UDC 001.81: 681.3

Information Sharing Using WWW in WHIST

●Hiroshi Tsuda ●Kanji Uchino ●Kunio Matsui

(Manuscript received June 2,1997)

Document information sharing has become important to cope with the increasing amount of digital documents in the information network society. We developed WHIST (Web-Human Information Seeking Tools) as an environment to share, retrieve, and organize document information in an intranet.
There are two processes in document information sharing in WHIST: document sharing and information retrieval. WHIST integrates and shares documents such as e-mail, news, and HTMLs bottom-up using the WWW. WHIST also provides various information retrieval methods to shared documents including full-text document searching, user adaptation, and multiple views based on the information extraction using natural language processing.

1. Introduction

Thanks to the rapid growth of the Internet, we can now easily access huge numbers of documents using e-mail, network news groups, and the WWW. However, several new problems have arisen concerning information retrieval, organizing, and sharing.

For example, the 6th Internet User Survey of the GVU (Graphics, Visualization and Usability Center) in October 1996 found that the top four problems that users have with the WWW are:

- Access speed (76.55%),
- finding known information (34.09%),
- organizing collected information (31.03%), and
- finding pages already visited (13.41%).

(http://www.cc.gatech.edu/gvu/user_surveys/)

The first problem is a quantitative problem that will be solved in the near future with the support of the network infrastructure. The remaining problems are all concerned with information retrieval and organization and will become increasingly important.

Consider document information organization. Internet documents include a large amount of junk knowledge. When reading network news and WWW pages, we select useful information; the selected information can be regarded as secondary information of the Internet documents. The process of creating the secondary information is simply information organization. There are, however, few systems that can organize documents in terms of their contents.

One of the central functions of an office intranet is to share information at a low cost to make business efficient. One way to implement an intranet is to attach WWW interfaces to LAN-based groupware systems. Traditional groupware such as Lotus Notes, Novel Groupwise, and Fujitsu TeamWARE has become intranet software. Another approach is to combine existing Internet media such as the WWW, e-mail, and network news. Producers of traditional groupware, however, concentrate on providing methods for sharing documents, as opposed to methods for searching and organizing shared documents. For example, users have to attach additional information such as keywords to reuse documents efficiently.

The main interest of this paper is "document information sharing" or simply "information sharing". We use the term "document information sharing" to mean the following processes.

1) Document Sharing: Document sharing is a process that makes documents accessible by group members.
2) Information Retrieval: Information retrieval consists of document retrieval and information organization to reuse shared documents.

For information sharing systems in an intranet, we consider the following to be especially important:

- To reduce the user's tasks for document sharing,
- to integrate various document media such as e-mail, news, and the WWW in document sharing,
- to provide various methods of accessing the contents of shared documents, and
- to work on multiple platforms such as UNIX and Windows.

We developed WHIST (Web-Human Information Seeking Tools) to meet the above requirements for an intranet. WHIST offers a framework for document sharing using the WWW in which digital documents such as e-mail, news, and HTMLs are integrated so they can be shared among group members. It also provides various information retrieval methods for shared documents, including full-text searching, user adaptation, and multiple views with document organization. WHIST uses natural language processing to extract secondary information automatically from the contents of shared documents. WHIST consists of a set of tools to retrieve, maintain, and share document information in UNIX Workstations and Windows PCs.

Chapter 2 explains our motivations for developing WHIST and describes a model of information sharing. Chapter 3 outlines the tools of WHIST, Chapter 4 explains the document sharing features, and Chapter 5 explains the information retrieval features. Finally, Chapter 6 gives our concluding remarks.

2. Motivations

2.1 Motivations for developing WHIST

There were two motivations for developing WHIST.

The first motivation was to create a WWW-based environment in which to control document life cycles. Digital documents such as e-mail, news, WWW pages, and daily reports have their own life cycles. Documents are created in editors, mail readers, and browsers. Then, they are stored into a DB or a file system in a certain format. Stored documents are retrieved, organized, and maintained for recycling. Some of the documents can be accessed by anyone, while access to other documents is restricted to certain groups or individuals. When documents are no longer needed, they are discarded. Until now, the life cycle has often been controlled by individuals working on a desk (paper documents) [1] or on personal PCs with news/mail readers and bookmarks of WWW browsers. To advance information sharing, however, the cycle should be controlled on a network.

We consider the WWW to currently be the most suitable framework with which to control the cycle because it provides the following important features for document sharing:

- Unique pointers to documents: URL,
- a common document format: HTML, and
- a common document transformation protocol: HTTP.

Although the WWW is growing rapidly, it is still mainly a browsing tool and is not often used as a publishing tool. Supporting document life cycles also leads to more active use of the WWW.

The second motivation was to use natural language processing in information retrieval from free-format documents. As we explained in Chapter 1, it is becoming more difficult to acquire wanted documents from the WWW, previously visited WWW pages, e-mail folders, and so on. Simple full-text searching is inadequate if the search results are still too large. Document organization and visualization are candidate methods for resolving this situation. However, a difficult prob-

lem with these methods is that Internet documents are in a free format. Even in the WWW, HTML tags are freely used for visual effect and often have nothing to do with the documents' semantic structure. To organize these free-format documents, current groupware software lets its users add optional information. We use natural language processing techniques to minimize the load of the organizing process.

2.2 Document Information Sharing Models

To clarify the goal of WHIST, this section explains our models of document information sharing. We regard document information sharing as consisting of the following two processes:

1) Document Sharing: Document sharing makes documents accessible by group members.
2) Information Retrieval: Information retrieval consists of document retrieval and information organization for reuse of shared documents.

Before the growth of the Internet, the document sharing process was more important. Most of the information to be shared was local information processed by individual members. The size of shared information is also relatively small. As illustrated in **Fig. 1**, every LAN-based groupware system provides a common DB and a common environment to register and browse shared information. We call this type of information sharing "local information sharing."

Recently, however, the growth of the Internet has made the latter process more important. Internet documents such as mailing list articles, news, and WWW documents are already being accessed by people all over the world. But, as was mentioned in Chapter 1, this does not mean that those documents are instantly reusable.

For example, suppose a member of a group finds an important page by using search engines, directory services, and clicking many anchors. If that member does not organize the URLs in the member's home page, other members will have difficulty in finding the information among the huge amount of junk information. Even the member may not be able to reaccess the page at a later date without registering it to the member's own bookmark. What is required here is an environment to facilitate access to shared documents based on document retrieval and organization. We call this second type of information sharing "partial global-information sharing"; this type is illustrated in **Fig. 2**.

As discussed in Section 2.1, WHIST provides a WWW-based document information sharing environment to support not only document sharing but also information retrieval methods.

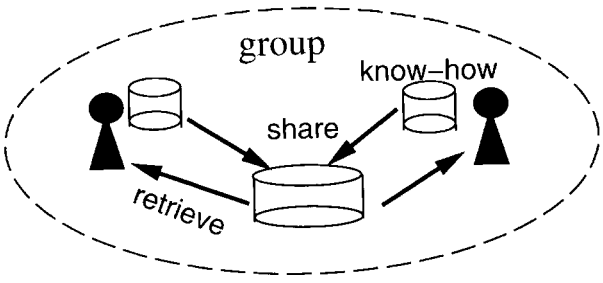

Fig.1– Local information sharing.

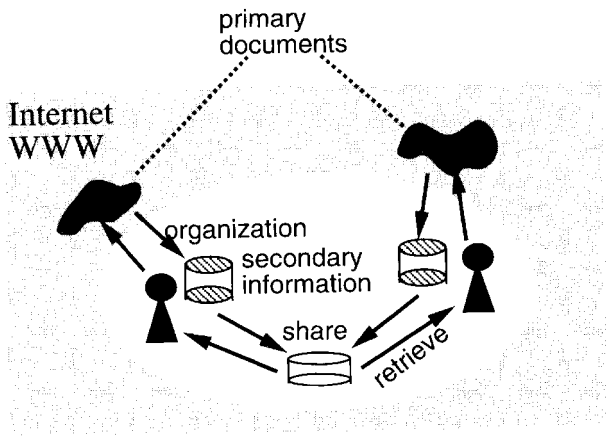

Fig.2– Partial global-information sharing.

3. WHIST

3.1 Tools in WHIST

WHIST consists of the following set of tools to support document information sharing in an intranet:

- A-Kore[note)] : document sharing environment,
- HyperScheduler: group scheduler,
- Web-Search: full-text searcher of local WWW files,
- WEDGE: group editor of WWW documents, and
- Automatic What's New: maintainer of group home pages.

The following subsections outline the first three tools, which are closely related with document information sharing.

3.2 Information Sharing Tools

A-Kore and HyperScheduler are designed for document information sharing. They are explained in detail in Chapters 4 and 5.

3.2.1 Overview of A-Kore

A-Kore provides an environment in which to share and reuse general documents such as e-mail, news, and WWW HTMLs. **Figure 3** shows an overview of A-Kore. A-Kore eliminates the troublesome steps needed to register documents to the WWW in the document sharing process. Users only need to specify a document that they are reading using WWW browsers, news readers, e-mail readers, or editors. Users can specify additional information such as a title, genre, and the groups that will be allowed to access the document. Then, the document is converted into an HTML file and sent to the WHIST server through the network. In the WHIST server, the document is stored in the user's personal document base together with the secondary information that is automatically extracted from the contents.

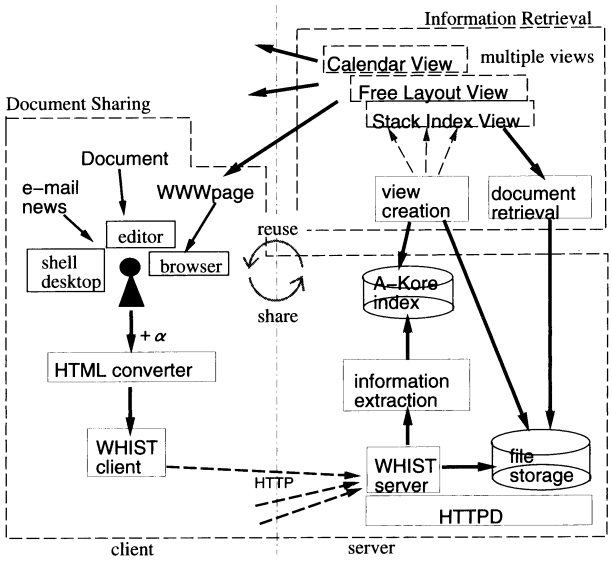

Fig.3– Overview of A-Kore.

note) "Ah Kore" is a Japanese expression meaning "Oh!, this is it".

A-Kore also offers multiple views of shared documents; for example, it offers a stack-index view, calendar view, trend view, and free layout view. Users can access documents in organized formats using these views.

3.2.2 Overview of HyperScheduler

HyperScheduler is a WWW-based group scheduler. Its document sharing mechanism is the same as that of A-Kore, except that the document contents are schedule events such as the one shown in **Fig. 4**.

HyperScheduler offers various views of the shared schedule. Several kinds of anchors to shared information are also embedded in the views. Compared with traditional schedule software, HyperScheduler has the following advantages:

- Schedules are written in free-format documents (input does not require special software),
- only WWW browsers are required to view schedules,
- it supports a group hierarchy, and

```
9/2       9:30      MI) meeting
96/9/3    9:30      >Kyoto
          13:30–15:30      IR) IR Group meeting
9/19/96   AM        UI meeting (Main Bldg.)
96/10/3   18:00     (party)    # private schedule
1/22      10:00     first meeting
          13:30     discussion with Dr.K
```

Fig.4– Schedule document.

- it is an entrance to various shared information items.

3.3 Document Retrieval Tool

3.3.1 Web-Search

Web-Search is a document retrieval tool of WHIST that attaches a full-text search feature to WWW servers. Web-Search provides the following three kinds of interfaces:

- An interface with a search engine: Web-Search uses a fast file search engine called Aleph[2] that has a similar function to that of UNIX command grep. Web-Search contains CGI scripts to invoke Aleph from the WWW.
- An interface with WWW authors: WWW authors can create search pages by specifying several parameters in an HTML file. There is no need to write additional CGI scripts. Two kinds of search results are available: URLs and lines. The URL results list the URLs that contain the specified keywords in their titles. The line results return a part of the lines that include the specified keywords.
- An interface with WWW readers: From the search pages created automatically by Web-Search, readers can specify search keywords with AND, OR, and NOT operators and can specify instructions for case sensitivity and treatment of white characters.

3.3.2 Example of using Web-Search

Let us explain how Web-Search works using a simple example. A WWW author who would like to establish a book finding service only needs to specify several parameters in an HTML file as **Fig. 5**. Here, the URL of book information is /bu/part/*.html., and posnum and negnum are the maximum numbers of positive and negative keywords, respectively. The FORM tag creates a Search Books button.

```
<FORM METHOD=POST ACTION="/cgi–bin/web–search.cgi">
  <INPUT TYPE=HIDDEN NAME=userid VALUE="nlp">
  <INPUT TYPE=HIDDEN NAME=subdir VALUE=/bu/part>
  <INPUT TYPE=HIDDEN NAME=fileselect VALUE="*.html">
  <INPUT TYPE=HIDDEN NAME=title VALUE="NL Group Books">
  <INPUT TYPE=HIDDEN NAME=posnum VALUE=3>
  <INPUT TYPE=HIDDEN NAME=negnum VALUE=2>
  <INPUT TYPE="submit" name="search" VALUE="Search Books" >
<FORM>
```

Fig.5– Parameter specification of Web-Search.

When a WWW reader clicks the Search Books button, the page shown in **Fig. 6** is displayed to enable input of the keywords. After specifying the keywords (semantics) and search conditions (white space characters are not ignored), clicking the Search Start button returns the search result, an example of which is shown in **Fig. 7**. If the

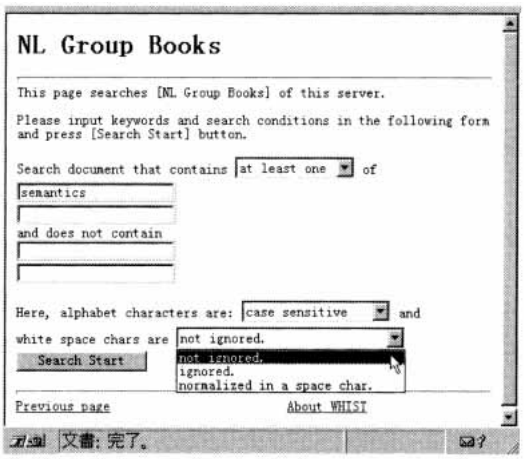

Fig.6– Web-Search page for search condition input.

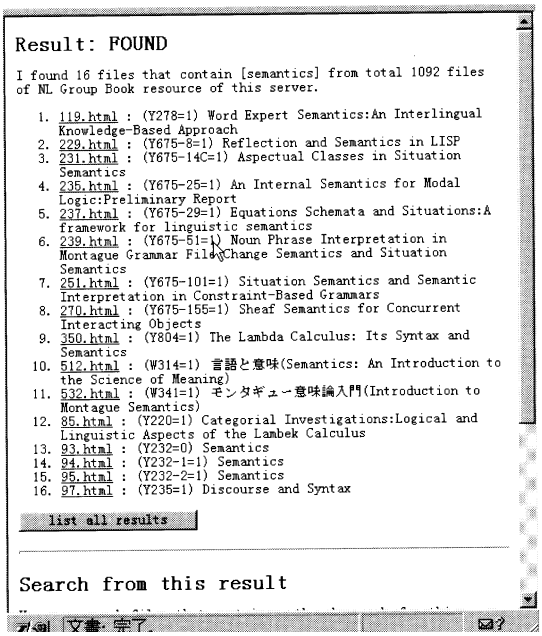

Fig.7– Web-Search search result.

result contains too many files, the user can add other words to search within the result.

3.3.3 Comparison with Internet search engines

Most of the robot-based Internet search engines create indices of large amounts of WWW pages and offer services to retrieve the indices. On the other hand, Web-Search gives a search ability to each WWW server and enables search activities using local and distributed multiple servers. Web-Search resembles distributed DBs and static software agent systems such as KQML (Knowledge Query and Manipulating Language)[3] and cooperative problem solving.[4)-6)] Distributed resource searching using Web-Search is explained in Section 5.1.

4. Document Sharing in WHIST

This chapter explains the document sharing features of A-Kore and HyperScheduler. As was explained in Section 2.2, document sharing is the first part of information sharing.

4.1 Capturing Step of A-Kore

There are two processes in the capturing step of A-Kore. One is direct manipulation and the other is a process to extract secondary information automatically.

4.1.1 Direct capturing

In A-Kore, users can directly capture documents from the UNIX command line, Mule or Emacs (which are popular editors in UNIX), WWW browsers, or Windows.

In the Mule interface, the `akore-buffer` function captures the document in the current buffer. There is also a capturing interface in a mail reader (mh) and news reader (gnus) on Mule. By using the capturing interface, users can register e-mail and news that they are reading simply by pressing a key. In the Netscape interface, users simply click the capturing button in a frame to capture the HTML document in the other frame. This is implemented in JavaScript and FRAME tags of HTML. There are two kinds of Windows interfaces. One is an icon on the desktop to which users drag and drop text files. The other is an editor like the Windows notepad with a capturing button.

With any of the above interfaces, capturers can set additional information to documents such as a title, genre, and range of sharing. At first, titles are automatically extracted from the subjects of e-mail/news, the TITLE tag of HTMLs, or the first non-null line. Users can change the initial title. The default document genres are "None", "Computer", "News", "Hobby", and "etc." These genres are customizable. The range specifies how far the document can be shared; the default ranges are "Personal", "Group", "Laboratory", "Company", and "General". Personal documents can be read only by their owner; that is, the view mechanism of A-Kore (see Section 5.2) also works as a firewall of personal DBs.

Selected documents are copied and converted into HTML files (see Subsection 4.1.2). A-Kore copies original documents. Another approach to creating personal DBs is, for example, to collect references of documents such as bookmarks of WWW browsers. The later approach is appropriate for frequently updated documents. The copying approach, however, has advantages in document content processing, which is one of the aims of A-Kore.

As shown in Fig. 3, documents are sent to the WHIST server. In the WHIST server, secondary information such as keywords and related dates (see Section 4.1.3) are automatically extracted from the document contents.

4.1.2 Common Format of Shared Documents

In A-Kore, shared documents are stored as HTML files. In the header part of the HTML, various meta-level and secondary information is embedded manually or automatically. **Table 1** is a list of parameters of the information. Genre and Range are specified manually in the A-Kore client by captures. Only the last two parameters, RelDate and KeyWords, are set in the WHIST serv-

Table 1. Parameters of meta information of A-Kore documents.

Parameter	Explanation
Current Time	Capture time
Src Time	Creation time
Doc Type	Document type
Media	Document media
From	Creator
User ID	UID of capturer
User Name	User name of capturer
Title	Title
Genre	Genre
Range	Range of information
File Header	File name in WHIST server
System Name	Machine name of WHIST client
RelDate	Extracted related dates
Key Words	Extracted keywords

Table 2. Examples of dates in English.

23 March 1996	(basic)
24 Mar. 1996	(abbreviated month)
23 March 96	(last two digits of year)
15th May 1996	(day expression)
17th of May in 1996	("of", "in" are inserted)
Thu. 11 May 1995	(day of week)
23 March	(year is dropped)
September 2, 1996	(basic)
Sep. 3, 1996	(abbreviated month)
May 16th 1996	(day expression)
September 2 in 1996	("in" is inserted)
September 2	(year is dropped)
22-25 September 1997	(basic)
January 11-12, 1997	(basic)
Jan. 28—Feb. 10, 1997	(start and end in different months)
Dec. 27, 1996-Jan. 14, 1997	(start and end in different years)

er automatically by the information extraction modules. The secondary information extracted in the WHIST server is used to build the A-Kore index from which multiple views are later created.

4.1.3 Date information extraction

The information extraction module of A-Kore extracts date information from the contents using natural language processing.

Many documents around us, especially in the office, are related to dates such as deadlines, days of meetings, and so on. We studied more than 300 e-mail, news, and WWW documents written in English and Japanese and found more than 100 different ways of representing the date. **Table 2** shows some examples of date patterns used in English.

The information extraction module extracts all the date expressions contained in the shared documents. The module extends a regular expression parser to deal with number expressions in English and Japanese. Extracted dates are embedded as values of the RelDate parameter shown in Table 1.

4.2 Schedule Document Sharing in HyperScheduler

The document sharing mechanism of HyperScheduler is the same as the one in A-Kore except that shared documents are a set of schedule events like the one shown in Fig. 4. The document format is almost free, but an event is in a line containing the date, time, and description of an event.

As a group scheduler, HyperScheduler supports hierarchies among groups and members. Schedules of a group are inherited to the group's subgroups and members. Multiple inheritance is also allowed. In the schedule document shown in Fig. 4, the events that begin with a Group ID (first and third line) are treated as group events.

5. Information Retrieval in WHIST

This chapter describes the information retrieval methods for shared documents in A-Kore and HyperScheduler that are used in the information retrieval process described in Section 2.2. The methods are "document retrieval" and "creation of multiple views."

5.1 Document Retrieval

5.1.1 Search documents captured with A-Kore

The first and general step to access document information is full-text searching. As explained in Section 3.3, Web-Search gives a general full-text search procedure to WWW servers.

In A-Kore, shared documents are a set of HTML files in a WWW server. To enable full-text searching, Web-Search is embedded in A-Kore

views such as the stack index view shown in **Fig. 8**.

5.1.2 Distributed searching using multiple Web-Search engines

It is common for even single group members to use multiple WWW servers. This requires a method for searching distributed document resources. Web-Search contains a meta-search module called Intranet Search Navigator (ISN) to combine multiple distributed Web-Search engines. The mechanism is just like the one used in MetaSearch (http://www.metasearch.com/).

Recently, various search methods for multiple resources have become available in intranets, for example, directory services, search engines, DB interface, and WWW local search engines such as Web-Search. However, it is becoming difficult to find an appropriate resource for the required genre. ISN provides a meta search interface to deal with this problem. ISN can call not only distributed Web-Search engines but also general search engines with FORM tags.

Figure 9 shows an example screen of ISN. The user inputs keywords and the required genre of search resource. Genres are given by the WWW authors in advance. ISN searches multiple search engines registered under the selected genre in parallel or sequentially to gather the result. ISN invokes multiple search engines through HTTP using the GET and POST protocols. The main features of ISN are as follows:

- The user need not be aware of the search resource, and
- the user need not know the format (e.g., operators of AND and OR keywords) of each search page.

For WWW authors, ISN offers a robot program to access the URLs and analyze FORM tags. With specified pairs of URLs and genres, the robot gathers the information required for the meta search.

5.2 Multiple Views

Multiple views are central features of information accessing in WHIST.

In DB research, a view or user view is a presentation of data to the user in a format that differs from that of the original data. Tanaka[7] indicates the need to introduce a view function to the WWW to make the WWW more rich and personalized. In the Dexter model,[8),9)] which is a basic model of hypertexts, hypertexts logically consist of the Run-time layer (user's action), Storage layer (definition of nodes and links), and Within-Com-

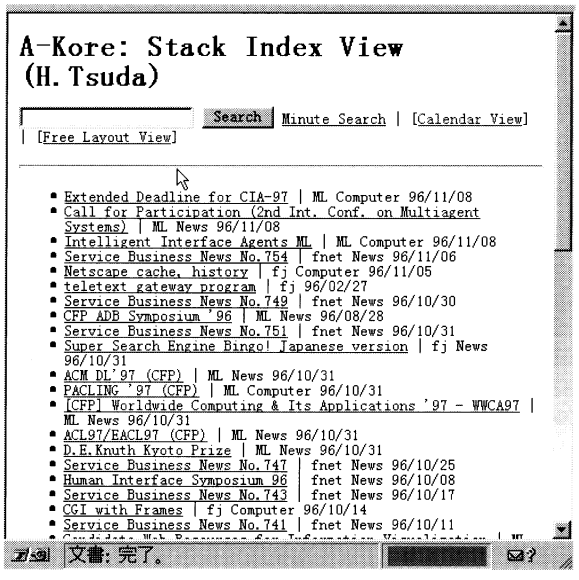

Fig.8– Stack index view.

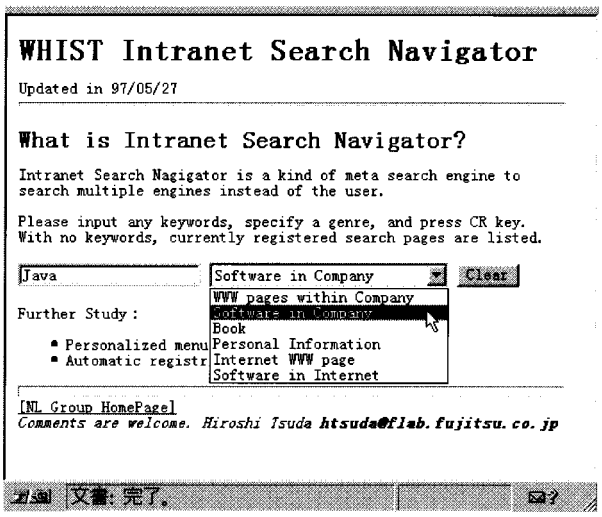

Fig.9– Intranet Search Navigator.

ponent layer (contents). When seen from the Dexter model, HTML merges information of all three layers. Current HTML has only one view, which is the one given by the author, and hence it is not, by itself, a suitable framework with which to implement flexible views.

The basic idea of accessing documents in WHIST is also to give multiple views to shared documents. As shown in Fig. 3, the view creation module of A-Kore creates multiple views from A-Kore index. The module is implemented in CGI scripts to distinguish between contents and link information.

The rest of this section explains four A-Kore views that are linked to each other: the stack index view, calendar view in relation with HyperScheduler, free layout view, and trend view.

5.2.1 Stack index view

The stack index view is a list of document titles sorted according to when the documents were last accessed by their owner. The view is an implementation of Noguchi's "Tyou Seiri Hou" (Super organizing method).[1] The super organizing method is as follows:

1) Put each document in a fixed-size envelope and put the envelopes together in a row like books on a shelf,
2) put new documents and documents that have just been read at the left end of the row, and
3) look for documents starting from the left end.

This method makes it easy to find new and frequently accessed documents because they eventually accumulate at the left end of the row.

Figure 8 shows an example of the stack index view of A-Kore. Titles of previously captured documents are aligned from top to bottom. By clicking a title, users can read the contents of the document, and the title moves to the top of the stack index view the next time the user accesses the view. New and frequently accessed documents are placed at the top of the stack according to the access actions. In other words, user adaptation is automatically realized in the view. Only the owner can use the updating process; other people can access the view but they cannot change the ordering of the stack.

5.2.2 Calendar view and HyperScheduler

The calendar view is a link between A-Kore and HyperScheduler. This subsection explains HyperScheduler and the calendar view.

There are multiple views to schedules in terms of the time periods (week, month, several weeks either side of the current week) and the group hierarchy. Several HTML links are embedded in the views. In Hyper-Scheduler, by clicking HTML anchors, users can not only change the views of HyperScheduler but also visit various WWW pages such as other members' home pages and A-Kore shared documents. Thus, HyperScheduler works as an entrance to group-shared information.

Figure 10 shows an example of the group view. Group views show native events of the group, inherited events of the super groups, and the initials of the members/subgroups who have a schedule. Native events and inherited events are shown in different colors; for an example, see the two events for June 3 in Fig. 10.

The calendar view of A-Kore connects A-Kore documents with the related dates extracted in the capturing step. Anchors of A-Kore documents are embedded in HyperScheduler. **Figure 11** shows an example of a calendar view. This view is a public view.

There are many kinds of documents related to dates, for example, "Call for Papers" (deadline, camera ready deadline, and conference dates) and "Call for Participants" (registration, and conference dates). The calendar view makes the user aware of important dates in the user's schedule. We call the mechanism "temporal push technology." It is also possible to view all the documents captured by multiple members in a calendar view.

5.2.3 Trend view

The trend view shows the temporal frequency of documents.

Information organization is very effective

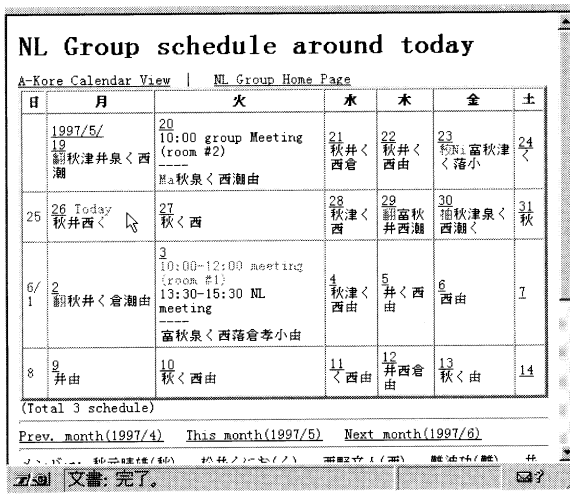

Fig.10– HyperScheduler: group view.

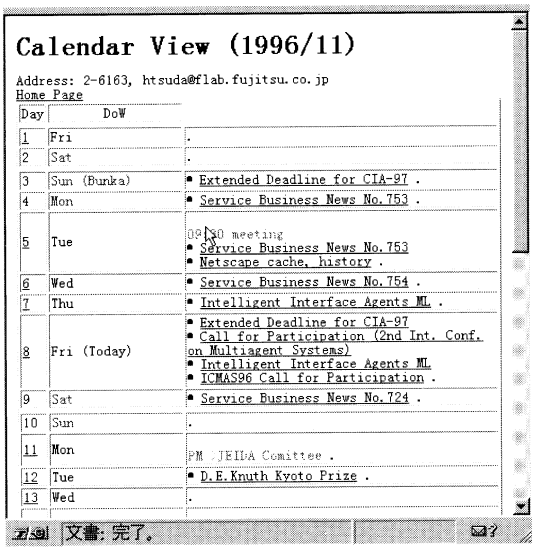

Fig.11– Calendar view.

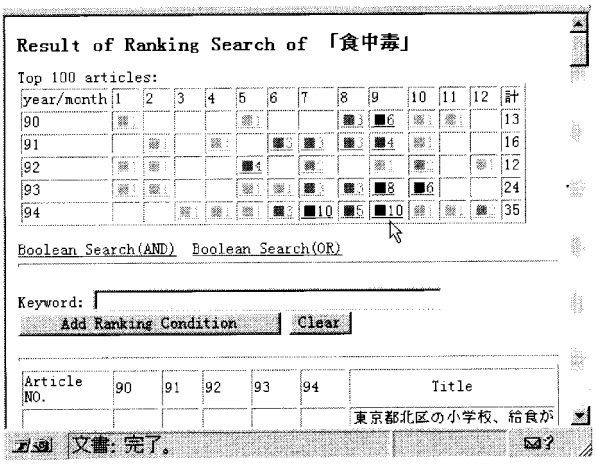

Fig.12– Trend view of search results.

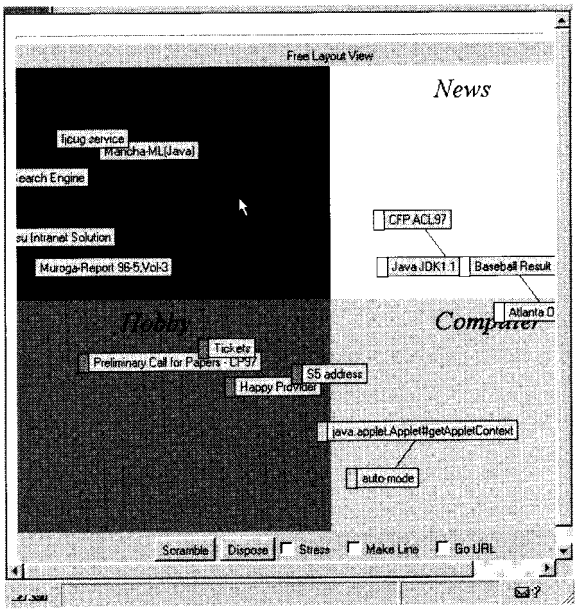

Fig.13– Free layout view

when retrieving large amounts of resources. **Figure 12** shows the trend view for a search for articles containing the string "Syoku Tyuu Doku" (food poisoning) in a newspaper over the 5-year period from 1990 to 1994. The table shows the monthly frequencies of the result documents. The dark squares (in monochrome) or brighter colored squares indicate a high frequency. Using this view, searchers can grasp the trend of search results and select the article of a particular time period by clicking the corresponding square. Figure 12 suggests, not surprisingly that food poisoning is more common in summer. The trend view helps in the browsing step of information seeking.[10)]

5.2.4 Free layout view

In the free layout view, users can freely place titles (links) to A-Kore documents in a two-dimensional space. This view is a personal view. **Figure 13** shows an example of this view. Initially the background is divided into four areas corresponding to the capturing genres, and documents of the same genre are linked to each other.

Unlinked documents inherently repel each other, while linked documents attract each other.

The view is implemented in a Java applet based on a graph layout applet[11] that automatically lays out graphs. The semantics of the links and background are not fixed in this view. Users can display their own background picture to create new relations.

6. Concluding Remarks

This paper discussed the requirements for document information sharing in the current information network society and explained the WHIST environment. In document information sharing, we have to consider not only the environment for document sharing but also the methods used to access shared documents. In these access methods, information organization, which creates secondary information from primary documents, plays an important role.

Compared with existing information sharing environments, one of the distinctive features of WHIST is that most of the secondary information is automatically extracted from document contents by natural language processing and users' action-histories. WHIST also provides multiple views of shared documents.

It is expected that natural language processing will be increasingly applied to document information retrieval, organizing, and sharing.[12] Marchionini[10] stresses the need for information seeking, which is a human-oriented process consisting of information retrieval, analytical strategies, browsing strategies, and learning. WHIST can be seen as a realization of information seeking using the WWW.

As an extension of A-Kore and HyperScheduler, we are designing a small-scale groupware system called WorkWare. WorkWare connects documents and schedule information organically. To extend the temporal push technology of the calendar view explained in Subsection 5.2.2, WorkWare extracts date and related information from the contents of documents and automatically makes connections between people, documents, and schedule information. WorkWare helps its users to access document information with the help of their schedulers.

Lastly, we will briefly describe some issues regarding the implementation of WHIST and WHIST's current status. WHIST is based on Internet standard protocols and document formats such as HTTP, URL, HTML, e-mail, and network news. Most of the WWW server programs of WHIST are implemented in CGI scripts in the Perl and C languages. Client programs of A-Kore are implemented in Emacs-Lisp for Mule, Java applications for Windows, and JavaScript for WWW browsers such as Netscape. Multiple view browsing programs of A-Kore are implemented in Perl and Java applets. All of the WHIST tools have been used and evaluated in the authors' laboratory. Some of the members of our research team are using A-Kore instead of the bookmarks of WWW browsers, e-mail folders, and news logs. With little effort, information such as URLs, e-mail, and news articles can be shared through group views. Some of the tools of WHIST are freely available within Fujitsu and related companies.

References

1) Noguchi, Y.: Tyou Seiri Hou (Super Organizing Method). (In Japanese), Tyuukou Sinsyo, 1993.
2) Namba, I., Igata, N., and Matsui, K.: An Efficient Aho Corasick and Extended Boyer Moore Pattern Matching Machine. Proc. of the Workshop on Information Retrieval with Oriental Languages, pp. 87-95, 1996.
3) Finin, T., Fritzson, R., McKay, D., and McEntire, R.: KQML as an Agent Communication Language. Proc. of the 3rd Int. Conference on Information and Knowledge Management (CIKM94), 1994 (URL: http://www.cs.umbc.edu/kqml/)
4) Papazoglou, M.P. and Laufmann, S.T.: An Organizational Framework for Cooperating Intelligent Information Systems. *Int. J. of Intelligent and Cooperative Information Systems*, **1**, 1, pp. 169-202 (1992).

5) Aiba, A., Yokota, K., and Tsuda, H.: Heterogeneous Distributed Cooperative Problem Solving System Helios and Its Cooperation Mechanisms. *Int. J. of Cooperative Information Systems*, **4**, 4, pp.369-385 (1995).
6) Tsuda, H. and Aiba, A.: Natural Language Understanding with Heterogeneous Constraint Solvers in HELIOS. (in Japanese), *Computer Software*, 13, 6, pp.43-52 (1996).
7) Tanaka, K.: Network Society and Multimedia Databases. (in Japanese), *J. of Information Processing Society of Japan*, **38**, 1, pp.24-29 (1997).
8) Halasz, F. and Schwartz, M.: The Dexter Hypertext Reference Model. *Communications of ACM*, **37**, 2, pp.30-39 (1994).
9) Nielsen, J.: HYPER Text and HYPER Media. Academic Press, 1990.
10) Marchionini, G.: Information Seeking in Electronic Environments. Cambridge Series on Human-Computer Interaction. Cambridge University Press, 1995.
11) van Hoff, A., Shaio, S., and Starbuck, O.: HOOKED on JAVA. Addison-Wesley, 1996.
12) Takeda, H.: Network Enhanced Intelligent Information Integration. (in Japanese), *J. of Japanese Soc. for Artificial Intelligence*, **5**, 11, pp.680-688 (1996).

Hiroshi Tsuda received the B.S., M.S., and Dr. degrees in Information Science from the University of Tokyo, Japan in 1987, 1989, and 1997, respectively. He joined Fujitsu Laboratories Ltd., Kawasaki in 1989 and has been engaged in research and development of natural language processing, logic programming, and network programming. He sojourned at ICOT from 1989 to 1995 for the Japanese fifth generation computer project. He is a member of the IPSJ, JSSST, and ACL.

Kunio Matsui received the B.S. degree in Information Engineering from Shizuoka University, Japan in 1980. He joined Fujitsu Laboratories Ltd., Kawasaki in 1980 and has been engaged in research and development of natural language processing systems. He is a member of the IPSJ.

Kanji Uchino received the B.S., M.S., and Dr. degrees in Information Engineering from the University of Tsukuba, Japan in 1991, 1993, and 1996, respectively. He joined Fujitsu Laboratories Ltd., Kawasaki in 1996 and has been engaged in research and development of information retrieval systems. He is a member of the IPSJ and JSAI.