NCSTRL: Design and Deployment of a Globally Distributed Digital Library

NCSTRL is the largest distributed digital library on the Internet.
We describe how it grew and the lessons we learned by growing it.

James R. Davis
Xerox Corporation
Carl Lagoze
Department of
Computer Science
Cornell University

The World Wide Web provides unprecedented access to globally distributed content. The extent and uniform accessibility of the Web has proven beneficial for research, education, commerce, entertainment, and numerous other uses. Ironically, the fact that the Web is an information space without boundaries has also proven to be its biggest flaw. Key aspects of libraries such as selectivity of content, customization of tools and services relative to collection and patron characteristics, and management of content and services are noticeably absent.

Over the past four years we have been researching the technology and deployment of a digital library architecture that makes it possible to create managed information spaces, *digital libraries*, within the World Wide Web. Our work has taken place in the context of NCSTRL, the Networked Computer Science Technical Research Library (pronounced "ancestral"), a digital library of computer science research reports. The technical foundation of NCSTRL is Dienst, a protocol and architecture for distributed digital libraries that we developed as part of the DARPA-funded Computer Science Technical Reports Project (CSTR). At the time of the writing of this paper, the NCSTRL collection consists of papers from over 100 research institutions residing in servers distributed across the United States, Europe, and Asia. In addition, the Dienst protocol and implementation has been successfully adopted by a number of other distributed collections.

In this paper, we review our experiences with NCSTRL and Dienst, describe some of the lessons we have learned from the deployment experience, and define some directions for the future.

Dienst - A Protocol and Architecture for Distributed Document Libraries

Our work on digital library architecture is based on the following principles:

- *Open Architecture* - Following well-known software engineering principles, the functionality of a digital library system is available in the form of distinct functional units, each of which has operational semantics exposed through an open protocol.
- *Federation* - Digital Library Systems are compositions of these functional units (or services). New functionality can be added to systems through the implementation of value-added services, which interact with existing services using established protocols.
- *Distribution* - The components (and content) of a digital library may be spread over the global Internet, but presented to the user as a single uniform system.

Dienst, an architecture and protocol for distributed document libraries, is a working demonstration of these concepts. The architecture combines the following features. First, it specifies the operational characteristics of *core digital library services*. Second, it has a structured *document model.* Third, it mandates an open, extensible *protocol* for communicating with digital library services and accessing these documents. Finally, it provides a mechanism for definition and *administration of a distributed collection*. These features are described in more detail in [1]. We summarize them briefly here and describe how they are deployed in NCSTRL.

The Dienst architecture specifies four core digital library services. *User interface services* provide a human-friendly gateway to the information obtained from other services. *Repository services* store and provide access to documents, according the Dienst document model, which is described below. *Index services* provide search capabilities, accepting a query and returning a list of document identifiers that match the query. *Collection services* define the components, services and documents, of the digital library collection, making it possible for user interface services to interact with them.

Of these four services, only the first is used directly by a human, via a World Wide Web browser. The others are used by programs, in particular Dienst services, but also by other digital library or publishing systems. This modular design allows easy integration of higher-level digital library services with existing Dienst services, or evolution of existing services as the architecture matures.

In the Dienst document model, each document has a globally unique identifier, a kind of Uniform Resource Name, or URN. (A URN differs from the more-familiar URL by being *location independent*. Where a URL includes a path on a specific server host and port, a URN is merely a name.) Dienst uses the CNRI handle system (http://www.handle.net) for its identifiers. A handle has two components: a *naming authority* that is unique to each institution or publisher, and a *document name*, which is assigned by the naming authority.

A document is not only location-independent, it is also format-independent. Each Dienst document is available in multiple *representations*, which are alternate formats with different expressive capacity, e.g. PostScript, TIFF page images, etc. In addition, each document is available in multiple *decompositions*, such as physical pages or those associated with the logical structure of the document.

Clients use Dienst services via the Dienst protocol. The protocol is defined as a set of service requests, or *verbs*, each of which communicates with a specific service. For example, the `SearchBoolean` verb of the Index service accepts a query specification and replies with the handles of documents that match the query. The `Disseminate` verb of the Repository service accepts a document specification (a named representation such as `Postscript` and a decomposition, such as `chapter 3;page 4`), and returns the dissemination that matches that specification.

Figure 1 shows how Dienst services cooperate, using the Dienst protocol, to allow discovery of and access to Dienst documents in repositories. A user connects to a user interface (UI) service (of which there may be many), and issues a search request, specifying search keys such as title or abstract words, optionally joined by Boolean operators. The UI service sends a search request to a set of index servers (actual routing depends on characteristics of the query). Each index server returns a list of handles for documents matching the query, then the UI server formats and presents them to the user as a hypertext list of links. The user can then select a document, and a request is then sent to the appropriate repository to transmit a dissemination of the document for presentation by the user interface.

The Dienst architecture includes a *collection service*, which provides information on the services and content that are part of the NCSTRL collection. In Dienst we adopt the definition that content is *in* a digital library's collection if it can be directly *discovered* using the resource discovery tools (index services) defined by the library. The collection service implements this concept by providing the following information via protocol requests:

- The list of organizations that are part of the collection
- the network location(s) of index servers that store indexing information for each organization, and
- meta-information about each of the index servers that aids in routing queries.

In operation, each user interface server periodically queries the collection service, and uses the information returned to route queries to appropriate index servers. Figure 2 illustrates this interaction.

The growth of NCSTRL outside the United States raised reliability and performance problems due to connectivity characteristics of the global Internet. To obtain good performance, we defined a set of *connectivity regions*[2], which are sets of servers (network nodes) with relatively good mutual network connectivity. Indexing information from servers outside the region is replicated onto servers within the region. Regional query routing is implemented by *collection views*, collection metadata customized for a specific regions, and *regional collection servers*, which distribute a collection view for their respective region. Each user interface service is then assigned, at configuration time, to a regional collection server so that its queries are routed within the connectivity region.

The collection service also provides a mechanism for managing membership in the NCSTRL collection. The process of joining NCSTRL is as follows. A site applies by filling out a form at the NCSTRL home site. An applicant site must provide an administrative contact, state its policy on copyright, and agree to maintain their collection in a reasonable manner. This form goes to the NCSTRL gatekeeper, who verifies that the institution conforms to our membership policy, and then assigns them a handle naming authority. Meanwhile, the site downloads and installs the NCSTRL software, and creates bibliographic records for their collection. When the software is working, an NCSTRL administrator tests it remotely, and, if found good, adds the site to the Central Collection Server. This information is then, in a short time, downloaded by the regional collection servers, and subsequently to the user interface services. Within an hour, the new site is available to users through all NCSTRL user interfaces.

How NCSTRL Grew

Approximately three years after its inception, the NCSTRL collection contains about 22,000 documents from 118 different institutions. This section discusses how the collection grew, then mentions some of the problems we faced in making it grow.

History

The roots of NCSTRL lie in two earlier projects, the above mentioned CSTR project and the NSF sponsored WATERS[3] (Wide Area Technical Report Service) project. The CSTR project had two aims: to make the technical reports of five universities accessible online, and to carry out research in digital libraries using this corpus. This double focus on both creating a working collection for daily use while simultaneously supporting research using that collection gave rise to occasional tensions. It required us to take pains to keep the system reliable even as it grew and changed. These tensions continued into the NCSTRL project, as we shall see.

Dienst built upon an earlier result from the CSTR project, a standardized format for exchange of bibliographic records (RFC1357, later replaced by RFC1807[4]). CSTR participants considered and rejected the BibTex and Refer formats, even though they were widely known to the computer science community, because both lacked key fields deemed necessary (e.g. ACM Computing Reviews categories, grant numbers) and neither was easily extended to add them. The USMARC format was considered and discarded because its great complexity allows it to be used only by professional catalogers. Also, there was, as of then, no organized effort to develop network descriptive metadata standards for electronic resources, such as the Dublin Core[5] (with which RFC1807 has a number of similarities).

We began working on Dienst in the fall of 1993, and began installing it at the various participant sites. As we gained experience with it, it became apparent that Dienst could run equally well at non-CSTR sites, and that the distributed collection thus created would be even more valuable for it. By May of 1994, Dienst was running at all five CSTR sites, and at Princeton and Dartmouth.

Through most of 1994, the CSTR and WATERS projects operated independently of one another. By December of 1994 it became clear that the computer science community was ill served by having two projects in seeming competition, and the two projects met to discuss a merger in April 1995. A technical solution was developed, known as NCSTRL *Lite*, that allows sites to contribute to the collection without running a full Dienst server. A Lite site stores its metadata on an FTP site and makes its documents available via FTP or HTTP. A dedicated Lite *Gateway* site periodically polls this metadata and indexes it, thus making the Lite collection searchable to other Dienst sites. The gateway also translates Dienst document requests into the appropriate FTP or HTTP request to the Lite site. New NCSTRL institutions choose which version of NCSTRL to operate. Figure 3 illustrates the growth of both NCSTRL Standard and NCSTRL Lite sites.

From its beginnings, the NCSTRL project attempted to identify and remove the obstacles to growth of the collection. The first two were uncertainty and ignorance. Some sites were uncertain which project might "win" in the long run, and hence were reluctant to join either. Many others were simply unaware of the effort or the benefits of participating. We removed the first simply by merging, and the second through a publicity and outreach campaign. We invited the Computing Research Association (CRA), the umbrella organization for academic computing research in the USA to our founding meeting, and in July we asked them to endorse the project to its member institutions. We also planned for a series of demonstrations at general computer science conferences such as the ACM 50th anniversary computer science conference.

Our project became international in August of 1995, when the European Research Consortium for Informatics and Mathematics (ERCIM) asked to join. ERCIM was already pursuing an agenda similar to the CSTR project, and found the NCSTRL architecture to be sufficiently good that they wanted to adopt it. Expanding the scale of NCSTRL raised some technical problems we'll discuss below.

Difficulties encountered

At the time of this writing, NCSTRL has been in operation for roughly three years, and the predecessor projects a year or two before that. In that time, we've encountered a number of technical and social problems.

Metadata quality

NCSTRL search uses only document metadata, not the document text, so the quality of this metadata is very important. Each participating site is responsible for its own metadata, which in practice means that contributing authors provide it. This contrasts with the traditional library where cataloging is a centralized task performed by highly trained professionals. Sadly, the author-generated metadata is of uneven quality, negatively impacting the tools for resource discovery and content browsing in NCSTRL.

Consider author names. In the current collection of roughly 22,000 documents, there are about 37,000 author entries, of which 15,040 are unique. The NSCTRL record format mandates that author names be stored in inverted form, but 503 names (3.3%) are uninverted. An additional 293 (1.9%) have some other problem — the most common being the presence of additional information (e.g. an author affiliation) or use of non-ASCII characters or surrogates such as LaTex or HTML entities. To some extent, the presence of these non-ASCII characters is really an indication that we need a richer character format (e.g., Unicode), not a flaw in the cataloging, but the consequence is the same: Search and display of words not expressible in seven bit ASCII is flawed or impossible.

Harder to measure is the diversity in spelling for author names. For example, there are entries for "Abelson, H.", "Abelson, Hal", "Abelson, Harold", and "Abelson, Harold J.". Librarians have long coped with such diversity by employing authority files. Some NCSTRL sites employ similar measures, but not all do, and in any case, no single site is able to ensure collection-wide consistency.

Perhaps more ominously, 2498 records have a malformed version field, which indicates that the creators of these records did not understand the standard, and thus that other, as yet unnoticed errors may also exist.

Connectivity

After ERCIM joined NCSTRL, we discovered just how bad Internet connectivity could be. European Internet connections, unlike those in the USA, often have asymmetric quality — the bandwidth in may be different from the bandwidth out, and the connectivity from one country to another has little or no relation to geographic proximity, but more reflects long standing cultural or economic ties. Our solution, devised in cooperation with ERCIM, and first proposed at a May 1996 workshop at 5th WWW conference in Paris, was to create the regional divisions described above. This architecture was first deployed in July 1996, and has been in use ever since.

Server Quality

A distributed system such as NCSTRL is only as good as its weakest link. Some servers seem to be chronically down or slow. Some are running old versions of Dienst. Of the 48 sites running Dienst, four sites are (or were) running a very old version (3.5), which predates NCSTRL, and 14 more are running the first NCSTRL release (4.0). This is particularly a problem for the sites running 3.5, which had a less expressive search protocol than we now employ. If we were still using a fully distributed search, we'd have to limit all searches to only the syntax of the oldest server. In practice, we handle this by replicating the metadata from these old servers, and bypass them altogether. What is perhaps more problematic is that the antiquity of the software probably shows a lack of commitment to the operation of the server and maintenance of the collection content. Interoperability of different software versions is certainly a well-known, but unsolved, problem in the commercial world. It is certainly one that we can not afford to ignore as we continue to develop distributed information systems.

Permanence

One problem that continues to be troublesome for NCSTRL is the perceived permanence of the service. Institutions have been reluctant to join a service that doesn't have long-term certainty, and librarians (who have a special focus on longevity issues) are deservedly skeptical about NCSTRL as a resource on par with the materials maintained in traditional collections. Administration and maintenance of NCSTRL require substantial resources - to process membership applications, to fix bugs, and to extend the system (and this is in addition to the effort each contributing site expends in maintaining their local contributions). Permanence of the collection is dependent on making it self-supporting and, as of this date, economic models for long-term support of Internet infrastructure have not been established. At present, NCSTRL, like some other electronic scholarly archives[1], relies on government funding.

---

[1]The Los Alamos preprint server (http://xxx.lanl.gov) is sponsored by the US

Long term support for NCSTRL remains an open issue.

Lessons and Future Directions

In this final section, we review some of the lessons learned from the project, both decisions we changed and those we remain convinced of, then conclude with expected future developments of NCSTRL

Decisions we changed

Distributed search

In our initial use of Dienst, search was fully distributed. Each institution ran both an Index service and a Repository service, with the Index service indexing all and only the holdings of the Repository. We wish to emphasize that this tight coupling between Index and Repository was an artifact of the initial implementation, and in no way required by the Dienst model or protocol. Every query was sent to each and every index server (unless the query designated a single publisher). In retrospect this was a poor design decision, because as the collection expanded, the probability that every server would be both up and reachable decreased. Even when all servers were up, it meant that each search took as look to complete as the time taken by the slowest server, and when even one was down, it took as long as the (preset) timeout time. This was already bad when all NCSTRL servers were within the United States, but transatlantic connections made the problem intolerable. Our redesign using connectivity regions, explained above, handles these problems nicely.

Linkage of User Interface to Document

In the original Dienst design users viewed a document through the User Interface server of the site storing that document, which was not usually the same as the server they used to start the search. Although a standard UI server is included in the Dienst software release, sites could customize it, and thus the UI appearance changed as users moved from document to document. In the most recent version of the software, users view documents using their originating UI *regardless* of the location of the document. As in the original design, the user interface is still decoupled from other services, making it possible for individual researchers to experiment with presentation interfaces, but users maintain the same "look and feel" throughout their session.

Decisions we still like

Several years of experience with Dienst and NCSTRL have proved the value of a number of the decisions we made. Here we list the most important design choices that still seem correct. Most are technical, but some relate to policy and administration.

An open protocol

Our adherence to the principle that the functionality of Dienst should be exposed via a published protocol has made it possible for a number of research projects to independently exploit the NCSTRL corpus. For example, the protocol allowed the Stanford GlOSS[6] system to collect document metadata and facilitated interoperation with the Stanford Infobus protocol[7]. We are also aware of unpublished experiments that relied on the multiple renditions supported in the document model.

Architectural division of services

The basic architectural division of services — Repository, Index, Collection, and UI — has stood up well, and provides a demonstrable model of how distributed digital library systems should be designed. While we have redistributed the work among services (in particular, replicating indexing data), this has required little change to the protocol, and none to the core architecture. Our recent introduction of collection regions required no protocol changes whatsoever.

Document architecture

The Dienst document architecture has proven its flexibility for a variety of document types and formats. Within NCSTRL, documents are available in a number of formats such as TIFF, GIF, PDF, PostScript, and HTML. We have experimented with various document decompositions including physical pages and logical divisions such as sections, chapters, and the like. Other researchers have used the document model for other forms of content such as journals (The Making of America Project, http://moa.cit.cornell.edu) and environmental data[8].

National Science Foundation and Department of Energy. The New Zealand Digital Library (http://www.nzdl.org/) is funded by the New Zealand Foundation for Research, Science and Technology and the Lotteries Grants Board.

Flexible intellectual property rights policy

To participate in NCSTRL, an institution must agree to just three conditions. First, all document metadata is to be freely distributable. This allows NCSTRL to replicate indexing data without constraint. Second, each institution agrees to state its policy on intellectual property rights (IPR), which is displayed by the UI service when a reader accesses a report. The NCSTRL home page shows, for information purposes only, the IPR policies from two of our member institutions, but does not provide any legal advice. Finally, the institution certifies that the documents it adds to NCSTRL meet its (local) selection policy. Otherwise, we set no conditions. This flexibility allows participation by sites that charge for reports either individually or by subscription, as for example the Chicago Journal of Theoretical Computer Science. These conditions are stated informally. We could have set stricter or more formal conditions (e.g., with a contract), but this would have raised a barrier for participation far too high For most, if not all our members, the expense of having a corporate attorney examine and execute the contract would have outweighed the benefits of joining.

Flexible mutation policy

NCSTRL sets no policy whatsoever on withdrawal or alteration of contributed technical reports, and it provides technical means for a contributing institution to withdraw a document from the collection. An institution may wish to withdraw a document to correct errors or as a condition of publication in another venue. This is accomplished by marking the bibliographic record as "withdrawn". If need be, even the author and title can be removed from the record. Each site is free to alter documents in its repository, although we do not encourage this practice.

Use of HTTP as a transport for the protocol

The protocol is logically distinct from, and a level of abstraction above HTTP. Dienst requests are *encoded into* URLs, but they are not themselves URLs. This choice of encoding has been surprisingly trouble-free. Aside from imposing a few restrictions on reserved characters, the HTTP protocol and the URL syntax are quite easy to use, and the ubiquity of the Web (and HTTP proxies and firewalls) make it easy to deploy and access Dienst through the Web. It also allowed us to use the same code for the User Interface server (which *is* a Web application) as for the other services. Thus when the UI server wishes to display an inline GIF image of a page, it can simply construct a URL that is handled by the Repository. None of this would have been possible had we expressed Dienst directly as a TCP service.

This encoding was not wholly cost-free. First, we had to minimize the chance that the encoding for a Dienst request would collide with a URL that might have another meaning to a Web server. Second, the encoding was sometimes more cumbersome than we'd like for those encoded URLs used within an HTML Anchor or IMG tags. In such usage, there is no way to specify HTTP Headers, so all necessary information must be encoded into the URL itself. The only real problem we had with URLs arose because of our over-scrupulousness in encoding handles into URLs. Handles use the slash character to separate the naming authority from the local string, but the rules for http URLs (RFC 1738) state that the slash is to be used only to separate hierarchical levels. If one considers the whole handle to be a single item, then RFC 1738 seems to imply that the slash ought to be encoded (as %2F). This encoding, while correct, is not correctly supported by some popular Web servers. On reflection, had we flouted the law, we would have had wider use.

Design for change

Since version 4 of the Dienst protocol, all protocol requests and replies have contained explicit version numbers, allowing servers to support multiple versions of the requests, and thus preserve interoperability among sites, regardless of version. Over time, we have changed the on-the-wire format for queries, search results, and collection data, yet all services continue to interoperate

Postscript as a common document format

The Dienst package for adding documents to the collection requires that new documents be provided as PostScript files. We chose PostScript for two reasons:

1. Most of our readers can at least print a PostScript document if not display it on their screen.
2. It enables browsing. On-screen browsing is a vital aspect of any digital library. While some users download and print the document as a whole, others browse figures, abstracts, or bibliographies. Usage studies such as Bishop's[9] show that this is important. PostScript is readily converted to inline GIF page images and thumbnails to support browsing.

PostScript has turned out to be less universal than previously thought. Documents in Level 2 PostScript can't be printed on Level 1 printers. Worse, there appears to be wide variation in the quality of PostScript generated by some popular document processing tools, e.g. Microsoft Word, Framemaker. Jensen[10] reports that adding new documents to Berkeley's Dienst site often requires several time-consuming iterations with the author until the PostScript is acceptable. We note that LANL XXX site reports similar problems.

Despite these difficulties, we do not consider this a mistake. At the time, there was no alternative to PostScript. While PDF is probably more reliably printed, and free PDF viewers are widely available, there is still no free means of creating PDF, nor are there means to extract full text from PDF for full text searching.

The future of Dienst and NCSTRL

NCSTRL has been a unique experiment in both the technical and policy issues in distributed digital libraries. Its technical foundation, Dienst, continues to be a prominent example of an architecture for digital library interoperability. Over its history, NCSTRL made several large leaps in coverage and functionality, all of which were facilitated by the existence of the open and expandable Dienst architecture. We continue to work on expanding the functionality of the Dienst protocol, and are applying its concepts in more advanced research in digital library architecture.[11]

On the other hand, the original focus of NCSTRL on technical reports is threatened from both below and above. The ubiquitous Web now serves one of the functions of the technical report (rapid sharing of new or informal results) through personal home pages. We can also see trends of scholarly journals adopting a more fluid boundary between informal reports and refereed articles. Some have envisioned a world where publications begin as unrefereed postings and come to acquire endorsement and credibility through the refereeing process, and eventually settle into permanent archives maintained for posterity.

These developments have led us recently to expand the scope of NCSTRL. For example, we have worked with the Los Alamos e-Print archive to establish the Electronic Publication Repository for Computer Science, which interoperates with NCSTRL through the Dienst protocol, and we have expanded the NCSTRL collection with electronic journal content through our collaboration with D-Lib magazine. We hope to further expand the scope of NCSTRL in the future, with the possibility of including scholarly materials from other disciplines.

Acknowledgments

This work was supported in part by the Advanced Research Projects Agency under Grant No. MDA972-92-J-1029 with the Corporation for National Research Initiatives (CNRI), by the National Science Foundation under Grant No NSF-CDA-9308259, and by Xerox. Its content does not necessarily reflect the position or the policy of the Government or CNRI, and no official endorsement should be inferred.

We thank Naomi Dushay, Jim French and Rebecca Wesley for providing helpful data, our collaborators in the CSTR and NCSTRL projects for many ideas and helpful criticisms during the project, and the anonymous reviewers.

Figures

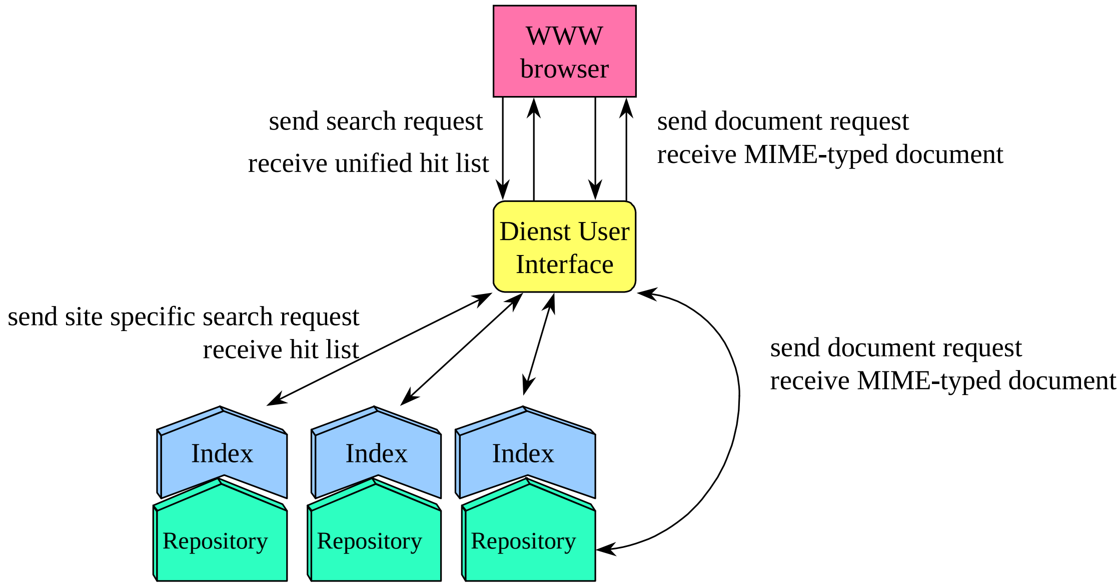

Figure 1 - Interaction of Dienst Services. The UI service communicates with the user with standard HTTP and HTML, and with the other Dienst services using Dienst protocol.

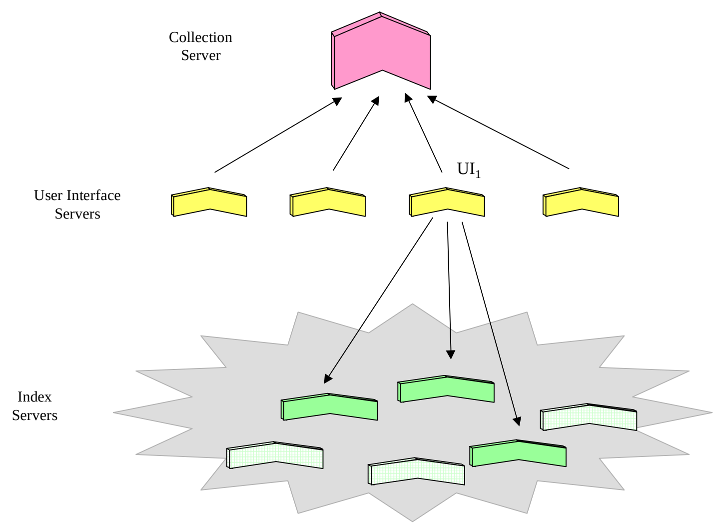

Figure 2 - Query routing using the Collection Service. A User Interface server obtains the set of Index servers from the Collection Service, then sends the query to the appropriate servers. A Region consists of a set of mutually well-connected servers, thus different UI servers will contact different sets of Index servers.

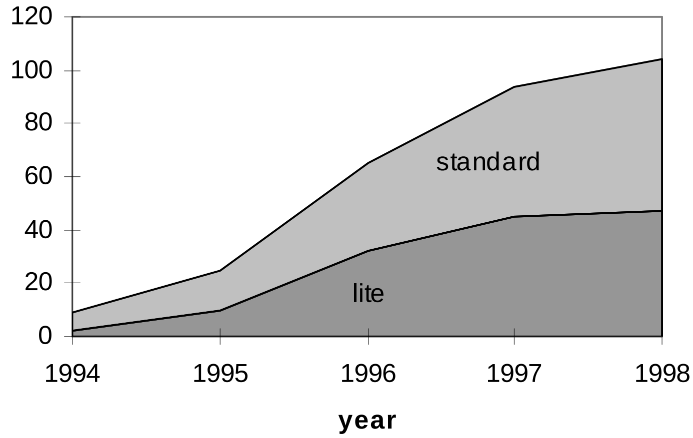

Figure 3 - Cumulative number of institutions in NCSTRL. NCSTRL began operation in mid 1995. Previous data represents participation in WATERS (later, NCSTRL Lite) or Dienst (NCSTRL standard) systems.

References

[1] C. Lagoze, E. Shaw, J. R. Davis, and D. B. Krafft, "Dienst: Implementation Reference Manual", Cornell Computer Science Technical Report TR95-1514, May 1995.

[2] C. Lagoze, D. Fielding, and S. Payette, "Making Global Digital Libraries Work: Collection Services, Connectivity Regions, and Collection Views", Third ACM Conference on Digital Libraries, Pittsburgh, PA, June 24-27, 1998.

[3] K. Maly, J. French, A. Selman, and E. Fox. "Wide Area Technical Report Service: Final Report", Technical Report, Department of Computer Science, Old Dominion University, June 6, 1994.

[4] R Lasher and D. Cohen, "A Format for Bibliographic Records", RFC 1807. June 1995.

[5] S. L. Weibel and C. Lagoze, "An Element Set to Support Resource Discovery", *International Journal of Digital Libraries*, January 1997.

[6] L Gravano,, H. Garcia-Molina, and A. Tomasic, "The Effectiveness of GlOSS for the Text-Database Discovery Problem", *Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data*, 1994. See also http://gloss.stanford.edu/.

[7] S. W. Hassan and A. Paepcke, "Stanford Digital Library Interoperability Protocol", Stanford Digital Library Project Working Paper SIDL-WP-1997-0054, 1997.

[8] C. Houstis, C. Nikolaou, M. Marazakis, et. al, "THETIS: Design of a Data Repositories Collection and Data Visualization System for Coastal Zone management of the Mediterranean Sea", *D-Lib Magazine* , November 1997

[9] Bishop, A. P. (1998, in press). "Digital Libraries and Knowledge Disaggregation: The Use of Journal Article Components." In *Proceedings of the ACM Digital Libraries 1998 Conference*. New York: ACM.

[10] A. Jensen. "Taking Local Resources Global: The NCSTRL Experience at UC Berkeley Library", *Issues in Science and Technology Librarianship* 18 (Spring 1998).

[11] S. Payette and C. Lagoze, "Flexible and Extensible Digital Object and Repository Architecture (FEDORA)", Second European Conference on Research and Advanced Technology for Digital Libraries, Heraklion, Crete, September 1998.