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

[12] CoWord, (2004, Mar.) [Online]. http://reduce.qpsf.edu.au/coword/

[13] P. Dourish and V. Bellotti, "Awareness and coordination in shared workspaces", Proc. CSCW'92, pp.197-214, 1992

[14] L. Efimova, (2004, Mar.). Wikis and blogs: convergent and divergent conversations, [Online]. http://blog.mathemagenic.com/2004/01/20.html#a913

[15] N. Ferguson and B. Schneier, Practical Cryptography, New York: John Wiley & Sons, 2003.

[16] D. Gordon, Cliki, (2004, Mar.) [Online], http://www.mcs.vuw.ac.nz/~donald/?Cliki

[17] S. Greenberg and D. Marwood, “Real time groupware as a distributed system: Concurrency control and its effect on the interface”, Proc. Conf. Computer Supported Cooperative Work, 1994.
[18] S. Greenberg, M. Roseman, D. Webster and R. Bohnet, “Issues and experiences designing and implementing two group drawing tools”. Proc. Int. Conf. System Sciences, Vol. 4, pp. 138-150, 1992.
[19] M.D. Gross, “Graphical constraints in CoDraw”, Proc. IEEE Workshop on Visual Languages, pp. 81-87, 1992.
[20] J. Grudin, “Why CSCW applications fail: Problems in the design and evaluation of organizational interfaces”, in Groupware: Software for Computer-Supported Cooperative Work (D. Marca and G. Bock, Eds.), pp. 552-560. Los Alamitos, CA: IEEE Computer Society Press, 1992.
[21] J. Grudin, “Groupware and social dynamics: Eight challenges for developers”. Communications of the ACM, vol. 37, pp. 92-105, 1994.
[22] C.-L. Ignat and M.C. Norrie, “Grouping/ungrouping in graphical collaborative editing systems”, Proc. 5th Int. Workshop on Collaborative Editing Systems, Helsinki, Finland, 2003.
[23] R. Johansen, D. Sibbet, S. Benson, A. Martin, R. Mittman and P. Saffo, Leading Business Teams: How Teams can use Technology and Group Process Tools to Enhance Performance. Reading, Mass.: Addison-Wesley, 1991.
[24] Karsenty and M. Beaudouin-Lafon, “An algorithm for distributed groupware applications”, Proc. 13th Int. Conf. Distributed Computing Systems, pp. 195-202, 1993.
[25] E. Kim and K. Severinson Eklundh, “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, 1998.
[26] E. Kim and K. Severinson Eklundh, ”Change Representation in Collaborative Writing”. Report #TRITA-NA-P0005, Royal Institute of Technology (KTH), Sweden, 2000.
[27] Leuf and W. Cunningham, The Wiki Way: Quick Collaboration. Addison-Wesley, 2001.
[28] Mitchell, Communication and Shared Understanding in Collaborative Writing. Unpublished Master’s Thesis. Computer Science Department, University of Toronto, 1996.
[29] Myspreadsheet, (2004, Mar.) [Online] http://www.myspreadsheet.com
[30] J.C. Nash and M. Nash, “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.
[31] J.C. Nash and M. Nash, “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.
[32] J.C. Nash, A. Adler and N. Smith, “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. 2003.
[33] J. Newman and R. Newman, “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, 1992.

[34] R.E. Newman-Wolfe, M. Webb, and M. Montes, “Implicit locking in the Ensemble concurrent object-oriented graphics editor”, Proce. ACM Conf. Computer Supported Cooperative Work (CSCW'92), pp. 265-272, New York, 1992.
[35] S. Noël “Comment assister l’écriture collective sur le Web”, Unpublished Master’s Thesis, Ecole polytechnique de Montréal, 2001.
[36] S. Noël and J.-M. Robert, “How the Web is used to support collaborative writing”. Behaviour & Information Technology, Vol. 22, pp. 245-262, 2003.
[37] S. Noël and J.-M. Robert, “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, 2004.
[38] Oasis-Open.org, (2004, Mar.) [Online], http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=office
[39] OPEN_Studio, (2004, Mar.) [Online]. http://draw.artcontext.net
[40] C.R. Palmer and G.V. Cormack, “Lock-free distributed objects: A shared spreadsheet”. Department of Computer Science, University of Waterloo. Technical Report CS-98-04, 1998.
[41] C.R. Palmer and G.V. Cormack, “Operation transforms for a distributed shared spreadsheet”, Proc. CSCW'98, Seattle, WA, 1998.
[42] I.R. Posner and R.M. Baecker, “How people write together“, in Readings in Groupare and Computer-Supported Cooperative Work: Assisting Human-Human Collaboration (R.M. Baecker, Ed.), pp. 239-250, 1993.
[43] R. Ramloll and J. Mariani, “Do localised auditory cues in group drawing environments matter?”, Proc. ICAD'98, Glasgow, UK, 1998.
[44] T. Richardson, Q. Stafford-Fraser and K.R. Wood, K.R., ”Virtual Network Computing”, IEEE Internet Computing, Vol. 2, pp. 33-38, 1998.
[45] C.B. Rizzi, C.M.M.C. Alonso, L.M.J. De Seixas, J.S. Costa, F.R. Tamusiunas and A. Da Rosa Martins, “Collaborative writing via Web – EquiText“, 7th Cong. Int. Informatica en Educacion., 2000.
[46] C.B. Rizzi, C.M.M.C. Alonso, E.B. Hassan, L.M.R. Tarouco and L.M.J. De Seixas, “EquiText: a helping tool in the elaboration of collaborative texts“, Proc. 11th Int. Conf. SITE'2000, 2000.
[47] SubEthaEdit, (2004, Mar.) [Online], http://www.codingmonkeys.de/subethaedit/.
[48] Sun and D. Chen, “Consistency maintenance in real-time collaborative graphics editing systems“. ACM Trans. Computer-Human Interaction, Vol. 9, pp. 1-41. 2002.
[49] J.C. Tang and S.L. Minneman, “VideoDraw: A video interface for collaborative drawing“. Proc. SIGCHI Conference on Human Factors in Computing Systems: Empowering People, pp. 313-320, Seattle, WA., 1990.
[50] WARP Project, “Design and Implementation of a Shared Spreadsheet“. Division of Computer Science, University of St Andrews, WARP Report W5-95, 1995.