

**Final
Repository Plan Report
for the
OSD CALS IWSDB PROJECT**

An MVP Joint Venture

August 10, 1994

Submitted by Task 1 Team

ManTech International Corporation
Technology Applications Operations Center
1313 Locust Avenue
Fairmont, West Virginia

In support of
Contract DAAB10-89-D-0503
And in compliance with
CDRL Sequence Number A010

SOW Numbers 3.1, 3.2



Robert S. Kidwell
Technical Director

Jack G. Richman
ManTech International
Corp.

Robert E. Burkewitz, ACOR
USACIMMC

William C. Gorham
CALS Requirements and

OSD CALS

OSD CALS Project
Manager

Technical Assessment

TABLE OF CONTENTS

| | |
|---|-----|
| LIST OF FIGURES | iii |
| LIST OF TABLES | iv |
| 1.0 INTRODUCTION | 1 |
| 2.0 SUMMARY | 3 |
| 2.1 Core Repository Services | 5 |
| 2.2 Resources/Ancillary Services | 6 |
| 2.2.1 Systems Administration | 6 |
| 2.2.2 Qualification Team | 6 |
| 2.2.3 Technical/User Support | 6 |
| 2.2.4 Marketing Function | 7 |
| 2.2.5 Hardware/Software for Repository | 7 |
| 2.3 Choice of Repository Infrastructure | 8 |
| 2.3.1 Flexibility of Platforms for Browsers | 8 |
| 2.3.2 Choice in Platforms for Servers | 8 |
| 2.3.3 Information Navigation is Fast, Interactive and Highly Intuitive | 8 |
| 2.3.4 The WWW Works on Standards and is Extensible | 9 |
| 2.3.5 The Design of the WWW Provides for the Transfer of a Variety of Data Formats | 9 |
| 3.0 CONCLUSION | 10 |
| 4.0 TECHNICAL INFORMATION | 12 |
| 4.1 Overview of the WWW | 12 |
| 4.1.1 HTML and URLs | 14 |
| 4.1.2 WWW and Other Information Discovery Services | 14 |
| 4.1.3 Text-Based Browsers | 16 |
| 4.1.4 Graphical WWW Browsers | 16 |
| 4.2 Surveyed List of Browsers | 17 |
| 4.3 Servers | 19 |
| 4.4 Domain Stakeholders | 20 |
| 4.4.1 Use Group | 21 |
| 4.4.2 Create Group | 21 |
| 4.4.3 Manage Group | 21 |
| APPENDIX A: RESPONSES TO ACOR SELIM-CTM COMMENTS | A-1 |

LIST OF FIGURES

| Figure | Title | Page Number |
|---------------|---|--------------------|
| Figure 1.0-1 | Multi-Platform Access to CALS Test and Validation Tool Repository | 3 |
| Figure 2.0-1 | Virtual Schema of CALS Test and Validation Tool Repository | 4 |
| Figure 3.0-1 | Tools Repository Overview | 11 |

LIST OF TABLES

| Table | Title | Page Number |
|---------------|----------------------|--------------------|
| Table 2.2.5-1 | Server Activity | 7 |
| Table 2.2.5-2 | Manpower Requirement | 8 |

1.0 INTRODUCTION

This plan presents a broad vision and strategy to implement a CALS Test and Validation Tools Reuse Repository. The primary purpose of this repository is to enable acquisition personnel to determine whether a deliverable conforms to the relevant CALS standard. However, it is imperative that the repository offers services that all stakeholders of the CALS domain can utilize to ensure and support the long-term viability of the CALS initiative. For this report, a stakeholder is a person, group, or organization that has a vested interest in creating, maintaining, and using the re-use repository. Based on our survey of existing information discovery services, the World Wide Web (WWW) project supports the desire of establishing an open and virtual repository, because it provides cost-effective access to relevant services to a wide spectrum of users.

Each domain or area of concern has several stakeholders that have interests in generating, maintaining, and using the assets within the context of that domain. There are three main stakeholders for any given domain: users, creators, and maintainers/managers. An organization can play the roles of more than one stakeholder. For example, an organization may create, manage, and use the assets in a domain. These three functions encompass the life-cycle of domain assets. In the domain of CALS test and validation tools, vendors (or other developers) of test and validation tools fulfill the **create** function, libraries such as CALS Test Network (CTN) fulfill the **manage** function, and the organizations that utilize repositories to achieve mission requirements denote the **use** function.

The infrastructure for the CALS Test and Validation Tools Repository is based on the services that the repository offers to its users. In order to establish these types of specific services, the methods by which a repository can be practically implemented have been surveyed. The repository will be accessed by members of all the three groups mentioned above, namely the use group, the create group, and the manage group. The key aspect of the repository services will be the transfer of information to and from the repository to the user, whichever group the user may belong to. The repository will also have the added features of interfacing to news groups, transferring of files, accessing test and validation tools, accessing information on reuse qualification information, and other related services.

After surveying information discovery resources, we have decided on the WWW concept as the foundation for the repository infrastructure because information is distributed at different geographical locations such as the National Institute of Standards and Technology (NIST), CALS Test Network, CALS Shared Resource Centers (CSRC), etc. Also, such an implementation would provide the user virtual access to data. The hypertext and hypermedia interface would link all the information sources, and the entire system would be a virtual web of CALS test and validation tools information. A major factor leading to the selection of the WWW is the extent of hardware and software support available for the WWW and the relative simplicity of linking to such a vast web of information.

It is essential to note that there are existing programs that are intended to serve the needs of the CALS user community. To this end, our plan espouses an open and virtual repository that will

provide seamless access (wherever possible) to existing/emerging services within the CALS domain. Thus, the users from multiple platforms can be provided with a set of comprehensive services without incurring unnecessary costs from duplication. The following figure depicts a basic scenario for the virtual CALS test and validation tools reuse repository using multiple platforms with the WWW.

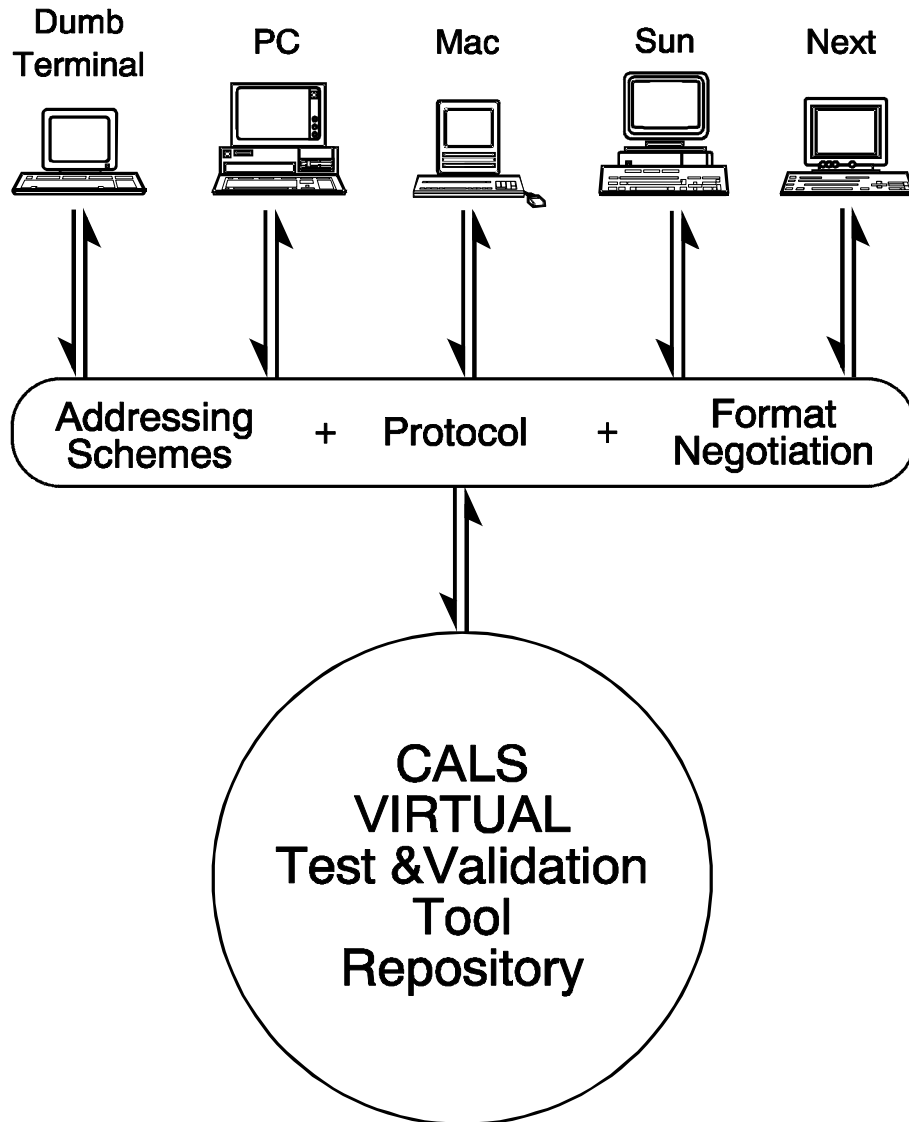


Figure 1.0-1 Multi-Platform Access to CALS Test and Validation Tool Repository

2.0 SUMMARY

The infrastructure for the CALS test and validation tools repository is based on the services that the repository offers to its users. The services offered by the repository are in turn predicated on the users' needs which the repository is intended to serve. Key user profiles in the CALS test and validation tools domain have been identified. These functions form the basis for the repository services. After surveying existing information discovery services, we chose the WWW to act as the underlying operating principle for the repository.

Any given domain has multiple stakeholders. Three key stakeholders fulfill the functions of use-create-manage. (In the jargon of marketplace, these functions are identified as supply side, demand side, and brokering function, respectively.) The end-users of the repository may belong to one or more of the above stakeholder groups.

The **use** group is typified by the "design with reuse" paradigm. In the **create** group, existing assets of the domain are utilized in creating new applications. The **manage** group function provides the necessary bridge between the use and create functions and encompasses the tasks of domain assets maintenance and evolution.

The CALS test and validation tools repository will use the WWW Hypertext Transfer Protocol (HTTP) protocol for data transfer. The repository will provide the user with hyperlinks that connect with other relevant CALS domains such as the CALS Shared Resource Centers, CALS Test Network, National Institute of Standards and Technology, and others. Figure 2.0-1 shows the expected connectivity of the CALS test and validation tools repository.

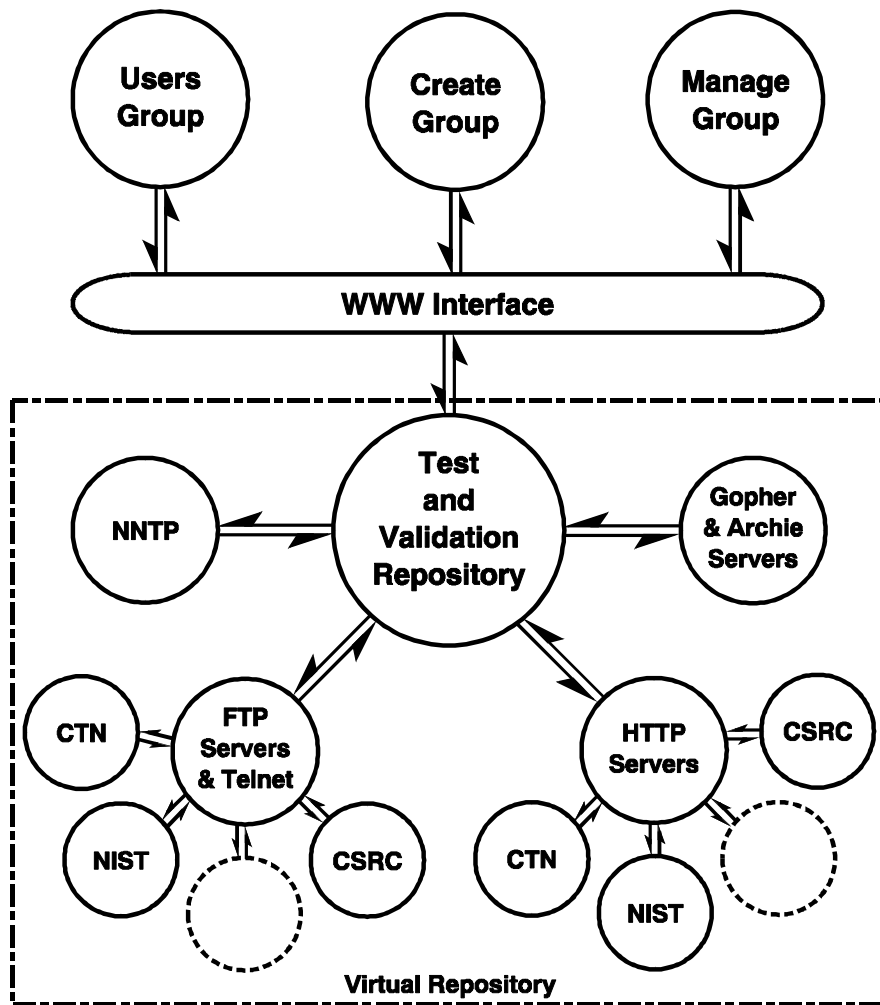


Figure 2.0-1 Virtual Schema of CALS Test and Validation Tool Repository

In this report, the issues concerning the CALS Test and Validation Tools Repository have been summarized into three categories, namely, Core Repository services, Ancillary Services/Resources, and Choice of Repository Infrastructure. These services are described next in detail.

2.1 Core Repository Services

Based on the user profiles identified earlier, the CALS test and validation tools repository will offer the following services. The following general services apply to all groups of end-users (use-create-manage). Specific services to each of these groups are also listed.

General

- Access to the repository from multiple platforms (DOS, MAC, UNIX, VMS, etc.).
- Access to information that is pertinent to the CALS test and validation domain (technical/research papers, upcoming conferences, updates to CALS standards, etc.). Discussion groups to critique existing/emerging CALS standards.
- Browsing and retrieving mechanisms to facilitate identification and use of assets.
- Training appropriate to each user group (may be on cost-reimbursable or cost-sharing basis).

Use Group

- Use of CALS test and validation tools to test data conformance to various standards (subject to licensing and access restrictions).
- Pre-qualified CALS test and validation tools that comply with repository's reuse standards. Repository's qualification process made available to the users.
- Membership in news groups that specialize with CALS test and validation tools issues. Appropriate cross-postings to/from other pertinent news groups in the CALS domain.
- Seamless access (where possible) to other pertinent services in the CALS domain.

Create Group

- Access to repository's qualification process and the various domain criteria specific to each standard and the tool class. These criteria can be used by the creators/vendors as requirements when developing new tools.
- Membership to news groups that specialize in the creation/development of CALS test and validation tools. Appropriate cross-postings to/from other pertinent news groups in the CALS domain.
- Access to information/literature about reusability issues, "design for reuse" guidelines, domain analysis guidelines, etc.

Manage Group

- Access to information relating to setting-up and operating a domain-specific repository such as Technical Concept of Operation, Operational Guidelines, infrastructure requirements (personnel, software, hardware, etc.), security/access restrictions, licensing and data rights issues, legal responsibilities, and the concomitant recommended courses of action, et al.

2.2 Resources/Ancillary Services

In support of the essential services provided to the different stakeholders as outlined above, ancillary services are necessary to maintain the smooth flow of operations in the repository. These services will mainly be involved in the general working, maintenance, and upkeep of the repository and ensure that the goals envisioned are reached. The anticipated ancillary services and their functions are described below.

2.2.1 Systems Administration

The systems administration group is the name for a person or persons responsible for the administration of the WWW site. The systems administrators are also known as webmasters. Webmasters typically set up and maintain the local WWW server and often handle support and questions regarding the site. A webmaster might also install other WWW related programs. The webmaster is responsible for the site activity and performs quality assurance checks and general upkeep of the server information. A small company or organization can set up a good site with a part-time webmaster. Intermediate sized sites should have a full-time employee to work on the web and related information resources. For a large organization, setting up a professional, well-maintained site requires at least three people sharing duties and coordinating skills. The scope of the intended repository demands a full-time Webmaster.

2.2.2 Qualification Team

The qualification team would consist of several persons whose primary responsibility would be to generate qualification criteria and evaluate new software tools against the established guidelines. The framework for the qualification of tools will be presented in the Criteria and Test Procedures report (CDRL #A003/A006). The data entering the system must also be validated with the tools in the repository. The team will assist the stakeholders in the use-create-manage groups on handling data. The team will update and keep information in the repository in synchronization with the changes in the industry and standards. Another responsibility would be to interface with vendors of the tools in the repository to keep current versions of software in the libraries.

2.2.3 Technical/User Support

The technical and user support group is responsible for handling queries from repository users. They will answer questions regarding the data in the repository, configuration of the tools for user specific needs, and the qualification criteria, and/or methodology of the information interchange. The group will interface with members of the qualification team on a regular basis to eradicate inconsistencies and to initiate updates in the qualification process. Members of the technical support group will be dedicated to keeping themselves abreast of any technological advancements and standards changes that may take place. Each sub-domain of the test and validation tools repository will have a person or persons who have a thorough understanding of the intricacies of the standards and operation of the tools for which they are responsible.

2.2.4 Marketing Function

A marketing group is necessary in a virtual repository environment. The group will work in conjunction with the qualification team in procuring new tools and identifying external entities which can be a part of the virtual repository of information. The group will also make the CALS industry aware of the repository facility and encourage its use for validation of CALS data in a test tools environment. Members of the marketing and technical groups will also act as liaisons with existing repository services and influence emerging services within the domain.

2.2.5 Hardware/Software for Repository

The virtual CALS test and validation tools repository will basically consist of a WWW server configured to access other relevant servers in the domain of CALS testing and validation tools. Document creation, organization, and maintenance play a major part in creating a good WWW service. Deciding on the platform on which the server resides depends on the load expected to be serviced by the server. The installation, upkeep, and maintenance will be handled by the systems administration functional team. The average amount of traffic handled by a server depends upon the nature of the organization and the audience to the information stored. Table 2.2.5-1 shown below gives a reasonable approximation of the necessary server for different workload environments.

Table 2.2.5-1 Server Activity

| Activity | Requests/Day | Organization Type | Platform |
|-----------------|---------------------|---|---|
| Light | 0-2000 | Small Campus, College Departments, Small Company | Low end computer platform |
| Intermediate | 2,000-5,000 | Midsize Company, Small University, Small Company LAN | 486 PCs, Macintosh II, UNIX Workstations |
| Mid-Heavy | 10,000-20000 | Large Company, Major University, Mid-range Service Providers | Pentium PCs, PowerMac, SparcStation 10, DEC or HP Machines |
| Heavy | 30,000 and up | Major Providers | Higher end servers like SparcCenter 2000 |

A request is considered as a data transfer in the HTTP protocol. Typically there will be more than one such request in a user session. The number of such requests in a user session cannot really be determined until the repository is completely established, the depth of data in the repository is setup, and the data transfer is logged. For the CALS test and validation tools domain, it is anticipated that a server capable of handling the mid to heavy workload will be necessary considering the volume of transactions that will take place on a day-to-day basis in the CALS domain in the future.

The estimated full-time manpower requirements of the ancillary services are tabulated below.

Table 2.2.5-2 Manpower Requirement

| Ancillary Service | Manpower (Full Time Employees) |
|-----------------------------|---------------------------------------|
| Systems Administration | 1 |
| Qualification Team | 3 |
| User/Technical Support Team | 2 |
| Marketing Function | 1 |

2.3 Choice of Repository Infrastructure

The WWW has been chosen as the front end of the CALS test and validation tools repository for many reasons. Many companies and organizations have set up WWW servers that allow them to distribute hypermedia documents across the Internet and local networks. The essential reasons for choosing the WWW are listed below.

2.3.1 Flexibility of Platforms for Browsers

The WWW is designed to convey information in a device independent manner. Hypermedia documents can be shown on many different types of computers. In other words, documents on the WWW are shown to users in a way that is best suited to their display. Documents with graphics and formatting may be shown in full color on an X-terminal, while only the basic formatting and text will be shown on a text-only screen. Section 4.2 contains a comprehensive list of the browsers currently available.

2.3.2 Choice in Platforms for Servers

WWW server software is also available for different types of hardware platforms. This makes it easier to establish a WWW server without the need for specific hardware requirements. Section 4.2 contains a list of the platforms available. Section 4.3 contains a list of the servers available.

2.3.3 Information Navigation is Fast, Interactive and Highly Intuitive

Documents on the WWW contain hot-spots in the form of hyperlinks. Forms and menus allow users to search quickly through archives and documents. Multimedia presentations, on-line forms, and database interfaces can be created so that users can interact and control information

rather than just view it. Such a facility is well suited to the test and validation repository where information transfer and modification need to be performed.

2.3.4 The WWW Works on Standards and is Extensible

Most WWW clients and servers have been designed to communicate using Transmission Control Protocol/Internet Protocol (TCP/IP). Many web clients also speak the Gopher protocol, File Transfer Protocol (FTP), and Network News Transfer Protocol (NNTP) so that hypermedia browsers can also act as Gopher, FTP, and USENET client applications.

2.3.5 The Design of the WWW Provides for the Transfer of a Variety of Data Formats

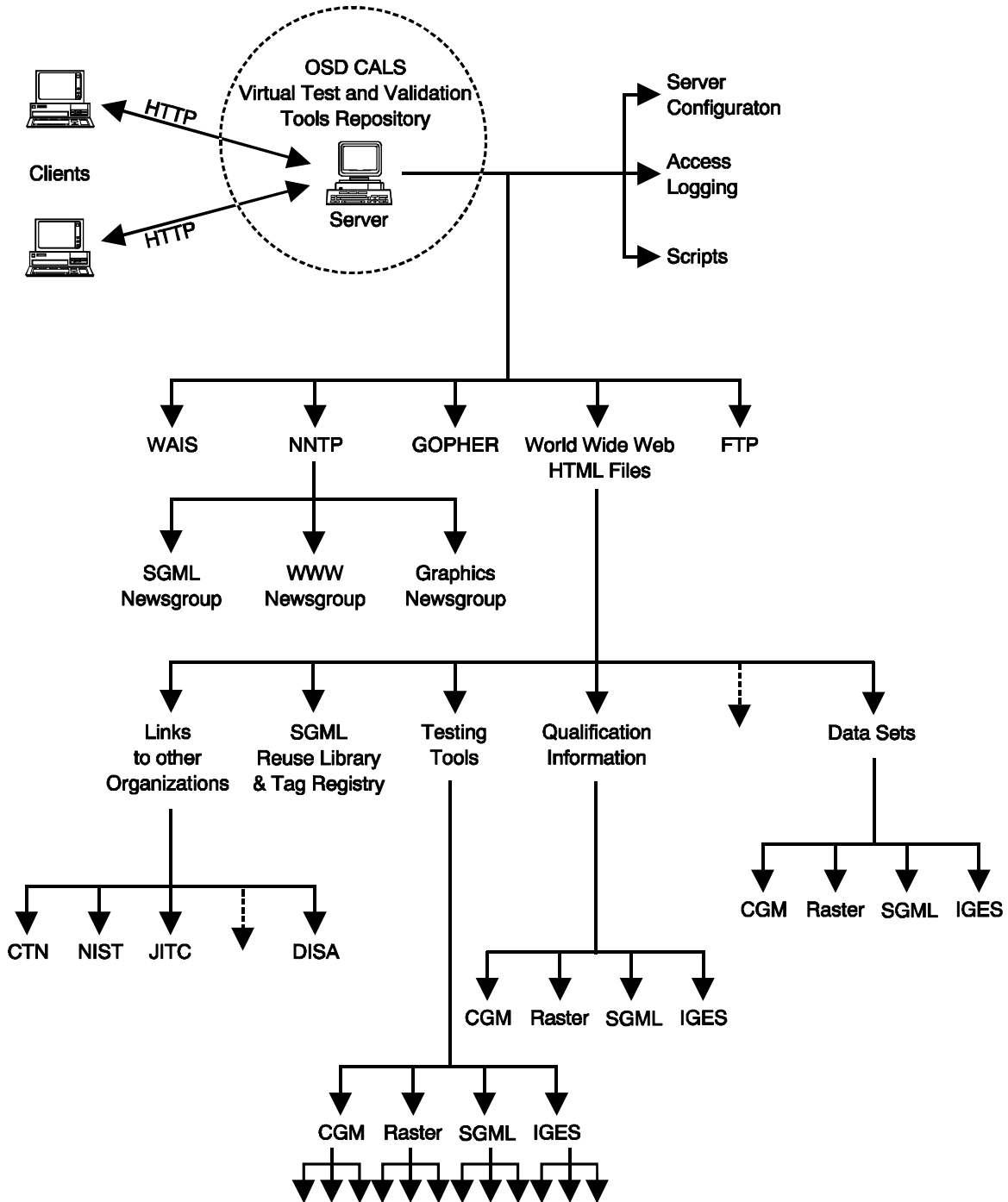
Audio, video, extended character sets, and interactive hot-spot graphics are some of the fundamental types of data that can be transferred using the WWW. There are actually no barriers to the types of data sent by the server as long as the client or browser has an application associated with the file data type to read it. This is essentially a very important feature that can be taken advantage of for file data types that are not a standard as of yet. As and when the formats become pertinent, the clients can have the respective software to access up the file. This feature has implications in the domain of CALS test and validation tools. Data files for Standard Generalized Markup Language (SGML), Computer Graphics Metafile (CGM), RASTER and Initial Graphics Exchange Specification (IGES) can be ported across the Internet while keeping the file format intact. The files will then be submitted to the respective application by the browser.

Our team has successfully installed WWW servers on a variety of platforms at the ManTech Services Corporation office complex in Fairmont, West Virginia. The server software was installed and checked on UNIX, Linux and Windows-NT host platforms. The server software was obtained by anonymous FTP to the National Center for Supercomputing Applications (NCSA). The Mosaic WWW browsers also were installed on our computers. Preliminary trial front end applications for a CALS test and validation tools repository were developed in the Hypertext Markup Language (HTML). The servers were checked by allowing persons to login to the HTTP WWW server from external nodes on the Internet. Further development is underway.

3.0 CONCLUSION

The test and validation tools repository will offer comprehensive services that all the stakeholders (demanders, suppliers, and brokers of CALS test and validation tools marketplace) can utilize. The final repository plan espouses a virtual repository concept (using the WWW) so as not to duplicate existing/emerging services in this CALS domain. The repository infrastructure is intended to serve users from differing operating environments. Finally, the repository is conceived to be a centerpiece in assimilating, disseminating, nurturing, evolving, and transitioning CALS standards and technologies to the user community.

Figure 3.0-1, Tools Repository Overview, depicts the services, test suites, newsgroups and other organizations accessible to the repository clients. Given the explosive growth of the WWW, we anticipate continuing improvement in both the breadth and depth of services offered and information made available through the OSD CALS Virtual Test and Validation Tools Repository.



Legend:

CGM = Computer Graphics Metafile
 CTN = CALS Test Network
 DISA = Defense Information Systems Agency
 FTP = File Transfer Protocol
 HTML = HyperText Markup Language
 HTTP = HyperText Transport Protocol

IGES = Initial Graphics Exchange Specification
 JITC = Joint Interoperability Testing Center
 NIST = National Institute of Standards and Technology
 SGML = Standard Generalized Markup Language
 WAIS = Wide Area Information Server
 WWW = World Wide Web

Figure 3.0-1 Tools Repository Overview

4.0 TECHNICAL INFORMATION

This section details the WWW concept, the HTML format, other information discovery services, WWW browsers, WWW servers, and an outline of domain stakeholders.

4.1 Overview of the WWW

The WWW is officially described as a "wide area hypermedia information retrieval initiative aimed at giving global access to a large universe of documents." The WWW has accomplished the task of providing users on computer networks a consistent means of accessing a variety of media in a simplified manner.

The WWW relies mainly on hypertext as its means of interacting with users. Hypertext is basically the same as regular text because it can be stored, read, searched, or edited. However, an important exception is that hypertext contains connections within the text to other documents. These connections are known as hyperlinks which create pathways to other documents and links. Hyperlinks, therefore, facilitate an intricate and complex web of connections between documents. Hypermedia is hypertext with a difference: hypermedia documents contain links not only to other pieces of text, but also to other forms of media like sounds, images, and movies. In other words, hypermedia simply combines hypertext and multimedia.

The WWW is used on the Internet. The WWW refers to a whole body of information (an abstract space of knowledge) while the Internet refers to the physical side of the global network (the giant mass of cables and computers). The WWW uses the Internet to transmit hypermedia between computer users internationally. People and organizations are responsible for the documents they author and make publicly available to millions of users around the world on the Internet.

The WWW project was initiated in March 1989, by a collective of European high energy physics researchers who proposed the project to be used as a means of transporting research documents and ideas effectively throughout their international organization. The initial project proposal outlined a simple system of using networked hypertext to transmit documents. Since then, hundreds of people around the globe have participated in contributing their time to the writing of WWW software. The project reached global proportions and is now one of the premier forms of information exchange in the world. It is extremely popular because of its simplicity of use and excellent user interface capacity. The WWW offers a very user-friendly interface to the traditionally hard to master resources of the Internet. It is probably this ease of use as well as the popularity of many graphical interfaces to the WWW that caused the explosion of the WWW traffic in 1993. The potential of using networked hypertext and multimedia has prompted many users to create and explore countless innovative applications on the Internet.

The WWW software is designed around a distributed client-server architecture. The WWW client (also called a WWW Browser) is a program that can send requests for a document to any WWW server. A WWW server is a program that, upon receipt of a request, sends the document back to the requesting client. Using a distributed architecture means that a client program may be running on a completely different and separate machine from the server. The task of

document storage is left to the server and that of retrieval is local to the client. Therefore, each program can concentrate on its individual duties and work independently of each other. Because servers operate only when documents are requested, they put a minimal amount of workload on the computers they run on for each transaction. The WWW is composed of thousands of these virtual transactions taking place per hour throughout the world, creating a web of information flow.

The language that WWW clients and servers use for communications is called the HTTP. The standard language used for creating and recognizing hypermedia documents is the HTML. It is loosely related to the SGML. Start and end tags (markup) that precede and follow each logical portion of a document are added to the standard text to represent different attributes of the enclosed text. The entities are defined by the HTML Document Type Definition (DTD). The HTML DTD defines the elements and tags of the SGML subset. WWW documents are written in HTML. HTML documents are nothing more than standard ASCII files with formatting codes that contain information about layout (text styles, document styles, paragraphs, lists) and, of course, hyperlinks. A new version of HTML called HTML+ will be ready by the end of 1994 and will support more interactive forms, hot-spots in images, more versatile layout, and formatting options and formatted tables. Related to HTML is HyTime, a proposed standard language for representing the structure of multimedia, hypertext, hypermedia, and time and space based documents. HyTime, Hypermedia/Time based Document Structuring Language is also based on SGML as information content is packaged using standard markup. HyTime was basically created for the digital information publishing industry. HyTime can be used as a means of integrating the information management of a multifaceted enterprise, facilitate concurrent engineering and other forms of collaborative research.

The WWW uses Uniform Resource Locators (URL) to represent hypermedia links and links to network services within HTML documents. It is possible to represent any file or service on the Internet with a URL. Most WWW browsers allow the user to specify a URL and connect to that document or service. When a hyperlink is selected within a document, the user is actually sending a request to open a URL. In this way, hyperlinks can be made not only to other texts and media but also to other network services. WWW browsers are typically not just WWW clients but are also full-featured FTP, Gopher, and Telnet clients. HTML+ will allow the use of URL's to perform E-mail functions automatically.

The WWW exists virtually. There is no standard way of viewing it or navigating around it. Many software interfaces to the WWW exist that have similar functions and work in the same way, no matter what computer or type of display is used. A WWW browser, as mentioned earlier, is used to navigate a user through the WWW. The browser is a software program on any computer with a graphical or textual interface, such as a Macintosh, an X-Windows systems, or even an IBM-compatible with Windows. WWW clients or browsers are available for different varieties of platforms and environments.

The explosive growth of the WWW brings close to realizations the concept of everyone having access to any document, sound recording, or video image on a computer screen. The underlying principle of the WWW is a simple concept: the combination of a HTML and the URL. The

HTML file can be created with any standard word processor adding special HTML pointers, markers, and style tags. The encoded information is used by the user's WWW software to interpret the layout and style and to make internal and external hyperlinks. The URL addresses enable the HTML to link to any available resource around the Internet.

4.1.1 HTML and URLs

HTML is a markup language related to SGML. SGML grew out of a decade of work addressing the need for capturing the logical elements of documents as opposed to the processing functions to be performed on those elements. SGML is essentially an extensible document description language, based on a notation for embedding tags into the body of a document's text. It is defined by the international standard International Organization for Standardization (ISO) 8879. The markup structure permitted for each class of documents is defined by an SGML Document Type Definition (DTD). It is impractical to design a DTD to meet the needs of all possible users. Instead, the markup has been tailored to the needs of a specific community.

In HTML, the need to support a wide range of display types and to keep browser software as simple as possible limits the complexity that can be handled. Text is tagged with marks that pass information to a software reader. The readers use the tag information to format the text and other media for the viewer. As mentioned earlier, HTML is soon to be superseded by HTML+ which contains additional features and enhancements. HTML+ has grown out of several years of experience with the HTML document format of the WWW community. Browser writers are experimenting with extensions to HTML and it is appropriate to draw these ideas together into a revised document format. The new format is designed to allow a gradual roll over from HTML, adding features like tables, captioned figures, and fill-out forms for querying remote databases or mailing questionnaires.

Basically, HTML uses tags to tell the WWW browser how to layout the text and also how to make links to other parts of the same document or documents. The location of the document is specified by a Uniform Resource Locator. The URL is a system of identifying where a resource resides anywhere on the Internet. In the same way that a local computer needs to know where a file is stored in order to open it, the URL locates a file as long as the entire path of the file is specified. The data types that the URL can address are not limited to files alone. Databases, images, news groups, Gopher and Archie servers, and many other services are capable of being addressed via URLs.

The URL is a simple concept and can be grasped quite easily. The most straightforward use of URLs is to point at other WWW documents on HTTP servers. If the URL of a document is known, a browser can be pointed there. Generally, however, the navigation of the WWW is through other WWW documents; and the URLs of these documents need not be known. Most browsers inform the user of the URL to which a certain link will take them.

4.1.2 WWW and Other Information Discovery Services

The WWW is intuitive in use and exists on many platforms, from simple text-based browsers to full graphical implementations. It is an excellent way of cruising the information space offered by

the Internet resources. The WWW is currently expanding at an estimated 4 times as fast as the Internet explosion in the late 1980's.

The WWW is an attempt to unify the enormous amount of information available from the global networks, with a very easy-to-use front end interface and a server protocol. It may be thought of as a single tool to combine individual network tools dispensing the need to run these tools independently. Some of the more widely used tools are Anonymous FTP, Archie, Wide Area Information Server (WAIS), and Gopher.

Anonymous FTP is one of the most useful of Internet facilities, establishing the original archiving concept, making it possible to make information available across the globe. The addition of a document search engine, namely Archie, has made it possible to locate files in the entire information space. However, Archie can give only a location. It is still necessary to use an FTP client to retrieve the file. This problem was partially addressed by Gopher by providing on-line viewing and retrieval facilities within a client package. The addition of Veronica to Gopher filled the need for a search engine within the abundance of Gopherspace. WAIS brought the power of full text searches and questioning and of resources. Even though these tools and utilities allowed the searching of the Internet and the retrieval of information and resources found therein, searches and all finds continued to exist generally independently of each other.

The WWW is based on the concept of links. The underlying language is powerful and yet extremely simple. Within any WWW HTML document, there can be links to other documents, links to marks within the same document, and links to other resources such as FTP sites, Gopher servers, WAIS servers, news groups, and finally to the fast growing list of experimental services being developed.

The WWW relates to existing resources in a simple manner: pointers can be easily built into text documents to specify the location of the target. The target can be a text file, an image, a specific location in a text file, a Gopher server, FTP site, or various other types of information. The work of locating and displaying the target information is done within the local application browser. The information is transferred across the Internet and is displayed as per the rules of the local client browser application. All formatting and layout information is constructed locally. For this reason, WWW browsers return different results depending on their sophistication. WWW browsers will also interpret information from FTP and Gopher sites easily. The format of data transfer used in the WWW is called the HTTP. All the WWW servers have to do is honor this communications protocol when information is transferred.

HTTP/WWW works by making the embedded hyperlinks manifest. These hyperlinks are inserted into the documents using the HTML. The hyperlinks are not seen directly by the user, but the effects are shown in the browsers. Browsers vary depending upon the sophistication and power of the platform, from simple line mode browsers to a full Graphical User Interface (GUI). However, all browsers have a common purpose which is to allow a user to navigate around the WWW from link to link, with as little trouble as possible. The software makes these connections in the background. Text-based browsers show a number next to each hyperlink which can be chosen to activate that link and initiate information transfer. A browser such as Lynx highlights

links and allows the user to move to different points on the screen using arrows. Full graphical interface browsers use colors or underlining to show the hyperlinks and mark text. Backtracking to previously accessed locations, going back to a home page starting point, or hot listing (collecting favorite places to return in later sessions), keeping track of places visited in current session, and searching by keyword are some of the features that make using such browsers productive tools.

All these features combined with the global nature of the WWW accessing methods make it a natural front end choice in a virtual CALS Test and Validation Tools Repository where information is distributed across many sites across the globe. A closer look is taken into some of the more popular browser and server software available currently. The browsers are mainly divided into two categories: text-based and graphical-based browser systems.

4.1.3 Text-Based Browsers

Line Mode Browser

This is the most basic program for WWW access. It is a general purpose information retrieval tool that gives WWW readership to anyone with a dumb terminal. Although it may not be as flashy as a window implementation, it covers a wide class of users who still do not have window facilities. It is important to realize that a user is still viewing the exact same document as would a user with full graphical (minus images) interface. There are no cut-down versions for text-based systems. Links are marked by a number in a square bracket; pressing the number on the keypad will initiate the link. The line mode browser will run on a variety of systems including UNIX systems, VMS with any flavor of TCP/IP, VM/CMS, DOS-PC, MVS, and the Macintosh.

Lynx Full Screen Browser

This is a hypertext browser for vt100 monitors using full screen, arrow keys, and highlighting. It is similar to the Line Mode Browser, but it uses highlighting to mark hypertext rather than numbers. These are navigated through by using the cursor and the arrow keys. Up and down arrow keys move the user back and forth sequentially through the links. The right and left arrow keys allow the user to jump back through links. In a situation where searching is allowed, pressing the forward slash key lets the user search for a text string.

4.1.4 Graphical WWW Browsers

Cello

Cello is a multi-purpose MS Windows Internet browser which permits a user to access information from many sources in different types of formats. Cello can access data from the WWW, Gopher, FTP, CSO, X.500 directory servers, WAIS servers, Hytelnet, techInfo, and others through external gateways. Cello can be used with the WWW HTML hypertext markup standard to build local hypertext systems on Local Area Networks (LANs) or on single machines. Cello also permits the post processing of any file to a Windows application through the file manager.

Mosaic

Mosaic was developed by the Software Design Group of the NCSA to interface with the WWW. Initial versions of Mosaic have been developed for X-Windows, Microsoft Windows, and Apple Macintosh. This is freely available, simple to use and has contributed greatly to the explosive use of the WWW. It has a great level of consistency across the different platforms and handles services such as FTP, Usenet, Gopher, and WAIS excellently. The Mosaic project defined the ideal set of features, with the aim of consistency across platforms. Mosaic has the ability to display:

- Hypertext and Hypermedia documents.
- Electronic text in a variety of forms.
- Text in bold and Italic.
- Layout elements such as paragraphs, bulleted lists, and quoted paragraphs.

The capability to support hypermedia - audio, video, graphics, extended character sets, and interactive hot-spot graphics is fundamental to Mosaic. These processes are accomplished using external software that is linked to Mosaic. When a special hypermedia link is initiated, the external software is called, and the file needing to use the application is submitted for execution. Mosaic has the capability to support and make hypermedia links to FTP, Gopher, Telnet, NNTP, and WAIS servers. Mosaic will also keep track of where a user has been, offering a list of visits which enables quick backtracking, useful when a dead end appears at the end of a long line of link traversals. Mosaic allows the user to add sites to hot lists of URLs, perfect for those must visit sites. Mosaic also offers the capability to cache copy of pages the user visits, so during backtracking, downloads are not necessary.

Due to its consistency across different platforms, using Mosaic varies very little from one variety to another. It is quite straightforward and intuitive in nature. Shown next is a comprehensive list of available browsers. It is important to note that browsers are available for all the important host platforms available.

4.2 Surveyed List of Browsers

Microsoft Windows browsers

Cello Browser from Cornell Legal Information Institute.

Mosaic for Windows from NCSA.

MSDOS browsers

DosLynx

A text-based browser.

Macintosh browsers

Mosaic for Macintosh From NCSA. Full-featured.

Samba From CERN.

Amiga browsers

A **Mosaic Browser for AmigaOS**, based on NCSA's Mosaic. Supports older Amigas as well as the newer machines in the latest versions.

NeXT Step browsers

OmniWeb A WWW browser for NeXTStep.

WWW, CERN's NeXT Browser-Editor. A browser/editor for NeXTStep.

X/DecWindows (graphical UNIX, VMS) browsers

NCSA Mosaic for X UNIX browser using X11/Motif. Multimedia magic. Full http 1.0 support including PUT-method forms, image maps, etc.

NCSA Mosaic for VMS browser using X11/DecWindows/Motif. For the VMS operating system. Multimedia magic. Full http 1.0 support including PUT-method forms, image maps, etc.

tk UNIX WWW Browser/Editor for X11 (Beta test version).

MidasWWW Browser A UNIX/X browser from Tony Johnson (Beta, works well).

Viola for X (Beta) Viola has two versions for UNIX/X: one using Motif, one using Xlib (no Motif). Handles HTML+ forms and tables.

Chimera UNIX/X Browser using Athena (does not require Motif). Supports forms, in-line images, etc.; closest to Mosaic in feel of the non-Motif X11 browsers.

Text-mode UNIX and VMS browsers

Line Mode Browser This program gives W3 readership to anyone with a dumb terminal. A general purpose information retrieval tool.

Lynx full screen browser. This is a hypertext browser for vt100s using full screen, arrow keys, highlighting, etc.

PerlWWW A tty-based browser written in perl.

VMS Dudu Rashty's full screen client based on VMS's SMG screen management routines.

Emacs w3-mode W3 browse mode for emacs. Uses multiple fonts when used with Lemacs or Epoch. See the documentation.

The variety of platforms for which browsers are available is shown below.

| Text Only Browsers | Graphical Interface Browsers |
|--|---|
| Dumb terminal, any UNIX platform | Sun 4/Sun OS 4.1.x |
| Text only, Sun OS, IBM AIX, DEC Ultrix, VAX Multinet | Silicon Graphics IRIX 4.x |
| Macintosh text-only, Mac SE's Sys 7.x | VMS |
| Perl | Linux |
| emacs | DEC MIPS Ultrix, DEC Alpha AXP, OSF/1 |
| | IBM RS/6000, AIX 3.2 |
| | HP 9000/700, HP/UX 9.x |
| | NeXT, NeXTStep 3.0 |
| | Commodore Amiga, AmigaOS 3.0 |
| | IBM compatibles, 386 and above with Windows 3.1 |
| | Macintosh computers System 7.x |
| | Power Macintosh |

4.3 Servers

Servers are available for UNIX, Macintosh, MS Windows, and VMS systems.

UNIX Servers

NCSA httpd

NCSA has released a server, known as the NCSA httpd; it is available at the URL <ftp://ftp.ncsa.uiuc.edu/pub/web/>.

CERN httpd

GN Gopher/HTTP server

The GN server is unique in that it can serve both WWW and Gopher clients (in their native modes). This is a good server for those migrating from Gopher to WWW, although it does not have the server-side-script capabilities of the NCSA and CERN servers.

Perl server

There is also a server written in the Perl scripting language, called Plexus.

Macintosh Servers

MacHTTP

This is a server for the Macintosh family.

MS Windows and Windows NT Servers

HTTPS (Windows NT)

HTTPS is a server for Windows NT systems, both Intel and Alpha-based.

NCSA httpd for Windows

The NCSA httpd for Windows has most of the features of the UNIX version, including scripts.

SerWeb

A simple, effective server for Windows. There is also a Windows NT version of SerWeb.

WEB4HAM

Another Windows-based server.

VMS Servers

CERN HTTP for VMS

A port of the CERN server to VMS.

Threaded HTTP Server

A native VMS server which uses DECthreads(tm). This is a potentially major performance advantage because VMS has a high overhead for each process.

| WWW Server Software |
|--|
| Most flavors of UNIX. |
| HP, SGI, and SUN systems. |
| DEC MIPS Ultrix and DEC Alpha AXP. |
| Perl. |
| Macintosh, 68020 or better, Power Macintosh, and System 7.x. |
| NeXTStep. |
| VM, VM/CMS, VM/XA, and VMS. |
| Windows 3.1 and Windows NT. |

4.4 Domain Stakeholders

This section further describes the domain stakeholders and their functions.

4.4.1 Use Group

The use group is typified by the "design with reuse" paradigm. Existing assets of the domain are utilized by this group in creating new applications. A large fraction of existing repositories serve this user group by cataloging/classifying assets that can be retrieved by the users so as to build new applications. Program libraries, math libraries, etc., are examples of repositories serving this group.

4.4.2 Create Group

The create group is typified by the "design for reuse" paradigm. Assets are created by this group so that these assets can later be reused in building other applications. These assets may be created while developing an application in the domain or may be created solely for the purpose of supporting new applications in the future. When creating these reusable assets, the create group must anticipate and support the requirements of future applications. The create group has to derive and anticipate the requirements of the user group in order to generate reusable assets. In essence, the success of "design for reuse" paradigm determines the efficiency of "design with reuse" paradigm. Thus, it is important for a repository to offer services that support the functions of this group.

4.4.3 Manage Group

The manage function is typified by a repository infrastructure. This function provides the necessary bridge between the use and create functions and includes the tasks of domain asset maintenance and evolution. As the underlying hardware/software technology changes and as the requirements of new applications evolve, the reusable assets have to necessarily evolve so as to be useful to the user community. A repository centralizes the tasks of asset maintenance and evolution and offers consistent and quick access to the latest advances in the domain to the user community. The repository serves as a forum to exchange information among users of the same group as well as users between different groups. This will enhance the quality of the assets/information within the repository.

APPENDIX A: RESPONSES TO ACOR SELIM-CTM COMMENTS

[Author's Note: It is noteworthy that the information used to formulate responses to the government's comments was, for the most part, found by utilizing the tools we are recommending for the repository, i.e., Mosaic, WWW, and the Internet.]

This appendix is intended to provide backup information addressing the comments received from the ACOR SELIM-CTM on July 26, 1994, re the Preliminary Repository Plan Report. The comments, responses, and references are listed in the pages that follow.

4. *Preliminary Repository Plan Report comments:*

a. *Page 9: Define Z39-50 protocol in Section 5.1.2.*

The Preliminary Repository Plan Report looked at a variety of approaches to creating a virtual reuse repository. One of the protocols investigated was Z39-50. This is a standard entitled "**Z39-50-1992 Information Retrieval Application Service Definition and Protocol Specification for Open Systems Interconnection.**" This specification defines a standard protocol for use in automated library systems. Implementation of this standard allows a given library to extend its reach beyond its own collection to tap the resources of remote collections and databases. The WAIS communicate through an extension of this protocol standard. Although useful in particular domains, Z39-50 was eliminated from consideration early on for the purposes of our study. The following references are provided for additional information:

1. Clifford A. Lynch. The Z39-50 information retrieval protocol: An overview and status report. *ACM Computer Communication Review*, 21(1):58-70, 1991.
2. Martin L. Schoffstall and Wengyik Yeong. A critique of Z39-50 based on implementation experience. *ACM Computer Communication Review*, 20(2):22-29, April 1990.

(Reference 2 is included at the end of this appendix for your information.)

4. *Preliminary Repository Plan Report comments:*

b. *Page 12: Expand upon the loose comparison of HTML to SGML, and the HyTime standard.*

HTML and SGML

HTML is based on the SGML which is an international standard for document markup that is becoming increasingly important. The term markup derives from the way proof-readers have traditionally penciled in marks that indicate how a document is to be revised.

SGML grew out of a decade of work addressing the need for capturing the logical elements of documents as opposed to the processing functions to be performed on those elements. SGML is essentially an extensible document description language, based on a notation for embedding tags into the body of a document's text. It is defined by the international standard ISO 8879. The markup structure permitted for each class of documents is defined by an SGML Document Type Definition, usually abbreviated to DTD.

Working groups in ISO have produced a range of SGML DTDs for documents: e.g., ISO 12083 defines DTDs for books, and ISO 10744 defines the HyTime architectural forms standard for hypermedia/time-based documents. The Text Encoding Initiative (TEI) is an international research project for SGML-based document exchange in the humanities. Publishers are cooperating to produce common DTDs for computer manuals, e.g., the DocBook DTD. The US Department of Defense (DoD) CALS Program defines SGML DTDs for documentation for defense procurement contracts. These standards are large and complex, and perhaps best suited as interchange standards that facilitate conversion between proprietary document formats.

So what differentiates HTML from these efforts? HTML provides a lightweight delivery format that can be rendered by relatively simple browsers, and which has grown out of the practical experience with wide-area hypertext information systems in the Internet community. It is impractical to design a DTD to meet the needs of all possible users. Instead, the markup has to be tailored to the needs of a specific community. HTML is aimed at fulfilling the dream of a web of information freely accessible over the Internet with links between documents spanning continents. The need to support a very wide range of display types and to keep browser software as simple as possible limits the complexity that can be handled. Similarly, the disparate needs of authors has led to the inclusion of limited rendering hints. The features supported and the enhancements in work arise from several years experience with the WWW and the existing HTML format usage.

HTML and HyTime

The HyTime standard provides a rich range of architectural forms but is not aimed at run-time efficiency. Suggestions have been made as to how the HTML DTD could be adapted to comply with HyTime's clink architectural form [Kimber 93]. This would necessitate documents declaring links as external entities and the use of local names in link definitions, but in the absence of any immediate benefit, there has been little enthusiasm for this within the WWW community. Instead, it is believed that a straightforward filter program should be used to map HTML and HTML+ documents into a format that is strictly compliant with HyTime, when this becomes appropriate.

Notes from the "First International Conference on the Application of HyTime," July 26-27, 1994, in Vancouver, British Columbia, Canada (Attended by Don Reynolds, ManTech, for OSD CALS Task 5 - IETM's):

The explosive growth of the WWW on the Internet has caught the attention of everyone involved in SGML, HyTime, and general hypermedia research. There was a lot of discussion concerning the HTML and its relationship to SGML and HyTime at the First International Conference on the Application of HyTime in Vancouver, Canada, on July 26-27, 1994, HyTime94. The incorporation of HyTime into HTML has been investigated from both the HTML perspective [1] and the HyTime perspective [2].

SGML is used to define markup languages (tag sets) for any desired document structure. The document structure (allowable set of tags) is defined in a DTD. HTML is a markup language (or set of tags) that was created in accordance with the conventions outlined in the SGML ISO

standard. HTML is, in fact, not a new markup methodology but rather a specific instantiation of a particular SGML document type.

Although there currently exists a DTD for creation of HTML documents, certain commonly used aspects of SGML are not supported by HTTP/WWW servers and therefore cannot be used in HTML documents. Also, the DTD for HTML does not reference or use any hyperlinking architectural forms from the HyTime standard.

HyTime, unlike HTML, is not itself a markup language. The HyTime standard defines a particular set of architectural forms (in the form of SGML elements and attributes) that can be used within SGML documents for linking data contained in or referenced by other arbitrary SGML documents. HTML currently uses a hyperlinking and addressing scheme that does not follow the HyTime standard and also does not follow commonly used SGML practices.

Making HTML an accepted and fully conforming application of both SGML and HyTime is addressed to some extent in [2]. In terms of linking, hyperlinks in HTML documents are represented with the <a> tag (where "a" means anchor). The target of the link is then specified in an "href" attribute for that <a> tag. In HTML, the target may appear within the same document or it may be within some other document somewhere else on the Internet. Linking to targets across the Internet is accomplished using the URL syntax. The URL provides a simple yet powerful way of linking to any file anywhere in the world on the Internet. The difference between the way links are specified in HTML and the way they would be handled in a HyTime-compliant format begins with the way URLs are used in the href attribute. Specifically, in SGML/HyTime, external files are conventionally accessed through entity references. In other words, instead of including a path and filename as an attribute of an element (as HTML does with href and URLs), the path and filename would instead be given within an entity. In a HyTime-compliant form, that entity could then have some unique name that would be used as the value for a docorsub attribute within a nmlist element (defined in HyTime), that in turn could be used within a nameloc element whose ID would then become the attribute for href (which, in HyTime, would be renamed linkend).

From the previous example, it is not too much of a stretch to convert all forms of hyperlinking (both internal to the document and external) from HTMLs formulation to a HyTime-compliant format. It should also be noted that HTMLs hyperlinking formulation does not support an anchor with multiple targets were as the HyTime formulation can easily do this.

Although HTML could be modified to comply with HyTime, at this time, it is not clear when or if future versions of HTML will ever be made to do this. It is noted in [1] that there has been little support within the WWW community for the inclusion of HyTime because of the size and complexity of the standard itself. This opinion was reinforced by some developers of SGML applications at HyTime94. It is envisioned that at some point in the future, if necessary, translation programs could be written that would convert HTML documents into a HyTime compliant format. Writing the conversion programs should not be an overly difficult task.

It should also be noted that HTML is currently undergoing revision. There are currently three levels of HTML. HTML level one is mandatory for all WWW clients. HTML level 2 is a superset of level 1 and includes forms for user input. It is not clear how wide the current support is for level 2. HTML level 3 is a superset of level 2 and among other things will include support for tables, figures, and equations. As with level 2, it is not clear how wide the current support is for level 3.

1. Raggett, David. "HTML+ (Hypertext markup language)." Hewlett Packard, July 23, 1993.
 2. Kimber, Elliot. "Making HTML a HyTime Application." First International Conference on the Application of HyTime, Vancouver, British Columbia. July 26-27, 1994.
4. *Preliminary Repository Plan Report comments:*
- c. *Pages 15 & 17: Expand upon the relationship between the WWW browser as a generic interface and any ongoing approaches within the Defense Information Systems Agency (DISA) Software Reuse Program.*

DISA Software Reuse Program has set up a DoD Software Repository System (DSRS) that contains approximately 4000 assets mostly in the domain of Management Information Systems (MIS). The repository is currently implemented on Sun platforms and is accessible to users via the Internet as well as modems. Currently, two versions of front-ends are available to the users: character-based and X version. In the future, access via Microsoft Windows is also contemplated. The repository is based on a multifaceted classification scheme whereby users can search and retrieve assets based on multiple indices.

In the coming weeks, we will meet with representatives of the DISA Software Reuse Program to continue our initial dialogues and to explore the possibility of synergistic relations between the two programs, especially in the areas of repository services.

*Reference [2]:

Critique of Z39.50 Based On Implementation Experience

Martin L. Schoffstall
Performance Systems International Inc./NYSERNet Inc.

Wengyik Yeong
NYSERNet Inc.

January 31, 1990

Abstract

The Z39.50 Information Retrieval Protocol is a proposed ANSI standard for the retrieval of bibliographic information that originates from the library community in the United States. Unlike previous interlibrary loan protocols designed by the library community, Z39.50 is intended to be the basis of an end-user service.

This paper provides a brief introduction to Z39.50, then discusses the shortcomings of the protocol that were discovered as a result of implementation experience.

Keywords: bibliographic retrieval, Open Systems Interconnection, TCP/IP, Internet, TCP/IP to OSI transition.

Introduction

Avenues that were not even dreamed of a generation ago have been opened by the growing degree of interconnectivity between computer systems. Without leaving the office, researchers can now gain access to super computers and other specialized systems in geographically remote locations; electronic mail has made scholarly collaboration possible on a grander scale than ever thought possible; and emerging teleconferencing technologies may one day make conference travel a relic of more primitive days.

It is reasonable to expect that, in light of these advances, researchers should also have convenient electronic access to publications. The Z39.50 Information Retrieval Protocol (Z39.50) [1] attempts to solve part of this problem, providing the ability to retrieve bibliographic information, if not the publications themselves. Bibliographic information such as author, title, and publisher may be retrieved from the databases of major bibliographic information providers such as the On-line Computer Library Center (OCLC) and the Library of Congress.

Z39.50 conforms to the OSI Reference Model [2] and is layered directly on top of the Presentation Layer [3,4] of the OSI stack. The only Application Layer services used by Z39.50 are those provided by the Association Control Service Element (ACSE) [5,6].

The functional model used by Z39.50 is that of origins (clients) retrieving information from databases made available for access by organizations through targets (servers). A database is

viewed as a collection of records with no other inherent infrastructure. Records are in turn simply a collection of fields that may occur in any order and may be repeated within the record.

This model contains a relatively large number of origins compared to targets, with each target providing access to a few databases consisting of a large number of records. The bibliographic retrieval process is consequently made up of two steps: a search to reduce the amount of data to be retrieved down to the subset desired, and a retrieval to actually obtain the data.

The search process provides origins with the ability to request a search on one or more databases, specifying a set of index terms to be used in conducting a search. The index terms that may be used to search a database are not defined in Z39.50, but are subject to private agreement between origin and target implementors.

Upon completion of a search, the target creates a result set that is conceptually made up of the subset of records in the databases which obey all search criteria. Records in a result set are a logically ordered list numbered from 1 to N, the number of records in the result set. Z39.50 allows the naming of result sets so that multiple result sets may coexist, while not requiring that targets support the result set naming facility. If a target does not support result set naming, the target must at least support the single result set default.

Bibliographic information may be retrieved by reference to the result set, and the logical numbers of records in the result set. Z39.50 provides a facility to retrieve only a subset of the fields in a record from a result set: if an element set consisting of a predetermined subset of the fields found in a bibliographic record is specified during the retrieve operation, only that subset of the fields in the bibliographic record that is defined by the element set will be returned. Element sets are not defined in the Z39.50 specification, but are instead subject to private agreement between origin and target implementors.

In addition to the search and retrieval capabilities required for bibliographic retrieval. Z39.50 also provides facilities to initiate and tear down an Information Retrieval Session, to perform resource and access control at an extremely fine granularity, and to delete result sets whose contents are no longer of interest.

NYSERNet Inc. has implemented a Z39.50 origin that currently interoperates with a Z39.50 target implementation performed by OCLC. As a result of the experience gained by implementing Z39.50, a number of flaws were perceived. This paper describes these flaws.

Layering and ROS

The Remote Operations Service (ROS) [7,8] provides facilities for an application entity to request the remote performance of predefined operations by another protocol entity. The generic structure of a ROS operation is that of a request/response transaction between an invoking application entity and a performing application entity. Classes of ROS service are defined to support both synchronous and asynchronous operation. There is no limitation in ROS preventing operation over duplex communication facilities: operation invocations are not limited to half-duplex facilities.

Z39.50 ignores ROS, not using any of the services which ROS offers. This is a mistake, for ROS offers much functionality that is useful to Z39.50.

Z39.50 is inherently transactional, with origins requesting that targets perform such operations as searching, retrieving, and deleting. This request/response mode of operation is identical to the transactional paradigm represented by ROS. It is significant to note also that all protocol elements of Z39.50 contain a reference id parameter which serves the same function as the ROS INVOKE-ID. This bolsters the assertion that the exclusion of ROS services in the definition of Z39.50 is a regrettable oversight.

Arguments have been made that ROS was excluded from the definition of Z39.50 because ROS was not capable of asynchronous communication, operated only in half-duplex mode, and was in general "too ambiguous." These arguments are fallacious. As documented in Reference 7, ROS Operation Classes 2 through 5 are asynchronous, and there is no restriction in the Reference requiring ROS to operate only in a half-duplex environment.

As to the assertion that ROS is "too ambiguous," it has to be noted that such applications as the OSI Directory [9,10] and MHS [11,12] have been successfully implemented on top of ROS. The successful deployment of these technologies, some for over four years, clearly speaks as to the validity of the claim that ROS is "too ambiguous."

Protocol Transfer Syntax

The designers of Z39.50 chose to encode Z39.50 protocol packets in a unique format by defining their own transfer syntax and encoding rules. This decision is somewhat questionable, for an internationally recognized standard exists for the definition of data transfer syntaxes: the ASN.1/BER standards [13,14].

The ASN.1 standard is an extensible language for defining device independent data structures for transmission across data networks. Much effort has been expended to create automatic tools to encode, decode, and manipulate structures defined in ASN.1 and encoded with the BER. ASN.1 is also the transfer syntax, and the BER the encoding rule of choice in the Presentation Layer of the OSI stack. ASN.1/BER is the only transfer syntax supported by vendors of OSI hardware and software.

In contrast, the syntax used by Z39.50 for Protocol Data Unit (PDU) formats is specific to both Z39.50 and the library community in the United States. This is not great auspices for a purportedly international standard.

Although Z39.50 is not alone in having a unique transfer syntax, it has to be pointed out that other protocol definitions having unique transfer syntaxes precede the common acceptance of ASN.1/BER, making Z39.50 behind the times, at best.

The Manpower Requirements Criteria (MARC) Format

The MARC format [15] is the least common denominator for the representation of bibliographic information during interlibrary communication in the United States. Within Z39.50 protocol packets, bibliographic data is encoded in the MARC format. Unfortunately, MARC is extremely haphazard and redundant on a grand scale. For example, a random search conducted by opening Reference 15 at an arbitrary page reveals that at least three different MARC fields can be used to define the attribute title: 110t, 111t, and 130t. In fairness, there are subtle distinctions between these three MARC fields, but such distinctions are too subtle to be of relevance to any but a career librarian.

Far be it, however, for such a criticism to be offered of so hallowed an institution as the MARC format without also suggesting a substitute. It is proposed that, for the purposes of providing end-user service at least, if not interlibrary communication, the MARC format be replaced with a systematic cataloging method free of redundant specification. A system based on the ASN.1/BER standards discussed in the previous section is recommended.

A point to ponder for the library community is that future library access will be predominantly by means of data networks. The explosive growth of personal workstation use, coupled with the remote access concepts pioneered by Z39.50 itself, makes a future of electronic remote access inevitable. In this light, because library information is essentially kept in databases that have a native (non-MARC) format, it is perhaps time that librarians consider a standard for data representation based on ASN.1, not MARC. A format for the future of library access, so to speak.

It has also to be noted that Z39.50 is intended for a purpose different than any other protocol specified by the library community. Z39.50 is the basis of an end-user service, not a protocol for use in an interlibrary loan application. In contrast, the MARC format is designed for interlibrary loan purposes. Given the differences in the needs of end-user applications as compared to interlibrary loan needs, it is questionable whether the MARC format is even suitable from a philosophical standpoint. Procrustean methods to mold MARC into a form suitable for an application never envisioned by its authors can only detract from the Z39.50 protocol.

Z39.50 Query Formats

Z39.50 defines not one, but two formats that may be used to specify the search queries used to locate bibliographic information. It is unfortunate that both are problematic.

The Type 1 Query is currently the standard format defined in Z39.50 for the structuring of search queries during a search. A standard format for querying is of utmost importance in an information retrieval protocol such as Z39, and it is unfortunate that the Type 1 Query format is flawed. Specifically, the Type 1 Query is an ambiguous grammar. The constructs

query ::= argument | argument + argument + Operator
argument ::= operand | query

allow the infinite expansion of the query and argument non-terminals

query --> argument --> query --> argument -->

Thus, the Type 1 grammar does not lend itself well to implementation without making an assumption about the limits of the complexity of a search query. Fortunately, a slight modification renders the grammar suitable for implementation. All that is required is that the construct

query ::= argument | argument + argument + Operator

be changed to

query ::= operand | argument + argument + Operator

In the case of the Type 0 Query, the flaw lies in its very existence: the Type 0 Query is an "escape hatch" for implementors not wishing to conform to the standard in the area of search query construction. Unfortunately, its existence is also one more obstacle in the way of transparent application interworking. Hence, in the interests of interworking, it is recommended that the Type 0 Query be excised from the protocol and that all implementors be required to use a common query format.

Element Sets and Index Terms

The element sets used in bibliographic information retrieval by Z39.50 are not standardized within the protocol, but are one of the many ambiguities in the protocol calling for "private agreement outside of the standard." As with all other instances when "private agreements" are called for, this requires a priori knowledge on the part of the origin.

It is often argued that bibliographic data is inherently unstructured, and hence it is impossible to standardize on a legal set of element sets which may be used in all retrieves. This may be the case, but it is still no reason for "private agreements," because a facility could easily be provided within the Z39.50 protocol for origins to discover legal element sets as part of the protocol interaction. For example, the target could pass to the origin a list of

<database-name, element-set>

pairs during the initialization of the Z39.50 Information Retrieval Session.

Similarly, index terms used in search queries are left to "private agreement," and thus a priori knowledge is required. Again, a facility similar to that for element sets should be provided for the discovery of legal index terms that may be used in a search by origins.

The Diagnostic Record

If the origin requests the return of a sequence of bibliographic records, and the target is unable to return one or more of the records requested, diagnostic records are returned instead. Neither the format nor the number of such records to be returned is specified in Z39.50.

The format of the diagnostic record to be returned should be specified, this being a case of any definition being better than none at all. It is less important what the format of a diagnostic record is, than that there is a common format, providing origins with the ability to uniformly process error conditions from all database providers.

However, even though the format of diagnostic records is less important than the existence of a common format, it is worthwhile to explore what a diagnostic record should be. A diagnostic record should ideally be susceptible to machine processing without the need for human intervention. Hence, a diagnostic format consisting simply of a character string error message, while a vast improvement over the current vacuum of definition, would be inferior to a competing format that provided a standard set of error returns. The error returns themselves should of course be defined as part of the Z39.50 standard, and exhaustively cover all possible error conditions.

If the target is unable to return one or more of the bibliographic records requested by the origin, the number of diagnostic records to be returned should also be specified. Currently, a Z39.50 target may return any number of diagnostic records it chooses to, as long as at least one record is returned. This ambiguity should be resolved in a uniform way. Ideally, a diagnostic record should be returned for each bibliographic record that could not be returned. This allows for the return of error information on a record-by-record basis, providing the finest possible granularity in error processing.

Result Set Naming

There are ambiguities in the area of result set naming as well. Z39.50 allows result sets to be named but does not require target implementations to support result set naming. There is also no way within the protocol for origins to determine if a target will support result set naming. A priori knowledge is again required.

The area of result set naming should be clarified. Either Z39.50 should require that all targets support result set naming, the preferred option, or a mechanism should be provided within the protocol to allow origins to discover if a target will support result set naming.

Cramming Functionality Into Service Elements

There is a tendency within Z39.50 to cram too much functionality into a single service element. This results in overly complex PDUs that have to provide a variety of options to cope with the many functions that each PDU has to perform. In other words, each PDU is too complex, because each service element tries to do too much.

This is most apparent in the Delete Service Element. Z39.50 provides the ability to delete one or all existing result sets in an Information Retrieval Session. Unfortunately, both the single delete and the delete-all functions are crammed into a single protocol element, instead of being two distinct service elements, DeleteSingle and DeleteAll. PDU realization of the two service elements would be far simpler than the current PDU associated with the Delete Service Element,

which is essentially a massive branch: half of the fields are used if the deletion of a single result set is to be performed, and the other half of the fields are used if all result sets are to be deleted.

The Search Service Element suffers from a similar problem, albeit to a lesser extent. Z39.50 actually offers two search services, to allow the searching of either a single database or multiple database simultaneously. As with the Delete Service Element, the functionality of both searches is crammed into a single service element to the detriment of the protocol in general and the Search Service Element in particular. A case may perhaps be made here that since both search functions share common parameters, they should rightfully be two options of a common service element. Although this argument has merit, it has to be noted that the two search functions also have differing parameters and that the form of the PDUs can be greatly simplified if the two search functions were separate.

Small Sets, Medium Sets, Large Sets and Piggybacking

Z39.50 allows the piggybacking of data in the Search Service Element. This is accomplished by the appropriate initialization of three service parameters: the small set upper bound, the large set lower bound, and the medium set present number. Piggybacking is then accomplished as follows: if the number of records in the result set is less than or equal to small set upper bound, then all the records in the result set are returned. If the number of records in the result set is greater than or equal to large set lower bound, then no records are returned. Last, if the number of records in the result set is between small set upper bound and large set lower bound, then the number of records to be returned is given by medium set present number.

While piggybacking is a time-honored protocol optimization, it is questionable if this functionality will actually ever be used. It is an unfortunate fact of life that the bibliographic information providers in the United States charge for information retrieved. As piggybacking is essentially a look ahead mechanism allowing applications to anticipate the needs of the end-user, it is somewhat doubtful as to whether, given the information retrieval charges, applications should be allowed to risk anticipating wrongly, and incurring unnecessary usage charges on behalf of the end user.

Aside from this philosophical doubt, the method by which piggybacking is accomplished is also in question. The method used by Z39.50 to decide on the number of records to be piggybacked is clumsy at best. The division of result sets into small, medium, and large sets is completely arbitrary, and is indeed left to the origin. Because the purpose of all this protocol gymnastics is to decide how many records to piggyback, it would be far simpler to replace the small set upper bound, the large set lower bound, and the medium set present number fields with a single service parameter to be used to determine how many records to piggyback.

Nits

Apart from the problems mentioned above, there are a few nits to be picked with Z39.50.

PDUs are tagged with values beginning at 20. There appears to be no reason for such an arbitrary value. Tags could just as easily begin with 0 or 1. This is a small detail, except that by

beginning with 0 or 1, tag values are smaller in size and require fewer bits to encode, thus incurring less transmission overhead.

As discussed above, result sets may have unique identifiers associated with them. These identifiers are referred to as both result set names and result set ids in the protocol specification although there appears to be no difference between a result set name and a result set id. This would be a triviality, except that in the protocol specification, result set names and result set ids have distinct tags.

If an origin attempts to retrieve bibliographic information on a nonexistent result set, the action to be performed by the target is undefined.

The value of the result set name service parameter of the Delete Service Element is undefined if a single result set delete is to be performed on a target that does not support result set naming. It is also not clear from the specification whether the special result set default can be deleted from targets that do not support result set naming, and if so, whether further searching can be performed: how is the result set named default to be created again on a target that does not support result set naming?

The Z39.50 protocol machine tends to transition into unknown states in general. This tendency has been demonstrated time and again in this paper, particularly in the last few paragraphs of the Nits section.

Conclusions

The shortcomings of Z39.50 are reviewed in this paper. The protocol suffers in general from haphazard definition, with some areas being over defined and consequently overly complex, while other areas prove to be vacuums of definition.

Bearing in mind that Z39.50 is different from any protocol ever specified by the library community, it is speculated that the library community may not yet understand the needs of an end-user protocol, or the differences between a protocol intended for widespread independent implementation and a protocol to be implemented by a small number of organizations.

Historically, the library community has been small and closed. In this environment, out-of-band communication for the purposes of achieving interworking is a viable option. As all previous protocols defined by the library community have been for interlibrary loan and hence susceptible to out-of-band communication methods for achieving interworking because they are implemented by only a small group, it is entirely possible that the library community may not understand the need for a protocol to be completely specified in the protocol definition. This ignorance on the part of the library community would go far in explaining the way in which Z39.50 is defined.

It is also to be wondered if the library community even understands that an end-user protocol is inherently different from an interlibrary loan protocol. Although the two basically accomplish the same thing, the transfer of bibliographic information, an interlibrary loan protocol is inherently batch-oriented and optimized for the transfer of large amounts of information. Furthermore, it

would be permissible for an interlibrary loan protocol to depend on the oral traditions of librarianship to work, because both peers in any protocol exchange would be members of the library community.

In contrast, an end-user protocol is inherently different in very important ways. By definition, one peer in the protocol exchange of an end-user protocol is an end-user, not a member of the library community. This implies that dependency on oral traditions of any sort is a recipe for disaster, and every nuance of the protocol must be exhaustively specified. Also, the interactive nature of bibliographic retrieval requires a protocol optimized for responsiveness, not ability to transfer large amounts of data.

It is concluded that Z39.50 is inadequate as the basis for an end-user bibliographic retrieval service. The concept of electronic access to bibliographic databases is, however, an excellent one and should be pursued in the future. Clearly, a bibliographic retrieval protocol cannot be defined without the specialized knowledge of the library community, but it is equally clear that the protocol design experience of the Internet community is needed to ensure an optimal design. Hence, it is proposed in closing that the future success of bibliographic retrieval may require the cooperation of two disparate communities, drawing on the strengths of the two communities.

Bibliographies

[1]

National Information Standards Organization,
Proposed American National Standard for Information Sciences -- Information
Retrieval Service Definition and Protocol Specification for Library
Applications Z39.50-1988, not published.

[2]

International Organization for Standardization and International
Electrotechnical Committee,
Information processing systems -- Open Systems Interconnection -- Basic
Reference Model,
International Standard 7498, October 15, 1984.

[3]

International Organization for Standardization and International
Electrotechnical Committee,
Information processing systems -- Open Systems Interconnection --
Connection Oriented Presentation Service Definition,
Final Text of Draft International Standard 8822, April 1988.

[4]

International Organization for Standardization and International
Electrotechnical Committee,
Information processing systems -- Open Systems Interconnection --
Connection Oriented Presentation Protocol Specification,
Final Text of Draft International Standard 8823, April 1988.

[5]

International Organization for Standardization and International
Electrotechnical Committee,
Information processing systems -- Open Systems Interconnection --
Service Definition for the Association Control Service Element,
Revised Final Text of Draft International Standard 8649, April 1988.

[6]

International Organization for Standardization and International
Electrotechnical Committee,
Information processing systems -- Open Systems Interconnection --
Protocol Specification for the Association Control Service Element,
Revised Final Text of Draft International Standard 8649, April 1988.

[7]
International Organization for Standardization and International
Electrotechnical Committee,
Information processing systems -- Text Communication --
Remote Operations Part 1: Model, Notation and Service Definition,
Working Document for International Standard 9072-1, March 1988.

[8]
International Organization for Standardization and International
Electrotechnical Committee,
Information processing systems -- Text Communication --
Remote Operations Part 2: Protocol Specification,
Working Document for International Standard 9072-2, March 1988.

[9]
International Organization for Standardization and International
Electrotechnical Committee,
Information processing systems -- Open Systems Interconnection --
The Directory -- Overview of Concepts, Model and Service
International Standard 9594-1, December 1988.

[10]
International Telegraph and Telephone Consultative Committee,
The Directory -- Overview of Concepts, Models and Service,
Recommendation X.500, December 1988.

[11]
International Organization for Standardization and International
Electrotechnical Committee,
Information processing systems -- Text Communication -- MOTIS --
Message Handling: System and Service Overview,
International Standard 10021-1, December 1988.

[12]
International Telegraph and Telephone Consultative Committee,
Message Handling: System and Service Overview
Recommendation X.400, 1988.

[13]
International Organization for Standardization and International
Electrotechnical Committee,
Information processing systems -- Open Systems Interconnection --
Specification of Abstract Syntax Notation One (ASN.1),
International Standard 8824, December 1987.

[14]

International Organization for Standardization and International
Electrotechnical Committee,
Information processing systems -- Open Systems Interconnection --
Specification of Basic Encoding Rules for Abstract Syntax Notation One
(ASN.1), International Standard 8825, November 1987.

[15]

On-line Computer Library Center
OCLC-MARC Tape Format,
ISBN 0-933418-62-0, September 1984.