Shared Books: Collaborative Publication Management for an Office Information System

Brian T. Lewis

Acorn Research Centre
5 Palo Alto Square, Suite 910
Palo Alto, California 94306
acornrc!lewis@decwrl.dec.com

Jeffrey D. Hodges

Xerox Corporation
475 Oakmead Parkway
Sunnyvale, California 94086
Hodges.osbunorth@Xerox.com

Abstract

A *Shared Book* helps the users of an office information system create a multiple-part publication and manage it throughout its life cycle. The Shared Book supports simultaneous collaboration both by allowing different workers to work on different parts at the same time and by ensuring that workers use the current revision of each part. It protects publication information by providing locking and access control. The Shared Book communicates the publication's current state to each worker through a What-You-See-Is-What-I-See (WYSIWIS) display. Early users report that the Shared Books application has improved their productivity by simplifying their file management and by helping them track the production status of their publications. Shared Books will be part of the next release of the Xerox ViewPoint document processing system. In this paper we present the results of our study of publication management needs and describe how Shared Books supports them. We also describe some interesting aspects of its user interface and implementation.

I. Introduction

The result of information processing in offices is often a document. Managing these documents often requires extra effort by office workers either because they are produced by a team of people or because they are revised frequently. Examples of these "difficult" publications include service manuals and programming manuals. Part of the difficulty lies in locating the correct *revisions* or *specific incarnations* of these documents. Ensuring that these correct revisions are used, updated, printed, and distributed takes effort. When several people work together to produce a document, coordinating and tracking the progress of their work also requires effort. Even in the "electronic office", this *publication management* is still primarily done manually. Mistakes lead to increased cost, missed deadlines, and lost information.

This paper describes Shared Books, which is an application we developed to support collaborative publication management in the Xerox ViewPoint document processing system. A ViewPoint installation consists of several workstations linked to shared file, communication, and print services. Each workstation has its own local file system. Sharing is done through files stored on file services, but users must copy these files between workstations and services. ViewPoint also has a distributed electronic mail system. These existing facilities help in preparing documents, but they still require manual effort and therefore are error prone. This is especially true when several workers are involved and when the publication consists of several independently-edited parts. For example, two people may simultaneously edit separate copies of the same document, or one may be unable to find all the correct parts of a multiple-part publication.

The goal of Shared Books is to help ViewPoint users concentrate more on their publication and less on the production process. We have accomplished this by modeling Shared Books after *Job Dockets*, a common publishing shop tool, and by building on our experience with tools for maintaining large software systems [Lewis 83].

Job Dockets are manila folders that contain the materials needed to produce a publication. This includes drafts of text and illustrations, galleys and page proofs,

design specifications, correspondence, and production schedules. A *job sheet* is often pasted on a docket's cover to identify the job and list its production schedule: a list of tasks, dates, and worker assignments. Dockets are passed from one worker to another as each job step is completed. Unlike a docket, however, a Shared Book is intended to allow several users to work on a publication at the same time.

Shared Books is currently being tested within Xerox and will be included in the next major release of ViewPoint. It is being used by its implementors and by a technical writing group within Xerox.

The next section presents the results of our study of publication management functions. The third section describes systems functionally similar to Shared Books. Section Four outlines Shared Books' user interface. Section Five discusses how Shared Books meet publication management needs. The sixth section describes experience with the use of Shared Books. Section Seven summarizes this paper and outlines possible future work.

II. Publication Management

As a result of interviews with publishing professionals, we found the three key categories of publication management to be *storage and retrieval*, *document binding*, and *job management*.

Storage and retrieval

A key requirement for any publication management system is support for efficiently finding, retrieving, and storing documents. This must include *all* document information. First, the system must manage all the information necessary to reproduce the document. Besides content, this might include style rules, spelling dictionaries, and boilerplate content shared by several documents. Second, the system should also manage such related information as contracts, specifications, correspondence, production schedules, and accounting data.

Publications have a revision history: they are changed over time, which results in a sequence of revisions. It is highly desirable for a publication management system to be able to reproduce older revisions of a document. Some publications also have multiple parallel *versions*, each with its own independent revision history. If old revisions are seldom retrieved, archiving might be provided to store them more cheaply.

The publication management system must reliably store information despite computer or network failures or attempts by unauthorized people to modify information. Loss or corruption of information can have financial, and sometimes legal, consequences. The system must also prevent two different people from attempting to modify the same information at the same time.

Document binding

Binding aids in the assembly of a multiple-part publication by providing automatic numbering and cross referencing across all parts. It may also include generation of front and back matter such as a table-of-contents or an index.

When several people work on a publication, there is a need to check that all parts, including the front and back matter, are numbered and cross referenced consistently. The parts must each refer to the same revisions of such auxiliary information as style descriptions, whether that information belongs to the publication or is "imported" from elsewhere.

Job management

A job management facility may provide support for organizing the production of a document. This includes assistance not only in planning tasks and responsibilities but also in tracking the progress of work on the publication. It may also include support for estimating the cost of the job and for maintaining accounting and statistical information about its production.

Besides help for producing individual documents, a job management system may also provide between-job support. This may include work schedules for people, peak-load reports for production departments, and reports of all jobs currently in progress.

The job management system should recognize that there are different work roles involved in publishing: for example, writing, designing, and copy editing. These roles often have different requirements. For example, an illustrator probably should not be allowed to revise a job's accounting information. As noted in [Greif and Sarin 86] however, the same person will sometimes have different roles at different times and the system should accommodate this.

Flexibility is another requirement. A job management system must accommodate different production styles. Most publishing groups are highly structured and publishers may refuse to bend their existing procedures to fit a new job management system.

III. Related Work

The Shared Books facility is an extension of ViewPoint's existing (non-shared) Books facility. Both Shared Books and (non-shared) Books aid in the assembly and formatting of complex publications by serving as containers for multiple document parts. They assist document binding tools in automatically numbering

objects and in generating indexes and tables of contents. The two facilities differ, however, in that (non-shared) Books allow only a single person to work on a publication at a time.

Shared Books is closely related to work on system models [Schmidt 82, Marzullo and Wiebe 86, Leblang and Chase 84] which describe software systems. Shared Books and system models both contain revision and construction information but system models also contain module interconnection information. Shared Books is an outgrowth of the Description File (DF) system originally developed by Schmidt [Schmidt 82] and later modified to better support groups of closely cooperating software developers at Xerox [Lewis 83].

The Books facility of Interleaf Inc.'s Technical Publishing Software (TPS) [Interleaf 86] is similar to ViewPoint's (non-shared) Books facility. TPS's Books hold multiple-part publications and provide support for automatic numbering, index generation, and table-of-contents generation. Additionally, users may manually create "shared" TPS Books by using TPS's generic "desktop links" feature. A link icon points to a real file stored elsewhere on a user's workstation or on the network. Using links allows multiple users to simultaneously change different parts of a shared publication and gives TPS's Books functionality similar to Shared Books. The two key differences between a Shared Book and a TPS Book are: (1) a Shared Book provides explicit and automatic support for collaboration, and (2) a Shared Book provides status and tracking information about a publication in a What-You-See-Is-What-I-See (WYSIWIS) display whereas a TPS Book does not display such information.

The Collaborative Editing System (CES) [Seliger 85] has functionality that significantly overlaps that of Shared Books. CES enables a group of authors to work together on a document at the same time. Different authors can work concurrently on different sections. Each author views, and may edit, the document's structure in a display that resembles a table-of-contents. This display is similar to that provided by a Shared Book but does not include revision or status information. Like Shared Books, CES provides read and write access control although CES does not allow privileges to be set separately for each section. CES also provides locking.

Although Shared Books is not a hypertext system, it does have collaboration features that resemble those of such hypertext systems as NoteCards [Trigg et al 86], TEXTNET [Trigg and Weisner 86], Neptune [Delisle and Schwartz 86], and Intermedia [Garrett et al 86]. These allow a group of people to share a network of linked documents that they can browse, annotate, modify, and link. These systems provide locking. Some systems, such as Intermedia, provide access controls. Neptune also supports the storage and retrieval of multiple revisions.

IV. User Model Overview

In the ViewPoint system, a closed Shared Book appears as an icon on the desktop of a user's workstation. Each worker collaborating on a publication has a copy of that icon on their desktop.

A Shared Book is a ViewPoint *container*. Containers are common ViewPoint objects. A container holds a list of *entries* and allows users to manipulate the entries as a unit. Examples include file folders and electronic mail in-baskets [Smith et al 82]. Because a Shared Book is a container, ViewPoint users already know the basic operations on it and its entries. For example, a user may copy an icon into a Shared Book in the same way he copies it into a folder.

The Shared Book window

When the user opens the icon, a window appears that displays the Shared Book's entries (see Figure 1). This window resembles a table-of-contents. Each entry, which corresponds to a part of the publication, is displayed in a row. There are two classes of entries: body and auxiliary. Body entries provide the publication's content, while auxiliary entries may hold additional information such as correspondence.

The Shared Book window displays most of the information about each entry. A separate "Entry Details" property sheet, opened when requested by the user, displays infrequently needed information such as an entry's access controls. The intent is to keep only the most important information at the user's focus of attention. Besides an entry's name and class, information displayed in a Shared Book's window includes:

- Lock status: a small icon showing whether the entry is locked by the user (solid black), locked by someone else (gray), or is currently not locked (no icon).
- Revision number: a number that is automatically incremented when an edited entry is *Saved*, which makes the changes available to others.
- Creation date: timestamp of the most recently saved revision.
- Notes: status information or comments entered by a worker.

The information displayed is kept up to date in a *relaxed WYSIWIS* fashion [Stefik et al 86]. A change on one workstation, such as adding a new entry, is not immediately broadcast to all others. Instead, new

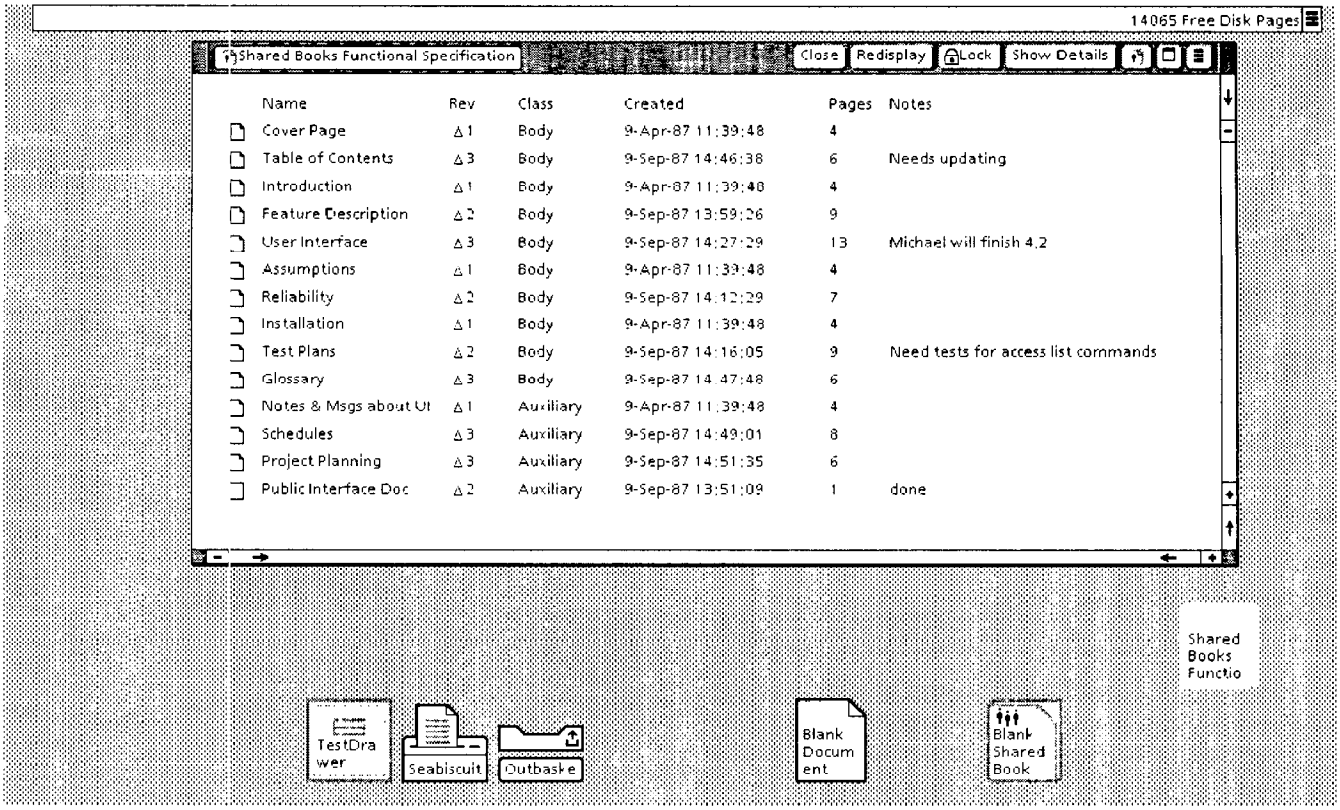

Figure 1. An open Shared Book window.

information is automatically retrieved and the window updated on the next user action. Updating only as needed is cheaper to implement. There is no danger of doing an operation with out-of-date information since the Shared Books facility ensures that a user has up-to-date information before allowing an action to proceed. This method of demand updating is similar to the "no-notification" scheme described in [Garrett et al 86].

Entry details property sheet

Detailed information about an entry is shown when the user selects an entry and invokes the "Show Details" command in the Shared Book's window header (see Figure 2). The Entry Details property sheet shows the entry's access list, and if it is locked, who locked it, when, and why (the "Reasons" field). The user who locks an entry may edit its Reasons field to explain to other collaborators what he is doing. The user may also change the entry's access list, notes, and class. He may also unlock the entry if he changes his mind.

Manipulating entries

Standard ViewPoint system commands such as Open, Copy, and Paginate may be applied to entries in a Shared Book by selecting one entry, or a range of adjacent entries, and invoking the command. Users must have appropriate access privileges.

In addition, Shared Books provides two new operations on entries: Lock and Save. Lock gives a user the exclusive right to modify an entry. The WYSIWIS displays on the workstations of other collaborators will show that the entry is locked. No time limits are imposed on locks and locks are granted on a first-come, first-served basis. The two Save commands, "Save Minor Revision" and "Save Major Revision", unlock entries and make new revisions available to other workers. The two commands differ only in how they increment the revision number of the Shared Book itself.

Manipulating an entire Shared Book

Operations on the entire publication are done by selecting the Shared Book icon and invoking the operation. For example, copying the Shared Book icon to a printer icon will paginate the publication if necessary and print it.

V. Shared Books' Support for Publication Management

In this section we discuss in detail how Shared Books provides for the three main publication management functions: storage and retrieval, document binding, and job management.

Storage and retrieval

A Shared Book provides a container in which users work on a multiple-part publication. Users do not need to keep track of where the parts are stored. They only need an icon for the Shared Book on their desktop. This icon acts as a reference to the "real" Shared Book container remotely stored on a network file service.

Besides the entries, the remote Shared Book contains a data file that lists the names of all entries, the timestamps of their latest revisions, and other information. This file supplies most of the data displayed in the Shared Book window. It is managed (transferred, updated, and locked) automatically and users are unaware of its existence.

We chose to implement Shared Books with a centrally-stored data file that is read and manipulated by copies of the Shared Book software running on each workstation for two reasons. First, no general database service was available on the XNS network when we started our development. Second, we did not want to develop a new network service to manage shared book information since at that time such services were relatively difficult to write and to integrate with other service software.

Files are transferred on demand from the file service and cached on the workstation. For example, opening an uncached entry will automatically retrieve the entry before the open operation proceeds. It is not necessary to have all the Shared Book's entries on the workstation all the time. This caching of whole files on the workstation is similar to the Caching File System (CFS) [Schroeder et al 85] and the Andrew File System [Howard et al 87].

Since editing entries will change only the cached files, users must save them to make the new revisions available to other workers. The Save commands check each entry locked by the user to see if it was changed. Revised entries are both stored in the remote Shared Book and unlocked; unchanged entries are only unlocked. Users do not have to remember what entries they have changed. Because the Save commands store these entries on a file service, Shared Books provides users with a convenient way to back up their work.

The Shared Book window displays current information about each entry to all users collaborating on a publication. More precisely, it displays information

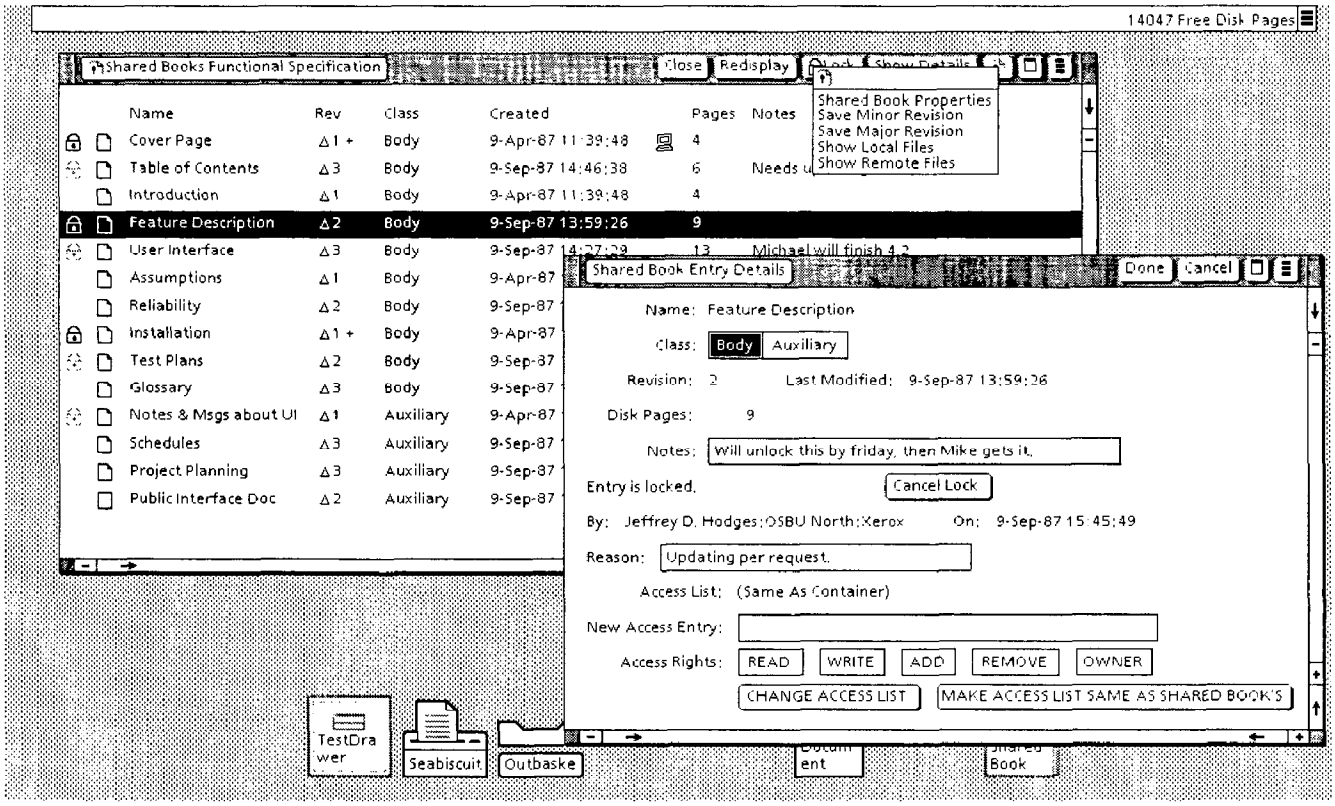

Figure 2. The user has opened the Entry Details property sheet for the entry *Feature Description* and filled out the notes and reasons. Shared Books will apply these changes when the user selects **Done** in the property sheet header. Also, the user has made changes to *Cover Page* and *Installation*, but has not yet Saved them. The "+" indicators in the Rev column indicate this. Only the user making the changes sees this indicator; notice that the entries locked by other workers (indicated by the gray lock icons) do not display them. The open auxiliary menu displays the two Save commands plus other commands used to manipulate the Shared Book's files.

about the most recently saved revision. Even when someone else has locked an entry, the user may browse the most recently saved revision. This allows users to share stable intermediate results. [Greif and Sarin 86] describes the usefulness of this capability. Shared Books does not currently display or automatically retrieve previous revisions of entries.

The Shared Books facility reliably manages information by providing locks and access controls. Besides the lock for each entry, there is a lock for each Shared Book. Updates to a Shared Book's data file are serialized by transparently acquiring its lock. For example, invoking a command that will change the data file, such as inserting an entry into a Shared Book, causes the Shared Books facility to attempt acquiring the lock. If successful, the command is allowed to proceed and the lock is held for the duration of the command. Otherwise, the user is told that the Shared Book is currently in use and is shown the other user's name. Locks are implemented by a *librarian service* on the network. This method of updating the shared data file is similar to the centralized-lock control regime described in [Stefik et al 87].

Access controls may be specified for the Shared Book itself and for each entry. The Shared Book's access list establishes both the default and the maximum permissions for all its entries. Individual entries may have more restricted permissions.

The file service provides reliable transfer of single files. Shared Books builds on this to provide reliable execution of its commands. It guarantees that if a fault occurs during execution of a command, in particular a Save command, corrupted data will be detected. Further operations on a Shared Book will not be allowed to proceed until the Shared Book is recovered. Currently, users must recover a corrupted Shared Book manually. This model of "on demand" crash recovery by software running on workstations rather than servers is similar to that presented in [Paxton 80].

Document binding functions

The ViewPoint Document Editor and associated applications implement pagination, table-of-contents generation and index generation. They will operate on both Shared Book icons and (non-shared) book icons. However, using a Shared Book guarantees publication consistency. Pagination *edits* documents since assigning page numbers modifies document content. Shared Books automatically locks and caches correct revisions of all body entries at the start of a binding operation. It will not allow the operation to proceed if any body entries are locked by another user.

Additionally, the user may select one or more entries in an open Shared Book window and invoke a binding function. The entries are processed regardless of their class (body or auxiliary). For example, this allows users to paginate auxiliary entries, which are ignored when the entire Shared Book is paginated.

Job management functions

Shared Books provides simple but flexible job management support on which users may build their own procedures. Shared Books helps users track the current state of a publication by displaying revision numbers, timestamps, lock indicators, and the identities of users holding locks. Shared Books does not have integrated support for task scheduling or cost estimation. However, users may use auxiliary entries as a convenient way to store this job information along with the publication.

The Notes and Reasons fields provide additional basic job management support. Since users may enter any text into them, they may be used to hold both procedural and annotative information [Trigg et al 86]. Procedural messages refer to the production process while annotative messages refer to the publication's content.

Access lists and user identities can be used to support different roles for users. A user may have several user identities, each with different access rights to a Shared Book. One identity ("production editor") may be able to change the Shared Book's access list, while another identity ("author") is able to edit the Shared Book's entries, and a third ("reader") may only read entries.

VI. Experience Using Shared Books

A group of five technical writers, responsible for writing and updating the Xerox Network Services (XNS) reference documentation, has used a preliminary version of the Shared Books facility since October 1986. This documentation consists of six publications and 1800 printed pages. The writers report that they have found the locking and backup support of the Shared Books facility especially useful. Before using Shared Books, they had trouble with more than one person editing the same part simultaneously. They also nearly lost several days of work when two workstations failed without warning. The users also report that they are increasingly using Shared Books in their daily work.

The writers have used Shared Books in a way that we did not anticipate. Instead of placing the material for each publication in a separate Shared Book, they divide up the parts of the six publications into four shared books with the following titles:

Chapters
Quick Reference Guides
Glossary Submissions
Release Deliverables

This reflects the way in which they did their work before starting to use Shared Books. The writers primarily use Shared Books as containers with locking and simplified backup. Although their style of using Shared Books is unexpected, it tends to validate one of our design goals: that the system must be flexible in accommodating different styles of work.

We, the Shared Books implementors, use Shared Books to manage our own documentation. Our experience is that they simplify file management and make it easier to coordinate changes. Also, performance is adequate despite the latency caused by the transfer of whole files.

VII. Summary and Conclusions

The goal of Shared Books is to assist groups of workers with their publication storage and retrieval, document binding, and job management tasks. Our intent is to help users concentrate more on their publication and less on the process. Although we did not have the resources to implement complete support for each function, we believe that Shared Books does provide significant publication management help for collaborators.

Preliminary experience with Shared Books shows that it supports key publication management functions in a useful and flexible manner. It is a productivity tool: by reducing the effort and number of mistakes involved in the creation, revision, and management of large multi-part documents, Shared Books improves its users' productivity. Users find Shared Books flexible; the technical writers adapted Shared Books to their methods as opposed to adapting to Shared Books. They also report finding Shared Books easy to use since it is an extension of existing ViewPoint facilities.

Shared Books demonstrates the effectiveness of a WYSIWIS (What-You-See-Is-What-I-See) user interface in multiperson publication management. Each collaborator working on a Shared Book can determine the publication's status and that of each part. The Notes and Reasons fields allow workers to communicate procedural and annotative information to each other. Users find the table-of-contents-like display intuitive and easy to use.

In the future, we plan to work closely with customers who are using Shared Books to refine our understanding of their publication management needs. We hope to provide better support for the display and recreation of previous revisions of a publication. We also hope to provide publication archiving and improved document binding support. We expect to study the ways in which workers use Shared Books in order to improve its WYSIWIS user interface.

Acknowledgements

Shared Books was implemented by the authors and Mike Kupfer, Bill Maybury, Jenny Richlin, Michael Tallan, and Steve Tom. Sara Bly and Brady Farrand worked on the user interface design. Sara in particular made significant contributions to clarifying the Shared Books user model. The following people contributed to improving this paper: Dan Craft, Brady Farrand, Dave Fylstra, Rachel Rutherford, Dave Schroit, Marion Sturtevant, Michael Tallan, and Don Woods. We especially thank both Mike Kupfer, who contributed key ideas to early drafts, and Bill Mallgren, who coaxed us into clarifying the paper's contribution. We thank Kathryn Lewark for tirelessly tracking down our references.

References

[Delisle and Schwartz 86] Norman Delisle and Mayer Schwartz. "Contexts--A Partitioning Concept for Hypertext". Proceedings of the Conference on Computer-Supported Collaborative Work, Austin, Texas, December 3-5 1986.

[Garrett et al 86] L. Nancy Garrett, Karen E. Smith, Norman Meyrowitz. "Intermedia: Issues, Strategies, and Tactics in the Design of a Hypermedia Document System". Proceedings of the Conference on Computer-Supported Collaborative Work, Austin, Texas, December 3-5 1986.

[Greif and Sarin 86] Irene Greif and Sunil Sarin. "Data Sharing in Group Work". Proceedings of the Conference on Computer-Supported Collaborative Work, Austin, Texas, December 3-5 1986.

[Howard et al 87] J. Howard, M. Kazar, S. Menees, D. Nichols, M. Satyanarayanan, R. Sidebotham, M. West, "Scale and Performance in a Distributed File System". Proceedings of the Eleventh ACM Symposium on Operating Systems Principles, Austin Texas, November 8-11 1987.

[Interleaf 86] "Technical Publishing Software Reference Manual". Volume 1, Interleaf Inc., Cambridge Massachusetts. 1986.

[Leblang and Chase 84] David B. Leblang and Robert P. Chase, Jr. "Computer-Aided Software Engineering in a Distributed Workstation Environment". Proceedings of the ACM SIGSOFT/SIGPLAN Software Engineering Symposium on Practical Software Development Environments, Pittsburgh, Penn., April 1984.

[Lewis 83] Brian T. Lewis. "Experience with a System for Controlling Software Versions In a Distributed Environment". Proceedings of the Symposium on

Application and Assessment of Automated Tools for Software Development, IEEE and University of Texas at Austin, San Francisco, California, November 1-3, 1983.

[Marzullo and Wiebe 86] K. Marzullo and D. Wiebe, "Jasmine: A Software System Modelling Facility". Proceedings of the ACM Symposium on Practical Software Development Environments, Palo Alto, California, December 9-11, 1986.

[Paxton 80] William H. Paxton. "A Client-Based Transaction System To Maintain Data Integrity". Xerox Palo Alto Research Center Report No. CSL-80-3, March 1980.

[Seliger 85] Robert Seliger. "Design and Implementation of a Distributed Program for Collaborative Editing". Masters Thesis, MIT, Sept. 1985, Also available as Laboratory for Computer Science Technical Report TR-350.

[Schmidt 82] Eric Emerson Schmidt. "Controlling Large Software Development in a Distributed Environment". Xerox Palo Alto Research Center Report No. CSL-82-7, December 1982.

[Schroeder et al 85] Michael D. Schroeder, David K. Gifford, and Rodger M. Needham. "A Caching File System for a Programmer's Workstation". Xerox Palo Alto Research Center Report No. CSL-85-7, November 1985.

[Smith et al 82] D. C. Smith, E. Harslem, C. Irby, R. Kimball. "The Star User Interface: An Overview". Proceedings of the National Computer Conference 1982, Houston, pp. 515-528.

[Stefik et al 86] M. Stefik, D. G. Bobrow, S. Lanning, D. Tatar. "WYSIWIS Revised: Early Experiences with Multi-User Interfaces". Proceedings of the Conference on Computer-Supported Collaborative Work, Austin, Texas, December 1986.

[Stefik et al 87] Mark Stefik, Gregg Foster, Daniel G. Bobrow, Kenneth Kahn, Stan Lanning, and Lucy Suchman. "Beyond the Chalkboard: Computer Support for Collaboration and Problem Solving in Meetings". *Communications of the ACM*, January 1987, pp. 32-47.

[Trigg et al 86] Randall Trigg, Lucy Suchman, and Frank Halasz. "Supporting Collaboration in NoteCards". Proceedings of the Conference on Computer-Supported Collaborative Work, Austin, Texas, December 1986.

[Trigg and Weiser 86] Randall Trigg and Mark Weiser. "TEXTNET: A Network-Based Approach to Text Handling". *ACM Transactions on Office Information Systems* 4, 1 (January 1986), pp. 1-23.