# Digital Libraries Based on Full-Text Retrieval

Ian H. Witten, Craig G. Nevill-Manning and Sally Jo Cunningham
Department of Computer Science, University of Waikato,
Hamilton, New Zealand.
{ihw, cgn, sallyjo}@cs.waikato.ac.nz

**Abstract:** Because digital libraries are expensive to create and maintain, Internet analogs of public libraries—reliable, quality, community services—have only recently begun to appear. A serious obstacle to their creation is the provision of appropriate cataloguing information. Without a database of titles, authors and subjects, it is hard to offer the searching and browsing facilities normally available in physical libraries. Full-text retrieval provides a way of approximating these services without a concomitant investment of human resources. This presentation will discuss the indexing, collection and maintenance processes, and the retrieval interface, to public digital libraries.

Creating a publicly-available digital library presents some interesting challenges, particularly if the library is intended for serious professional work rather than casual browsing. First, the raw material: the collection must comprise text that can be placed in the public domain. Second, the selection of material: although a huge amount of public-domain text is present on the Internet, its quality is extremely uneven and only a tiny fraction is appropriate for inclusion in a library. Third, the format of material: it is helpful if the collection resembles the traditional form of typeset pages rather than raw electronic text. Fourth, cataloguing: appropriate bibliographic information in a usable format may be difficult to find and onerous to provide manually. Fifth, information retrieval: a uniform, easy-to-use, publicly-accessible interface is necessary.

There are two essentially opposite approaches to solving these problems. One is to arrange for authors, or their representatives, to contribute information to the library, so that a traditional author/title/keyword catalogue can be built through which readers access the collection. Because authors have a vested interest in seeing their work fully represented and properly catalogued, they will presumably be prepared to invest time and effort to ensure that full information is supplied. An extreme example of this philosophy is the HYPATIA project (`http://hypatia.dcs.qmw.ac.uk`) which is creating a database of people, research interests, bibliography entries, and papers to which people offer information by subscribing individually and by academic department.

The second approach, pursued in this paper, is to gather material from public repositories without any active participation by authors or their institutions. This has the potential to create a vastly larger collection, but requires a new mechanism for searching the library because conventional bibliographic information is not available. An extreme example of this philosophy is the well-known ALTAVISTA search engine which provides a full-text index to a vast universe of indiscriminately-garnered Web pages (`http://altavista.digital.com`).

Although the second approach provides the basis for the search engines that are rapidly becoming our primary access point to the Web, few digital library projects have adopted this philosophy. One that has is the New Zealand Digital Libraries project. This is unique in several respects. First, it provides a full-text index of the entire contents of each document, whereas other digital libraries index on user-supplied document descriptions, abstracts, or other document surrogates. Second, it makes a minimum of assumptions about conventions adopted by document repositories. Other schemes rely for their information on the index file that is present by convention in most directories of technical reports, or on other information provided explicitly for indexing purposes. Third, the work directly addresses the problem of building the library in a geographically remote location with high Internet costs—an environment in which the benefits of networked library technology are especially striking. Finally, the scheme is extremely economical in local disk resources.

Like many other digital library projects, we have chosen the domain of computer science technical reports to exemplify our approach. Much high-quality information already exists in digital form and is freely accessible on the Internet, and because the time value of information is high, computer science already relies more than most on pre-publication in the form of technical reports.

This paper begins by briefly surveying both Internet technical report collections and Web search engines. These exemplify the two different philosophies to collection-building outlined above. We then go on to describe our prototype digital library. We describe the indexing and retrieval mechanism, including the full-text index, and summarize the types of search provided, the indexes that are needed to support them, and the query interface through which they are made accessible. We go on to discuss the process of building the collection, and the crucial issue of what information is stored centrally and what can remain distributed. We then briefly summarize how the information is gathered, consider the factors that limit the size of the library, and review future directions for networked digital libraries.

## Internet libraries for technical reports

Several technical report searching and indexing systems have been developed in the past; some are summarized in [Tab. 1]. The physics e-print archive, operating since 1991, has already supplanted journals and pre-print mailings as the primary information dissemination point for several areas of physics. Documents are submitted by e-mail in TeX format, along with bibliographic information which is used for indexing. NASA has a large index to publications since 1962, but most references are to documents that are only available in physical form.

Computer Science is well endowed with indexes. The HARVEST system contains a large number of documents, many of which are Web pages rather than technical reports. The documents are indexed on limited information extracted from the original files. UCSTRI allows searching of files of abstracts, which often appear in technical report FTP archives. While this information is limited in scope, the collection includes 185 sites and 14,000 documents. The DIENST and WATERS systems have been superseded by NCSTRL (see below). DIENST is a distributed library architecture, developed in a project involving several universities, that originally accommodated scanned images and indexed on text produced by OCR. WATERS included technical reports from several universities and provides bibliographic searches based on information provided by participating departments The Karlsruhe bibliography service indexes a large collection of computer science reference lists.

Recently, the NCSTRL (for "Networked Computer Science Technical Reports Library") project has arisen as a development of the DIENST and WATERS systems. NCSTRL is a collection of computer science technical reports made available for non-commercial and educational use. There are two levels at which organizations can participate. "Standard" participants run indexing, searching, and user interface software locally, whereas "Lite" participants maintain an ordinary FTP site of technical reports and supply bibliographic information to a special server. Lite connection is easiest but Standard connection allows a site to maintain its own search engines, to provide access to documents in multiple formats, and to provide customized user interfaces—thumbnail browsing, page zooming, page selection, support for logical document structure, etc. NCSTRL currently (July 1996) has 55 participants, of which nearly two-thirds are Lite. Standard NCSTRL sites use the DIENST software, an open, distributed, implementation of digital library servers. DIENST provides three basic services at each site: a repository, indexes and a search engine, and a user interface. The repository can hold documents in different formats (PostScript, TIFF, GIF, HTML) and can handle the same document in multiple formats. Sites can use home-grown software for any of these components. For example, they can supply their own browsers (and some do). DIENST makes a distributed library appear as a single collection at the top level, even though the repositories, search engines, and browsing interfaces are distributed.

| **Name** | **Domain** | **Reference** | **Sites** | **Documents** | **Indexed on** |
|---|---|---|---|---|---|
| e-print archive | Physics | [Ginsparg 1994] | | several million | user-supplied title, author, abstract |
| NTRS | NASA | [Nelson *et al*. 1994] | | 3 000 000 | abstracts |
| HARVEST | CS | [Bowman *et al*. 1994] | 380 | 37 000 | limited info from several file types |
| UCSTRI | CS | [Van Heyningen 1994] | 185 | 14260 | file of abstracts at ftp site |
| DIENST | CS | [Davis & Lagoze 1994] | 15 | – | bibliographic information provided |
| WATERS | CS | [Maly *et al*. 1994] | 15 | 2 000 | citation and possibly abstract |
| NCSTRL | CS | `www.ncstrl.org` | 55 | – | bibliographic information provided |
| Karlsruhe | CS bib | `liinwww.ira.uka.de/bibliography` | | 560 000 | full text of bibliographic entry |

**Table 1:** Technical report indexes on the Internet

## Web search engines

At the other end of the spectrum from contributory technical report libraries are search engines that seek and index Web pages. Typically, they permit a full-text search on the contents, or partial contents, of Web pages and some of the files that they point to—although most non-HTML file types, including PostScript, are excluded.

It is interesting to trace the historical development of such systems. In the pre-Web era, a system called VERONICA claimed virtually complete coverage of the documents pointed to by the GOPHER menu structure—although no attempt was made to index the document text. Early Web indexes such as ALIWEB were based on specially formatted submissions from service providers. JUMPSTATION provided three separate indexes to HTML pages: one for titles, a second for headings, and the third for the most frequent words in the text. The LYCOS system (`www.lycos.com`), at least in its original version, restricted the text searched to the first few lines of each page, and certain high-information-content keywords that appear later in the page. By 1995 LYCOS had encountered several million pointers to Web pages, but 75% of these were unexplored; at that time new documents were being indexed at the rate of 15,000 per day.

Although these early Web indexes were academic efforts, most present-day search engines are either funded by advertising or created by commercial firms that want to advertise their services and products in the information retrieval area. ALTAVISTA, for example, is a high-profile operation of Digital Equipment Corporation, and is exceptionally well-resourced to ensure that it provides impressive performance. Other search engines offer advertising, and in some cases it has been possible for Web page owners to pay to ensure that their pages figure prominently in the information retrieved by searches. Thus the indexes are competitive in nature; consequently all claim to provide the best service and it is very difficult to say with any accuracy how many pages they index—all tend to mention figures of around 50 million Web pages.

INFOSEEK (`www.infoseek.com`) and WEBCRAWLER (`webcrawler.com`) are commercial systems that resemble LYCOS, although at least at the beginning they offered considerably less coverage. OPENTEXT (`www.opentext.com`) is a more recent addition to the field. It offers ranked, Boolean and proximity searches that can be confined to summaries, titles, first headings, links, and body text. A year ago its index covered around one million Web pages, and around 50,000 pages per day were being added. ALTAVISTA is another, more recent, large index. It offers a basic ranked search facility in which one can additionally specify that certain terms are mandatory and others are prohibited. Terms can take the form of words, phrases and partial words. An advanced search facility allows Boolean expressions of search terms as well, separating the query specification from the way in which the results are to be listed. Finally, the EXCITE search engine site (`www.excite.com`) is eager to announce that its index is 50% bigger than its nearest competitor, that its indexing technology produces more relevant hits, and that its speed is unmatched by other search facilities.

## The NZ digital library for computer science research

The New Zealand Digital Library for Computer Science Research is essentially an application of Web indexing technology to a digital library system. It incorporates features of both approaches, combining the unobtrusiveness of Web search engines with the selectivity afforded by confining search to a limited set of information archives. It currently provides access to 22,000 research documents on 250 sites worldwide (550,000 pages, 220 million words). This corresponds to 17 Gb of PostScript files containing technical reports in the computer science area; some relevant statistics are given in the upper block of [Tab. 2]. The library went on-line in May 1995 with an initial collection of approximately 2000 documents. Although until recently it has been publicized only within New Zealand universities, the library has been accessed from over 700 sites worldwide. The current implementation is clearly both useful and seeing use, and user feedback is overwhelmingly positive.

## Indexing and retrieval

The key innovation in this project is that it dispenses with the traditional library catalogue, which relies on information about each item in the collection being supplied along with the item itself. If information providers must supply such information, the scope of the collection is thereby restricted. The alternative, cataloguing by library staff, means that the collection is limited by human resources. It was resolved that although the collection would be formed entirely from publicly-available repositories of text, the system should not require any effort on the part of participating repositories; indeed there should be no need for these information providers to be aware of their inclusion in the index. No special software, archive organizations, or file formats are required of the providers. The only information used for cataloguing is derived from the documents themselves.

**Full-text index**

These considerations led to the use of a full-text index instead of a library catalog as the main retrieval mechanism. The index makes the entire text of all documents available for retrieval, rather than some much more restricted list of keywords as is the case in many computer-based text retrieval systems. From the point of view of building the system, this has the advantage that nothing more is needed to construct the index than the plain text of the documents in the library: it is a way of finessing the usual requirement for traditional bibliographic database information such as author, title, publisher, and so on. From the point of view of the user, it provides a powerful tool for searching for information: we consider below whether this can supplant the more traditional forms of library access—by author, title, date, subject, and so on.

The search engine for the library is the public-domain system MG. Tailored for highly efficient storage of full-text databases, MG can pack an index to a large collection of text into only 5% of the size of the original text [Witten *et al.* 1994]. Further, it responds rapidly to queries: experiments with the 750,000 document TREC collection produce ranked output for queries of forty to fifty terms within three to five seconds.

**Types of search**

MG supports the usual Boolean and ranked keyword searches over the full text of the document. In the digital library, different kinds of search are implemented as follows.

*Author/title*. In most reports, the first page gives bibliographic information such as title and author. By limiting attention to this page, the user can approximate a search based on such information. For example, an initial page search for documents authored by *Knuth* will not retrieve documents that merely cite his work.

*Publication date*. Most reports also include publication date on the initial page, and the same technique serves for publication date searches. Alternatively, this facility could be simulated by permitting the user to search on the date in which a technical report was entered into its repository, although such a strategy is likely to produce uneven results because some reports are placed in repositories long after they were originally produced.

*Page searching*. The digital library stores the full text of technical reports, and supports searching over the complete document text. This is very useful in performing very general searches with high recall, that is, ones that retrieve a high proportion of the relevant documents in the collection. However, a large number of irrelevant documents ("false drops") can be expected as well. For example, in a search for *information retrieval*, many

| | | *total* | *per report* |
|---|---|---|---|
| **Source documents** | **Sites** | 250 | |
| | **Documents** | 22 000 (550 million words) | 10 000 words |
| | | 550 000 pages | 25 pages |
| | **Original PostScript** | 17 Gb | 800 kb |
| | *compressed with gzip* | *6 Gb* | *270 kb* |
| **Extraction process** | **Text** | 1.3 Gb | 60 kb |
| | *compressed with gzip* | *430 Mb* | *20 kb* |
| | **First page facsimiles** | 29 000 pages | 1.5 pages |
| | | 300 Mb | 14 kb |
| **Indexed collection** | **Text** | 375 Mb | |
| | **Index** | 300 Mb | |
| | **First-page images** | 305 Mb | |
| | **Total** | 980 Mb | |

**Table 2:** Current size of the New Zealand digital library for computer science research

technical reports will contain these terms somewhere in the document, perhaps widely separated. The greater the physical distance between the two words, the less likely that the document is about the subject *information retrieval*. A limited kind of proximity searching is supported by page-level indices, which require the query terms to appear on the same physical page of the document. A still higher degree of precision can be achieved by specifying an exact co-location of words in a phrase. Phrase searching is implemented within MG by post-processing query results; a string search for phrases can be performed on documents returned by any query.

*Case folding/truncation/exact match*. Retrieval systems commonly allow users to decide whether or not query terms should be exactly matched in the document returned. Exact matches are useful, if not crucial, in constructing some types of search: for example, in locating the author *Gray* rather than the color *gray*, or for finding documents about *NeXT* computers or the *SMART* system. On the other hand, case folding helps to avoid artificial distinctions between capitalized and non-capitalized forms of the same word (such as *Information* and *information*) or between different forms of a word (*retrieval*, *retrieve*, *retrieving*). These truncated terms of the root word should be sought automatically, without forcing the user to list all alternatives. The digital library allows the user to alternate between these types of search by selecting either exact match or case folding/stemming.

### Query interface

The World Wide Web is the access medium for the digital library. [Fig. 1] shows the query page and a typical response. Because of the Web's page-oriented nature, the document list, query options, and returned documents are placed on separate pages. The maximum number of matches returned for a ranked query is a potential transmission bottleneck, for it is easy to issue a query that returns many thousands of documents. This is initially set to 40, which gives good performance over the Internet, but can be increased by the user. Because WWW transactions are stateless and the user is unknown to the server, it is impossible to provide services such as remembering queries, displaying a history, or saving profiles, without requiring users to identify themselves explicitly.

It is clearly necessary to share the query engine between users. MG was not designed to deal with simultaneous multiuser access, and while it is possible to launch a process for each query, the startup overhead makes this very inefficient. Consequently we have arranged for access to one persistent copy of the query engine to be shared between incoming queries. One copy runs continually, and additional software provides exclusive access for the duration of each query.

## Building the collection

The lowest common denominator for representing information in conventional libraries is paper, and many digital library efforts involve scanned paper documents (e.g. [Crocca & Anderson 1995; Van House 1995]). In the world of electronic information, PostScript, rather than plain, unformatted text, is the closest analog to paper

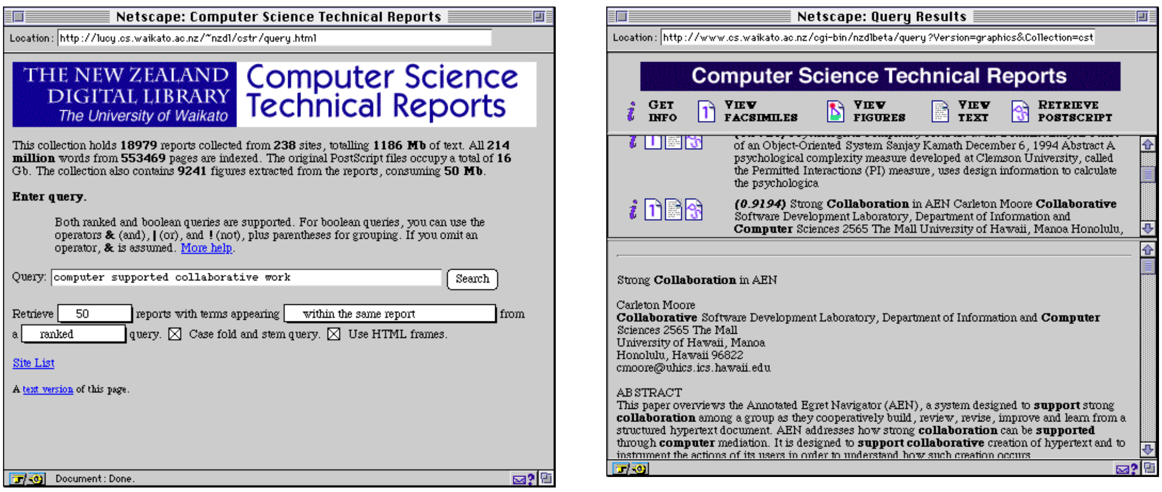


**Figure 1:** Interface to the digital library: the query page and a typical response

as a document storage medium, and its page-based nature turns out to be helpful in structuring the collection. In order to build the index, it is necessary to be able to extract plain, unformatted, text automatically from the documents. In fact, the system design accommodates not just PostScript but any format from which paginated ASCII text can be extracted—for example DVI, RTF or HTML files. Document images can be accommodated by OCR-ing them for indexing purposes: the inevitable recognition errors will reduce the quality of the index, but this can be ameliorated by using ranked queries containing redundant terms. However, from an initial investigation it appeared that PostScript files are almost universal in computer science technical report archives, and the system currently deals only with this format.

Archives of technical reports can be located through several lists maintained on the Internet, and recursively descending the directory hierarchy looking for (possibly compressed) PostScript files. Each file is downloaded, along with its size and date, and the appropriate information is extracted.

**Information stored centrally**

With an Internet-based digital library, a crucial question is how much information to store centrally. It was decided that the library would comprise an index and search engine, and the documents themselves would remain in their original repositories. Periodically, all repositories are scanned to refresh the index, adding any new documents and removing references to those that have been deleted. There is no intention to provide an archiving service that will retain documents that are removed from their original sites.

In addition to the index, a facsimile image of each document's first page or two is retained locally so that users can read the title and abstract, and sample the look and feel of the original. The plain, unformatted, text must be extracted from the documents to build the index, and it proves expedient to retain a full copy of this text locally as well. This is useful in its own right for browsing: users can examine the text without going to the trouble of

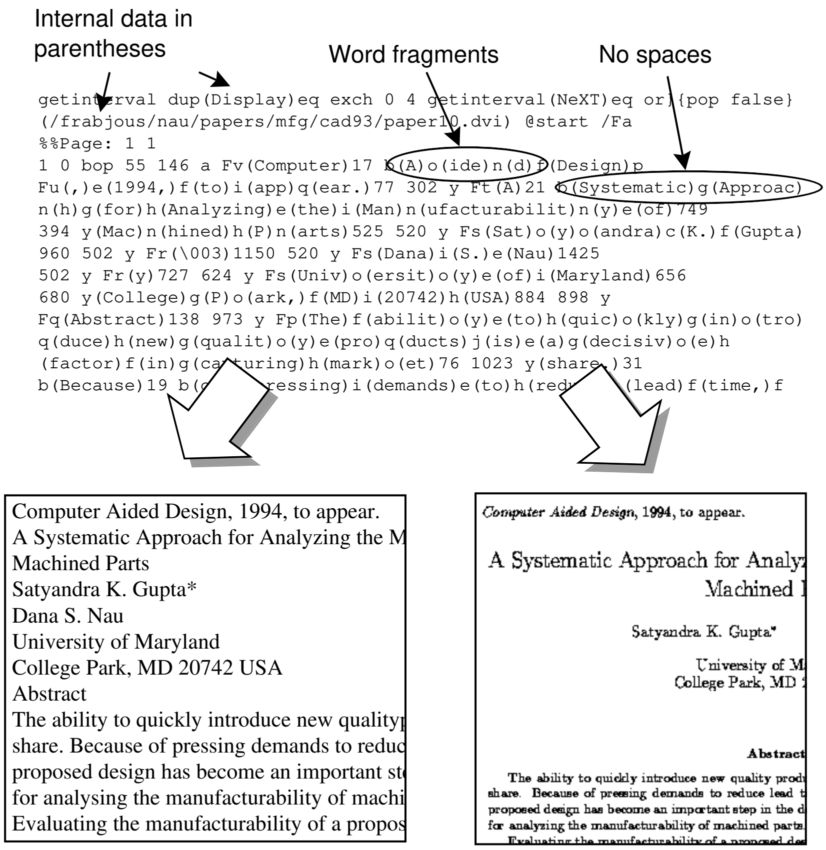


**Figure 2:** Conversion from PostScript: a PostScript file, the text extracted from it, and a facsimile image

downloading PostScript. Moreover, the text of the collection provides an excellent foundation for bibliometric research. [Fig. 2] illustrates (at the top) an original document in PostScript form, and (at the bottom) the two files extracted from it: on the left the full ASCII text, and on the right a facsimile image of the first page.

**Text extraction**

Although the words of a technical report usually appear as plain text within a PostScript file, they are thoroughly intermixed with PostScript language commands and internal data. Words appear within parentheses, but so does internal information such as font names and error messages. Spaces are not explicit, but are coded implicitly in terms of the placement of words on the page. Finally, whole words are not always bracketed together: to give greater control over spacing, word fragments are often placed individually. For these reasons, text cannot be extracted reliably by syntactic analysis of the PostScript file. [Fig. 2] illustrates some of these problems.

The solution is to prepend a PostScript prologue that redefines the operators responsible for placing text on the page, and execute the resulting file using a PostScript interpreter. The redefined operators write the text fragments to a file, taking note of their position on the page in order to insert spaces and new lines where appropriate. This technique is based on part of the GhostScript distribution (the *ps2ascii* program), but is simpler and far more robust in terms of the variety of PostScript files it can handle. In fact, *ps2ascii* failed on 75% of 144 reports chosen at random (each from a different site), while another converter, that used in the Harvest system [Bowman *et al.* 1994], failed on 40%.

[Fig. 3] shows the a simplified version of the PostScript prologue used for this process. It extracts not only the body of the report, but also any text in figures and tables. Although an absolute specification of the minimum inter-word spacing (5 points) is shown, the actual implementation sets this threshold by analyzing the report itself. Moreover, although for indexing purposes only the stream of words needs to be extracted, the Digital Library provides a HTML approximation to the original document for browsing purposes. We have developed simple heuristics to recognize paragraph boundaries, and to solve other practical problems with the output of the extraction process—such as dealing with non-ASCII characters, dehyphenation, and page reversal in case the PostScript file holds the document back to front (these situations are not easy to detect).

**First-page facsimile**

The facsimile image of the first page shows the user the beginning of the document as it actually appears, complementing the raw text of the full paper. This provides an analogous service to a library supplying users with fax copies of the first page or two of a document.

One approach is to store the PostScript version of the first page. However, PostScript documents invariably contain a lengthy prologue that defines fonts and special functions. This prologue is essential to interpret the first page, and it turns out that in most cases the size of the file required to reproduce the first page is nearly as large as the entire document. Consequently the facsimile is stored as a screen-resolution bitmap, so that the file size is bounded by the size of a standard page. The facsimile images are produced by a PostScript interpreter as a 75 dpi bitmap, and are saved as graphics files in GIF format.

The first page of many technical reports shows little more than an institutional logo along with the title and authors' names. The second page may also give little information. In order to ensure that a page with some content is imaged, the information in a page is estimated by the size of the GIF file that represents it. Because these images are compressed, pages with large areas of white space and big letters compress well, to a few kb, while pages full of actual text occupy over 10 kb. The extraction system renders pages in sequence until the sum of the file sizes exceeds this figure, a simple heuristic that seems to work well in practice.

| | |
|---|---|
| `/show {` | *redefine the show operator to be:* |
| `  currentpoint pop` | *get the current x coordinate* |
| `  X sub 5 gt { ( ) print } if` | *output a space if x has increased by at least 5 points* |
| `  dup print` | *output the string* |
| `  systemdict /show get exec` | *execute the original show to update the current point* |
| `  currentpoint pop /X exch def` | *record the new x position* |
| `} def` | |

**Figure 3:** The prologue used to extract text from PostScript documents

### Collection maintenance

All that is necessary to maintain the library's collection is to ensure that the documents indexed continue to exist. This can be checked by periodically examining the technical report repositories for changes and updating the collection accordingly. Considerable benefit is obtained by not relying on any cataloguing information stored with the repository itself: for example, frequent maintenance problems in technical report servers are caused by changes in the bibliography file format, and inconsistencies in the information, in the repositories that they index [Van Heyningen 1994].

One source of growth for the collection is when new documents in known repositories are located during a routine examination of currently indexed sites. New sites can be detected by various means: monitoring standard Internet lists for new additions; manually scanning the newsgroups that announce them; and encouraging users to email suggestions to a central coordinator.

## Gathering the Information

One of the motivations for a New Zealand digital library is efficient use of the expensive transpacific Internet link. Computer scientists can search for and preview technical reports locally before downloading the full document file, thus encouraging exploration without concern for network charges. However, to build the full-text index it is necessary to examine the contents of each report. Transmitting all of them across the Pacific would negate any cost benefits the project might offer.

Consequently a distributed scheme is used to create the index. In the first stage, a computer in North America downloads each technical report, extracts the facsimile images and raw text, sends them to New Zealand, and deletes the report. This process is illustrated in [Fig. 4].

This operation significantly reduces transmission cost on the Pacific link. The 17 Gb of uncompressed PostScript currently indexed by the library is actually stored in repositories in compressed form, occupying approximately 6 Gb. In this form, each report takes about 13 seconds to download to the North American site, whereas it would take have taken about 36 seconds to download to New Zealand because of the low bandwidth of the Pacific link. In actuality, only 1300 Mb of uncompressed text, which reduces to 430 Mb when compressed, need be transmitted, which takes about 4 seconds per report. Added to this is the cost of transmitting facsimile images: 305 Mb in total, or another 4 seconds per report.

## Size of the library

The lower block of [Tab. 2] summarizes the size of the document collection and the storage requirements for the MG indexes that comprise the library. The total space occupied by all indexes is 300 Mb, a mere 2% of the total size of the PostScript files that are indexed. The text of the technical reports occupies 375 Mb when compressed by MG. Finally, the first-page facsimiles occupy an additional 305 Mb, giving a total disk storage requirement of

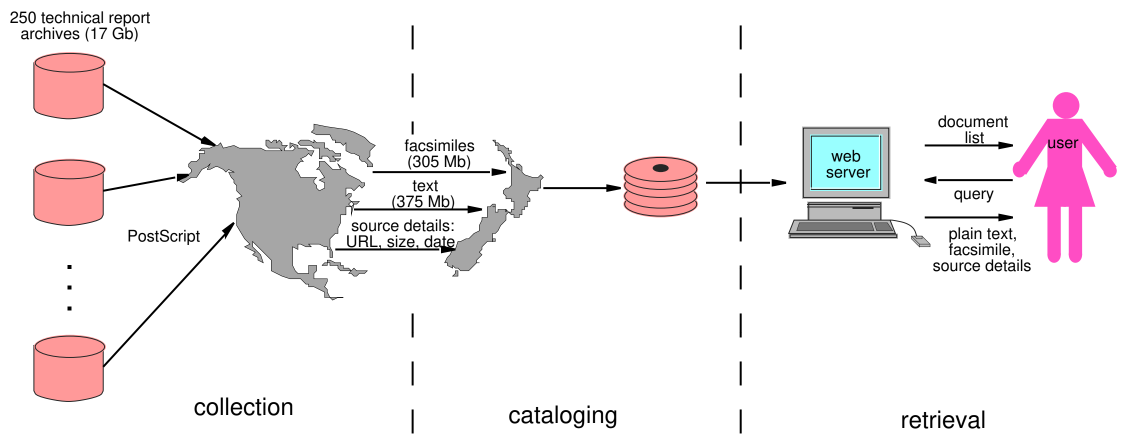


**Figure 4:** Architecture of the digital library

just under 1 Gb, 6% of the total PostScript size.

What are the limits to the size of a library of this nature, using current technology? All sizes involved increase linearly with the volume of text, and a factor of ten growth would only involve a few Gb of disk storage. Retrieval time is virtually independent of database size: it takes two disk seeks per query term and two per document retrieved. The database inversion process is more likely to be a limiting factor. An improved algorithm has recently been reported that is faster and requires less main memory than MG: extrapolating from experiments with a 2 Gb text collection it is estimated that 5 Gb of raw text can be inverted in twelve hours with only 40 Mb of main memory [Moffat & Bell 1995]. This corresponds to growth of the digital library by a factor of four. It provides an indication of what is certainly possible, although there is no reason to believe that it represents a limit. The resulting collection would index 80,000 documents comprising two million pages or 45 Gb of PostScript, and the size of the centralized information, including plain text, indexes, and first-page facsimiles, would amount to just 4.5 Gb.

Nevertheless, an all-inclusive information collection policy is basically unscalable and will become infeasible as the number and size of technical report repositories grows. There are several possibilities for culling the collection when that becomes necessary. One is to monitor participating users' access, and note the repositories that receive the least use. This regulates the growth of the collection by the size, diversity, and level of activity of the user population rather than by the growth of the bibliographic universe. Another is to use the characteristics of the documents themselves to determine what to cull from the collection. For example, citation studies could identify documents that do not appear to contribute to the literature.

## Future Directions

The operation of the digital library hinges on the idea of extracting information automatically from PostScript. Presently, words and page boundaries are extracted, and the process is entirely algorithmic. The collection would be enhanced if reliable ways could be devised for determining bibliographic details of the reports so that a proper catalogue could be built, rather than relying on their appearance on the first page (which may be a cover page). Moreover, the text collection provides an ideal basis for bibliographic studies in computer science. If it were possible to locate bibliographic reference sections within documents, a separate database could be constructed that would be extremely useful for citation analysis. Hence we need to consider how to extend the extraction process by incorporating heuristics to identify some document structure.

Users find it hard to choose between document- and page-level retrieval because it is not clear what the basis of the choice should be. It may be that the page-level index suffices for all retrieval, and the document-level one should be abandoned. Perhaps the problem can be solved by a more sophisticated information display mechanism: for example, TileBars [Hearst 1995] provide a way to visualize the distribution of query terms throughout a document that should be easy to integrate into our system. A number of other searching mechanisms would be very useful. Mixed Boolean and ranked queries would allow users greater control over the documents that are retrieved. Browsing by location needs some human pre-processing to identify machine locations by institution rather than by network address. Even simpler, and also useful, would be to allow users to look at the directory that a particular report comes from, because sometimes multipart documents are stored as separate files. Finally, co-location queries allow users to examine which words occur together; this is useful for textual analysis purposes.

## Conclusion

We have distinguished two opposing philosophies for the construction of networked digital libraries. One is to have contributors supply bibliographic details for the information that they place in the collection. The other is to garner the collection automatically and use a full-text index for access instead of an ordinary library catalogue.

This paper has focused on the second approach. With it, public digital libraries can be constructed on the Internet entirely automatically, in any area for which repositories of suitable text can be located. Extracting all index information from the documents themselves is feasible if full-text indexing is used, and eliminates the manual cataloguing effort involved in creating and maintaining the library. The material in such a library can be distributed globally. The central database in our prototype library, which comprises a full-text index, facsimile images of the first page or two of each document, and the plain text of all documents, represents only 10% of the collection size.

Looking into the future, it is likely that a new generation of digital libraries will marry the two approaches. The first offers high-quality cataloguing information, while the second provides significantly increased coverage. Improved techniques for information extraction from text, along with large public-domain bibliographies, offer the possibility of being able to match reports in a collection with items in a bibliography file, thus providing cataloguing information at no additional cost. Moreover, personal contributions to people-oriented research databases like Hypatia are likely to provide more authoritative reference information than do general bibliographies, so bibliographic quality can be increased by amalgamating information from different sources, tagged according to likely reliability.

Finally there is the problem of distribution. Architectures like HARVEST and NCSTRL provide a distributed infrastructure that underpin all aspects of the collection. However, there is a danger in being *too* distributed: whereas users want to see a unified system, these schemes allow sites to provide their own browsing software through which their repository must be viewed, and this nonuniformity can be a great annoyance in practice. Web search engines, in contrast, are not distributed, though they may involve multiprocessors accessing the same database in order to provide adequate power. Again we will probably see an amalgamation of the two approaches, and indeed distributed indexes expressly designed for multi-collection environments are a current research topic in the information retrieval community.

Digital libraries represent one way of dealing with the new reality of Internet publishing. Making a minimum of assumptions, a library based on full-text retrieval imposes structure on a fundamentally anarchic, uncatalogued, system, giving information consumers a tool to find what they need.

## References

[Bowman *et al*. 1994] Bowman, C.M., Danzig, P.B., Manber, U., and Schwartz, M.F. (1994). Scalable Internet resource discovery: Research problems and approaches. Communications of the ACM, 37 (8), 98–107.

[Crocca & Anderson 1995] Crocca, W.T. and Anderson, W.L. (1995). Delivering Technology for digital libraries: experiences as vendors. Proc. Digital Libraries '95, 1995, Austin, TX. 1–8.

[Davis, & Lagoze 1994] Davis, J. & Lagoze, C. (1994). A protocol and server for a distributed digital technical report library, Technical Report 94-1418, Computer Science Department, Cornell University.

[Ginsparg 1994] Ginsparg, P. (1994). First steps towards electronic research communication, *Computers in Physics 8*(4), 390–401.

[Hearst 1995] Hearst, M.A. (1995). TileBars: visualization of term distribution information in full text information access. Proc. CHI'95, 56–66.

[Maly *et al*. 1994] Maly, K., Fox, E.A., French, J.C., and Selman, A.L. (1994). Wide area technical report server, Technical Report, Dept. of Computer Science, Old Dominion University.

[Moffat & Bell 1995] Moffat, A. and Bell, T.A.H. (1995). In situ generation of compressed inverted files. J American Society for Information Science, 46 (7), 537–550.

[Nelson *et al*. 1994] Nelson, M.L., Gottlich, G.L., and Bianco, D.J. (1994). World Wide Web implementation of the Langley Technical Report Server, NASA Technical Memorandum 109162, Hampton, Virginia.

[Van Heyningen 1994] Van Heyningen, M. (1994) The Unified Computer Science Technical Report Index: Lessons in indexing diverse resources. Proc. Second International WWW Conference, Chicago.

[Van House 1995] Van House, N.A. (1995) User needs assessment and evaluation for the UC Berkeley electronic environmental library project: a preliminary report. Proc. Digital Libraries '95, Austin, TX. 71–76.

[Witten *et al*. 1994] Witten, I.H., Moffat, A., and Bell, T.C. (1994) Managing Gigabytes: Compressing and indexing documents and images. New York: Van Nostrand Reinhold.

**Acknowledgments:**

Thanks to Todd Reed, Mike Vallabh and Brent Summers for technical assistance, and to Tim Bell and Mark Apperley for encouragement and advice. Bruce McKenzie provided the prototype WWW interface to MG, and the Computer Science Department at Calgary generously served as our North American site. This research was supported by the NZ Foundation for Research, Science and Technology and by the NZ Lotteries Grants Board.