Z39.50 – implications and implementation at the AT&T library network.

Robert K. Waldstein

AT&T Bell Laboratories
Murray Hill, NJ 07974

Abstract

The AT&T library organization has developed an interest in Z39.50 for a number of diverse reasons. It is hoped that eventually Z39.50 will help with or solve several classes of problems, ranging from behind the scenes issues resulting from distributed computing architectures to diversity of user interfaces. In addition to helping with known problems, we hope that Z39.50 will give us a flexibility required for a constantly evolving library organization in an international corporate environment.

For Z39.50 to meet our needs the main requirement is that the protocol itself incorporate all the functionality of our existing information retrieval environment. The 1992 (version 2) version of the standard was a major start, but the newer version (1995 version 3) comes much closer to incorporating existing functionality.

The next major requirement is proven interoperability and transparency of database provider to our users. Issues of indexing style, default operations and ways to override defaults, database coverage and loading characteristics become even more apparent in Z39.50 than in the traditional online world.

Our end users, like users everywhere, are expecting interfaces integrated into their regular computing environment. A solution to this problem is a well accepted search and retrieval protocol. Z39.50 is well positioned to become this protocol, and in this belief we have focused our attention on developing a high-quality server for our internal resources.

1. Introduction

The AT&T library organization has changed considerably, and continuously, along with the rest of AT&T in the years since divestiture in 1984. We provide world-class library services to the employees of AT&T worldwide including: technical information needed for research and development, business and marketing needs, as well as manufacturing information and internal newsletters. Some significant user requirements that we must meet as a corporate information service provider are:

- Cost reductions. This is both in terms of our budget and what people are willing to pay for information services and resources.
- Information provided in the user's environment. e.g. integrated workstation, fax.
- Information on demand. Users often want the desired information at the time of request. In addition, the user wants to control the depth and format of the desired information.
- User access to information. User demand for direct access to information is growing, with or without an intermediary's involvement.

The computing environment in which we provide our services has been evolving rapidly. Distributed computing has become a requirement for a flexible environment, both in terms of costs and functionality. In addition, workstations – powerful computer resources at the user's desk – have nearly replaced terminals as access tools. Workstations have not only opened new opportunities, they have changed users' expectations about the "look and feel" of the information presented to them.

The information market has also been rapidly evolving during this time period. Interest in new forms of access licenses, especially site licenses, has been growing among database providers. Databases have begun to expand beyond flat text, into "multimedia" – particularly scanned images. Finally, special interest database providers are appearing on the Internet, covering an entire range of corpuses. These include the human genome, ftp-able files, congressional bills, acronyms, and Library of Congress exhibits. Although providing access (as well as meaning and organization) to all this information may be pushing what some may consider the library's role, our users do come to us expecting our service to include these resources.

Finally, of course, the means by which information is accessed and used is changing. Our users have multiple or no offices. Telecommuting is a growing practice. Information requests and requirements arise while waiting in airports. The growing international user population and information environment make time zones and export laws increasingly important. The importance of security, of computers, information, and customers interests has been growing, especially as the Internet becomes both a carrier and a source of information.

This paper presents technical concerns related to these issues. It will focus on the author's perceptions, and in particular on where and why Z39.50 presents a flexible means of approaching a diverse set of these issues.

2. Overview of the AT&T Library Network and Retrieval Environment

The AT&T library network is from many perspectives based on information retrieval. Our end users search the databases we provide, our information professionals search both internal and external database resources, our publications are based primarily on searching database resources, and our library automation systems are based on database searches and modifications. Our internal database setup since the early 1980s has been Unix–based, with an internally developed database engine called SLIMMER[1]. SLIMMER has been designed to work as a filter – for searching, retrieving, and formatting records from databases; as well as for updating databases. Our library automation systems, including circulation systems, table of contents (TOC) alerting, billing, and photocopy tracking, consist primarily of scripts tying together database retrievals and database updates. These scripts are written primarily in high level languages such as perl[2], AWK[3], and Unix shell(s).

The above conveys a view of our library systems as modules which have, at their lowest level, database retrievals and updates. Until recently, a software module that used a database had to reside on the same computer as the database. From the 1960s through 1990, our library automation systems were mainframe-based. When applications resided on different computers they had the extreme limitation that they could not exchange information with other applications in real time. In the early 1980s this was solved by bringing most of our library applications together on a single computer. This was also when we switched to using the Unix operating system. But by the late 1980s, as our requirements grew, a single mainframe computer proved an inflexible and costly solution.

Our organization took the first step away from this architecture in 1991 when we began using Network File Systems (NFS). In 1991/1992 we moved to a cluster of minicomputers with a common file system using NFS. Unfortunately, this added complexities that we are still dealing with after 3 years:

- Shared database aspects such as shared memory and interprocess control require considerable care.
- Database and record locking has proved a continual problem in a multi-CPU environment.
- The load on the network and the file server is quite high in a large database environment. For example, an application needs to know the number of articles in *Byte* magazine that contain the word "computer". This can be expressed as a search on "Byte AND computer". The inverted file entries for "byte" and "computer" are brought to the computer where the application is running, intersected (ANDed), and the number of records in the intersection saved. To obtain one number, several million bytes of data flow over the network.
- Finally, as our network continues to grow and diversify other network problems are arising. Issues of security with exporting our file systems to computers geographically far away is a concern. In addition, as the computers are more dispersed the speed of the linking network (and the reliability) decreases.

Thus while NFS is good for many of our shared file applications, it has significant limitations for large database applications. A viable alternative approach is the use of database servers.

A database server has a database residing at one place with all the applications accessing the single copy via some robust, flexible technique. When the access technique is a network protocol, this makes the the database server and the application using the database nearly independent. Z39.50 is well-developed protocol that can help meet our internal needs as a database server. In addition, it allows us to use external database servers in a transparent manner.

Increasingly over the last 10 years we have also been mounting databases, both from internal as well as external sources. Most recently we have begun receiving newswire feeds, such as DowVision and AP wires. As our users' data requirements grow, and with them the demand on our organization resources, both computer (e.g. disk space, CPU, backups, security) and human (e.g. database administrators, help lines, tape handlers), our organization is looking constantly at buying database access. But our requirements are high: neither our end users nor staff can be expected to learn multiple interfaces or database setups, plus we require that the location of a database be nearly transparent.

The client-server model is becoming the 1990s' solution to the problems described above. In addition, and perhaps most importantly, it helps minimize the need for people (users at all levels) to learn new interfaces depending on the information resource being accessed. It also opens up opportunities for distributed library applications. It means applications that require database access can be built independent of where the database resource resides.

3. Z39.50 Version 2 Protocol Limitations

This section addresses issues that were of concern in Z39.50-1992 Version 2 – issues that version 3 has resolved. These are protocol limitations – that is, features the standard could not support in the 1992 version. Presenting a simplified user search is probably the easiest way of conveying the first set of issues that arose upon considering Z39.50. Note these issues are mostly resolved in the 1995 version of the protocol.

The following is a simplified user search interaction of a SLIMMER database.

1. The user is presented with a introductory screen presenting the database.
2. The user enters a search, for example "computer retrieval". SLIMMER searches all indexed fields, basically ANDs together the two terms, and tells the user:

   Term "computer" retrieved 21959 items -
   Term "retrieval" retrieved 919 items -
   now 315 in set

3. The user is now presented with ways of reducing the retrieved set; one common method is to restrict the search by field, for example "title".
4. Records are retrieved and displayed.

In the scenario above, the interface knows a fair amount about the database. It needs to know the database name and other relevant information to present the user on the introductory screen. This is probably the first embarrassment to a Z39.50 client implementor; all the interface really knows about the database is the network address; this does not make for a friendly welcome screen. But problems also exist lower in the interaction. Step 3 requires knowing both how to present indexed fields for a given database to a user and how to use them in a Z39.50 search. Step 4 requires knowing the content of a database record and how to present it to the user. A related requirement is knowing how to obtain a given field from a record. For example, if the user says "give me more records by this author" somehow the client software must be able to find the "author" field and know how to use it in a Z39.50 search.

All these issues are solved by the Explain facility in Z39.50-1995. Without Explain, implementations are constrained to conveying database information outside the retrieval session (e.g. by phone or documents). Since our system allows considerable database setup flexibility and change we needed Explain to get started. So we implemented the first stable Explain structure, as proposed in summer 1992. It has proved quite satisfactory for our main needs.

Step 2 above, the search step, also involves a number of protocol features beyond Z39.50-1992. By default SLIMMER searches all indexes. This capacity was not in the 1992 standard but was added shortly thereafter as the Use attribute "any". In addition, note the line:
Term "retrieval" retrieved 919 items - now 315 in set
The intermediate step information about "retrieved 919 items" cannot be conveyed in Z39.50-1992. This required the User Information Format features introduced in the 1995 version.

Finally, step 3 was probably the most controversal issue of a protocol deficiency in Z39.50-1992. SLIMMER carries along information about the fields of a record in which the retrieval terms were used. This allows, for example, a user to search on "einstein" and then based on the number of retrieved records either look at all the records or first reduce the set to those records where "einstein" was used in the "title or subject". This feature exists in many major database providers, as well as in the Common Command language (Z39.58); it can now be done interoperably done in version 3.

Step 4 includes fairly major requirements. It requires the ability to package a record in a Z39.50 message without loss of information. Since SLIMMER has elements (record pieces like author, title) with arbitrary string and numeric tags that can have diverse content, MARC is not an acceptable record package. John Kunze (University of California at Berkeley) proposed a flexible record structure called INFO-1 that supported the functionality required. We implemented and used this structure from 1992 through 1994. This record format evolved into the generic record syntax (GRS), which is part of the 1995 protocol specification. This new format even better suited our needs; in particular it has a clean way to carry a local record key and record dates, part of our basic SLIMMER record.

4. A Security Concern in Client/Server

There is a security requirement that arises once records are delivered into the control of client software. Currently, when our databases contain sensitive data (e.g. social security numbers, passwords) we simply do not display this content. This works quite well as long as we control the user display and interaction with the record. However, once the record has gone out to a user's program we lose control of how the record content is displayed and manipulated. This problem was solved by adding to a database setup a list of fields that are considered sensitive, and these are simply never sent to the client program.

This security approach has solved a number of problems as well as creating new ones. It solved a growing problem: as our interface gave greater control to the user it was becoming more difficult to protect against clever users gaining access to sensitive data. Simply not making the data available is one solution, but is too extreme for some cases. For example, if a personnel record has a special library access password we displayed on the screen

user has library access password.

That is, the password is sensitive, but the fact that it exists is of value and is not sensitive. Solutions to this problem are difficult and have uncertain return, so for now this functionality is an accepted loss in our Z39.50 implementation.

5. Z39.50 Extended Services

When the AT&T library network originally investigated Z39.50 a number of functions were ignored since interoperability was not necessarily a requirement. Many of these have since been proposed and incorporated into Z39.50-1995. Two noteworthy services, item order and database update, are discussed below.

5.1 Item Order

Item order is essential to our users – they complain when databases present records describing materials they cannot easily order. But this doesn't necessarily mean that we require it in Z39.50; it means we require it as a functionality of the client software. Usually this involves the client software gathering some amount of information about the item being requested (from the database record) and information about the user – both from the user and personnel database(s) – and delivering this information to a request handling system. In our present environment part of database setup is setting up what information is needed for a request and where the request is delivered and how. Presently we do not allow requests from distributed clients, since the request invokes request entry commands in our other systems. However, we will soon allow the following technique when a client program is connected to a database mounted at our server:

1. User says "I want the object described on my screen". This will cause an item order to be sent to the Z39.50 server. Technically this is an Extended Service item order package for a resultSetItem.
2. In most cases the server will just acknowledge the request and that will end the transaction (from the Z39.50 point-of-view). This works since the Z39.50 server knows who the user is (required to gain access to the database) plus which record is being requested. The Z39.50 server passes the request on to the correct request handler as set up by the database administrator.
3. When additional information is required; e.g. more billing information, permission to bill, or the size of requested item, this

information is obtained using Access Control via a PROMPT-1 access control format. PROMPT-1 allows the database provider to obtain any required information from the user. This approach is necessary because it is the database provider that knows what additional information is required with a given request, not the client designer.

This does not solve the problem of AT&T employees using distributed clients to search databases at non-AT&T information providers. In this situation we still want user requests to filter back through our request handling organizations. If we controlled the clients it would be easiest to set up our own private request format. In fact we do have a command distributed to many AT&T computers called *library* which sends electronic mail in a fixed format to request library materials and services[4]. If we could have the clients use this existing format our work is minimal. But we believe this approach has significant limitations. In our environment we need an accepted protocol for requests of material. Since we have many traditional library needs (e.g., book buying and borrowing, article photocopies), we are watching the development of ISO 10161 – the ILL protocol. In addition, that protocol is growing and developing to handle orders for diverse types of materials beyond traditional library needs.

The growth of the ILL and Z39.50 standards and their synergy[5] are of interest for other reasons. We acquire significant quantities of materials, especially books and photocopies, from external vendors. To give our customers the turnaround times and service required, we use electronic interaction with our vendors. This usually means setting up a new method of transmitting requests and information about requests with each vendor. If this could be standardized our organization, and we believe our vendors, would benefit.

5.2 Database Update

This, like item order, is a functionality we would like and are pleased will be supported in the context of Z39.50. We use a variety of technologies presently to ensure a single flow of database updates. As our computing environment becomes more complex, ensuring that only a single update process is running has become more complicated. As our environment becomes more distributed, security of data and especially of data updates is an ongoing concern.

Having a single network point for updates, in this case the Z39.50 server, simplifies issues of concurrent updates and security. In contrast with alternatives involving NFS, inter-process control (IPC), and other related network and operating system dependencies, single process control is a preferable solution. So although the availability of clients able to interoperably send database updates is not expected soon, this functionality may have internal application in the near future.

6. External Database Access

Accessing databases at other servers raises a new set of problems. To our users, at least theoretically, there should be no difference between accessing an external database and accessing an internal database. Unfortunately, the real world is not quite that simple. Our initial problems can be divided into several categories:

- Traditional issues of database loading.
- Issues of indexing and index access.
- Features of the remote Z39.50 server.
- Speed of the connection to the remote database.

It is interesting to think of the problems described below compared to issues of buying and locally mounting a database. When our organization buys information resource tapes, we investigate the best resource in terms of content for our customer needs. The assumption is made that our database administrators and systems can then make the data available in a way that will satisfy our customers. With the advent of Z39.50, acquiring a database requires answers to questions relating to whether the available Z39.50 access is sufficiently flexible to meet our requirements. This requires a new set of training and thinking in acquiring database access. The issues described have to be considered and handled before signing a contract.

6.1 Issues of Database Loading

An important advantage of Z39.50 is transparency of database access to the user. Z39.50 hides access differences, but the underlying database is still all important and different for each provider. These differences can be important, but often Z39.50 hides these as well. Issues of update frequency, completeness of loading (often all aspects of a database are not made available), quality and availability of full text, completeness of records (whether all fields loaded), and how record updates are handled are examples of important, basic issues. Librarians have long realized the importance of these factors in selecting and using a resource. However, it is clear that our end users cannot be expected to make similar judgment, especially when we have intentionally screened them from differences among database providers.

These are the first set of issues raised by the internal database administrator when we acquire and locally load a resource. It is easy to overlook these issues when acquiring Z39.50 access, particularly since most end users would not even notice these issues.

6.2 Issues of Indexing and Index Access

Indexing issues with remote servers break into two sets of major problems. The first set are issues of how the database provider indexed the records of the database, the second set concern the Z39.50 interaction.

AT&T library users have come to expect fairly complete indexing. That is, they expect indexing of most of the record content, and the ability to specify which record element is being accessed. How that data is indexed, and the depth to which it is indexed can vary considerably; librarians are trained to be aware of this factor.

SLIMMER allows considerable flexibility in indexing; our end users expect this, and our systems are designed using this functionality. For example, a "phone number" field may have content

123 456-7890

and we might index it so a user can search on "phone number"

123 456-7890 or 1234567890 or 7890 or 123.

The ability of the system to do this type of indexing is a combination of system flexibility, and equally importantly, local control. When database access is purchased indexes may be unchangeable, or at best changed via contract and interaction with the organization that makes the database available.

In addition to these issues are Z39.50 aspects of index access. The first issue often raised is coordinating the client and server to access the correct index. This can be done in three ways:

- Published lists of access points. By far the main such list is the BIB-1 attribute set which is part of the published standard.
- Out-of-band agreements on attributes. That is, the client author and the database provider can agree on values to be used for different access points.
- The Explain database to communicate the available attributes. A client using the Explain database can dynamically learn the access points for a given database and the Z39.50 method to convey using a particular access point. Explain is clearly the most flexible solution to this problem, but unfortunately there are not many existing Explain implementations.

Interpreting what is meant by an access point is a problem both in the Z39.50 environment and other search setups. The user who searches for "author" may or may not expect corporate authors or editors. These are problems in user interface but carry into the interface between the client and server developers.

6.3 Features of the Remote Z39.50 Server

Clients can be designed to compensate for some differences between remote database sources. For example, whether the server sends GRS or USMARC records is not something a user should notice. But some other server features are more difficult to hide from the user. Features (or lack thereof) we have had to cope with include:

- Whether proximity is supported. This feature is more complex, since the level of proximity, e.g., word, sentence, paragraph, is an issue as well.
- Whether Boolean operators are supported. Yes, it is possible to mount a database under Z39.50 and not support Boolean operators.
- How unspecified Use attributes are handled. This is a significant issue in our interface, since the default operation is to search all indexes. Our clients request the desired behavior, using Use attribute "any". However, many servers do not support this. For interoperability reasons our clients then switch for these servers to a Use attribute of "server choice" or send no Use attribute. The undefined behavior of the server at this point has caused some trouble and confusion.
- Whether the remote server supports named result sets. Some functions of our clients require creating and holding intermediate result sets. Without this capacity the clients

can still work but at a reduced functionality.

6.4 Speed of the Connection to the Remote Database

Z39.50 database resources are mainly available over TCP/IP networks (e.g. the Internet). When a user complains about our search response time, and the response time degradation is the result of someone doing real-time video at an unrelated site, this does not satisfy the user. The solution to this problem is the ability to purchase guaranteed band-width which is presently not available. In addition, firewalls and proxy servers can add considerably to response times – another overhead that is difficult for libraries to control.

This has been a major issue in our initial attempts to make Z39.50, as well as other Internet resources, available to our users. In the World Wide Web environment people may be willing to wait 1-2 minutes to hear the President's cat* or several minutes for an "archie" search. However, users expect external databases (which they are not even aware are external) to have response times like internally mounted databases; e.g. a few seconds. We frequently experience long delays in forming a connection, initializing the session, and retrieving records. Some of the user impact of these delays can be mitigated in the user interface and others may be avoided by caching data that the user may need to see or use again. But no complete solution that satisfies our users is presently available.

7. Workstation Issues – and the Future

Our users, and our libraries, are deploying more powerful workstations, and this raises expections about computer access in general.

7.1 User Expectations and Issues

The user expectations are starting to be met, partly through the explosion of client/server solutions coming via the World Wide Web and web browsers. Using existing (or arriving) WWW technology and clients many methods are available to present users with, and help them find, the desired information resource for their needs. These solutions range from standard HTML pages functioning as menus to searchable databases of resources which give back descriptions and pointers to relevant resources. A link on these pages can be either a pointer to text such as a president's speech, or a pointer to a database or set of databases using a Z39.50 URL. When a user selects a database search the web browser could open a Z39.50 session or invoke a companion application process to handle the link.

As appealing as the above scenario is, there are serious issues that need to be resolved. These issues have to do with who "owns" or handles the information connection.

- Who is responsible for the user's workstation as an information gathering tool? My organization is already getting calls from people who want network access, either to our services or to external information providers allowing network access. These range from people with dumb terminals to people with PCs who have never heard of nor want to hear of TCP, to people in restricted networks. Who, then, is responsible for the client software mounted on a user's computer? Although our view (and hope for client/server technology in general) is that these issues reside at the user's end, it is not clear users agree. In order to give users the greater control they desire, users presently must accept the burden of computer system administration as the overhead.
- Who is responsible for response time? Library literature has always claimed 3-5 seconds is desired response time. When our users follow an information link through the AT&T firewall out into the Internet, response time can be in minutes. Responsibility for this problem becomes murkier when the library is paying for the information access, and the user considers it unacceptable or unusable due to response time.
- Who is responsible for the functionality of the user's client software? This is further complicated since the functionality can vary depending on the remote server. There are issues of what functionality the client has (e.g., does it allow proximity searching), whether it interoperates correctly with the desired server, and whether the server supports the desired functionality. It is the author's belief that a well-designed client should present the user with

---

* Available at http://www.whitehouse.gov/White_House/Family/other/socks.au

functionality up to what it supports (e.g. Boolean operators, word proximity, USE attributes), and smoothly present the user with interoperability issues pertaining to a given server (e.g. the server fails the word proximity search request).

- Who is the user's point of contact? This issue intermixes with how databases are presented to the user: e.g. whose name and number appears on the screens, and who the user perceives is providing the information service. Setups where the user is presented with the information resource as coming from an internal provider as well as setups where the user is fully aware that a remote provider is involved are both used in the non-client/server world. Which setup is used is based on what the information provider and the intermediaries perceive as their role and relationship to the user.

- Who is responsible for the functionality and contract with the database server? This will be an early issue that needs to be resolved by my organization. If we purchase access to databases for a customer, we want the customer to help pay for the access and to be aware of the library's role, and we want user requests to be in our control and user feedback to come back to our organization. Whether we will be able to keep this degree of control in the new environment is uncertain – but the money issues are the bottom line. A number of solutions to this problem exist, but many details need to be resolved.

7.2 Library Expectations and Requirements

Our libraries have two points of interaction with databases; for searching and for systems access. In the case of searching, the staff have the issues (and desires) of users – they want a single integrated search environment. Library user wants are similar, but a single, consistent, user adaptable environment is most important since diverse users use the same system. Presently this goal has not been reached, as users are confronted with different interfaces for every CD-ROM product, laser-disk system, locally mounted OPAC-accessible databases, and, increasingly, OPAC-accessible externally mounted databases. Our vision is of a library OPAC that consists of client software accessing all the resources the library makes available through one common interface. Although we have attempted to achieve this in the past, without a common protocol and buy-in by the database providers this goal is basically unachievable.

Our library staff also accesses databases for all the basic functions of a library (e.g. for circulation, entering and tracking user photocopy requests, and for checking book status). It is not clear that workstation client/server access is an improvement for this functionality. However, if we decide to move in this direction, our environment combined with Z39.50 may make this a less painful move than might be expected. At a low level (below user interface) we should be able to take our present environment built on high level language scripts driving database access and updates and port it. That is, in theory we should be able to purchase a PC version of Perl, Awk, and shell and use this to provide our present functionality. Although we have little implementation experience at this level, we have reason to believe our environment has the desired flexibility and functionality to achieve a move of this scope.

8. Conclusion

The AT&T library organization developed an interest in Z39.50 for a number of diverse reasons. We hope that eventually Z39.50 will help with or solve several classes of problems ranging from behind-the-scenes issues resulting from distributed computing architectures to diversity of user interfaces. In addition to helping with known problems, we hope that Z39.50 will give us the flexibility we require for a constantly evolving library organization in an international corporate environment.

For Z39.50 to meet our needs, the main requirement is that the protocol itself incorporate all the functionality of our existing information retrieval environment. Version 2 of the standard was a major start, but version 3 comes much closer to incorporating existing functionality. In particular, Explain, generic record syntax (GRS), search restriction by attribute, new search information formats, and new access formats make the protocol viable in our environment with little or no non-conforming extensions to the protocol.

The next major requirement is proven interoperability and transparency of database providers to our users. This functionality is coming, though somewhat slower than hoped. Issues of indexing style, default operations and ways to override defaults, database coverage and loading characteristics become even more apparent in Z39.50 than in the traditional online world. However, we will soon be able to buy database access rather than mount tapes, with no loss of functionality or noticeable changes in access for our customers.

Our final requirement is that our end users, like users everywhere, are expecting interfaces integrated into their regular computing environment. Developing user interfaces for diverse user environments takes significant resources. Our users also expect the search tools to work the same, whether against internal or external resources. The clear solution to this problem is a well-accepted search and retrieval protocol. Z39.50 is well-positioned to become this protocol, and in this belief we have focused our attention on developing a high-quality server for our internal resources.

After three years of involvement in Z39.50, it appears that Z39.50 was the correct choice for flexible future growth of our organization. Z39.50 continues to gain acceptance and Z39.50 implementations continue to become more available. Acceptance by ISO is an important milestone. Increasingly, database suppliers (including CD-ROM vendors) offer Z39.50 access, making this technology an increasingly attractive alternative to mounting databases internally. The federal Government Information Locator Service (GILS) initiative makes access to government information, which is important at a corporation such as AT&T, a desired benefit. Finally, the growing interest in Z39.50 in the Internet community as demonstrated by development of Z39.50 URLs makes it likely that the less formal information resources of the Internet will also be available and searchable by a common protocol.

REFERENCES

1. Waldstein, Robert K. SLIMMER - a UNIX system based information retrieval system. *Reference Services Review* Vol 16, No 1-2, pp 69-76, 1988.
2. Wall, Larry and Schwartz, Randal L. *Programming perl*. O'Reilly & Associates, Inc. 1991.
3. Aho, Alfred V., Kernighan, Brian W, and Weinberger, Peter J. *The AWK Programming Language*. Addison-Wesley Publishing Company, 1988.
4. Waldstein, Robert K. *Library* an Electronic Ordering System. *Information Processing and Management* Vol. 22, No 1, pp. 39-44, 1986.
5. Turner,Fay *The ILL protocol and Z39.50*. Available via anonymous ftp from ftp.nlc.ca; in postscript (zillart.ps) or as ascii text (zillart.txt).