Structuring and Indexing the Internet

Ray Denenberg, Library of Congress
December, 1996

Abstract: *Indexing the Internet* (the *global index* model), currently the fashionable approach to information discovery, is at the bottom of the spectrum of semantic interoperability. A viable alternative is *distributed searching*, where sources maintain local indexes. *Structuring the Internet*, in a manner lending itself to effective navigation across distributed repositories (the *navigation* model), is perhaps highest along the spectrum of semantic interoperability.

The global index model has limitations; the most severe are lack of support for fielded searching, and the fact that mainly web pages get indexed while most real documents and objects do not.

Distributed searching overcomes problems of the global index model but has its own limitations, primarily: The *advertisement model*, popular in the global index scenario, can be defeated. Secondly, when raw, ranked results are merged, the resultant rankings are unlikely to be meaningful.

The navigation model, though highest along the semantic interoperability spectrum, comes at significant cost. Effective navigation requires coherently organized collections, which requires intellectual resources for aggregation, organization, and description. In addition, cataloging resources (human or automated) are required to create necessary descriptive records.

Z39.50 profiles are developed both for the distributed searching and navigation models.

Author's Note: *This paper was presented as the keynote address at the Workshop on Earth Observation Catalogue Interoperability sponsored by the European Space Agency and EC Centre for Earth Observation, 14-15 November 1996, in Ispra, Varese, Italy. When asked to speak on this topic,* Structuring and Indexing the Internet, *my initial response was that I felt it pretentious to address a topic so broad and nebulous. I was graciously offered the opportunity to change the title, but never got around to it. A more suitable title for this paper (though I was not cynical enough to suggest it) would be "The Futility of contemplating trying to Structure and/or Index the Internet".*

Indexing Vs. Structuring

Indexing and *structuring* are two different approaches to the problem of information discovery. This is not to say they cannot be used effectively in combination, but for simplicity, I will address them separately.

Indexing, as in "Indexing the Internet", is the approach to information discovery currently in vogue, but a viable alternative is *structuring* information, in a way that lends itself to effective navigation across distributed repositories, and is an approach that I think provides more satisfying results. Unfortunately, it is not an approach that has gained much favor within the Internet community, certainly not for web access.

For purposes of this discussion, *information discovery* means locating objects of interest when the population of object from which to choose is potentially widely distributed. The nature of the distribution can range from chaotic to highly-organized. At one extreme, there may be a wealth of information available on a topic of interest, but the information is scattered haphazardly and randomly, perhaps across the Internet.

On the other hand, objects may be aggregated into collections, organized thematically (by subject, creator, historical period, etc.). Although the objects in a collection may be distributed, the collection appears logically cohesive because metadata structures are associated with the collection, supporting coherent navigation.

Search Vs. Navigation

These two models of distribution, haphazard versus organized, are associated with different models of information discovery: *searching* and *navigation*.

In a typical search scenario, a user sends a query to an information source and is subsequently presented with a summary of results. The information source might be an intermediary, or *meta-searcher*, that selects and relays the query to several real sources, retrieves individual result sets, and merges them into a single set. Whether the query is sent to a single end-source (the *global index* model) or multiple sources (the *distributed search* model) is a crucial distinction, but it is not relevant to the typical search scenario, from an end-user point-of-view. In either case the user receives perhaps a one-line summary for each relevant document, then, perhaps based simply on the number of documents, narrows (or broadens) and re-submits the query. This process may continue iteratively until the result size is manageable. The user might then select a document for retrieval.

In a navigation scenario, a user maneuvers or is guided from node to node along a global information tree until arriving at a document of interest.

Browsing is probably a more familiar information discovery behavior than navigation, particularly on the web. Browsing refers to the unstructured, serendipitous process whereby a user, hoping to arrive at a document of interest, maneuvers from link-to-link (or page-to-page) without any informed tools to help decide what link to follow. Navigation is perhaps best described as value-added browsing, or browsing with tools.

For simplicity, I will consider the two models, search and navigation, separately, though in the real world, distribution models may be hybrids between haphazard and organized, and correspondingly, searching and navigation can be used effectively in combination.

Indexes Vs. Meta-Structures

Corresponding to these two behavioral models, search and navigation, are data models. The search model is based on indexes; the navigation model is based on meta-structures.

Indexes

Indexing is central to any search model, whether searching a single-source or distributed information. In a typical search model, a query is applied to one or more databases, and there are often indexes associated with each database, that facilitate or optimize searching. In a given implementation a physical index need not exist, but indexing is an essential modelling abstraction, so for this discussion we assume the existence of indexes.

Global Index Vs. Local Indexes

I will distill the discussion of indexing to two broad models: the single, *global* index, versus independently maintained *local* indexes. However, there are a few complications to dispense with first.

To begin, there is no *single*, global, index. Rather, there are several competing, redundant global indexes. For simplicity, assume just one.

Secondly, whether global or local, there may be a single, *flat* index supporting raw-text searching, or several indexes, corresponding to different fields, supporting *fielded* searching, for example to search on author, title, or subject. The current model of the web is a single, logical agglomeration of documents, with a single, logical flat index of its entire content. So for this discussion, assume no fielded searching for the global index model.

Finally, to further complicate matters, a global index might *logically* be a single, unified index, but *physically* distributed. This model, *distributed indexing*, is a special case of the global index model. It should not be confused with distributed searching, discussed below, which is based on the local index model.

Distributed Indexing

As I noted, distributed indexing may be dispensed with, as a special case of the global index model. First though, since distributed indexing has been a fashionable topic lately, I offer a few observations.

To begin, the issue of whether and how a global, shared index is to be distributed and managed is far from solved, and might not be solvable.

Secondly, distributed indexing is not just a technical problem. A large index may be a significant corporate asset, and the business implications of distribution are significant.

And finally, despite these barriers we cannot simply dismiss distributed indexing as an unsolvable problem, unless we also dismiss the global index model. Most experts agree that "indexing the Internet", or even any significant sector, into a single, physical index, is not a viable concept, and some form of distribution must be employed.

Three Models

To synthesize, we have three discovery models: global index, distributed searching, and navigation. In all cases, information is distributed. The *global index* model of searching absorbs the distributed index model. *Distributed searching* applies when each source maintains its own index. *Navigation* applies when there are coherent organizing structures imposed on the data.

I now consider each of these models separately, though I continue to stress, real-world approaches will likely be hybrids of these three models.

Global-index Issues

A global index is created through a process where some computer program collects indexing information from the various sources to be indexed. The program is referred to as a *crawler* (or *robot*, or *spider*). It traverses the web's global hypertext structure, recursively retrieving referenced pages. Actually, there are several of these crawlers, employing different traversal strategies. They create independent indexes, each, in a sense, duplicating the work of the others. (Note: by virtue of recursive traversal, some crawlers perform other functions besides creating indexes. These include HTML validation, statistics gathering, dead link detection, and file mirroring. We limit discussion of crawlers to indexing.)

There are dozens of technical issues associated with this process, for example:

- How *much* of a given document should be processed by a crawler? At the extremes, some crawlers process the title only, and others, the whole document. Between these extremes, some index an arbitrary number of lines at the beginning of a document (for example, the first 100 lines) and then, perhaps, the first line of each paragraph.
- How *often* should a given document be indexed? More frequent accesses do not always improve the quality of an index. Some documents need be indexed only once because they never change.

Some change, but infrequently. At the other end of the spectrum, some volatile documents should never be indexed because they change so rapidly (for example, a daily newspaper).

- How *deep* a traversal algorithm should be employed? Breadth-oriented traversal is more likely to locate a broader set of documents, though more superficially than depth-oriented traversal.

Along with technical complexities, there are clear deficiencies with the model.

- **Flat indexes**. Many experts feel that searching for raw text, as opposed to fielded searching, is one of the most crucial impediments both to semantic interoperability and relevance of search results. In the global index model, all of the words in a document (or all words in an arbitrary subset of the document) are indexed, without consideration to the context of the terms. In contrast, a sophisticated search engine may create many indexes, for example, individual title, author, and subject indexes, as well as an index for words in the "body of text", possibly an "abstract" index for words appearing in the abstract, and perhaps many others. The complexity is more than a matter of scale. Different search engines index differently and consequently may have completely different sets of indexes. Trying to create a global index supporting fielded searches becomes a near-futile task.
- **Distribution**. The problem of physical distribution of the single, logical index, may never be solved.
- **Web pages**. Mainly web pages get indexed, and most *real* documents and objects do not, because of the way the index is created (by traversing web pages).
- **Crawlers**. There are serious problems with the crawlers that create the indexes. I will now describe some of these problems.

Problems with Crawlers

Crawlers use vast amounts of *bandwidth*. They grab entire documents, many of which should not be indexed, for reasons of efficiency, suitability, or propriety. They index identical versions of the same document (for example by visiting mirror sites); they index the same documents repeatedly, even when the documents have not changed. And worst of all, there are many of these crawlers, all trying to index the Internet, redundantly. They create independent indexes, each duplicating the work of the others.

Crawlers can be very *annoying* to a server. They may create multiple concurrent tasks, consuming a large portion of available processing power. They may access a server repeatedly. They skew statistics, frustrating a server's attempts to obtain accurate information about user accesses.

Crawlers, essentially, are *dumb*: they do not know where to look, so they look everywhere; they download inappropriate data (a crawler might download an image and try to index it).

Crawlers cannot, in general, discern the relative *importance* of documents. A web server might have a home pages, pages with administrative information, and then pages with dynamic and useful information. A crawler does not know how to distinguish these pages.

Crawlers have problems with *context*. A web page describing the "Subway System of Washington D.C." points to a page titled "Departure Stations". "Departure stations", then, is indexed with no context.

Crawlers access mirror sites, and as I noted above this wastes bandwidth, but another problem with mirror sites is that they may be very much *out-of-date*.

Some servers are un-accessible when the crawler is run, so important information is simply missed. And closely related to the *missing information* problem is the problem of *latency*: the most current information usually is never indexed, because of the period between the time it is put up on the web and when it is first indexed.

And finally, there are *bad* crawlers. One of the ethical issues is balancing the high cost of indexing against the greater good. Perhaps a global index is created at exorbitant cost to the community at large, but maybe it is a "good" index and serves the community at large. On the other hand is the index created at high cost to the community, and it is a "bad" index, or its created for private use.

Proposed Solutions

There are various proposals and suggested architectures for solving these problems, based on cooperation: among the index providers, information resources, and information creators (authors and publishers). One suggestion is that these various web crawlers share, or partition, the work of indexing the web. While this would provide great efficiency improvement, it is not a practical suggestion, for reasons of complexity and business.

According to another proposed architecture, the information servers would provide indexing information, either for indexers to capture, or alternatively, to send to the indexers (the *pull* and *push* models, respectively) based on the theory that the server is in the best position to decide what should and should not be indexed. This model has appeal because not only would it improve the quality of the indexing information (as the theory goes), but would also dramatically reduce the amount of information transferred and thus save considerable bandwidth.

The problems with this approach is that it requires altogether *too much* cooperation (in contrast to a model where the server plays a passive role) and it defeats the proprietary index schemes employed by the indexing companies; they derive significant revenue from their proprietary algorithms.

Another suggestion is that the information creators themselves, the authors, provide metadata for their documents. That approach is not practical either, for a number of reasons: it, too, defeats the proprietary schemes of the indexers; it requires too much work of the authors and they are not willing to do it; and even if they were willing, they probably would do it wrong, and bad metadata is worse than no metadata at all. Another problem with author-created metadata is the temptation for the author to attach terms simply for the purpose of increasing the potential ranking of the document.

There are primitive mechanisms employed on the web for a server to designate files that should not be indexed, if and how often a document should be indexed, date of creation, last update, last verification, and (expected) next update. These are passive methods, and there is research towards development of more active mechanisms, for a server to *notify* indexers, when, for example, the content of a document has changed and it should be (re-)indexed.

Finally, perhaps the most radical of proposals is that search engines adopt a common standard for indexing. This suggestion is always rejected, because the proprietary indexing scheme is often the essence of a search engine.

Despite the problems cited above, "Indexing the web", or at least giving the impression of doing so, remains a lucrative enterprise, since there still are a number of companies doing it. But as the number of web documents continues to grow, almost exponentially, the percentage of the documents indexed will continue to drop.

Distributed Searching

Distributed searching is often advanced as an alternative to the global index model. In the distributed search model a client sends a query to a meta-search engine which relays the query to several real information sources, integrates the results (see note), and presents a single, logical result set to the client.

> Note: for simplicity, assume that *integration* is accomplished by retrieving individual result sets and merging them, though there are more efficient mechanisms.

Distributed searching overcomes many of the problems cited in the global index model, for example, fielded searching may be supported and searches may located real objects (not just web documents). On the other hand, distributed searching does have limitations.

Limitations of Distributed Searching

There are two major limitations to the distributed search model, one economic and the other technical.

The economic problem is the *advertisement model*. Often, one or more of the individual end-sources is a commercial service that searches free-of-charge but derives revenue from placing advertisements in the output results. This economic model based on advertising, popular in the global index model, can be defeated with distributed searching; the meta-searcher can strip out the advertisement.

A proposed solution to this problem is to develop a model by which advertisements are carried along with content. A more radical proposal is to abolish the advertisement model altogether (based on the sentiment that no economic force more strongly skews the free-enterprise model than advertisement). This is highly unlikely to happen because of the opportunities available for companies to reach customers at very low cost, and also because many users actually have come to *expect* advertising, and some feel cheated if advertisements are not present.

The technical limitation to distributed searching is the *merging* of results, and more importantly, *ranking* the merged results. Different search engines employ different ranking algorithms. When a meta-searcher merges raw, ranked results, the resultant merged rankings are unlikely to be very meaningful.

Normalizing Rankings

There are several techniques proposed or under study to allow a meta-searcher to produce normalize ranked results. Among these are the following.

(1) The simplest technique is for the metasearcher to request that each server employ a specific, public ranking algorithm, in lieu of its own native and proprietary algorithm. (In this scenario, each search engine informs the meta-searcher what algorithms it supports, by supplying ranking-algorithm-ids. The meta-searcher selects a common id, without necessarily knowing anything about the identified algorithm.)

A major (non-technical) problem with this approach is that the proprietary algorithm employed is often much more suited to the particular search engine and its indexing structure, so by asking the search engine to use a different algorithm, much of the potential native ranking power is lost.

(2) A more complex proposal is for the meta-engine to retrieve normalization information from each search engine, calibrate each set of ranked results, and produce a unified, ranked result set.

(3) Techniques have been developed to allow a client to specify ranking criteria along with a query, for example, what weight to associate with a given term.

(4) Another interesting though experimental approach is for the metasearcher to actually retrieve all of the documents located by the query, from the various servers, into a temporary (pseudo) database, and execute the original query against that database. This technique, though innovative, likely would be effective only on a small-scale.

Mechanisms (1), (2), and (3) are supported by the Z39.50 ZDSR profile described below.

Z39.50 Profile for Simple Distributed Search and Ranked Retrieval

The *Stanford Protocol for Internet Search and Retrieval* (STARTS), an initiative of the Stanford Digital Library Project, developed requirements for distributed searching and ranked retrieval during the Spring and Summer of 1996.

In July 1996, several Z39.50 experts collaborated with participants in the STARTS project to develop what was originally called the ZSTARTS profile (Z39.50 Profile for STARTS) renamed the *Z39.50 Profile for Simple Distributed Search and Ranked Retrieval*, ZDSR.

The profile assumes that queries pertain to text documents -- not just web pages but real documents (just documents though; not arbitrary objects).

It support searching by title, date-last-modified, author, language, url, and within the body of the text. It also supports relevant feedback searches, and stem and phonetic searching. Search results may be restricted by threshold score, or maximum number of documents.

The query includes a *restriction* component and a *ranking* component. The restriction component is the normal Z39.50 boolean query, used (in this profile) to specify the documents that qualify, for subsequent ranking. The ranking component is a list of terms each assigned a relative weight by the client.

Using this Z39.50 profile a client searches for *documents*, and retrieve document *descriptor*s containing *metadata* about the documents. For a given document, the *document descriptor* includes title, abstract, publication and creation date and date last modified, size of the document, the score that the server assigned to the document for the query, and per-term meta data: for each term that was in the query, its frequency and weight; and finally, a pointer (URL) which may be used to retrieve the document (document retrieval is otherwise out-of-scope for the ZDSR profile).

[Note: as of December 1996, the ZDSR search access points and the elements comprising the document descriptors are not completely determined.]

Z39.50 and Metadata

This leads to a brief digression on the general topic of metadata, in particular, Z39.50 and metadata.

Z39.50 deals intimately with metadata, though subtly so -- subtle for two reasons: first, the term "metadata" itself came into vogue recently (relative to Z39.50) so Z39.50 has addressed metadata without referring to it as such.

Secondly, elements that are metadata from one point-of-view might not be metadata from another. For example, bibliographic elements -- author, title, publisher -- are integral elements of a bibliographic *record*, so although they may be metadata for the object *described* by a bibliographic record, they are not metadata *for* the bibliographic record.

In other words, since metadata is data about the *subject* data, if bibliographic elements are the subject data, by definition they are not metadata. Z39.50 was originally used primarily in a bibliographic context. These ZDSR document descriptors, which are fashionably described as document metadata, are, from a Z39.50 perspective, really bibliographic records.

Z39.50 implicitly recognizes a number of different categories and levels of metadata. Most fundamentally, Z39.50 distinguishes search elements from retrieval elements, that is, between the search access points and the elements in a retrieved Z39.50 record; these may be, but are not necessarily, the same. For example, a bibliographic record for a book might include a spine title as a retrievable element of the bibliographic record; the record might be searchable by title but not by spine title. Another example: an object (say, an image) may be searchable by a unique identifier, but that identifier is not an integral part of the object.

Other categories of Z39.50 metadata are:

- *Transient* metadata associated with a query, for example the *score* of a document.
- Metadata associated with the specific *form* of an object available in multiple forms, for example, its *size*, *format*, and possibly even the *cost* to retrieve it (which may vary by format).
- Metadata that applies differently to different users: for example, *terms and conditions* (or *rights and restrictions*, etc.).
- *Server*-level metadata, *database*-level metadata, and various other classes of metadata available via the Z39.50 *Explain* facility.
- *Collection* level metadata, which Z39.50 has introduced for the *navigation* model.

Navigation

In contrast to the prevalent model of haphazard distribution of documents over the Internet, institutions are beginning to assemble compilations of documents, or more generally, *objects* into (relatively) coherently organized *collections*, which users may navigate to locate objects of interest.

Informally, a collection is an aggregation of related objects and subcollection. Collections are organized thematically, for example by subject, creator, or historical period; they may have diverse objects: text documents, images, audio, video, or arbitrary binary objects. These objects often have *Associated Descriptions* (described below). Collections are often hierarchical and may be distributed.

Support for effective navigation has two broad requirements: tools for semantic interoperability, and expenditure of resources, both intellectual and cataloging resources. *Semantic interoperability* in this context means standard navigational search semantics as well as meta-data structures, and clients designed to navigate, based on these semantics and structures.

Semantic interoperability is the easy part. Effective navigation requires coherently organized collections, which may require significant intellectual resources for aggregation, organization, and description. And in addition to the intellectual efforts, resources (either human or automated) are required to create records based on these meta-structures.

Collection Descriptive Record

An example of a navigational meta-structure is the *Collection Descriptive Record*, defined by the Z39.50 Collections profile, described below. The Collection Descriptive Record includes summary descriptive and navigational information for a collection. It includes a *brief description*, to help a user decide if a particular collection is of interest. Besides the brief description, there may be other descriptive items pertaining to the collection, that may be more comprehensive, or perhaps machine processible. These are termed *Associated Descriptions*. A user might view the brief description to decide whether to retrieve the Associated Description.

Associated Description

An Associated Description describes a collection or an object, and may take different forms for different categories of collections and objects. For example, it might be a finding aid, an SGML Encoded Archival Description, a cataloging record, an exhibition catalog (for museum collections); a GILS record, or even a web page. The Z39.50 *Catalog Interoperability Protocol* (CIP) profile defines structures including the CIP *Item Descriptor*, and CIP *Browse data*. These could be considered Associated Descriptions.

A collection may have several Associated Descriptions, and a client can retrieve descriptive information about them (essentially, description about description -- meta-meta data). This includes a brief description of the Associated Description itself, its type, size and format, and a pointer to the Associated Description, in case the user or client decides to retrieve it. This meta-meta information serves two purposes: it might help the end-user decide whether to retrieve the more comprehensive description, or the machine-processible *type* may allow the client to determine whether it is capable of processing it.

Related Collections

The descriptive record also includes a list of related collections, for example a parent (or otherwise superior collection) or a child (or otherwise subordinate collection) to which the user might wish to navigate (i.e. retrieve its descriptive record) if the collection that the record describes is too narrow or too broad.

A related collection may be a *context* collection, meaning it is the highest level superior collection likely to be of interest. It might be a sibling. Or it might not have any familial relationship: the descriptive record may point to a collection that simply "might be of interest" if the user is interested in the given collection.

For a given collection, the descriptive record lists a set of related collections, and for each, describes the relationship and provides a pointer to enable the client to retrieve the collection level descriptive record for that collection.

Enumerated Objects

The collection descriptive record also enumerates the members of the collection. For each object, there is a brief description, to help the user decide whether to retrieve the object, and a pointer, to retrieve the object.

Collections Profile

In 1995 the Library of Congress convened a team from several institutions to develop a Z39.50 profile for access to digital libraries. The scope was narrowed to apply to navigation of digital collections, and was named the *Z39.50 Profile for Access to Digital Collections*, informally, the *Collections* Profile. The larger problem of access to digital libraries was left to the province of other profiling efforts, one of which, led by the Library of Congress, developed the *Z39.50 Profile for Access to Digital Library Objects*, informally, the *Digital Library Objects* (DL) Profile. Other groups were initiating independent efforts to develop profiles aimed at specific types of objects and collections. The intention was to coordinate these efforts and that these latter profiles would be developed as compatible extensions or subsets of the Collections profile.

The Collections Suite of Profiles

The Collections profile is an *umbrella* profile for navigating collections. It defines the Collection Descriptive Record (described above) and provides the framework for the development of extensions for specific domains. These extensions are called *companion* profiles to the Collections profile: the CIMI profile for access to museum objects (developed by the Consortium for the Computer Interchange of Museum Information as part of its Project CHIO: Cultural Heritage Information Online), a profile for access to digital library objects, initiated by the Library of Congress, for access to the LC digital library and similar collections, and the *Catalog Interoperability Protocol* (CIP) profile, for access to Earth Observation Data and associated data resources being developed by the Protocol Task Team within the Committee on Earth Observation Satellites (CEOS).

Work began on the CIP profile independent of the Collections profile development. Technically, CIP is not yet a companion profile to the collections profile but there are efforts underway to align the Collections and CIP profile. The CIP profile is one of the more advanced and well-developed of the Z39.50 profiles and one that exploits the power and capability of Z39.50.

Both the Z39.50 community at large and the CIP community believe there is mutual benefit in harmonizing the Collections and CIP profiles, so that CIP would be a conformant companion profile. At present, work continues to achieve this harmonization.

There is currently investigation into Collection companion profiles for access to multimedia objects, and musical objects.