

Database Management for Multimedia Distributed Collaborative Writing

A. Wesley Wear, Yu Gong, Kai H. Chang
Department of Computer Science and Engineering
Auburn University, Auburn, AL 36849
{wwear, gong, kchang}@eng.auburn.edu

Abstract – *Traditional computer applications have been designed to be run by one user at a time who does some work in a single medium, such as ASCII text, and very little regard has been given to the fact that people often work together. With the recent development of computer networks and the widespread deployment of networked workstations, automating the group writing process for geographically distributed users has become feasible. In this paper, a software package which supports distributed, real-time, multimedia collaborative work, known as the Distributed Collaborative Writing Aid (DCWA), is described. The DCWA has five major parts but the emphasis of this paper is on the DCWA's distributed database, which is crucial to making the entire system work together both logically and consistently. The database is a specialized, distributed system based on the client-server model implemented in C++ running under 4.3BSD Unix and provides dynamic management capabilities unique for a multimedia collaborative working environment.*

1 Introduction

Computers are now familiar tools at work and at home and are increasingly influencing the way in which people interact due to recent advances in communication networks. Electronic mail, bulletin boards, newsgroups, distributed file systems, group schedulers, video conferencing, file transfer protocol (FTP), and “talk” services are but a few examples of computer-supported interactions. However, using computers to enhance people’s communication capability has made some of the current limitations of the technology very noticeable. In particular, traditional computer applications have been designed to be run by one user at a time who does some work in a single medium, such as ASCII text. Very little regard has been given to the fact that people often work together and even the scientific study of human-computer interaction (HCI) has emphasized the exploration of issues

when only a single user interacts with a computer [1].

Since modern technology is complex, it is unusual for an individual to tackle the design of a major project single-handedly and usually a team is assigned to work on the project. Today it is not uncommon for documents to include contributions from many people though usually one person is responsible for collecting the various parts and merging them together to form the (hopefully coherent) final set of documents [2]. In addition, single-user computer applications create barriers to group collaboration and they are normally used only passively by the group to store, retrieve, and present data. If group members need to work on the same item, for example a document, in most cases they must work in an interleaved fashion to prevent inconsistency. This method, however, means that all the other members are prevented from working on the document until the person is finished, even if their contribution is in a different section of the document. This method is clearly inefficient and as the need for improving productivity continues to grow, new technologies must be developed to provide active support for a team and overcome any geographical distribution of the team members [3].

There is thus a growing interest in a new generation of multi-user computer applications which assist groups of people working together, supporting collaboration even if the team members are geographically distributed. These new tools have been grouped together under the name of *Computer-Supported Cooperative Work* (CSCW) [4, 5, 6, 7]. The collection of hardware and software which supports CSCW is also known as *groupware* [8]. In the past, centralized groupware on both mainframes and PCs was explored as a means for achieving *office automation* [9] but now the focus has been shifted to the development of distributed, networked groupware for CSCW [9]. “Groupware” is also meant to be a more technically oriented label to differentiate it from “group-oriented” products, which are simply add-ons to single-user products so that groups of people may work together using these more traditional applications [1].

Another existing limitation to users has been the reliance on applications which support only one type of information medium. Thus a new field has emerged over the past few years known as *multimedia*. Multimedia applications allow the user to manipulate in-

formation in such diverse media as traditional ASCII text, graphical text, graphics, still images, and audio and video communication. The main reason for the growing interest in multimedia systems has been the rapid development of networking technology and with the completion of a Broadband Integrated Services Digital Network (B-ISDN) based on fiber optics, larger amounts of all kinds of data (such as text, audio and video) may be transmitted through a single channel efficiently and cheaply [10] as compared to the networks of just a few years ago.

This report introduces a tool known as the *Distributed Collaborative Writing Aid* (DCWA) which was developed to provide multimedia CSCW on workstations connected to the Internet and running the UNIX operating system. The DCWA can help users cooperate on any writing task (such as programming, report writing, note taking, etc.) logically, conveniently, and efficiently. The coding of the DCWA has been divided into five major parts and this report's emphasis will be on the DCWA's distributed database, which is crucial to making the entire system work together both logically and consistently. The next section explores the features of some related systems and then compares and contrasts the DCWA with them.

The paper is organized as follows. In the next section, an overview of research on CSCW in general and its database subsystems in particular is provided, then we describe the design of the new database system for the DCWA in Section 3. The implementation issues are briefly discussed in Section 4. In Section 5, we provide examples to demonstrate the capability of the system. Section 6 concludes the paper and discusses future work.

2 Literature Review

Groupware for geographically distributed users is usually based upon an architecture that allows multiple processes to communicate with each other [11]. Therefore, a fundamental requirement for CSCW is the provision for message exchange among the participants and it has been suggested that messages should be organized in a structured way to achieve efficiency [12]. That is, protocols must be designed to discriminate between cooperation information and messages of little relevance, e.g., the cognitive filtering in Information Lens [13] and the semi-automatic agents in Object Lens [14]. A second fundamental requirement is the provision of mechanisms for information sharing and cooperation [15]. It has been argued that information to be used in cooperative work needs to be taken out of the limit of a "personal wall", thus all users should have common access to all information [16].

2.1 The Group Collaboration Process

The group process may be divided into those behaviors (called *task behaviors*) which are aimed at achieving the group's task and those behaviors (called *maintenance behaviors*) which are aimed at maintaining the group as a cohesive unit. These two types of behavior are antagonistic and the group process usually involves both of them as progress is made towards the

goal. Using the current computing technology, these two processes are modeled by data flow and control flow in the CSCW system. Some researchers argue that groupware should also impose social models of interaction on the group. Others argue that social protocols should be determined by the group members independent of the CSCW system, and still others argue that a middle ground must be taken [11].

2.2 Multimedia and Its Solution

A salient characteristic of interactive, multimedia documents is the notion of supporting "nonsequential viewing", as well as "nonsequential authoring." As the consequence, there is not a predefined order in which the different segments of the document should be accessed. One solution of nonsequential document structure is to manage a collection of nodes and links [17]. In essence, the heterogeneous media are accessed via an homogeneous linking mechanism [2].

2.3 Database Systems for CSCW

The main challenges associated with designing a database for CSCW are:

- Implementing the partitioning method which will break the documents into manageable sections.
- Maintaining this partitioned structure as changes are made by each user.
- Updating the master copy as needed.
- Determining if two users are trying to modify the same section.

In traditional database systems, the information retrieval is primarily based on the notion of keys. However, these keys are static and may be extremely standardized or so personal that they are useless to all members of the group except for their creator [2]. Also, concurrency control (needed to mediate access to any sharable objects) is usually achieved through simple locking, though other systems use transaction mechanisms [11]. Floor control or a turn-taking mechanism is another way of providing access control to the shared objects. It has been recommended that systems support many such floor control policies so as to suit the various users' needs [11].

Until recently, most multimedia objects have been stored in files by using "document management systems" to store the documents' index and attribute information, where one of the attributes indicates the physical location of the file. However, these systems usually keep journal files to provide for recovery after system failures. Unfortunately, this presents a disadvantage to using commercial, relational database systems since the database journal files tend to be bottlenecks since every transaction is recorded in the journal [18]. It is also important to realize that "documents rather than simply numerical data will be the overriding user need (and database load) in the 1990s." [19]

2.4 Related CSCW Systems

Interest in the area of CSCW has grown rapidly since the first CSCW workshop in 1984 yet there are still relatively few successful products [2]. An experimental system providing asynchronous collaborative writing for a small group has been reported in [5]. The system uses email as the communication media and the text is represented in the ASCII format. The advantage of this approach is that the impact of the geographical locations and the computer variation among the team members is kept to a minimum. However, since only email is used, referencing comments or making changes to existing files may be difficult.

The NoteCards system of Xerox PARC allows multiple users to open and read the same node but only one of the users may modify the node's content at any given time. It is said that this limitation leads to drafts of the document being passed back and forth between the authors. [20]

CSCW can also be accomplished through multimedia communication channels. The MERMAID [21] and SPIN [22] systems are prototypes that provide real-time conferencing environments for geographically distributed participants by using synchronous textual, audio, and video communications. One of the drawbacks associated with these systems is that the cost is far beyond what most users can afford. A simplified version of a conferencing service is to use extended textual bulletin boards [15]. In addition to providing a conferencing service, the COGNOTER system [23] also collects and organizes ideas from its participants for discussion. It is one of the piloting systems that provides the WYSIWIS (What You See Is What I See) capability.

There are a few hypertext systems that are designed for cooperative work. Quilt [24] is a hypertext system that provides coauthoring services. The users are identified as either coauthors or commenters. It allows all users to comment on a document but only certain privileged users, i.e., the coauthors, can modify the document. The ForComment system [25] provides a group editing environment and allows up to 16 reviewers to comment on a document. However, only the original author can actually modify the document. The Aquanet system developed at Xerox PARC is a hypertext tool which facilitates collaborative knowledge structuring. It provides a "What You See Is What I Did" (WYSIWID) view on the shared structure but does not provide synchronous shared views or communication facilities [20].

The rIBIS [26] and SEPIA [20] are synchronous hypertext systems providing various collaboration modes. In the independent mode, users may work on their own tasks without interfering with each other. In the loosely-coupled mode, users may share certain public information while working on their own tasks and in the tightly-coupled mode, users share the same view and resources, e.g., mouse and file, which are strictly controlled to avoid conflicts. Major improvements of SEPIA over rIBIS include automatic mode switching and the use of composite nodes. SEPIA also provides an audio communication channel between participants.

Finally, some systems supporting CSCW collect information from the participants and then suggest appropriate courses of actions. The SYBIL system [27] collects and coordinates options about a specific design topic from members of a hardware platform project team, and then suggests goals and subgoals to them. The Office Works system [25] arranges meetings for a group of participants by checking each participant's schedule.

3 Design of the Database System

To reduce complexity, we decided to design the database component using the client-server model. All the database processes running on the group members' hosts will be considered "clients", except for one, which will be designated as the "server". The database server is only a "server" in the sense that it serves other databases. Users, and more importantly, the database's sibling processes (CR, UI, and NA) will not be able to distinguish whether their DB process is a client or the server. That is, the server has all the functionality that a client does, as well as additional capabilities. It is thus a centrally controlled database. However, data are distributed as much as possible. As such, actually only a few things need to be requested from this central site.

Most of the sibling processes' requests can be fulfilled locally and the main use of the server is to manage the edit locks of the nodes in the logical structure of the document (explained later in Section 4.1). It was decided that maintaining information consistency by managing the group's edits in this fashion would be easier than distributing this information and then running a "conflict-correction" algorithm. As before, a user will be able to select any leaf node to work on which is not currently locked by another group member.

In general, the database process should provide services for backup and recovery, definition of node attributes, searching the logical view to define a partial view, sharing and conflict resolution, and finally, run-time modification of the logical structure. Some details of these capabilities are discussed below.

3.1 Backup and Recovery

To implement backup and recovery, the DCWA database distinguishes between "saves" and "commits". Saves of a node will be local and may be undone (and even redone) until the user has the contents of the node like he wants it. Commits, on the other hand, are sent to the DB server and permanently replace the old version of the node. Only the content of that node will be replaced — the content of other nodes in the same physical file will not be affected by a commit, though the DB may need to update information about those nodes, such as their start and end points.

However, the ability to store multiple committed versions of a node to create a "history list" has been deferred to some future version of this project since this aspect needs to be discussed in more detail among the DCWA working group as how best to serve a group of users with such a feature. For example, when should the tool allow an old version to be deleted forever?

3.2 Node Attributes

Many other similar software tools allowed the users to assign a label to each node in the logical structure. Information that can be carried by a label is quite limited. The DCWA database allows the group to assign several attributes to each node. For example, in an academic writing application, a node may correspond to a segment of text which mentions a certain chemical compound. The user may want to record this fact in the corresponding logical node as a quick reminder for the future. More importantly, the attributes can be used by the search facility (see below) to gain immediate insight to the *semantical* structure, as opposed to the organizational structure of the logical view.

Therefore, in addition to the actual node content, each node of the logical structure will also contain attributes that describe the nature of the node. Some attributes will have their associated values assigned by the database, such as the node's id number and the time the contents of the node were last modified. Other attributes, such as the node's name (label) and any keywords, will have their associated values assigned by the group members. However, with two exceptions detailed later, there is no requirement for these types of attributes to have a value assigned by the group if they don't want or need it. Finally, the database will give the group the ability to define their own attribute/value pairs. In this case, the group not only assigns values but also specifies the names of the new attributes.

3.3 Search Facility

The documents, and their various sections, which the group is writing can become numerous and, thus, the logical structure would probably be a very complicated tree hierarchy. In order to help a user to locate a desired portion of the document quickly, search capabilities must be provided.

In group writing, especially in the case of "division of labor" writing, each writer only concentrates on his/her own part and repeated mentioning of the same point becomes possible. It has also been observed that collaborators need to know the details of a node written by another in order to use the node appropriately. If the node contains C program code, for example, the tediousness of the code often gives rise to the need for a reader to consult with the writer directly for clarification. In both these cases, an automatic search facility of the nodes' attributes would help a collaborator gain knowledge about what the others have written without having to actually read their work or ask them directly. It would also help a user quickly locate the node(s) he/she wants to edit.

The search facility allows a user to specify queries about the logical view. So far, few people have realized the definite need for a search facility with CSCW writing since in single user writing, the writer tends to know, and almost always remember, every detail of the document. In fact, search facilities are a common feature in most commercial databases and this feature should be available in a CSCW environment too. The use of node attributes and search techniques distinguishes the DCWA from other CSCW systems.

3.4 Definition of Views

This capability has been implemented in conjunction with the search facility discussed above. Nodes which meet the query criteria will form a "hit list" and information about these nodes will be sent to the UI for display, replacing the entire logical view with only those nodes. If a remote user creates a new node or modifies an old one, such that it also matches the criteria, it too will then be sent to the UI for display as soon as the local DB is informed about it.

3.5 Sharing and Conflict Resolution

The ability to maintain the consistency of the file as it is being modified by the users is handled by locking and unlocking the nodes of the logical structure. To maintain consistency, a DB server, as discussed above, will be responsible for controlling the locking mechanism.

There have been a number of real-time conferencing tools that include conflict resolutions, such as MERMAID [21], SPIN [22], and the textual bulletin board in [15]. However, their methods are not quite as effective for resolving conflict arising in document composition.

3.6 Run-Time Modification of the Logical Structure

It is possible that the group creates a primitive logical structure at the very beginning of their project and will need to refine it as the writing process proceeds. In any event, a group member should be provided with the capability to define new nodes or delete unneeded nodes during the writing, without disturbing the other users.

The DB server will also be responsible for this conflict resolution and treats it just like edits. After the server locks a subtree of the logical view for modification, the user may make the necessary changes, save, undo, and redo just as if she were editing the content of a node. When the user commits the modification, the DB server will merge the existing logical view with the modified subtree, write the entire logical view into the file on disk, and then send the new subtree to all the DB clients. The term *subtree* was used since all descendants of the selected node are locked as well. Should the user need to modify the entire logical view, the root node should be selected.

Finally, this procedure may also be used to change the values of those node attributes which are assigned by the group. The user should lock the node for logical view modification, change the attribute's value and then commit.

4 Implementation of the Database System

The database is a specialized, distributed program based on the client-server model implemented in C++ [28] running under 4.3BSD Unix.

A user should begin his/her DCWA editing session as follows:

Step 1: Open a file containing a logical view or create a new logical view.

Step 2: Determine which node to select, and how to use it (read, edit, or modify the subtree of the logical view that has that node as its root).

Step 3: Get the content of the document for the selected node and start to work.

To determine which node to select, the user may rely on past experience (i.e., the user knows which node he/she wants) or may make a queried search of the available nodes to limit his/her view of the tree.

When a user gets a node for read-only, he may be informed (when it is received or perhaps later) that another user is currently editing the node's contents. The user should decide whether or not to watch the changes as they are being made. This capability may be toggled on and off as the user chooses until the remote editor finishes. When a user gets a node for editing, the node becomes locked to other users who want to edit that node, though they may still read it, as described above. The same is true for users that lock subtrees of the logical view for modification.

In the remainder of this section, we briefly explain the implementation of important components in the DCWA database.

4.1 Logical View

Users are able to define a logical structure without any text or graphic object in the nodes. If the group doesn't want to get too complicated the logical view can be as simple as one root node followed by as many child nodes as necessary, for example:¹

```

Paper --
| - Abstract
| - Intro
| - Point 1
| - Point 2
| - Point 3
|   ...
| - Conclusion
| - Notes
| - Bibliography
| - Appendix I
| - Appendix II

```

The terms that the group uses to name nodes when defining the logical structure is unimportant to the database. A node's name is just one of many attributes that it maintains for each node. (See the subsection below on node attributes.) The database identifies and differentiates nodes by a node id which is maintained internally by the database itself and not assigned (perhaps erroneously) by the group.

Another example of a logical view is seen in Figure 1, which represents a possible structure for a C program file. Note that the logical structure is not the file dependency diagram used for writing a `Makefile`. It is merely an organizational structure showing the composition of the program. The file dependency in a `Makefile` deals with *semantic* dependencies, and is the reason for overlaying the semantic network (explained later) on the logical structure.

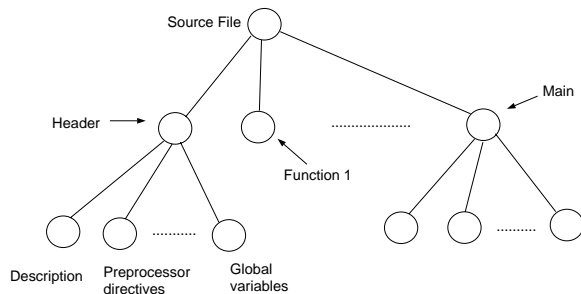


Figure 1: Another Graphical Example of a Logical Structure

4.2 Node Attributes

The basis for modeling semantic in the DCWA database is the capability of assigning attribute to each (organizational) node. Thus meanings are assigned to the node and it may then be related to other nodes through many relational operations such as classification, categorization, inheritance, etc. The following is the list of attributes that are maintained for each node, as well as an indication of the type of information each will hold:

- **Node Id:** An integer.
- **Name:** A character string label for the node.
- **Node Level:** One of Root, Intermediate, or Leaf.
- **Creation time:** A number of type `time_t`.
- **Creator:** A character string holding a user name.
- **Last Change time:** A number of type `time_t`.
- **Last Editor:** A character string holding a user name.
- **Topic:** A character string.
- **Description:** A character string.
- **Keywords:** A character string with each keyword separated by a space.
- **User Defined Attributes:** A character string with each attribute separated by a space.
- **User Defined Attribute Values:** A character string with each value separated by a space.
- **Read Access List:** A character string with each user name separated by a space – see the node access policy below.
- **Edit Access List:** A character string with each user name separated by a space.
- **Logical View Modification Access List:** A character string with each user name separated by a space.
- **Local User Reading:** A boolean.

¹ This is the way logical trees are displayed in the DCWA.

- Current Editor: The editor's database id number.
- Lock Mode: One of Edit or Logical View Modification.
- Media Type: One of Text, Graphics, Image, Audio, or Video if a leaf node, or Logical if not a leaf node.
- Filename : A character string with the machine name and full path name where the file is stored if a leaf node or NULL if a non-leaf node.
- Start point : An integer.
- End point : An integer.
- Number of Children : An integer.

Some of the above attributes function as labels only, but some are of semantic importance. The semantic attributes are indicated by italics.

4.3 Queries

A user may make a queried search of the logical structure to limit his/her personal view of the tree. There are two types of queries, new and revised. A new query searches the group's entire logical view for matching nodes but a revised query only searches the nodes that matched the previous query (either new or revised.) In either case, a "hit list" (perhaps empty) of matching nodes will be returned to the UI and the user may then select a node from among those in the hit list to read or edit or the user makes another query to further limit the view. If a query does not produce any nodes that the user is interested in, the user may start over by making a new query, which again searches the entire logical view.

- New Query: A message is sent with the data of the format:
 - DISPLAY: List of attributes to display, each separated by a comma
 - FOR: Query criterion of the form : Attribute Operator Test-Value

For example,

- DISPLAY: Node Name,Creator,Description, User Attribute-Value Pairs.
- FOR: Node Id \geq 3.

For the attributes specified in the DISPLAY field, the associated values are returned to the User Interface (another component in the DCWA). In addition, the node id is also always sent to the UI as the first attribute value so that the UI may identify which nodes are in the hit list in case further communication with the DB component concerning them is necessary. The other attributes are sent to the UI in the order the user specified in the DISPLAY field of the query, not their order as stored in the node itself. Note that the nodes in the hit list/personal view are no longer arranged

in a tree structure since such an organizational structure does not exist. That is, they are simply a list of nodes which meet the query criteria.

- Revised Query: The user may make a revised query based on what was obtained from the previous query. A message is sent and the data is of the same form as above, however:
 - The DISPLAY field may be different from what it was in the previous query, with either more or less attributes to display. If an attribute was displayed in the previous query it must be listed again if the user still wants that attribute's value displayed.
 - The FOR field will normally be different. If it is different, the resulting hit list is ANDed to the previous hit list to form the new hit list. If it is not different, the hit list will remain the same.

Some "derived attributes" not specifically stored by the database may also be displayed and/or used in the criterion, because the database can calculate the value. Namely, the date, day and time may be separated and the size of a leaf node (calculated from the start and end points) may be requested. The Current Editor and Lock Mode attributes may be used in queries but not the Local Reader attribute, since a user should know whether or not he/she is reading a node.

4.4 Semantic Network

Many existing CSCW tools only allow users to associate a label with each node of the organizational structure, however, the information that can be carried by one label is quite limited. Therefore, the node attributes as described above were designed and implemented. Once these attributes are assigned values, a search of the logical structure may be made to identify those nodes which the user is interested in. However, these attributes may also be used to develop a semantic network on top of the organizational structure of the logical view.

For example, a group of chemists writing a paper may have a node which corresponds to a segment of text that mentions a certain chemical compound C_1 . For future reference, the user may assign the following attribute/value pair to the corresponding node :

CHEMICAL_COMPOUND C_1

In addition to assigning attribute/value pairs to a node, the database also helps the user relate all these attributes in an *inheritance* structure. For example, the user knows that "CHEMICAL_COMPOUND" is a subconcept of "CHEMICAL." Therefore, the user may also want to add the following attribute/value pair to the node :

CHEMICAL CHEMICAL_COMPOUND

Clearly, the attribute/value pair discussed earlier can be used to encode the “is-a” semantic, but it may also be used for encoding the domain relationship among concepts by listing a user-defined attribute as the value of another user-defined attribute.

By associating each node with the user-specified attribute/value pairs, a semantic network is overlaid upon the organizational tree as seen in Figure 2. By using the semantic network and the search facility, users may compose queries to locate all nodes that satisfy a certain condition. The result of the query may also be used for a finer search until the needed information is found. This functionality is similar to what is provided by the Information Lens [13] and the Object Lens [14].

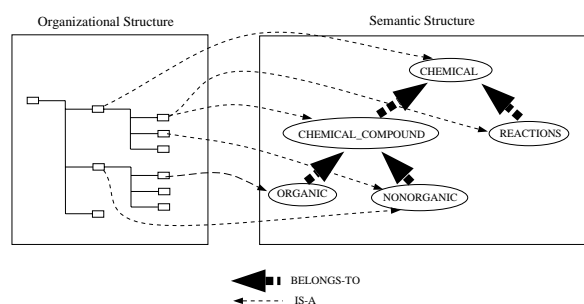


Figure 2: An Example Semantic Network

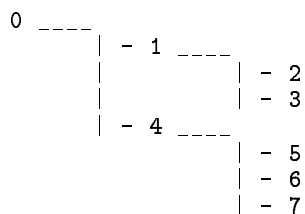
Finally, it is interesting to note that the hypertext solution in Quilt [24], ForComment [25], rIBIS [26], and SEPIA [20] is, in fact, a primitive form of overlaying a semantic net over the organizational structure.

5 Examples

This section gives two detailed examples of some of the database’s functionality, namely node access control in the logical structure and an example semantic network overlaid on the logical structure for identifying relevant nodes.

5.1 Node Access Control

The logical view tree used in the first example is as follows :



Assume that the logical view access log file contains the following information for the node id and the read, edit, and logical view modification access lists. (Note: what shown below are the output from the DCWA database program together with our explanations of each line of the output beginning with the // symbol.)

```
0 // node number
```

```

Cross Seidman // readers list
Gajiwala Ambati Dollar Lee Wear
                // editors list
Gong Chang    // l-structure modifiers list
%{           // node delimiter
1
Cross Seidman Ambati Dollar Lee Wear

Gong Chang Gajiwala
%{
2 // nodes 2 and 3 are closed
  // reflected by empty access lists.
%%
3
%}
4
Cross Seidman Gajiwala Ambati Dollar Lee
  // editors list is empty
Gong Chang Wear
%{
5 // all other nodes are closed
  // reflected by empty access lists.
%%
6
%%
7
%}
%}
  
```

The following requests for read, edit, or logical structure modification would then result in the associated answers :

1. Wear wants to read node 12.
Answer: No such node.
2. Gajiwala wants to read node 0. [Only leaf nodes contain readable and/or editable material.]
Answer: Not a leaf node.
3. Gajiwala wants to read node 2. [A user may read any leaf node for which he/she has permission.]
Answer: Ok.
4. Gajiwala wants to edit node 3. [The same user can have open as many different nodes as needed and in either mode : reading or editing.]
Answer :
Gajiwala has locked node 3 for editing.
5. Wear wants to edit node 5.
Answer:
Wear has locked node 5 for editing.
Gajiwala has locked node 3 for editing.
6. Gong wants to edit node 5.
Answer :
Gong can't lock node 5 because it is already locked for editing.
Wear has locked node 5 for editing.
Gajiwala has locked node 3 for editing.

7. Gong wants to edit node 6.

Answer :

Wear has locked node 5 for editing.
 Gajiwala has locked node 3 for editing.
 Gong has locked node 6 for editing.

8. Chang wants to read node 7.

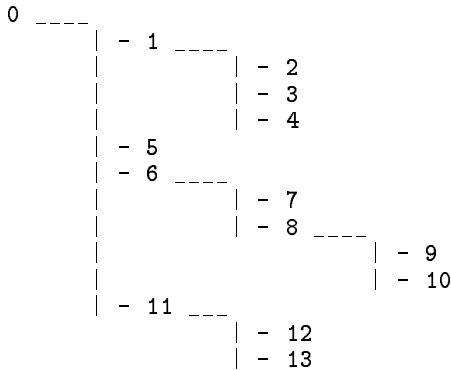
Answer :

Ok.
 Wear has locked node 5 for editing.
 Gajiwala has locked node 3 for editing.
 Gong has locked node 6 for editing.

Finally, though not shown in this example, the logical view modifiers may first acquire locks to any node and then modify the three lists associated with the node.

5.2 Semantic Network

This example shows how a semantic network about automobile safety may be overlaid upon a logical structure concerning the components of the car. The logical view tree used in this example is as follows :



Assume that the logical structure log file contains the following information for the node id and the user-defined attribute/value pairs :

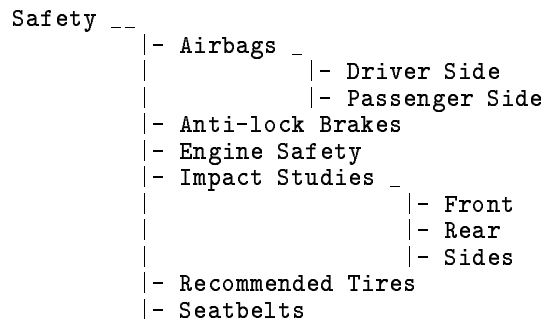
```

0
Make           Model           Year
Generic_Motors XV-1           1995
%{
1
Automobile     Safety
Frame          Impact_Studies
%{
2
Impact_Studies
Sides
%%
3
Impact_Studies
Rear
%%
4
Impact_Studies
Front
  
```

```

%}
5
Automobile     Safety
Engine         Engine_TBA
%%
6
Automobile
Interior
%{
7
Interior       Safety
Seats         Seatbelts
%%
8
Interior       Safety
Dash          Airbags
%{
9
Airbags
Driver_Side
%%
10
Airbags
Passenger_Side
%}
%}
11
Automobile
Wheels
%{
12
Wheels         Safety
Tires         Recommended_Tires
%%
13
Wheels         Safety
Brakes        Anti-lock_Brakes
%}
%}
  
```

The above semantic information, although recorded with the logical (i.e., organizational) structure, reflects the following semantic tree.



The following are sample queries showing a possible search of the safety information :

1. The user gets a list of leaf nodes by making a "New Query" which starts from scratch.

```
DISPLAY : User Attribute-Value Pairs
FOR : Node Level == LEAF
```

Answer :

```
2
Impact_Studies
Sides
%%
3
...
```

2. To determine which leaf nodes concern "Airbags", the user would then make the following "Revised Query" which search in the set of nodes returned by the immediate last query.

```
DISPLAY : User Attribute-Value Pairs
FOR : User Attributes == Airbags
```

Answer :

```
9
Airbags
Driver_Side
%%
10
Airbags
Passenger_Side
```

6 Conclusion and Future Work

The CSCW software tool known as the DCWA, will be useful to any group of people who have a need to coauthor documents, especially if they are separated geographically. The DCWA can help users cooperate on any writing task logically, conveniently, and efficiently.

The DCWA provides both textual and graphic editing, viewing, and coordination services to the group members. The design of the tool follows the principle of "division of labor" through the logical structure's locking mechanism. However, the "shared mind" approach can also be realized by viewing any other member's work space, as well as querying the node attributes and semantic network.

The DCWA database organizes textual and graphical information according to their structural relationships, as well as their semantical relationships. Users may navigate through the files logically by using the search facility, which queries the node attributes and limits the user's view to only those nodes which meet the criteria of the query.

Some possible extension to be considered in the next prototype are identified below.

- In addition to the current text and graphics, incorporate other media, such as audio and video communication, into the logical view.
- Make the database robust with respect to host (local and remote) failures.

- Include a new access list attribute (called `comment_access`) which holds the list of people who may comment upon the node. That is, they are more than passive readers of the node but they may not actually edit the node either, just post comments. Of course, a method for storing (and eventually deleting) these comments must be provided.
- Get rid of "logical" nodes and let each non-leaf node access the entire contents of its descendants. This will, of course, require the UI to be able to display the various media together.

References

- [1] Saul Greenberg. *Computer Supported Cooperative Work and Groupware*, chapter 1, pages 1–12. Academic Press, Ltd. London, 1991.
- [2] Kaj Gronbaek, Morten Kyng, and Preben Mogenssen. "CSCW challenges in large-scale technical projects - a case study". In *Proc. of the 1992 CSCW Conference*, pages 338–345, Nov. 1992.
- [3] J. Cuenca and A. Garcia-Serrano. "Intelligent Computer Support". *Cooperation among Organizations — The Potential of Computer Supported Cooperative Work*, pages 72–102, Power, R.J.D. (Ed.), Germany: Springer-Verlag, 1993.
- [4] R. J. D. (Ed.) Power. *Cooperation among Organizations — The Potential of Computer Supported Cooperative Work*, chapter 2,6, pages 13–25,103–118. Germany: Springer-Verlag, 1993.
- [5] D. Diaper. "Small-Scale Collaborative Writing Using Electronic Mail". *CSCW in Practice: An Introduction and Case Studies*, pages 72–102, Diaper and C. Sanger (Eds.), Germany: Springer-Verlag, 1993.
- [6] B. Hewitt and G. N. Gilbert. "Groupware Interface". *CSCW in Practice: An Introduction and Case Studies*, pages 31–38, Diaper and C. Sanger (Eds.), Germany: Springer-Verlag, 1993.
- [7] M. Sharples. "Adding a Little Structure to Collaborative Writing". *CSCW in Practice: An Introduction and Case Studies*, pages 51–68, Diaper and C. Sanger (Eds.), Germany: Springer-Verlag, 1993.
- [8] J. N. Orr. "Graphics and Groupware: Increasing Intimacy through Broadening Bandwidth". In *Groupware'92*, pages 73–76, Morgan Kaufmann Publishers, 1992.
- [9] J. Grudin. "Computer-Supported Cooperative Work: History and Focus". *IEEE Computer*, pages 19–26, May 1994.
- [10] Dimitri Bertsekas and Robert Gallager. *Data Networks*, chapter 1, pages 1–35. Prentice Hall, Inc., Englewood Cliffs, NJ, 1992.

- [11] Mark Roseman and Saul Greenberg. "GROUP-KIT : A Groupware Toolkit for Building Real-Time Conferencing Applications". In *Proc. of the 1992 CSCW Conference*, pages 43–50, Nov. 1992.
- [12] P. V. Rangan. "Managing Multimedia Collaboration". In *Groupware'92*, pages 418–420, Morgan Kaufmann Publishers, 1992.
- [13] T. W. Malone, K. R. Grant, and F. A. Turbak. "The Information Lens: an Intelligent System for Information Sharing in Organization". In *Proc. of the ACM Human Factors in Computing Systems Conference*, pages 1–8, Boston, MA, 1986.
- [14] T. W. Malone and K. Lai. "Object Lens: A Spreadsheet for Cooperative Work". In *Proc. of the Conference on Computer Supported Cooperative Work*, Portland, OR, 1988.
- [15] T. Rodden. "Technological Support for Cooperation". *CSCW in Practice: An Introduction and Case Studies*, pages 1–22, 1993.
- [16] J. Brooke. "User Interface for CSCW Systems". *CSCW in Practice: An Introduction and Case Studies*, pages 72–102, Diaper and C. Sanger (Eds.), Germany: Springer-Verlag, 1993.
- [17] Shoshana Loeb. "Delivering Interactive Multimedia Documents over Networks". *IEEE Communications Magazine*, pages 52–59, May 1992.
- [18] Mark F. Riley, James J. Feenan, John L. Janosik, and T.K. Rengarajan. "The Design of Multimedia Object Support in DEC Rdb". *Digital Technical Journal*, 5(2):50–64, Spring 1993.
- [19] Dimitris N. Chorafas and Heinrich Steinmann. *Solutions for Networked Databases : How to Move from Heterogeneous Structures to Federated Concepts*, chapter 5, pages 73–92. Academic Press, Inc. San Diego, 1993.
- [20] J. M. Haake and B. Wilson. "Supporting Collaborative Writing of Hyperdocuments in SEPIA". In *Proc. of the 1992 CSCW Conference*, pages 138–146, Nov. 1992.
- [21] K. Watabe, S. Sakada, K. Maeno, H. Fukuoka, and T. Ohomri. "Distributed Multi-party Desktop Conferencing: MERMAID". In *Proc. of the Conference on the Computer Support Cooperative Work*, Los Angeles, CA, 1990.
- [22] L. G. Palmer and R. S. Palmer. "DECSPIN: A Networked Desktop Videoconferencing Application". *Digital Technical Journal*, Vol. 5(No. 2):pp.65–76., Spring 1993.
- [23] M. Stefik, G. Foster, D. Babrow, K. Kahn, S. Lanning, and L. Suchman. "Beyond the Chalkboard: Computer Support for Collaboration and Problem Solving In Meeting". *Communications of the ACM*, Vol.30(No.1):pp.32–47, Jan. 1987.
- [24] R. S. Fish, R. E. Kraut, M.D. Leland, and M. Cohen. "Quilt: A Collaborative Tool for Cooperative Writing". In *Proc. of the Conference on Office Information Systems*, pages 30–37, Palo Alto, CA, 1988.
- [25] S. Opper. "A Groupware Tool Box". *BYTE*, pages 275–282, Dec. 1988.
- [26] G. L. Rein and C. A. Ellis. "rIBIS: A Real-Time Group Hypertext System". *Int. Journal of Man-Machine Studies*, pages 339–367, Aug. 1993.
- [27] J. Lee. "SIBYL: A Tool for Managing Design Rationale". In *Proc. of the Conference on the Computer Support Cooperative Work*, Los Angeles, CA, 1990.
- [28] Bjarne Stroustrup. *The C++ Programming Language*, chapter 1-7,R, pages 13–254,477–632. Addison-Wesley Publishing Co., Reading, MA, 1991.