CINDI: A Virtual Library Indexing and Discovery System

BIPIN C. DESAI, RAIJAN SHINGHAL, NADER R. SHAYAN, AND YOUQUAN ZHOU

ABSTRACT

THIS ARTICLE DESCRIBES A SYSTEM CALLED CINDI for cataloging and searching documents in a distributed virtual library. When putting a document in the library, the author provides and registers metadata in the form of a semantic header for the document. The semantic header contains information on both the syntactic and semantic content of the document. An expert system simulating the expertise of a cataloging librarian helps the provider fill the semantic header according to accepted library practice. Later, if someone is searching for documents in the library, then this searcher is helped by another component of the expert system in properly formulating the query. This component simulates the expertise of a reference librarian. The system then uses information provided by the semantic headers in locating and accessing documents wanted by the searcher.

INTRODUCTION

A virtual library is a collection of electronic documents and resources distributed across a computer communication network (Saunders, 1993). These documents must be cataloged adequately so that a future interested reader (searcher) can find and access them with relative ease. Many systems (Kahle, 1991; Pinkerton, 1994; Mauldin, 1995; Welcome, 1995) catalog a document on the basis of words selected from it. They do not use the document's semantic contents but generally use a program (called a robot, worm, spider, or crawler [Web robots, 1996]) which traverses the network accessing the documents to be cataloged.

Bipin C. Desai, Raijan Shinghal, Nader R. Shayan, and Youquan Zhou, Department of Computer Science, Concordia University, 1455 de Maisonneuve Blvd. West, Montreal, CANADA H3G 1M8
LIBRARY TRENDS, Vol. 48, No. 1, Summer 1999, pp. 209-233

An efficient cataloging system calls for a precise description of the semantic contents of documents. A number of systems have addressed the problem of cataloging among which CORE (Cromwell, 1994), MARC (Byrne, 1991; Crawford, 1984; Petersen & Molholt, 1990), MLC (Horny, 1985; Ross & West, 1985; Rhee, 1985), and TEI (Gaynor, 1994; Giordano, 1994) can be mentioned. These systems, however, are mainly designed for professional catalogers. Creating indexes based on search robots has the following disadvantages: repeated attempts by robots to find new resources would increase the traffic on the network; the number of these robots is increasing and system administrators would likely disallow visits by robots; a robot-based approach would become difficult to justify if the network switches to a fee-for-use mode of operation (Brody, 1995; Brownlee, 1995; Cocchi, Estrin, Shenker, & Zheng, 1991; MacKie-Mason, 1997). Searching with the more recent indexing systems (AltaVista, InfoSeek, Lycos, Yahoo) is cumbersome since the number of hits can be prohibitive due to poor selectivity of the supported search terms (Desai, 1997a).

Metadata should be designed so as to provide the semantic content of an information resource and be better suited to support its subsequent discovery than the resource itself. In many cases, the resource itself may not be able to provide its semantic contents by its nature, or it may do so only after a fairly extensive and time-consuming computation. Examples of such resources are the following forms of information: audio, video, and collections of program codes. Our metadata takes the form of a *semantic header* (SH) (Desai, 1994a). Details of SH and its comparison to the Dublin Metadata Element List (DMEL) are described by Desai (1997). The use of the DMEL in representing Web objects is given by Qin (1998).

When an author puts a document on the net, she is the one who knows the document well and can semantically describe it best. Accordingly, she fills in the slots in the semantic header. For an efficient search, the index is stored in database registries distributed across the network. Since the document provider fills her own semantic header, costly professional indexing is not required.

In this article, we describe an indexing and discovery system called CINDI (Concordia INdexing and DIscovery System), which helps a document provider fill in the semantic header for her document and register it on the net (see Figure 1). Once registered, CINDI provides the facility for a searcher to locate the semantic header and then the document. CINDI thus allows a document to be searched not only on its syntax but also on its semantics. In this article, we use the term "provider" for one who makes a document available on the Internet; a "searcher" is one who looks for document(s); a "user" can be a provider or a searcher.

The organization of this article includes a discussion of the knowledge discovery problem; an overview of CINDI; the registering and maintenance of the semantic header; the expert and database system used;

and the communication process. Owing to space limitation, the last two sections describe briefly the searching and annotation features of CINDI. The current implementation status of CINDI and our future plans are given in the conclusion.

Discovery on the Internet

In June 1995, we made a series of tests on a number of then existing Internet indexing systems; these were ALIWEB, DACLOD, EINet Galaxy, GNA Meta-Library, Harvest, InfoSeek, Lycos, Nikos, RBSE, World Wide Web Catalog, WebCrawler, WWW, and Yahoo. The intent of these tests was to determine how many URLs to documents containing the target search strings Bipin (AND) Desai were indexed by these systems. The results obtained are given in Table 1 which shows the number of hits, mis-hits, and misses (Desai, 1995a).

In this table and the following tables, the number of hits is the count of the documents found to be relevant to the query. The number of duplicates is the number of times the same document was retrieved by the indexing system using different components of the search criteria or when the same document is being served from more than one site. In the more recent search engines, the systems tend to eliminate the former form of duplicates; however, the same document accessible from more than one site is replicated in the result. The number of mis-hits is that of irrelevant documents, and the number of misses is the number of relevant documents missed by the search system.

Many of these pioneering indexing systems, existing in mid 1995, are no longer active. In the meantime, a number of new systems, such as Alta Vista, OpenText, Hotbot, and so on have emerged. Many workers in the domain of the digital virtual library feel that these newer systems have addressed many of the issues we raised in designing the CINDI System.

Table 1.
Search Statistics for Using the Search Term Bipin (AND) Desai: June 1995

Search System	*Number of Hits*	*Number of Duplicates*	*Number of Mis-hits*	*Number of Items Missed*
Aliweb	none	-	-	25
DACLOD	none	-	-	25
EINet	6	0	4	23
GNA Meta Lib.	none	-	-	25
Harvest	none	-	-	25
InfoSeek	7	0	0	18
Lycos	231	2	222	18
Nikos	none	-	-	25
RBSE	8	-	8	25
W3 Catalog	none	-	-	25
Web Crawler	7	3	0	21
WWW	2	0	0	23
Yahoo	none	-	-	25

The next series of tests was done from September through October 1997 to find the number of relevant documents that could be located by the then current search engines and to evaluate the usefulness of the index entries retrieved. Relevance of a document could be judged easily once the target set was known. We repeated the test performed in 1995 with the same search words. At the time of the test, some 325 URLs were known to contain the words "Bipin" and "Desai." These represent Web documents pertaining to one of the authors of this article. The complete list of these URLs can be retrieved from the following URL: http://www.cs.concordia.ca/$\sim$faculty/bcdesai/search-oct97/whereis-Desai.html.

The first set of tests, the results of which are given in Table 2, was done on the following search engines: Alta Vista, Excite, Hotbot, Infoseek, Lycos, OpenText, and Yahoo.

Table 2.
SEARCH STATISTICS FOR USING THE SEARCH TERM BIPIN (AND) DESAI: SEPT. 1997

Search System	*Number of Hits*	*Number of Duplicates*	*Number of Mis-hits*	*Number of Defunct*	*Number of Items Missed*
AltaVista/ Yahoo	97	9	23	4	264
Excite	114	10	29	7	247
InfoSeek	8	2	1	1	319
Lycos	57	7	15	14	297
Hotbot	247	28	58	19	155
OpenText	19	-	7	5	318

As in the 1995 series of tests, we have shown the results by noting the number of hits produced, the number of duplicates, number of mis-hits, and the number of relevant documents not listed in the result; we have also included a column for the number of defunct URLs (which do not lead to any valid target Web pages). The duplicates are either the same document being served from two sites or the same document from the same site listed more than once. The latter errors seem to have been corrected in most search engines which do sufficient pre-processing of the result to eliminate obvious duplicates before presenting it to users.

The documents missed could be due to the approximations used by engines such as Alta Vista when it finds a large number of hits. However, the fact that these search engines could not locate all documents indicates the difficulty of reaching isolated URLs by search robots.

The bigger problem is the lack of selectivity and the measure of usefulness of the documents found by the search engines. We have collated the results by following the trail of "next" sets of URLs, and these could be viewed by pressing on the number of hits for each search engine in the online version of Table 2 (Desai, 1997a). A glance at the abstract or sum-

mary presented by the search engine indicates that they are not very revealing and, except for the most pedestrian needs, following the pointers would result in a drain of the searcher's time.

Search Statistics for Using Various Search Strategies

In a third series of tests, we used a simple search with the search terms: Bipin Desai, the advanced search expressions "Bipin Desai," and "Bipin C. Desai" respectively. These tests were made only on Alta Vista/ Yahoo. The results of these tests are given in Table 3.

Table 3.
Search Statistics for Using the Various Search Terms: Sept. 1997

Search System	*Number of Hits*	*Number of Duplicates*	*Number of Mis-hits*	*Number of Defunct*	*Number of Items Missed*
Alta Vista/ Yahoo	4285	30-90%	10-80%		200+
Alta Vista/ Yahoo	29	2	13	3	312
Alta Vista/ Yahoo	128	14	-	10	201

The result for a simple search of Bipin Desai (row 1 of Table 3) shows a high number of hits (4,285 in the test reported here; there is a bit of variation due to Alta Vista's method of abandoning a search after a sufficiently large number of hits is made). However, the simple search produces very low selectivity and relevance. Most of the hits in the top 160 entries are irrelevant, and a large number of relevant documents are not located. Most searchers will not have the patience to go through more than a few pages of the result, there being some 214 pages of the result for 4,285 hits.

The result for an advanced search expression for "Bipin Desai" (row 2 of Table 3) gives a lower number of hits and relevance since the author prefers to include his middle initial in the name. Most searchers may not be aware of such details.

The result for an advanced search expression for "Bipin C. Desai" (row 3 of Table 3) gives a relatively large number of relevant documents, some of which are duplicates, being accessible from more than one site. Some of the defunct URLs are not deleted by the search engines, pointing to the maintenance problem of the underlying database. However, this search still missed about two-thirds of the documents.

These tests lead us to believe that a search system should support better semantics. It is our opinion that the semantic header-based system (see Figure 2) (Desai, 1997b), wherein the provider of the resource is responsible for generating the entry, would be a more useful scheme to

support discovery. The semantic header is designed to describe the semantic contents of the source information resource and is better suited to supporting knowledge discovery than the actual resource. Many formats of a resource may not be directly accessible electronically, be suitable for direct discovery, or may require a considerable amount of computation and extremely slow response. The semantic header could also be used as a surrogate to express semantic dependencies inherent in a collection, which is not possible to do with existing search engines.

The quality and the reliability of the document could be expressed by including reviewers' comments in the form of annotation with the semantic header. Such reviews are rarely accessible in traditional cataloging systems. However, in the CINDI system this, along with the abstract supplied by the authors, would be valuable in judging the suitability of a document to a searcher. It could also give feedback to the provider. The semantic header metadata also allow the server system to perform initial query processing and thus reduce the cost involved in accessing and processing irrelevant resources.

Overview of the CINDI System

The overall structure of the CINDI system is shown in Figure 1. The workstation at the provider's site contains the CINDI client software and a partial catalog. The client software is composed of a registering graphical user interface, the client portion of a distributed expert system, and the associated knowledge base. The semantic header information entered by the provider of a resource using this graphical interface is relayed from the user's workstation by a client process to the database server process at one of the nodes of the SH Distributed Database (SHDDB). The node is chosen based on its proximity to the workstation or on the subject of the index record. On receipt of the information, the server verifies the correctness and authenticity of the information and, on finding everything in order, sends an acknowledgment to the client. It also has a partial catalog of the thesaurus database. The function of these are described later in the section on the Semantic Header Registration System.

The server node is responsible for locating the partitions of the thesaurus for the subject hierarchy or the sites of the SHDDBs where the entry should be stored and forwards the replicated information to appropriate nodes. The server node is also responsible for providing the catalog information for the search system. In this way, the various sites of the database work in cooperation to maintain consistency of the replicated database. The replicated nature of the database also ensures distribution of load and continued access to the system when some sites are temporarily nonfunctional.

The user interface for the CINDI system consists of three graphical interfaces: the SH index registration system, the search system, and the

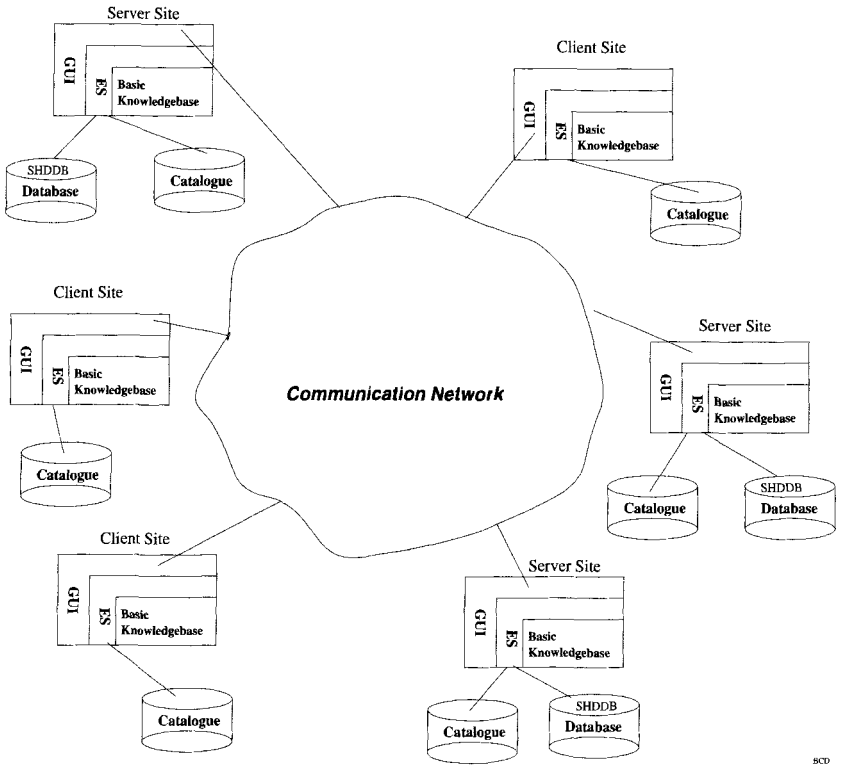

Figure 1. Overall Structure of the CINDI System.

annotation systems. The SH index registration allows a document provider to fill in the slots of the SH. The search interface is used by a searcher to locate documents. The annotation interface allows a user to insert comments on a document in its semantic header. The indexing and search systems have an associated expert system that helps the providers and searchers in selecting appropriate subject terms to best describe the source document or the query respectively. These functions are described briefly later in this article.

The SHDDB contains information on subject hierarchy, a thesaurus to help select controlled terminology from the subject hierarchy, the registered semantic headers, and the associated annotations. An expert system mimics the cataloging librarian and helps a provider make suitable controlled subject entries in the semantic header. When a searcher uses the expert system, it mimics a reference librarian in helping him locate the relevant subjects. The communication from a user workstation to the database site uses the client/server paradigm.

The slots of the SH, as seen in Figure 2, contain the title of the document, its authors, the subject(s), abstract, and so on. The intent of the semantic header is to include those elements that are most often used in the search for an information resource. Furthermore, the SH provides information on the organization of a document such as chapters, sections, or whether the document is part of or an actual collection. The registry containing all the semantic headers is much smaller than the actual collection of documents (Desai, 1997). A person searching for a document

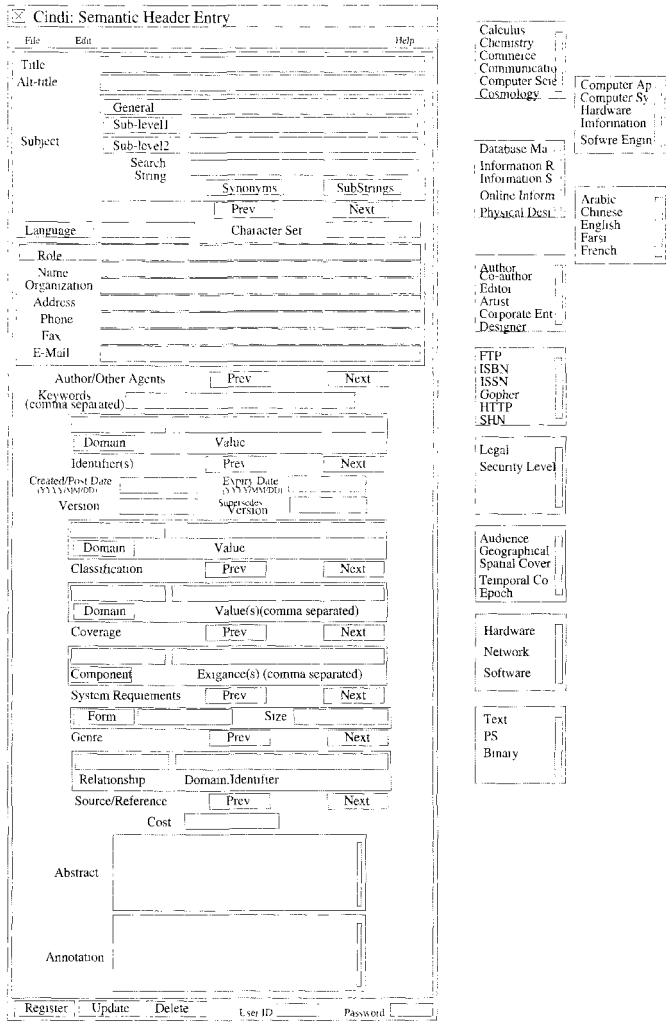

Figure 2. Components of the Semantic Header.

first locates the appropriate SHs. Once these are found, the actual documents can be easily accessed. Since the registry for the SH is smaller than the actual collection, and much of the query for the required search terms could be preprocessed, searching becomes faster.

Semantic Header Registration System

The Semantic Header entry and registration subsystem provide a graphical interface (similar to Figure 2) to facilitate the task of the provider (author/creator) of a resource to register the SH for the resource. The system also offers help by means of pop-up selection windows and an expert engine to suggest controlled terms. This expert engine is intended to bring some of the expertise of a catalog librarian to the ordinary user (Chander, Shinghal, Desai, & Radhakrishnan, 1997). Many of the elements in the SH can be extracted directly from the resource document if they are properly tagged (the Automatic Generation of Semantic Header project is currently underway at Concordia). Once the information is correctly entered, the provider can decide to register the SH entry in the database. When the SH information is accepted by the database, the provider is notified. A user ID and the associated password is required when the SH is first registered and for all changes made to it. Since the user ID and the password are not accessible by anyone other than the original registrar (usually the provider) of the index entry, the entry can only be updated by persons who are cognizant of them. Changes that may be made could be due to changes made in the resource or its migration from one system to another. A copy of the SH is stored at the provider's site for convenience in later updates.

The subject for the document being indexed is selected hierarchically. The provider first selects the general level, and a rule-based system thereafter guides the selection of the corresponding lower levels. In case the provider is unclear about the subject area of the document, she can seek help by entering a string in the "search string" slot and use either the synonym or substring push button. A rule-based system is invoked and guides the provider in selecting the appropriate subject. The provider is not allowed to enter the subject terms directly, thus restricting the subject terms to a controlled vocabulary from the subject headings.

Some of the slots can contain more than one value—e.g., the author slot can have more than one name and address to signify that the document has multiple authors. These multiple values are entered by using the NEXT and PREVIOUS buttons in the corresponding slots. Using this scheme, any number of values for such slots can be entered.

When a new provider first creates an SH, she chooses, with the help of CINDI, her own unique user ID and password. These values are stored

in the CINDI database and are associated with all SHs registered using this pair of values. The values of the SH slots can be updated only by the provider of the SH. The only exception is the annotation slot: here, any user can insert comments. Only the provider is allowed to delete her semantic header. The deletion of an SH would not affect its user ID and password.

Expert System Support for Registering

Expert and knowledge based systems have been used in various domains to provide users with the expertise of a domain expert. In our system, we need to guide the provider of a document in choosing the appropriate subject hierarchy from a controlled hierarchy (we use a hierarchy derived from ACM, INSPEC, and the Library of Congress Subject Headings [LCSH]). In many domains, the domain knowledge is encoded as a set of "if . . . then . . .else" rules. Encoding knowledge in such a manner and checking the user input against such encoded rules has been found to be fairly inefficient. Our initial approach using CLIPS (Giarratano & Riley, 1994) to encode these rules proved this observation. Furthermore, CLIPS imposed a considerable overhead on the system.

In registering an SH, and for later searching, it is essential to employ the knowledge and expertise of cataloging librarians. However, employing professional librarians may be costly, thus the need of an expert system to model librarians' expertise and guide users in cataloging and searching. The expert system would help users choose correct subject terms. It would also guide them to register, update, delete, and annotate SHs. The expert system is designed so that its query for resource searches facilitates efficient database access and reduces the number of incorrect results generated.

Expert systems have been used to encode the expertise of experts in well-defined domains. The system can then be used to guide users in reaching the same conclusion as the domain experts in a given situation. In our case, we need the expertise of a cataloging librarian to help users choose appropriate controlled terms using a knowledge-base encoded as a thesaurus of synonyms.

A typical user wanting to create a semantic header entry for her document usually does not have precise knowledge of the exact subject heading hierarchy under which the document should be classified. However, she has a very good idea of the exact topic(s) treated in her document and knows the usual terms used in the relevant literature. Such terms may not be the same as the controlled terms established by a cataloging authority such as the LCSH. The cataloging system, mimicking a cataloging librarian, should be able to guide the user in a search for controlled terms from a subject heading hierarchy.

Checking for all input combinations using direct encoding of knowledge has been found to be very inefficient (Chander, 1995). Our first attempt was to incorporate the expert system rules by embedding an expert system shell in the cataloging system. However, this approach was not only slow but increased the overall size of the system. Furthermore, the shell was not sensitive to the context under which a rule was to be tested and hence had to recompute the set of rules to be tested.

Our second attempt was to replace the expert system shell by distributing the rules in the appropriate components of the system. This not only increased the context sensitivity of the system and reduced the size of the program code, but it also reduced the number of rules that had to be considered for each subsystem. This distributed system was encoded directly as C/C++ functions, thus further improving the system performance. In this prototype version of the system, we allowed the user to enter a term at any of the three levels of the subject hierarchy. If the term entered was found to be a synonym of a controlled term at the same level as the one entered by the user, then the system would show the controlled term and the corresponding higher level term(s) and prompt the user to confirm them. However, should the term entered by the user be a synonym of a controlled term at a different level of some subject hierarchy, then the system requests the user to resolve this conflict and warns the user of this inconsistency. Also, a term entered by the user at a lower level may not be a synonym for any term at this level of the current subject hierarchy. To avoid such confusion and the possible need to backtrack, we revised our scheme to take advantage of the interactive and graphical nature of the interface.

In our final implementation, our strategy is to separate the subject hierarchy search into two orthogonal components: (1) a strictly hierarchical

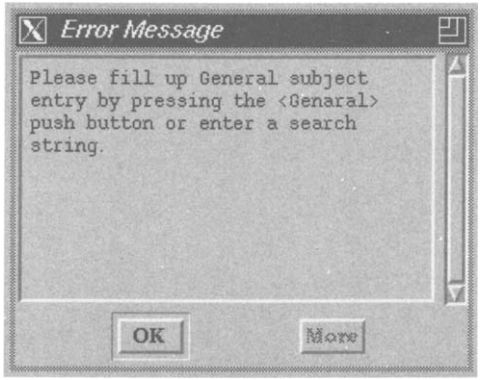

Figure 3. Selecting Subject Level-1 Before the General Level.

subject entry system using context sensitive pull down menus, and (2) finding controlled terms for a string or term entered by a user. In the first case, the user is guided by the system to first select a higher level subject before being allowed to select a lower level. In this component, the user is not allowed to directly enter a subject term as indicated in Figure 3. In this way, the search by the system for a lower-level is thus limited to the already selected controlled higher levels and avoids the confusion and backtracking.

In the event a resource provider has a basic idea of a term in the subject hierarchy that she or he wants to use as a resource, the term could be entered in the search slot of the GUI and use the synonym substring feature to query CINDI for a controlled term. Terms entered in this manner by a user could be synonymous at any level of a subject hierarchy. This could also occur when a provider enters terms that are part of controlled terms (substrings) and thus cause the term to match entries at multiple levels. If the provider enters a synonym such as *system*, which could occur at a large number of subject hierarchies, the expert system displays the result and suggests the provider enter a more specific synonym. The system, using a thesaurus of synonyms and controlled terms, finds subject hierarchies closest to a user's entered term and presents these for selection. The displayed term could be at any level of the hierarchy. The result of the search is presented to the provider in a pop-up window, as shown in Figure 4; the provider would then make a selection at one of the levels indicated by the system or reject the choice to try another term. The provider views the matching subject terms for each matching level by selecting the corresponding push button in Figure 4. If the entered term maps into a controlled term at a lower level of a subject hierarchy, the system automatically selects the higher levels of the subject hierarchy and displays them for the user to make a selection. These are illustrated in Figures 5 and 6.

The use of the GUI, along with the orthogonal separation of subject level entry in hierarchical order and by matching a synonym, has simpli-

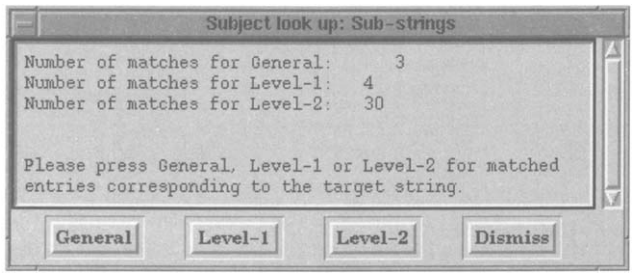

Figure 4. Example of Sub-String Look-Up: Search String "Com."

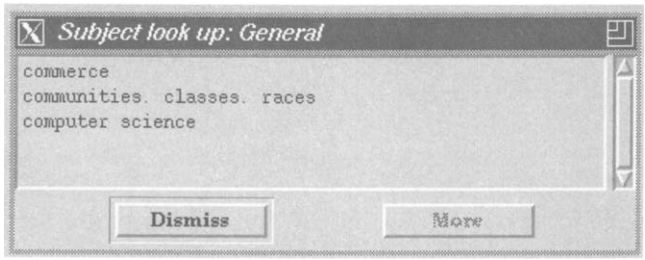

Figure 5. Example of Sub-String Look-Up: Display General Level.

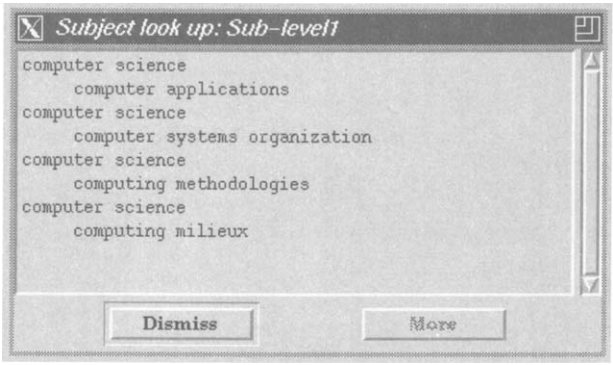

Figure 6. Example of Sub-String Look-Up: Display Sub-Level-1.

fied the implementation of both the cataloging system and the searching system. By limiting the provider to enter a higher level subject before a lower level subject (see Figure 3), we have avoided many pitfalls that occurred in our earlier implementation and hence simplified the set of rules and the data that have to be retrieved. This improved the performance of the system and allowed us to provide context-sensitive help for indexing, updating, and searching.

Some fields of the semantic header may have multiple or repeatable entry fields. The repeatable entry fields in the SH allow, for example, a document to be classified under more than one subject; an article could be written by more than one author; and the document can be identified by its HTTP, FTP, ISBN, and so on. We provide PREV and NEXT push buttons (Figure 2) at the bottom of each block in the user interface to accommodate these entries.

The PREV push button allows the user to view or modify the previous entry of a block and the NEXT push button allows the user to enter, view, or modify the next entry of a block. To proceed to the previous or next entry, certain rules are enforced. For example, the NEXT entry button

action is not allowed until the current entry is completed and it has a valid value. The validity of all slot values is verified at the client site before the system allows the semantic header entry to be registered.

CINDI Database System

The index entries registered by a provider of a resource in CINDI are stored in SHDDB. From the point of view of the users of the system, the underlying database may be considered to be a monolithic system. In reality, it could be distributed and replicated, allowing for reliable and failure-tolerant operations. The interface hides the distributed and replicated nature of the database. The distribution is based on subject areas and, as such, the database is considered to be horizontally partitioned (Desai, 1990).

The database on different subjects in the CINDI system is to be maintained at different sites of the communication network (as illustrated in Figure 1). The locations of such nodes need only be known by the intrinsic interface and the database catalog used to distribute this information. Catalogs would also be used to store information about the location of the subject areas maintained in the SHDDB so that the client process at the user workstation can select subject hierarchies for indexing and retrieving semantic headers.

The semantic header entered by the provider of the resource using a graphical interface is relayed from the user's workstation by a client process to the database server process at one of the nodes of the SHDDB. The node is chosen based on its proximity to the workstation or on the subject of the index record. On receipt of the information, the server verifies the correctness and authenticity of the information and, on finding everything in order, sends an acknowledgment to the provider at the client site.

The thesaurus database contains four object classes which represent the general subject of the subject hierarchy, the sub-subject and the sub-sub-subject, and finally the *synonym* which contains those subject terms (at any level) synonymous with the controlled terms. The registration subsystem at the server site is responsible for registering, deleting, updating, and annotating semantic headers. This subsystem uses an SHDDB area to store the SHs and other related objects. The database is an aggregation of three objects: SH, user ID, and word. SH contains all fields of the semantic header. Some of these fields are included in the SH object as attributes; others are objects which are components of the SH object. As an example, the *author* object is a part of the SH object which should be (partially) ordered. This is because, in the GUI, the first author field entered by the user would take part in the construction of the semantic header name (SHN). The SHN is derived from the following required elements in the semantic header: title, name of first author (or name of

organization, if the document or resource being registered is attributable to a corporate or organizational entity), first subject, creation date, and version. The SH also includes information in the Identifier object to access the corresponding online information resource. The UserID object contains both a user ID and a password entered by the user in the GUI.

The SHDDB also contains the *word* object. It stores non-noise words appearing in those fields of an SH which may be used during the search operation. The *word* object corresponds to the SH objects where the value of the *word* object appears in the semantic headers. This object contains a fixed number of *context* objects equal to the number of search GUI fields.

The semantic header database and the catalog database are implemented using the ODE (Object Database Environment) database system (Arlein, Gava, Gehani, & Lieuwen, 1993; Agrawal & Gehani, 1989; Agrawal, Dar, & Gehani 1993; Biliris & Panagos, 1993).

Registering the Semantic Header

The graphical interface (see Figure 2) facilitates the provider (author/creator) of a resource to fill in and subsequently register the bibliographic information about the resource. Once the information is entered, the provider can decide to register the semantic header entry in the database. The REGISTER push button allows providers to register the current semantic header into the CINDI database. To register a new semantic header, a user ID and password are required. Before an actual registration request is made to the server, the client system would check the SH entry to ensure that all the required fields are entered. This validation ensures that the standard indexing scheme is enforced.

When a semantic header is received at the database (server) site, the system performs a number of operations to register an SH. The first step is for a parser to verify the syntax of the input file and ensure that the mandatory fields of the SH have been entered. The non-noise words of the semantic header are stored in temporary variables and data structures for later use. The next step is for the database module to verify the status of the user ID and the password and ascertain that the SH does not already exist in the database. Finally, the words and the semantic header are indexed into the database. The non-noise words would be added to the database and all attributes of the SH object would be initialized. Finally, the unique SHN identifier is assigned to the newly added SH object. In a case where an error occurs, an error code would be sent to the client site which would be used by the client expert system to guide the user to correct the problem.

When an SH is registered by the server, a copy of it is stored locally at the client site. Later, this could be loaded to update the semantic header as discussed in the section on Updating a Registered Semantic Header.

Client-Server Communication

The communication between the user interface and the database is made using the TCP/IP protocol written in C. The server daemon runs at the server site. When a client, at the user site, is called by the user interface, it connects to the server and sends data in a file containing the query request. The server calls appropriate functions for parsing the file and transforming it into a database specific query (or queries). This query is sent to the database for processing. Finally, the server receives the result of the query in a file created by the database module and sends it back to the client using the TCP/IP protocol.

Since a server may provide services to more than one client at a time, the server assigns a unique client ID to each file received from the client site. Each client ID is a concatenation of three fields. The first field is a fixed string used in all files. The second field is the value of time in seconds. The third one is the process ID of the child process responsible for serving a specified client. Thus possible client ID collisions at the server site are avoided.

In a case where a network problem prevents data transmission, the server program provides a timeout mechanism to prevent the GUI at the user site from waiting for the server to respond indefinitely. If, after a specified period of time, the server fails to complete the process of transmitting data from/to the client, the server sends an appropriate message to the client process and disconnects from the client process. Subsequently, the user interface receives the error code from the client process and displays an error message to the user. This way, the GUI does not freeze, and the user can carry on making other requests.

Once a client has established a connection to a server, it issues a series of transactions each of which is invoked by a function call. One such transaction actually performs the connection between client and server. With the exception of the connect transaction, each transaction generates a sequence of actions as follows: a message identifying the transaction to be executed and the associate data are sent by the client to the server; the server process identifies the transaction from the associate data; and a message containing the response to the transaction is sent by the server to the client along with the ID of the original transaction. This is repeated until the client initiates a terminating transaction.

Updating a Registered Semantic Header

The UPDATE push button (see Figure 2) allows a provider of resources to update an existing semantic header. When there is a need to update the SH, it is loaded from a local saved copy into the GUI for registration through the use of the OPEN menu item in the FILE menu. The provider is not allowed to change fields corresponding to the SHN nor the reviewers' annotations (except for the annotations made at the time of

registration; the others are not stored locally in the original copy of the registered SH at the client site). The GUI rule system enforces this requirement by making these fields non-editable during the update phase and by giving appropriate warning messages if the provider tries to modify these fields. To register the updated semantic header, the provider has to enter the user ID and password that were used when initially registering it.

The UPDATE button is disabled when a new semantic header is being entered. The REGISTER push button is disabled when a provider opens an existing semantic header for modification.

Deleting a Registered Semantic Header

The DELETE push button is used to remove a semantic header from the system. An SH can be deleted by the provider who registered it only if no public annotation were made in it after the SH was registered. The procedure of deleting an SH is similar to that of updating it. To delete an SH, the provider enters both a user ID and the corresponding password that match those entered when the SH was registered. Otherwise, an error message is relayed to the provider. The SH and the SHN maintained in the word object with the value corresponding to each non-noise word are deleted from the database. If a word object, after such deletion, is found to be associated with no SHNs, it would be deleted from the database as well. The DELETE button is disabled when the user enters a new SH.

SEARCHING

In the current search system, we have incorporated the elementary expertise used by a reference librarian. Reference librarians are aware of the conventions used by cataloging librarians. They are conversant with the classification schemes, terms, indexes, structures, and resources available for a user's particular need. This basic expertise of the librarian is replicated to assist the users of our application in discovery and guides the user in entering the various search items in a graphical user interface similar to the one used by the registering subsystem (see Figure 7). The system is designed so that its query for document search facilitates efficient database access and reduces the number of incorrect results generated. For example, the system aids the user in completing a given field entry based on the contents of the other search fields. As in the case of the registering sub-system, the expert system provides context-sensitive help in choosing appropriate search terms for index entries such as the subject, sub-subject, sub-sub-subject, and so on (Chander et al., 1997). When no SHs are found, the system suggests other alternatives to the user.

The CINDI system offers a wide range of search criteria to allow precisely targeted resource retrieval. Most widely used search fields are

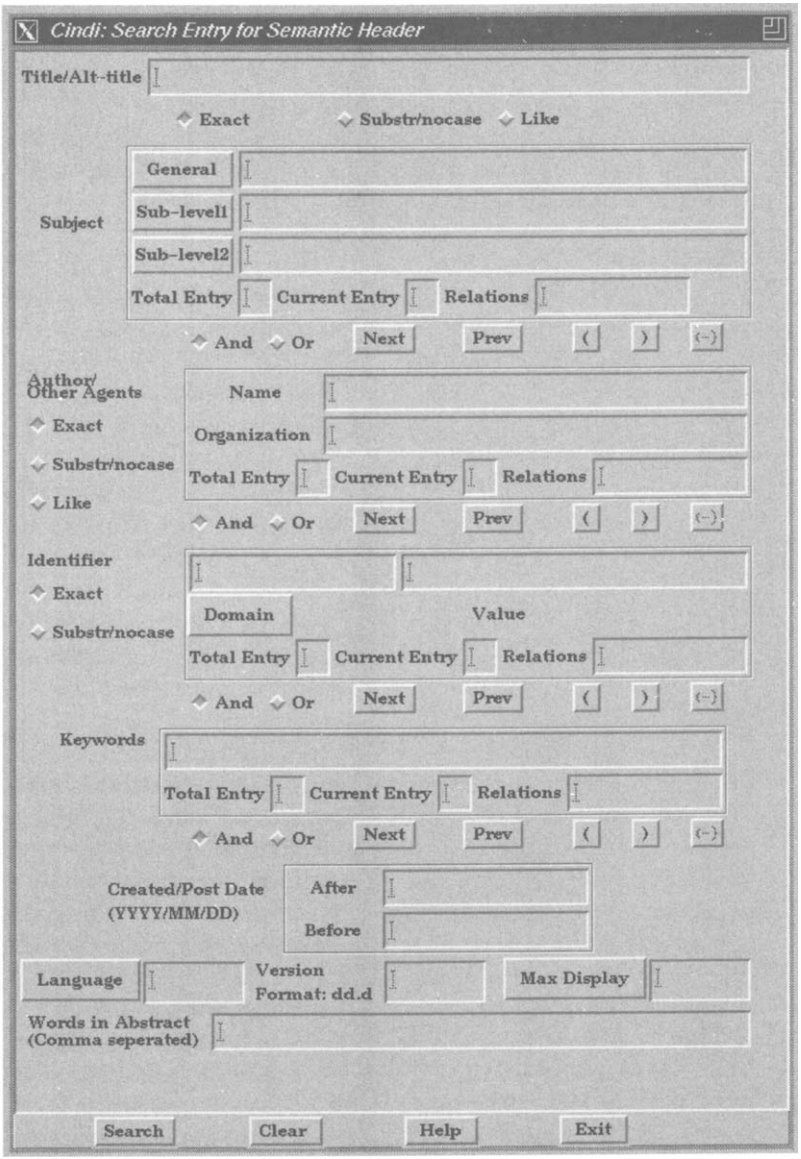

Figure 7. Graphical User Interface: Search System.

included to allow the user to tailor the search and discovery needs. In most of these fields, the search can be specified to be performed using an exact match or substring of the word entered by the user. Most of these fields allow the logical operations *and* and *or* to refine the search. Parentheses are also provided to allow nested logical search predicates. To transform the user-defined queries received from the client into database queries, we employ the reverse Polish (postfix) notation.

Once the user has entered a search request, the client process communicates with the nearest server which determines the appropriate sites of the SHDDB. Subsequently, the server communicates with these sites and retrieves one or more SHs. The results of the query can then be collected and a list in user specified block size is sent to the user's workstation as illustrated in Figure 8. The contents of any of these semantic headers are displayed on demand by clicking on the title in the list. This is illustrated in Figure 9. The user can navigate to the other semantic headers by pressing the appropriate push buttons in this display. The system allows the user the facility to access one or more of the actual resources by connecting to a browser. A user can access the actual resource only if the selected item for access has an identifier which allows online access via a browser such as Netscape which could be used to display the resource.

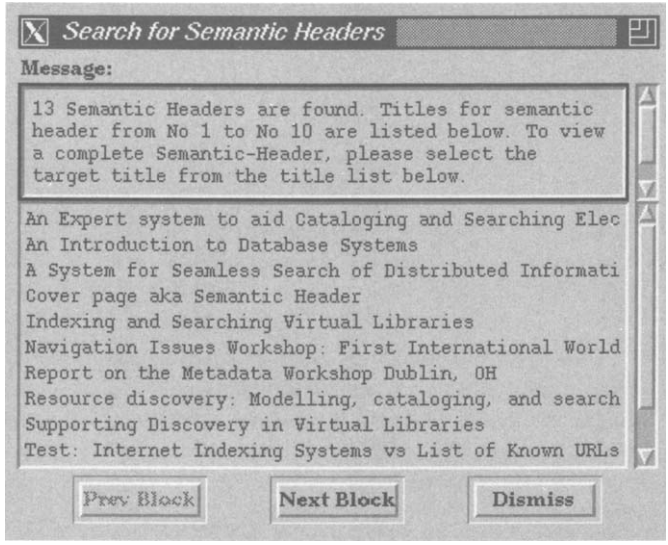

Figure 8. Result of a Search Query: List of Semantic Headers.

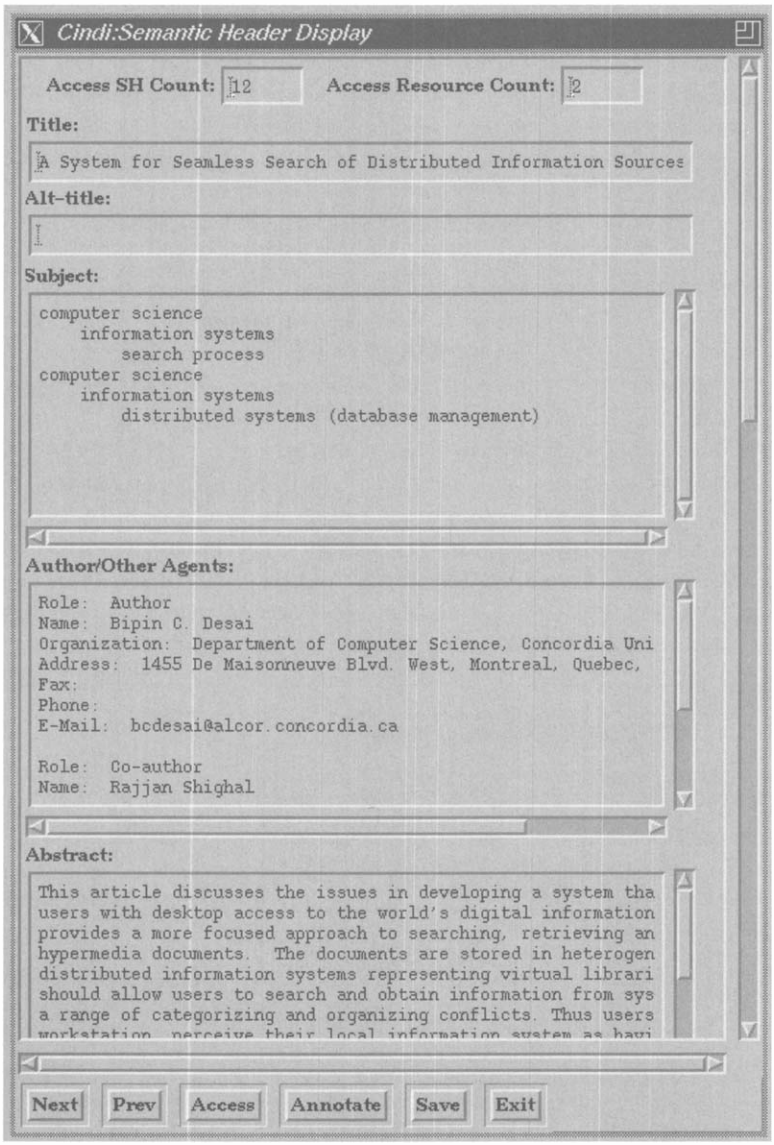

Figure 9. Display of a Selected Semantic Header.

We associate two counters with each SH to measure the extent of the dissemination of the SH and the resource it describes. One counter records the number of times a given SH was accessed in searches made by users of the system. The second counter indicates the number of accesses made via CINDI to the actual resource corresponding to a semantic header.

ANNOTATION

The research community depends on peer review of documents submitted for publication. Such review or annotation is often not published. However, comments to the editor made by readers of journals are usually published and are accessible to the community. Since many of the resources on the Internet tend not to be reviewed, it would be beneficial for a user to have access to annotations made by other users for a given resource. The proposed system allows users to add annotations to an existing resource.

These annotations are stored along with the index in the SHDDB (Desai, 1996). Peer reviews of electronically submitted papers could be implemented using such annotations. Authentication of reviews has to be done by an appropriate editorial board.

The graphical user interface of the annotation subsystem is shown in Figure 10. The annotation subsystem is similar to the indexing subsystem. However, only a few of the indexing entries that uniquely identify the resource in question are required. An annotation made by any user can be entered and would be registered with the identity of the user. Such annotations could be valuable guides for future users.

To avoid nonserious entries to the annotation by unscrupulous readers, we have separated the annotation entry from the search subsystem. In order to add annotations, the user has to save the semantic header when viewing it in the search subsystem using the SAVE push button. It is expected that the user would actually access the resource corresponding to the SH. If the user decides to make an annotation, he loads the saved SH from the local file system by pressing on the LOAD SH push button in the annotation form.

The newly entered annotations, together with the annotator's information, as well as his login name and host name, would be concatenated to the existing annotations when the annotator registers them.

CONCLUSION

Current index systems are based on harvesting the network for new documents. Such documents are retrieved and their contents used to provide terms for the index. The big disadvantage with this scheme is the unreliability of the index entries produced and the lack of an authentic abstract for the item. The current Dublin Metadata Element (Desai, 1995) list also suffers from the absence of the abstract. Furthermore, current index schemes are relevant for resources of limited protocol and are not applicable to other resources. CINDI has addressed these problems, giving rise to the following advantages: CINDI allows the indexing of resources accessible online or offline; CINDI requires that the provider of the resource use controlled terms and provide an abstract (this is an improvement over extracting phrases from the first part of a resource or by simply

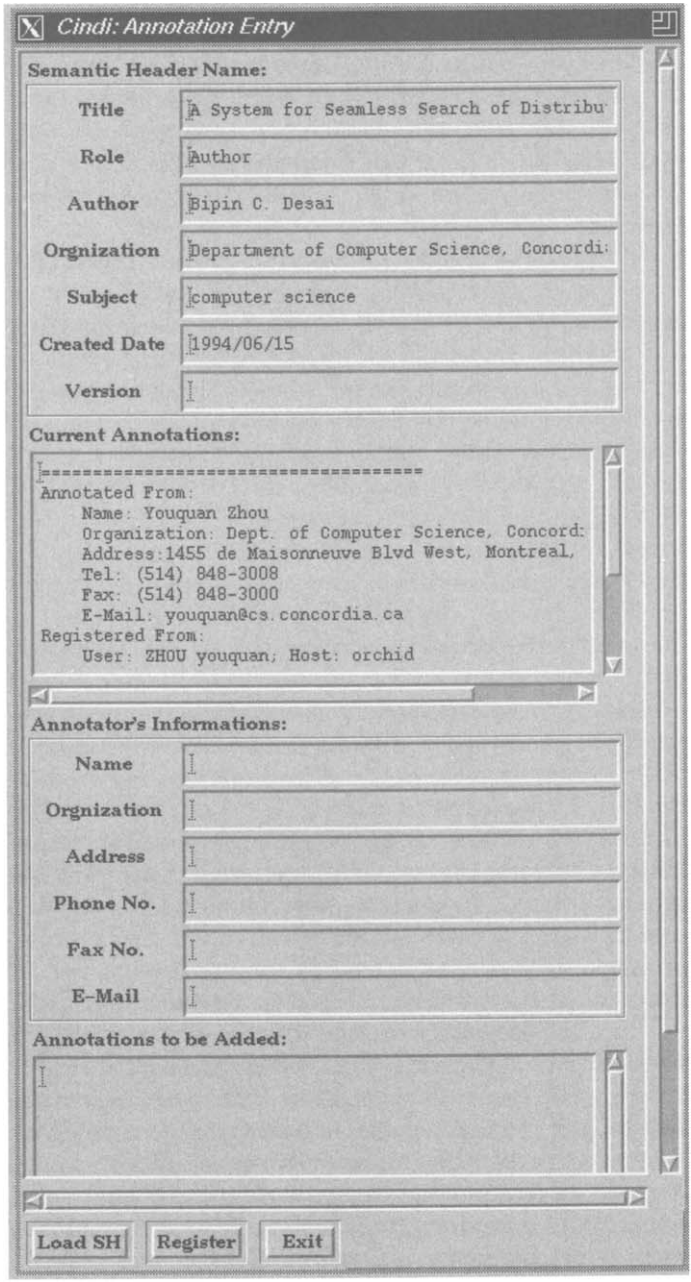

Figure 10. Graphical User Interface: Annotation System.

picking up terms by scanning a resource as is done by some of the existing systems), since the registration of the semantic header in the database is performed by the provider of the resource, it improves cost, accuracy, and efficiency; CINDI allows annotations by reviewers which enable future users to make better informed decisions regarding the relevance of the source resource; the size of the CINDI database is not limited since the database is distributed among a number of sites. Furthermore, the system lends itself to the well researched distributed query processing techniques to support discovery from these distributed database systems in parallel.

The expert system support provided in our implementation is obtained by distributing the rules to be enforced in various parts of the system including the GUI. This method was chosen over using expert system shells such as CLIPS. While such shells facilitate knowledge engineering and rule encoding, we found in our trial implementation that they incurred a significant overhead. For example, for every rule firing, if a system such as CLIPS were used, its inference engine would recompute the set of rules that can fire. The distribution of the rules in CINDI improved performance since only a small number of rules had to be developed in such an environment.

CINDI, in its current implementation, uses an ODE database and an X-window based Motif interface (Heller, 1994). The client software for ULTRIX for Sun can be downloaded from: <http://cindi.cs.concordia.ca/cindi>. We plan to port the client software to Linux. The initial decision to use a Motif-based interface was due to the limitation of HTML (Hypertext, 1997) in providing unspecified numbers of repeating fields. With the newer tools, such as XML and Java, we have undertaken porting CINDI to the Web. We will also port SHDDB to a robust commercial DBMS.

CINDI, as do other self-indexing systems, requires the active participation of the provider. To make this task easier, we are providing an automatic semantic header generation system. Preliminary results of this work in progress is encouraging, and it will be incorporated in our Web-based version.

ACKNOWLEDGMENTS

We gratefully acknowledge the contributions of P. G. Chandra for developing the initial expert systems for cataloging librarians. Concordia University reference librarians Carol Coughlin and Lee Harris were always available when we had questions about cataloging and reference librarian practice. This contributed greatly to the rules developed for the expert system and the thesaurus. We are also grateful to the editors, Jian Qin and M. Jay Norton, for their very constructive comments which helped improve the paper. In addition, this work is partially supported by grants from NSERC.

REFERENCES

Agrawal, R., & Gehani, N. (1989). ODE (object database and environment): The language and the data model. In *Proceedings of the 1989 ACM-SIGMOD International Conference on the Management of Data* (Portland, Oregon, 2 June 1989) (pp. 36-45). New York: Association for Computing Machinery.

Agrawal, R.; Dar, S.; & Gehani, N. (1993). The O++ Database programming language: Implementation and Experience. In *Proceedings of the Ninth International Conference on Data Engineering* (April 19-23, 1993, Vienna, Austria) (pp. 61-70). Los Alamitos, CA: IEEE Computer Society Press.

Arlein, R.; Gava J.; Gehani, N.; & Lieuwen, D. (1993). *Ode 4.1* (user manual). AT&T Bell Laboratories.

Biliris, A., & Panagos, E. (1993). *EOS user's guide* (Release 2.0). AT&T Bell Laboratories.

Brody, H. (1995). Internet@crossroads.$$$. *Technology Review, 98*(May/June), 24-31. Retrieved from the World Wide Web January 11, 1999: http://www.techreview.com/articles/may95/Brody.html.

Brownlee, N. (1995). *New Zealand experiences with network traffic charging.* Retrieved January 11, 1999 from the World Wide Web: http://www.auckland.ac.nz/net/Accounting/nze.html.

Byrne, D. J. (1991). *MARC manual: Understanding and using MARC records.* Englewood, CO: Libraries Unlimited.

Chander, P. G.; Shinghal, R.; & Radhakrishnan, T. (1995). Goal supported knowledge base restructuring for verification of rule bases. In R. F. Gamble (Ed.), *IJCAI'95 Workshop on Verification and Validation of Knowledge-Based Systems, Montreal* (pp. 15-21). Somerset, NJ: IJCAI, Inc.

Chander, P. G.; Shinghal, R.; Desai, B. C.; & Radhakrishnan T. (1997). An expert system to aid cataloging and searching electronic documents on digital libraries. *Expert Systems with Applications, 12*(4), 405-416.

Cocchi, R.; Estrin, D.; Shenker, S.; & Zhang, L. (1991). A study of priority pricing in multiple service class networks. Retrieved January 11, 1999: ftp://parcftp.xerox.com/pub/net-reserch/pricing1.ps.Z

Crawford, W. (1984). *MARC for library use: Understanding USMARC formats.* Boston, MA: G. K. Hall.

Cromwell, W. (1994). The core record: A new bibliographic standard. *Library Resources & Technical Services, 38*(4), 415-424.

Desai, B. C. (1990). *Introduction to database systems* St. Paul, MN: West.

Desai, B. C. (1994a). *The semantic header and indexing and searching on the Internet.* Retrieved January 11, 1999 from the World Wide Web: http://www.cs.concordia.ca/~faculty/bcdesai/cindi-system-1.0.html.

Desai, B. C. (1995a). *Test: Internet indexing systems vs list of known URLs.* Retrieved January 11, 1999 from the World Wide Web: http://www.cs.concordia.cs/~faculty/bcdesai/test-of-index-systems.html.

Desai, B. C. (1995b). *Report of the Metadata Workshop, Dublin, OH (March 1995).* Retrieved January 11, 1999 from the World Wide Web: http://www.cs.concordia.ca/~faculty/bcdesai/metadata/metadata-workshop-report.html.

Desai, B. C. (1997a). *Test: Internet indexing systems vs list of known URLs: Revisited.* Retrieved January 11, 1999 from the World Wide Web: http://www.cs.concordia.ca/~faculty/bcdesai/revisited.html.

Desai, B. C. (1997b). Supporting discovery in virtual libraries. *Journal of the American Society of Information Science, 48*(3), 190-204.

Desai, B. C., & Shinghal, R. (1996). Resource discovery: Modelling, cataloging, and searching. In *DEXA '96* (Seventh International Workshop on Database and Expert Systems Applications, September 9-10, 1996, Zurich, Switzerland) (pp. 70-75). Los Alamitos, CA: IEEE Press, Zurich, Switzerland.

Gaynor, E. (1994). Cataloguing electronic texts: The University of Virginia library experience. *Library Resources & Technical Services, 38*(4), 403-413.

Giarratano, J., & Riley, G. (1994). *Expert systems: Principles and programming* (2d ed.). Boston, MA: PWS Publishing.

Giordano, R. (1994). The documentation of electronic texts using Text Encoding Initiative headers: An introduction. *Library Resources & Technical Services, 38*(4), 389-401.

Heller, D., & Ferguson, P. M. (1994). *Motif reference manual.* Sebastopol, CA: O'Reilly & Associates.

Horny, K. L. (1985). Minimal-level cataloguing: A look at the issues. *Journal of Academic Librarianship, 11*(6), 332-334.

HyperText Markup Language homepage. (1997). Retrieved January 11, 1999 from the World Wide Web: http://www.w3.org/pub/WWW/MarkUp/.

Kahle, B. (1991). *An information system for corporate users: Wide area information servers* (Thinking Machines Technical Rep. No. TMC-199).

MacKie-Mason, J., & Varian, H. (1997). *Usage-based pricing: Analyses of various pricing mechanisms.* Retrieved September 1997 from the World Wide Web: http://gopher.econ.lsa.umich.edu/EconInternet/Pricing.html.

Mauldin, M. L. (1995). *Measuring the Web with Lycos.* In B. C. Desai & B. Pinkerton (Eds.), *Proceedings of the WWWII Workshop on Web-wide indexing/semantic header or cover page* (Darmstadt, Germany, April 1995) (pp. 26-29). Retrieved August 4, 1999 from the World Wide Web: http://www.cs.concordia.ca/~faculty/bcdesai/www3-wrkA/workshop-a.html

Petersen, T., & Molholt, P. (Eds.). (1990). *Beyond the book: Extending MARC for subject access.* Boston, MA: G. K. Hall.

Qin, J. (1998). *Computational representation of Web objects in an interdisciplinary digital library: A survey and experiment in polymer science.* Retrieved February 11, 1999 from the World Wide Web: http://www-dept.usm.edu//~slis/qin/metadata.

Rhee, S. (1985). Minimal-level cataloging: Is it the best local solution to a national problem? *Journal of Academic Librarianship, 11*(6), 336-337.

Ross, R. M., & West, L. (1985). MLC: A contrary viewpoint. *Journal of Academic Librarianship, 11*(6), 334-336.

Saunders, L. M. (1993). *The virtual library: Visions and realities.* Westport, CT: Meckler.

The Web robots pages. (1996). Retrieved January 11, 1999 from the World Wide Web: http://info.webcrawler.com/mak/projects/robots/robots.html.

Welcome to ALIWEB. (1995). Retrieved January 11, 1999 from the World Wide Web: http://www.nexor.co.uk/public/aliweb/aliweb.html.

Additional References

Desai, B. C., & Shinghal, R. (1994b). *A system for seamless search of distributed information sources.* Retrieved January 11, 1999 from the World Wide Web: http://www.cs.concordia.ca/old/w3-paper.html.

TEI guidelines for electronic text encoding and interchange. (n.d.). Retrieved September 1997 from the World Wide Web: http://etext.virginia.edu/bin/tei-tocs.