

Design Considerations in Instrumenting and Monitoring Web-Based Information Retrieval Systems

Michael D. Cooper

*School of Information Management and Systems, University of California, Berkeley, Berkeley, CA 94720-4600.
E-mail: cooper@socrates.berkeley.edu*

The Internet Web environment opens up extraordinary opportunities for user access to information. Techniques for monitoring users and systems and for evaluating system design and performance have not kept pace with Web development. This article reviews concepts of Web operations (including browsers, clients, information retrieval applications, servers, and data communications systems) with specific attention given to how monitoring should take place and how privacy can be protected. It examines monitoring needs of users, systems designers, managers, and customer support staff and outlines measures of workload, capacity, and performance for hardware, software, and data communications systems. Finally, the article proposes a client-server design for monitoring, which involves creation of a series of server and client systems to obtain and process transaction and computer performance information. These systems include: A Log Server, which captures all levels of transactions and packets on the network; a Monitor Server, which synthesizes the log and packet data; an Assistance Server, which processes requests for information and help from the Web Server in real time; and an Accounting Server, which authenticates user access to the system. A special System Administrator Client is proposed to control the monitoring system, as is a System Information Client to receive real-time and on-demand reports of system activity.

Introduction

This article examines the design issues facing the developer of an integrated instrumentation and monitoring package for Web-based information retrieval systems in a client-server environment that would enable evaluation of user and system performance. This is not an article about the evaluation of these systems but rather a discussion of design considerations in instrument development that will facilitate later evaluation.

The methods and procedures by which users obtain

information are changing rapidly. Daily, hundreds of thousands of individuals use the Internet and Web browsers like Netscape to obtain a much broader range of information than has previously been possible. This information consists not only of bibliographic citations but of full text, data, images, sound, programs, and the like. The manner of searching has changed as has the content of search results, but the design of transaction logging programs has not caught up with these changes.

Despite the changing nature of the content of information retrieval systems, the user interface, and the data communications interface, a basic research question remains: Is the system effective in helping users find information? It is growing more difficult to answer this question because interface and communications systems are more complex. However, there is much useful information ideally suited for logging (such as user actions and system performance) and these logs can be used to supplement qualitative evaluations or confirm user uncertainty regarding actions taken.

It is extremely expensive to conduct personal interviews with users to find out if they are satisfied with the information retrieved and/or the information system itself. So efficient evaluation must rely on computerized methods to capture this data as unobtrusively and effectively as possible. This evaluation strategy does not preclude user interviews, usability studies, verbal protocols, questionnaires, or other evaluation strategies. It simply focuses on one method to ensure it is well designed to meet user and management requirements, and provides the necessary components to support these types of evaluations in an electronic environment.

Client-Server Information Retrieval System Configuration

A client-server information retrieval system is configured as shown in Figure 1, with many client computers able to connect via a communications network (such as

Received December 5, 1996; revised February 24, 1997; accepted August 29, 1997.

© 1998 John Wiley & Sons, Inc.

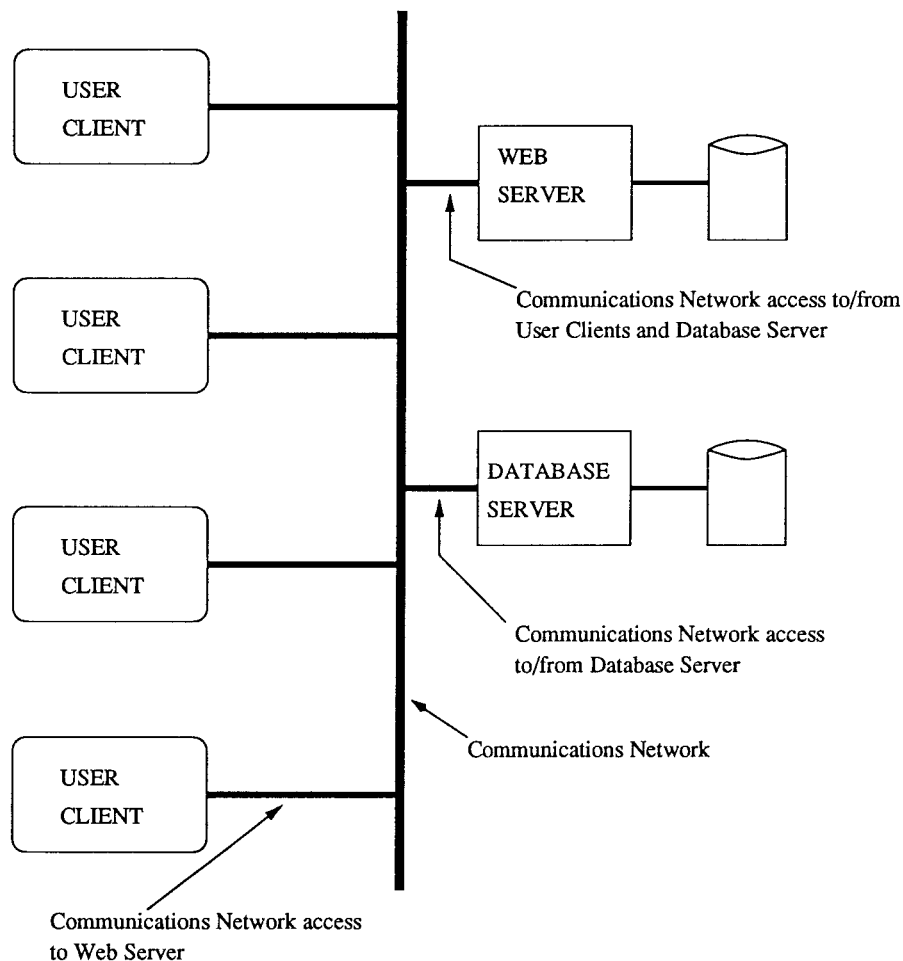


FIG. 1. Client-server system design.

the Internet) to a server computer using a communications protocol such as TCP/IP (Hunt, 1994; Washburn & Evans, 1993). Information not available on the Web server computer is obtained by having the Web server access other systems, such as a database server, also shown in the figure.

The nature of a client-server system is that the connections are not permanent; they are made for the life of the transaction and then terminated. For most client-server applications, the transient nature of the connection is not a problem since the server has no interest in maintaining a permanent connection to a client who, for example, wants to know the weather in San Francisco. However, when a library automation application such as an online catalog employs a client-server configuration, the server must continue a dialog with the client beyond one transaction.

The circumstances under which a database server is used are many. In a common scenario, a library provides a unified online catalog interface for its users to access a myriad of other information sources. To obtain the additional information, the library server communicates with other servers, reformats the data, and transmits it to the

library user's client computer. In another example, a library has an existing online catalog running on a large central computer and implements Web-based access. The library develops a server to handle communication with the clients and their Web browsers, and the server communicates with the mainframe, perhaps using the Z39.50 communications standard to obtain catalog information (American National Standards Organization, 1995; Cooper, 1996).

The Browser Program and Client Software Modules

A client computer is connected to a server over communications lines. Among the programs being executed on the client computer are an operating system, the telecommunications software to make the connection to the server, and a browser program, which provides access services for the client to information retrieval systems.

The browser is but one of many active programs; others include a word processor, text editor, spreadsheet program, database management application, or a time manager. A user conducting a bibliographic search using a

network browser may transfer material from the browser screen into a word processing package, use the client operating system to mail a document (that has been located through the browser) to another user, or use its print service to create hard copy of a document retrieved through the browser.

The browser is the main component of current information retrieval systems that has changed the nature of transaction logging. Currently, there are a number of widely used browsers including Netscape Navigator and Microsoft Explorer. Browsers are deceptively simple pieces of software for the end-user to employ but offers many functions as well as the potential to be augmented by third-party vendors.

With very little knowledge, a user can open a window, type a Universal Resource Locator (URL), and direct the browser to contact a Web server. Once a connection is established, the server transmits a response to the browser program in the form of a page of text, usually tagged using the HyperText Markup Language (HTML).¹

Behind the scenes, the scenario is considerably more complex. A Web-based client-server application relies on a data communications network, a data communications protocol (such as TCP/IP), and a data connection and transfer protocol (such as the Hypertext Transfer Protocol—HTTP) to operate.² The user's browser requests information from a server in a four stage operation: A connection is made between the client and the server, a request for information is made by the client, a response is delivered by the server, and a disconnection is made. The HTTP protocol governs the details of how these stages are conducted and the exact form of the commands that must be transmitted and received.

When the first page of information appears in the browser window, the user can employ the scroll bar to move up, down, left, or right through the display; request additional pages of information from the server; fill in a form and transmit it to the server; follow other URLs to other sites; cut and paste information from the browser screen to, say, a word processor window; print the material retrieved; save the material to a local file; and/or mail the material to another party. Browsers can be used to view local files, to view and follow threads and links in newsgroups, and to transfer files using FTP from other sites. For summaries of some current browser features, see Ayre and Mace (1996) and Berghal (1996).

A systematic exploration of browser menus makes it clear that a browser is far from a simple piece of software. And the basic functionality of a browser can be aug-

mented by so-called plug-ins and helper applications, and by programs (often written in the JAVA programming language or employing the JavaScript scripting language) that execute within the browser in the client computer. Examples include: Adobe Corporation's *Acrobat* program, which displays Portable Document Format (PDF) files (a popular multiplatform document format); *Real Audio Player*, which plays audio sound files found on the Web; and *Shockwave*, used to view interactive multimedia and graphics. These applications are invoked when a file with a designated file extension is delivered to the browser window for viewing or listening.

The Server Computer and Server Software Modules

A server computer provides information to its clients through a communications network, like the Internet, and has a number of software programs executing within it at the same time. These include an operating system, a package to handle TCP/IP communications, a client-server communications system to handle connections between the client browser and the server (and, possibly, a Z39.50 client and server communications system), and applications programs that retrieve data from the server databases.

The second new component in Web-based information retrieval systems is server software. As the name implies, it is the program that executes on the server computer and manages the dialog with the client. The server listens at a predefined port or ports on its hardware for signals to connect, and when a request is received, it makes a connection and begins the interchange of data with the client. The requests may involve transmission and receipt of an HTML-tagged document, CGI script (Gundavaram, 1996; Libes, 1995), a JAVA applet, images, video, or audio files.

Both public domain and commercial server software exist and are relatively simple to install.³ Liu, Peek, Jones, Buss, and Nye (1994) provide a review of Web server set-up procedures, and Lipschutz's (1996) article is an example of but one of many comparisons of commercial server software.

General Evaluation and Monitoring Issues

A logging and analysis facility for an information retrieval system must address these categories of questions:

1. User Information

An institution that manages an information retrieval system must ensure that the system is effectively helping

¹ The HTML language specifications can be found at URL: <http://www.w3.org/TR/Overview.html>. Many books on the subject are available. Their focus is generally on using HTML for the design of home pages on the Internet. Examples include books by Aronson (1995) and Wilson (1995).

² The specifications for version 1.0 of HTTP can be found at URL: <http://ds.internic.net/rfc/rfc1945.txt>.

³ Public domain HTTP daemons include the NSCA HTTPD and versions of Apache (<http://www.apache.org>).

users find needed information. A basic monitoring system is a set of tools used to accumulate data about system performance, but a truly dynamic monitoring system can be used in real-time to analyze user behavior, deduce patterns of behavior, compare such patterns to those previously stored, and immediately suggest to the user ways of modifying a search to improve its quality. The most obvious application of such a technique is to provide online help messages customized for typical problems or situations users encounter. More advanced applications include suggestions for search strategies based on the user's context, and reports to users on functions and features employed during a search session.

At a more basic level, a monitoring system is concerned with the activities and tasks the user is performing during a search. A logging system must be able to record some basic demographic information about users and facilitate the association of this information with user behavior, activities, and tasks.⁴ When user searching takes place, the system must be able to log measures of probable success and failure using indicators, such as zero results found for a search, large result sets, and errors incurred in the searching process. When hypertext linking is present in the retrieval system, the logging programs must be able to record links followed. If full text is viewed, the system should be able to track user movement through the text. When users finish reviewing their citations or text, the logging system should have some way of recording the user's perception of the usefulness of the material to their information need at the moment. This latter is not straightforward to implement but may be extremely important. Giving the user the option to give feedback could enhance the logging data considerably.

Another use of real-time monitoring is to provide information to the user on the status of a service request (that is, a search). The response time a system provides to its users is a function of the number of users on the system at any one time, the other tasks the system must perform, and the characteristics and quality of the hardware, software, and communications system. These factors can all be monitored and key indicators displayed to the user as needed or requested.

2. System Design

Is the design of the system and the interface to the system adequate and appropriate? The logs must allow for measurement of the usability of the system, the performance of the system, and comparisons of alternative designs for the system as a whole or variations within the system. It must also support the capture of data associated with specialized experiments, such as measuring response to changes in graphic design, component color, and on-

screen formatting to see if these affect usability and performance. A truly dynamic client-server environment with good monitoring tools and corresponding feedback from participants willing to evaluate the changes could support almost-immediate changes to an interface.⁵

3. Management Information

Information retrieval system managers, be they members of the operations staff or executives, need various types of information to help them understand how the systems are being used and how they are performing, so that systems can be managed effectively and rational planning can take place.

The logging system must be able to monitor computer system performance and integrate such reporting with feedback on information retrieval system performance. The log must track workload, performance, and capacity of the system. And it must be able to associate user commands and actions with computer resources used to ascertain the amount of these resources used. Measures used for this type of evaluation are discussed in detail in the section on *Computer Hardware, Software, and Data Communications Performance Monitoring* later in this article.

4. User Account Services

The logging system must be able to track individuals' usage of the system, if this is required or desired. The type of accounting ranges from tracking the amount of time the user is connected to a database, to recording the number of text pages viewed, number of bibliographic records downloaded or mailed, the disk space used for temporary storage of records, the types of commands issued, online orders for printed copies of materials, and the like.

5. Customer Support and Service

Customer support services benefit enormously from knowing how systems and features are actually used, as opposed to how designers think they will be used. User services staff should be able to use log files to discover user problems and obtain regular reports on system activities. A dynamic logging system that has sophisticated reporting and analysis tools should be able to synthesize user problems and store them in databases available for consultation by customer service representatives.

6. Research Questions

Any logging system must be adaptable enough to allow special experiments to be conducted without major

⁴ See the section on *Ethical and Privacy Issues in Transaction Monitoring*, below, for a discussion on the limitations of profiling.

⁵ The *frame* facility of recent versions of the HTML markup language makes dialog in one frame and display in another frame possible at the same time.

changes to its structure. In other words, it must be flexible and extendible without disruption to existing logging operations.

Ethical and Privacy Issues in Transaction Monitoring

Users, system managers, systems designers, researchers, and many other parties have an interest in the dialog that a user has with an information system. At one extreme, national and international law enforcement agencies might want to view one user's interactions to deduce that person's interests, whereabouts, queries, and the like. At the other end of the spectrum, a system manager, wanting to know what caused a system to blow up, could track the tasks being performed at the critical moment. Between these extremes are the users who can benefit from on-the-spot help in diagnosing problems they are having with their searches, system designers who would like to improve the interface to the system by studying user interaction, and researchers.

Another distinction comes from partitioning dialog between users within an organization (an intranet, or private Web site) from the dialog that takes place between users and a public site. It may be that different levels of accountability are called for in the monitoring that takes place in public versus private sites, or that both categories of sites should be held to the same ethical and privacy standard.

The ethics and privacy questions involved in transaction monitoring are legion. Even the popular press has become aware of the problem (Gomez, 1996; "We know you're reading this," 1996; Williams, 1996). Should a transaction log be maintained at all? The Cyber Rights Working Group of Computer Professionals for Social Responsibility suggests a number of guidelines for protecting the privacy of users (Computer Professionals for Social Responsibility, 1996). One is particularly relevant to logging policies:

Demographic or identifying information gathered at servers that is not actively provided by the user should not be used beyond the analysis of site activity; in particular, no attempt should be made to identify individual users or to pass this information on to other parties. Beyond the needs of system administration, information should not be collected or stored on usage patterns of individuals, such as time of day usage, sites visited, and downloads.

The American Library Association does not have any direct policy on transaction logging but is concerned about the confidentiality of library records (American Library Association, 1996). Their policy statement focuses on ensuring that information about what a patron "received . . . consulted, borrowed, acquired" is kept confidential and that personal identification is omitted from records of these transactions.

Some groups (CERT, 1992) have advised that a sys-

tem should announce that logging is taking place and give the user the option of logging off if they do not want to have their actions logged. This type of message is often displayed on contact with FTP sites.

What responsibility does the systems analyst or systems designer have in ensuring user privacy? Wood-Harper et al. argue that an information system may be considered good by society and yet be considered unethical by the employees of the organization that created it (Wood-Harper, Corder, Wood, & Watson, 1996). Kurth (1993) makes a similar point in the context of library transaction logging. They both argue for overtly specifying whose ethical perspective will be used to evaluate the system design (that is, user, corporation) because the perspective will definitely influence the design. In a similar vein, Johnson and Mulvey (1995) examine whether designers are responsible for all the uses of the systems they create. They suggest that it is the nature of professional behavior to create norms of design that will promote the public good and not result in public danger.

However, there are many systems in existence now for which it is difficult to judge social merits. *Computerized Performance Monitoring and Control Systems* (CPMCS) are used to track the performance of employees who do computer-mediated work, such as data entry, telemarketing, or order processing. Grant and Higgins suggest there are two parts to these automated systems, *sensors*, which collect the base data and *discriminators*, which evaluate and interpret the data. If the discriminator is known to be reliable, is intelligent enough to evaluate the sensed data, or "holds a view of the job consistent with that of the employee," it will be judged credible (Grant & Higgins, 1996, p. 220).

Monitoring and Evaluation of Information Retrieval Systems

Transaction monitoring of information retrieval systems, such as library catalogs, has been conducted for a considerable time. This methodology consists of a program that runs on the central computer system writing log records to a data file for each transaction the user performs with the system. Each record written is date- and time-stamped and assigned a unique session identification number so that all transactions associated with a session can be grouped together. This methodology is not unique; most database management systems create transaction logs to allow a database to be rolled back to its previous state should some disaster occur, or, for security purposes, to audit access and modifications to data (Elmasri & Navathe, 1994; Gray & Reuter, 1993).

A transaction file generally consists of a series of records in which each action the user performs (such as search, display, browse, or print) is written as a separate record to the log file along with the parameters associated with those actions.

A comprehensive review of transaction log analysis in

library automation systems was published in a special issue of the journal *Library Hi Tech*, which includes a number of articles which will be summarized here.

Both Kaske (1993) and Flaherty (1993) discuss the data elements typically found in a transaction log record, such as date and time of the transaction, terminal identification number, user input, and system response. Flaherty presents examples of transaction logs from a number of different systems and makes suggestions for the structure of an ideal log. Flaherty stresses the need for access to the transaction log through creation of a database of its records, prewritten programs that can produce standard reports from the records, prewritten codebooks for the records that allow easy use of statistical analysis systems such as SAS, and facilities for downloading records to workstations to facilitate their manipulation and analysis. This latter suggestion is reasonable for small data sets, but may be unreasonable to use for large operational systems such as Melvyl, RLIN, OCLC, or Dialog.

Peters' (1993) article summarizes previous studies that have used transaction logging data for decision making. These include studies of:

- Commands issued;
- system and user response time;
- length of sessions;
- sequences of commands issued;
- transition probabilities between search states;
- error analysis including spelling, zero hits, failures;
- use of assistance (help) facilities;
- printing and downloading;
- user persistence in completing a search;
- user preferences for interfaces within one system;
- same users between multiple systems;
- different user groups (that is, staff, users); and
- same set of users over time (longitudinal).

Peters' review makes it clear that an enormous number of studies have been conducted that look into many facets of information system use with the aid of transaction logs, but there has been little replication of these studies and little consistent evidence about user behavior deduced over time. Borgman et al. comes to similar conclusions (Borgman, Hirsh, & Hiller, 1996).

Kurth explores the problems inherent in transaction logging, including:

- Complexity of obtaining the data;
- large size of the data sets obtained;
- inability to replicate results because database or system may have changed;
- lack of full logging facilities from some vendors;
- no provision for adding additional logging to existing systems;
- limited or no ability to identify broad categories of users in the logs;
- no ability to record user perceptions of their searches;
- and

difficulty in recording many types of user errors or search problems.

He goes on to enumerate problems of analysis, such as defining the start and end of a search, defining what constitutes a search failure, and trying to determine whether a search that retrieves no citations is a success or a failure (Kurth, 1993).

One major problem for systems that do not require user identification numbers or names at log on (such as most library catalogs) is that it is sometimes very difficult to tell when one session begins and another ends. Terminals placed in public places (as in a library) are used continuously; as soon as one user leaves, another may start a search. Inferences about number of searches made, and length of time, commands issued, or computer resources used per search are consequently hard to make.

Computer Hardware, Software, and Data Communications Performance Monitoring

Transaction log analysis supplies some, but not all, information needed by users, managers, designers, and researchers. A major category of information that can influence user and system behavior originates from the computer and communications system on which the application is running.

Computer systems can be evaluated in three basic ways: Workload, performance, and capacity. Workload measures characterize the amount of work the computer system performs, such as the number of transactions processed per second or the number of online searches of a database performed within a certain time period. Performance measures characterize how well the computer system conducts its work. Perhaps the most important measure of performance from the user's perspective is response time—the time that elapses between the moment the user clicks a browser window submit button and the moment the first part of a response is received from the server and displayed. If the user does not receive a response in what is reasoned to be an appropriate amount of time, an activity may be suspended, the user may form a negative opinion about the system, or the user may make conclusions about the load on the system and continue waiting for service. From a system administrator's perspective, response time is an important measure of performance. When it rises too high or is erratic in its peaks, it is an indicator that the communications system, the hardware, and/or software may need fine tuning.

Capacity measures should evaluate when the computer system will be incapable of providing a predefined level of service; predict, for example, when there will be too many simultaneous users for the present hardware and software configuration to support; when existing telecommunications lines will be saturated; and/or when additional disk space, main memory, and/or central processors will be needed. Capacity measures may also figure

in purchasing decisions, such as when to acquire faster memory, central processors, or disks.

Data communication is an essential part of computer system operation, and most computer operating systems include tools for monitoring communications operations. In distributed information retrieval systems, client computers contact server systems and transmit and receive packets of information. Data communications logs of this flow of information may be very extensive or minimal depending on parameter settings (Washburn & Evans, 1993). At a minimum, the logs record a date- and time-stamp, an identification number, the IP address from which the message originates, and characteristics of the data transmitted. Data communications logs can be summarized to provide information on types of transactions processed, quantity of information transmitted to and from a terminal, and processing time per transaction. See Sunsoft (1994b) as an example.

General network communications must also be monitored. This information may be part of the data used in the monitoring of the information retrieval system, or aggregated separately because it is not transaction oriented. Information on network availability, circuit utilization, failure and error rates, and time distributions of network traffic all play an important part in the overall evaluation of a system.

From a historical perspective, much more emphasis was placed on computer performance evaluation (especially analytic models for capacity analysis) 20 or 30 years ago than now.⁶ The reasons for the shift in emphasis is time and cost. The effort of conducting a sophisticated analysis to determine whether a hardware and software configuration will meet specifications is very complex and may only need to be done in mission-critical applications. See Carmonia et al. for one example (Carmonia, Domingo, Macai, Puigjaner, & Rojo, 1994). It is often easier and less expensive to buy a faster disk drive, add additional central processors, or purchase additional data communications lines than to conduct an extensive investigation and then procure equipment or services.

Many computer hardware and operating system manufacturers include software to collect data on system performance. This software allows the system manager to monitor and log system usage. For many years, the IBM Corporation has supplied a sophisticated performance monitoring system called Resource Measurement Facility (RMF) for its MVS operating system.⁷ And it is not uncommon to find users of workstations displaying machine utilization statistics graphically in real time.⁸ Com-

mon measures include central processor utilization, disk utilization, number of pages of memory currently used, and number of page swaps. Even personal computer operating systems, such as Windows NT for workstations offer system monitoring tools (Microsoft Corporation, 1995).

Design Considerations for Monitoring

Design Considerations for Protecting User Privacy

There is no doubt that transaction log analysis is open to abuse. A number of issues must be addressed to protect user privacy:

1. *Browser cookie files.* Some hypertext transfer program server daemons have the ability to store in the client workstation file system a so-called *cookie* file. This file can be updated by a Web site that the user contacts. It is designed to be used to store identifying information about the user (such as a unique identification number) so that when a user returns to that same Web site, the presentation of information can be tailored based on what the system already knows about the user. The user must have the ability to disable the use of cookie files if it is desired.

2. *Server log files.* Some server programs can capture the dialog a client has with it. But just as with traditional transaction log records, privacy can be protected by omitting or encrypting data while still collecting needed information.

3. *User identification in log files.* A major concern is user identification. The benefit of a user identifying him- or herself to the server is that previously established profiles for criteria such as language selection, preferred location of materials, or preferred database can be established to streamline catalog or full-text searching. Identification is required when the user has to pay for online access and when billing records are created. But the user should be able to disable logging, and the transaction log does not need to store user identification, or, if it does store an ID, the ID can be a hashed version of the user's name or other identifier of the user's choice.⁹

4. *IP addresses.* Another concern in creating a log file is that the Internet Protocol (IP) address of the user's terminal can be tracked. It is possible to identify an individual user through the IP address with filtering methods. The IP address is critical during the session since it forms

⁶ See works by Ferrari (1978), Svobodova (1976), and Barnes (1979) written in the late 1970s. Typical of later works are those by McKerrow (1988) and Trivedi, Haverkort, Rindos, & Mainkar (1994).

⁷ See the IBM RMF home page at <http://www.s390.ibm.com/products/rmf>.

⁸ As an example, see the measurement tools provided in the Solaris operating environment in the SunOS operating system (SunSoft, 1994a).

⁹ See Kruse, Leung, and Tondo (1991) for an introductory discussion of hashing. The UNIX password system initially asks a user for an account password, hashes it, and stores the hashed password in a file. The hashing algorithm works only one way, from free-text password to encoded password. This hashed password (or user name) could form the basis for the ID number stored in the user transaction record.

the basis for the lowest level exchange of data, but it could be replaced with a more generic zone designation for logging purposes.

5. *Electronic mail addresses and postal addresses.*

Some information systems give the user the option to receive electronic mail from the server or printed copies of information via the post office. If the user wants his or her privacy protected, the system should be designed to process the user's request and then obscure or delete any reference to it in the transaction log.

Design Considerations for Information Retrieval System Logging

Information retrieval systems (such as library catalogs) are specialized applications that generally employ a well-defined series of steps to allow the user to search and view information. Table 1 summarizes these steps in the form of screens or views that the user sees. To perform a search, the user logs onto the system, enters a search, obtains search results, may have errors or help messages displayed, may download or print the results of a search, and finally leaves the system.

The table shows the type of monitoring that can take place for each activity. It is particularly important for the user, at login, to be able to disable all logging activities. While some of the monitoring activities proposed in this table are already in use, the emphasis in this design is on a much more comprehensive approach to capturing user actions: Being able to understand what other activities are going on at the client system, how the user navigates through the browser and other applications, and what the relationship is between the searching activity and other activities.

Another aspect of the design is inclusion of control devices that the user can set to evaluate the system and the results of the search. Examples include the ability to indicate (perhaps with a slider bar) whether the search entry screen is understandable, a citation is what the user wants, or a help message is understandable. These evaluation knobs, buttons, or slider widgets take up screen real-estate space, and may be ignored, but they have an important function of letting the user convey information about level of satisfaction with the results and the system.

Design Considerations for Client and Browser Logging

The added flexibility that a browser's graphical user interface offers to library automation applications is considerable. However, many transaction logging problems arise from this increased flexibility.

In a command-line interface, transaction logging consisted of recording what the user typed and the system response. The boundary of what is to be recorded becomes much more complex when a browser is employed. The simplest solution is to record the same information as was

recorded in the past: The HTML message that was sent by the client to the server and the HTML message that was transmitted to the client. For many purposes, this may be sufficient. But there are cases where it is not. Examples of potentially useful data are given in Table 2.

Some researchers may say that this is irrelevant information, while others will say they wish they had the data. Clearly, the decision of what to record depends on the purpose of the monitoring, but the choices are not as clear-cut as in a previous era.

The nature of most client-server applications is that they are stateless, that is, the client connects to the server application, receives information, and disconnects. Thus programming methods must be employed to obtain log information about a single client's full interaction (such as a search of a library catalog) however "full interaction" may have been defined. Two methods are currently available. The first uses a so-called *name, value* pair of hidden variables passed to a Common Gateway Interface (CGI) script to retain a continuity of dialog between the two sides (Gundavaram, 1996). When a user begins a transaction or search, the server assigns the user a unique session identification number and transmits it to the client. When the client returns a request for service, this same session ID is returned to the server, which can then make the connection with the client's previous dialog.

An alternate method of maintaining a logical connection between the user and the server is through *cookie* files which are stored in the client's browser subdirectory.¹⁰ If a cookie file is set by a server with the name or ID of the user, each transmission between the two will automatically involve exchange of this identification. Cookie files have privacy implications and this was discussed earlier.

Beyond the problem of maintaining a logical connection in a stateless environment, the programming problems of tracking what the user is doing within the browser are difficult to tackle. It should be possible soon to instrument the browser with calls to a JAVA program or script, to transmit all user browser actions.¹¹

Design Considerations for Server Logging

Almost all server software provides logging of requests, but what the Internet community calls logging bears no resemblance to the needs and requirements of

¹⁰ See *Persistent Client State HTTP Cookies* at the Netscape Corporation's Web site. URL: <http://www.netscape.com/newsref/std/cookie.spec.html>, as well as Gundavaram (1996).

¹¹ The Netscape Corporation's JavaScript and Sun Microsystems's JAVA language contains functions which allow the detection of many client actions to take place. They also contain functions which allow transmission of data between the client and the server under control of the function/applet. See Netscape's Plug-In Software Development Kit documentation at URL: <http://home.netscape.com/eng/mozilla/2.0/handbook/plugins/index.html> and Sun's Java documentation at URL: <http://java.sun.com>. Dragan and Mace (1996) provide a general introduction of how JavaScript works within Netscape.

TABLE 1. Monitoring activities in an information retrieval system application.*

Screen name	Function	Monitoring activity	User evaluation method
Login screen	Allow user to log onto information retrieval system by supplying user identification and password (if necessary).	Transmit user ID to accounting server for authentication	User can tell system to switch off logging, user identification, or both
Search entry screen	Allows user to enter search request or query	Transmit search query to server for processing and log the query	Allow user to indicate clarity of screen
Search results screens			
A. Browse results screen	Display results of search in compact form	Monitor size of browse window, scrollbar usage, mouse usage, citations displayed, citation(s) highlighted or selected, browse screen options selected (change format of browse display, find more or less, find similar). Transmit data to server	Allow user to set slider to indicate level of satisfaction with a citation or the search
B. Citation display screen	Display the citations retrieved as a result of the search	Monitor size of display window, scrollbar usage, mouse usage, citations displayed, options selected (change format of citation display, find more or less, find similar, find same author/title/word/subject), links followed (holdings information, another URL, text, sound, image, abstract). Transmit data to server	Allow user to set slider to indicate level of satisfaction with a citation
C. Text, citation, audio, video, image results screen	Display text, image, audio, or video on client system	Monitor size of display window, user control of the object, options selected. Transmit data to server	Allow user to set slider to indicate level of satisfaction with the object
Error and help messages screen	Display error and help messages on client system	Record error and help messages displayed. Transmit data to server	Allow user to indicate helpfulness of message
File/print/mail actions screens	Allows user to save a document to a file, and print or mail a document	Record these local actions and transmit information about them to the server	Allow user to indicate ease of use of these functions
Logout screen	Allow user to log out of information retrieval system	Transmit logout to server to close session	Allow user to evaluate the entire search. Include a "quit" or "logout" button on every screen to facilitate identification of the end of a session

* All monitoring activities are done with the explicit approval of the user of the system.

TABLE 2. Client and browser instrumentation.*

Device/function	Example activity monitored	Motivation for monitoring
Mouse	Mouse movements, button clicks, button used	User navigation on screen
Keyboard	Keystrokes, function keys, numeric keypad, control keys, arrow keys	User input
Window	Number of windows, window geometry, window foreground/background color, front type/size, tasks operating in window	Spatial arrangement of desktop, human factors of window layout and color
Window controls	Scroll up and down, scroll left and right, slider settings (if any), radio button settings (if any)	User operations at window
Browser functions	File activities, edit activities, bookmark usage and activities, page backward, page forward, go to home page, reload file, open new URL, print, find, mail, save file, stop	User operations with browser
Objects (text, citations, images, audio, video)	Movement through text, text pages viewed, citations viewed, length of play of audio and video, highlighted text/images selected, menu choices made, level of zoom, degree of rotation, level of sound, speed of playback, color balance, image resolution	How objects are manipulated
Browser internal activities	Plug-in applications available, plug-in applications executing, JAVA script executing, Helper applications executing, other applications executing, browser state variable information (disk and memory cache size, enabled/disabled, capacity; cookie file contents; bookmark file contents)	Internal and external state of the browser

* All monitoring activities are done with the explicit approval of the user of the system.

the library and information science community. Several different formats of log records are generated by HTTP daemons including those promulgated by the European Laboratory for Particle Physics (CERN) in Geneva, and the National Center for Supercomputing Applications (NCSA) in the United States. The fields in a log record typically include the name or the IP address of the user accessing the server, the authenticated name of the user, the date and time of the request (and the time offset from GMT), the type of request made (get a document, image, file), a code indicating the status of the request, and the number of bytes transferred as a result of the request. Essentially the log is content-free (from a library transaction log perspective).

Nevertheless, these logs provide the basis for a veritable cottage industry of software developers who offer both public domain and commercial products to analyze the logs.¹² Many of the products have attractive graphical displays of data by time of day/week/month of access, source of IP address, files examined, and links followed, but with such a minimal set of data fields in the log there is only so much one can do.

Some commercial HTTPD analysis tools imply considerably more sophistication in monitoring, requiring that the monitoring program begins execution and then invokes the HTTP daemon, thus putting a wrapper around the daemon and allowing capture of more information.

To implement transaction logging, as the library community understands it, requires intercepting each communication between the client and the server. As a message

is sent, a routine is invoked which writes relevant data to a file including a date- and time-stamp, the unique session ID number (see the discussion of browsers above), and the substance of the request. It must be stressed that this is not done automatically by existing daemons; a program must be written to do it.

If the server does not have its own store of, say, bibliographic information or full text in its database, it will query other computer systems. This inquiry may be made via the Z39.50 protocol (American National Standards Organization, 1995), by accessing a proprietary database through an institution-defined protocol, or perhaps via Structured Query Language (SQL) requests.¹³ Whatever the form of the access to the remote or local database, these requests are candidates for logging since they represent an intermediate response to the user's request and provide valuable information on how the user request was handled by the server.

Design Considerations for Computer Performance Logging

The goal of Web-based information retrieval systems logging should be to integrate existing transaction logging activities with existing computer performance monitoring to produce a log that contains both types of information. The following problems will be encountered in pursuit of this goal:

1. *Process vs. transaction orientation.* Most performance evaluation data is collected over time at predefined intervals. For example, to measure the usage of a data

¹² An important source of information about existing hypertext transfer protocol daemon analysis tools is maintained by Lars-Owe Ivarsson of Uppsala University in Sweden. URL: <http://www.hypernews.org/HyperNews/get/www/log-analyzers.html>.

¹³ Both Bowman, Emerson, and Darnovsky (1996) and Groff and Weinberg (1990) provide a good introduction to SQL.

communications line, samples of usage of the line might be made every 5 seconds and the results written to a log file. However, a transaction log is transaction-oriented, that is, content is associated with a users' search sessions, so it would be difficult to associate performance data with use data in any meaningful way.

2. *Process or task execution time.* Some types of performance data are associated with tasks or programs that are executed. For example, when a program is executed, it is possible to see the kind and quantity of resources it uses, such as central processor utilization, memory, or disk accesses. Unless special logging mechanisms are installed, there is only one HTTP daemon program executing in a server or one browser program executing in a client, so it is difficult to associate resources consumed with the requests of a single user.

3. *Organizational constraints.* Computer and communications system performance data is often collected by two separate organizational units, and transaction logs by a third. Ideally, one unit would be responsible for maintaining a unified log, but organizational difficulties and political constraints may prove a greater impediment to achieving this goal than technical problems.

4. *User exits from commercial monitoring software.* Many commercial software packages are available to perform computer performance evaluation.¹⁴ These packages produce very good reports with minor customization. Transaction logging requires extraction of the same data but integration of it into another format, requiring additional programming to achieve. User exits from commercial software packages are escape holes in the packages that allow the user to take advantage of the overall structure and operation of the package, and, at the same time, collect and analyze data customized for a specific application, such as Web information retrieval transaction logging.

Tables 3, 4, and 5 summarize computer and communications systems information that could be monitored. Monitoring characteristics of the operating system, data communications system, and applications programs (as in Table 3) helps define the environment within which the client is operating. Without this type of baseline information, it is hard to understand and/or control for all the variables that have an impact on what the user does. Table 4 carries this analysis further with specific reference to computer performance evaluation of a comprehensive

monitoring system, listing some of the components that must be monitored on both a server and client computer. Finally, Table 5 presents a number of measures of performance that can be used to characterize each component and some general measures of performance.

System Design

Previous sections of this article have shown that one must adopt a number of perspectives in evaluating information retrieval systems: The user, the manager, the systems designer, and the customer services representative. In addition, several key elements must be brought together to develop an integrated instrumentation and monitoring package for network-based information retrieval systems: Tools for measuring user evaluation of the information retrieved, user behavior, user preferences, the interface to the system, and the activities and performance of the server, the client, the browser, the communications network, and the database server.

The problem of integration is made apparent when one examines the number and type of monitoring records that would be generated in a Web configuration. Table 6 summarizes 12 different types of monitoring records that fall into these categories: System state, performance, transaction, and administrative records. The system state records include information about the configuration and capabilities of the client, the Web server, and the Database server. The performance records give workload and capacity measurements (like those outlined in Table 4 and Table 5). The transaction records include information from the information retrieval system (as in Table 1), as well as transaction data passing between the Client, Web, and Database servers. Finally, the demographic and accounting records capture specific data about the user.

It will be a significant challenge to integrate information from these sources into a unified framework that is flexible enough to meet current and future demands on multiplatform distributed hardware and software systems running in a client-server environment.

The proposed design is itself based on a client-server model. It assumes that there are six servers providing information to a number of different categories of clients. The "servers" are described as such to imply that they can be considered separate computer systems interconnected via a network. In fact, they are distinct programs or services that could run within the same machine or on separate machines, depending on the preference of the organization. The servers include a Web Server, Database Server, Log Server, Monitor Server, Accounting Server, and Assistance Server. There are several categories of clients in this design: A normal User Client, a System Administrator Client, and a System Information Client (see Fig. 2).

The model operates as follows: Transactions between the User Client and the Web Server are logged by the Log Server. The Log Server monitors all packets and

¹⁴ Examples include *Spectrum* from Cabletron Systems (URL: <http://www.cabletron.com/products/>), *Simulog* from a company with the same name (URL: <http://www.simulog.fr>), *OMEGAMON II* from Candle Corporation (URL: http://www.candle.com/product_info/solutions/), *COMNET III* from CACI Products Company (URL: <http://www.caci.com/Simulation.html>), and *RMF* from the IBM Corporation (URL: <http://www.s390.ibm.com/products/rmf>).

TABLE 3. Functions and activities performed at client computer.*

Function/activity	Example function/activity	Monitoring activity
Operating system functions	Task management, file management, print services, input/output management, interprocess communication	<ol style="list-style-type: none"> 1. Transmit to log server hardware characteristics such as CPU type, memory configuration, disk configuration, graphics board type, monitor type and size. 2. Transmit to log server performance information about client machine operations
Data communications tasks	Physical data link management, socket management, TCP/IP communications, mail transmission and receipt	Transmit to log server the characteristics of the physical data link (such as dial-up line, Ethernet, fiber optic), the form of communication (TCP/IP, ATM, PPP), speed of communication line
Application program activities	Execution of application programs such as spreadsheet, word processor, database manager, browser	Transmit to log server the names and versions of active applications, information about placement of application windows on the user's monitor, and computer resource consumption by the applications

* All monitoring activities are done with the explicit approval of the user of the system.

transactions on the information retrieval network, including those between the Web Server and the Database Server. The Monitor Server is a data synthesizer; it takes the raw data from the Log Server and organizes it. The Assistance Server is called by the Web Server to diagnose problems. The Accounting Server is used for user authentication. The System Administrator Client controls the monitoring process, and the System Information Client views synthesized monitoring data.

The capture of user interactions with an information retrieval system is a small subset of what the design permits. Logging of user interactions begins when a user makes initial contact with the Web Server information retrieval system. A log record is created indicating the initiation of a session, another log record is created with data describing the user, and still another with the user's first request for information to the system. These records are accumulated by the Log Server. As the user continues to interact with the server through a browser, more log records are created for each user request and for each Web Server response. As with traditional logging, the log records are captured (in this case by the Log Server), and then sent to the Monitor Server for synthesis, analysis, and ultimately archiving. In the traditional model, an analyst would use the data stored on the Monitor Server's database to produce reports on the system's operation. In this current model, the same activity is possible using a System Administrator Client, but tools may be available in the Assistance Server to do this automatically and also to conduct diagnosis of user errors in real time.

The Web Server

The Web Server is the standard machine supplying information to clients over a communications network. A browser program, executing on a client computer, makes a request to the Web Server which, in turn, supplies the information to the client. The Web Server may have data-

bases attached to it, so it can respond to information requests that do not then need to be passed to the Database Server.

The Database Server

The Database Server is a machine that can be on the same internal network as the Web Server or external to it. This machine supplies information to the Web server based on requests from a client's browser. The Database Server may supply data tables from its files, or may be executing a database management program such as Oracle, Informix, or Sybase to supply the Web Server's information requests. Queries to the Database Server may take the form of a home-grown protocol, Z39.50 requests for information retrieval systems, or SQL queries for more general types of information. Disk drives, which hold the database(s) are attached to the Database Server.

The Log Server

The Log Server is at the heart of the evaluation system. It acts as a focus point to gather data about the operation of the information system. It has the ability to capture all network traffic in the information system and to store it temporarily.

From the information presented earlier, it is clear that maintaining a transaction log of user requests and system responses only will not suffice if comprehensive evaluation studies are to be conducted. Two categories of data must be collected: High-level transaction log records, which permit the type of transaction log analysis conducted previously, and low-level records, which allow for much more detailed analysis of user and system interactions.¹⁵ The high-level data collection is straightforward

¹⁵ See the previous section of this article on *Monitoring and Evaluation of Information Retrieval Systems* which gives details on how information retrieval requests have been captured and analyzed.

TABLE 4. Characterization of computer system configuration.

Component	Description
Central processing unit	Manufacturer, clock speed, cache configuration, number of CPUs, machine cycles per instruction
Disks	Manufacturer, configuration (such as single, RAID), interface type (like SCSI), bandwidth, access time, rotational delay, capacity
Disk controller	Manufacturer, clock speed, bandwidth
File system	Type (such as NFS, NT, DOS), version, supplier
Network	Number of networks, network protocols (such as TCP/IP, PPP, X25), media type (such as Ethernet, Token ring, FDDI, ATM), input buffer size, maximum packet size, bandwidth
Operating system	Manufacturer, name, version, kernel configuration parameters (such as buffer size, swap space size, paging parameters)
Memory	Size, configuration, speed, type
Database management system	Manufacturer, name, version, configuration parameters (such as buffer size, cache size)
HTTPD daemon	Supplier, version, configuration parameters
Application software	Supplier, version, configuration parameters, compiler options when compiling (such as optimization level), algorithm efficiency, source code efficiency

to implement, and the log server is designed to intercept data between the User Client and the Web Server and to store this data (as described in Table 1). It does this by listening for packets being passed between the client and the server over the network.

Low-level data collection is needed to understand details of user interactions (such as the ones outlined in Tables 2 and 3). To capture this type of data, the Log Server must operate at the network interface level so that packets are intercepted as they come from the Ethernet

(or other type of media) and before they are processed by the lowest level of the TCP/IP code.¹⁶

Implementing the low-level data collection presents a number of problems:

¹⁶ This type of interception is done by so-called *packet sniffer* programs. One such program in the UNIX environment is called *snoop*. Another way to obtain the information is to configure the network interface to operate in *promiscuous* mode, whereby it captures all packets on the network regardless of their destination.

TABLE 5. Computer system performance measures.

Measurement category	Measures of performance
General measures	Average response time, transaction throughput per second, system availability, user load
System—CPU	CPU utilization, percent idle, percent busy, tasks/processes waiting for CPU to execute, load balance over multiple CPUs
System—Buffer/swapping/ paging/memory	Buffer cache read/write faults, processes swapped out of memory per second, page faults per second, pages moved in and out of memory per second
Processes/commands	Average execution time, length of run queue, frequency of use, CPU minutes used, real minutes used, blocks read
Disks	Average service time, percent of time device busy, average queue length for requests, number of read and write requests per second, bus utilization rate, available space on device
Files	Percent of file handles used, extent of file system fragmentation, file sizes, directory sizes
Input/output channel	Throughput per second, utilization rate
Network	Bytes transferred per second, network/line/port utilization percentage, packet collision rates, volume of packets sent and received, rate of network file system and remote procedure bad calls, errors per input/output packet, client time-outs
Database management system	Transaction throughput per second, query response time, average number of commands in run queue, query use (lack of use) of indexes/clusters/hash tables, disk and memory buffer cache utilization, cache misses, database task/data communications/input-output manager percent of time busy, percent of time query sorts take place on disk vs. memory, SQL query efficiency
Printer	Jobs in print queue, size of jobs, printer status (online, offline)
Application	CPU utilization per application, transaction throughput per second (if applicable), connect hours, uptime
HPPT daemon	Number of user sessions, frequency of viewing of pages, links followed, frequency of end use by time period, user IP address analysis, time required for downloads, type of browser and operating system used, thread creation time, response time, transaction throughput per second, frequency of abnormal terminations
Accounting	Number of user logins, number of root logins, length of time user connected, number of user sessions, file utilization, memory utilization, commands issued, time for each process forked by a user

TABLE 6. Categories of monitoring records.*

Record type	Sample record contents	Motivation for monitoring
Client system state	Operating system, data communications system, application program, and browser characteristics and configuration	Establish the computing environment in the client machine
Web server system state	Operating system, data communications system, application program, and Web daemon characteristics and configuration	Establish the computing environment of the Web server
Database server state	Operating system, data communications system, database management system, snapshot of the database	Ascertain the operating environment of the database server. Take snapshot of database to support retrospective evaluation (if feasible), or verification of search/retrieval problems
Web server performance records	Continual feed of data on computer performance of the server	Track performance and analyze capacity and workload
Client performance records	Continual feed of data on computer performance of the client including active applications running on client	Track performance and analyze capacity and workload
Database server performance records	Continual feed of data on computer performance of the database server	Track performance and analyze capacity and workload
Client messages to Web server	Content of the HTTP requests made by the client to the Web server	Log user activities
Web server messages to client	Information that the Web server sends to the client, including control data, content messages, cookie requests, JAVA applets sent, JavaScript files sent	Log system activities
Web server messages to database server	Queries and control information the Web server sends to the database server including SQL queries or Z39.50 requests	Log Web server messages to database server
Database server messages to Web server	Retrieved data and control information sent by database server back to Web server; could be text, images, audio, video	Log information supplied to Web server
Client demographic information	Age, gender, education, employment, purpose of visit	Log client data gathered from a guestbook or questionnaire
Accounting/login information	User name and/or ID, password, login time and date, IP address	Track individual use of system for authorization or accounting purposes

* All monitoring activities are done with the explicit approval of the user of the system. Not all applications will have a database server, but many will contact another machine (which may not be a server) to obtain information to complete a request by the client.

1. While the application program may see complete messages, the raw data formed into messages consists of a series of fragments (the nature of the IP protocol), and assembly must be performed to obtain a coherent packet.
2. Data collection must be extremely efficient because of the potentially high bandwidth of the network.
3. If an authentication system such as Kerberos is operating (Stevens, 1990, pp. 430–436), the log server must have the current session key to decode the packets.

Why is this low-level analysis necessary? Some types of measurement cannot take place at the application level because many details are filtered out by the time the transaction reaches this level. For example:

1. Servers only know what they think was sent to a user, not what was actually delivered. It may be important to know if the user stopped transmission of an HTML page or image mid-way for effective site evaluation.
2. The effective data transmission speed of packets to a client must be determined at a low level. If the Web Server knew the line speed at which the client was connected, it could tailor its messages to that speed by omitting complex graphics on, say, 38,400 bps lines.
3. User, system, and network response time can be more accurately measured with low-level analysis.

The Log Server would be configurable by the System Administrator Client to capture prescribed data and to forward various forms of the data to the Monitor Server. Likewise, the Log Server could be configured to send applets to Clients so as to be able to detect events occurring at the client. For example, an applet might be sent that would cause data about mouse movement in the Client to be forwarded to the Log Server.

The Monitor Server

The Log Server is the data capture system, while the Monitor Server is the watchdog for the monitoring data. The System Administrator Client specifies the nature of the information to be sent to the Monitor Server from the Log Server, the Monitor and Log Servers do handshaking, and the feed of data begins. But the Monitor Server performs other functions as well:

1. It stores log data in a database for further analysis. In all likelihood, the data would be stored in the form of an SQL database or perhaps a SAS database.¹⁷ Storing

¹⁷ SAS provides specific tools for analyzing system performance data through its SAS/CPE product. See <http://www.sas.com/software/it-vision/index.html>.

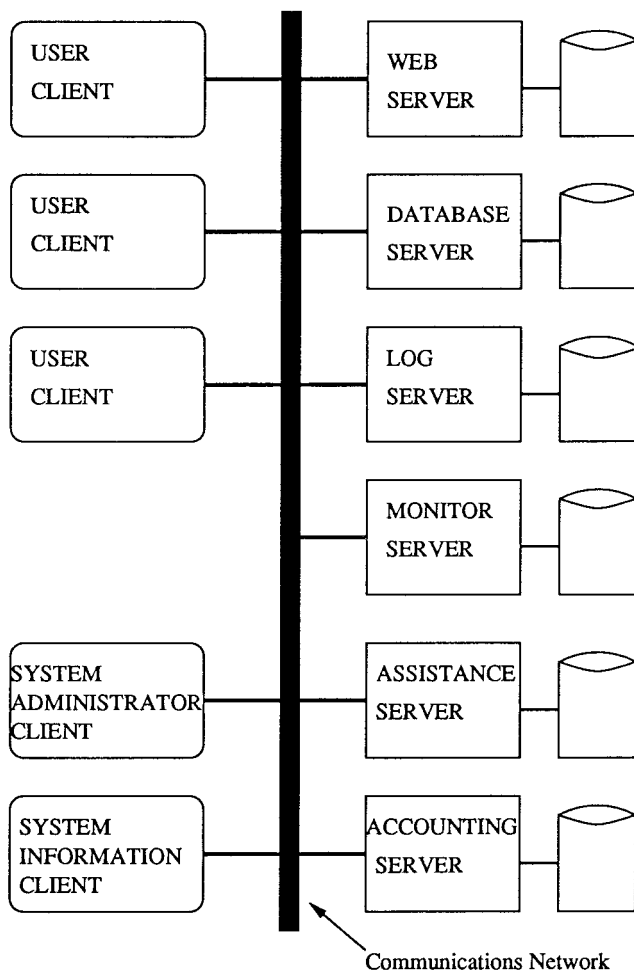


FIG. 2. Client-server monitoring system design.

the data in an SQL database makes it generally available to many client applications.

2. It monitors the log data for conditions established by the System Administrator and notifies the Administrator, the Web Server, or the Database Server of these.
3. It analyzes the log data and stores it in an archival database organized by day, week, month, and year, and subdivided by type of log record.
4. It maintains a data dictionary of the contents of the database.
5. It maintains other databases necessary for the control and analysis of the data. In the case of online catalogs, it is useful to know IP addresses of workstations that are in public areas. This aids in detecting in a transaction log file where a user session begins and ends. If users supply demographic information from a guestbook, it would be stored here. If standard reports were defined for evaluation, they would be stored here.
6. It archives the data in a hierarchy of storage systems as the quantity of log data grows and demand for it declines.

The Assistance Server

This server provides assistance to the end-user of the information system. When the Web Server detects a prob-

lem it calls on the Assistance Server with the specifications of the problem, and this server supplies needed data or advice. Many different levels and types of assistance can be offered, including:

1. Supplying the text of a help message to be displayed at the client by the Web Server.
2. Supplying the text of an error message.
3. Diagnosing a problem based on Web Server-supplied parameters.
4. Invoking an *agent* to help the user with a particular problem (Russell & Norvig, 1995).
5. Maintaining a technical support database to track problems and possible solutions. This database might originally be limited to information retrieval search problems but could be extended to hardware, software, and data communications problems.

The Accounting Server

Under some circumstances, clients using Web resources may have to use a login procedure to gain access to the Web Server and the Database Server. This login may be voluntary for some resources, required for others. Voluntary logins may be encouraged where none are required so that users can store profile information such as preferred electronic and paper-copy mailing addresses, search preferences (e.g., language=english, location of materials=my local library), and stored search statements for repeated execution.

The Accounting Server is designed to authenticate client users, system administrators, and those who wish to access the monitoring data. It stores user information and encrypted passwords and is called upon by the Web Server or Monitor Server when either of those machines gets requests for information.

The System Information Client

A typical client of the Web Server connects to retrieve information from the Web or Database Servers. In the current design configuration, another type of client exist: The System Information Client. This client connects to obtain information about the monitoring activities.

Among the possible functions this client could perform are:

1. Conducting computer performance evaluation studies of the entire system with data stored on the Monitor Server.
2. Providing real-time assistance to Client users through direct examination of log records passed from the Log Server to the Monitor Server (with the help of the Assistance Server). This assistance could be simple guidance with user search problems or more complex diagnosis of hardware or software problems. A sophisticated version of the Assistance Server might incorporate diagnostic agents that could be invoked by the System Information Client.

3. Analyzing Web Server and Database Server data to produce trend, geographic, visitor, page hit rate, path (URLs followed), task, and time-of-day analyses.
4. Performing transaction log analysis from the Monitor Server database using SQL queries against the database or specialized analyst-written programs.

Summary and Conclusions

Transaction log analysis of information retrieval systems has been an effective means of evaluation for years. Changes in technology, the computing paradigm of client-server architecture, and data communications methods call for a reexamination of how this monitoring should take place. The article has developed a client-server architecture for distributed monitoring and user assistance that relies on a series of specialized server computers to collect, process, and synthesize transaction log data, computer performance data, and user activity data.

Implementation of this conceptual design raises many problems and issues. Among the more difficult are managing the volume and level of data that would be accumulated by full instrumentation of a user client, developing a robust protocol by which the servers can negotiate the form of transfer of data between themselves, designing an instrumentation package to work within many different types of hardware and browser software, and most importantly insuring that user privacy is protected throughout any monitoring that occurs.

Acknowledgments

My thanks to Michael Berger, Barbara Norgard, Elisabeth Beller, and the referees for their thoughts on drafts of this article.

References

- American Library Association. (1996). *ALA policy manual—Section two (positions and public policy statements), section 52.4 (confidentiality of library records)*. Available: gopher://ala1.ala.org:70/00/alagophviii/policy.hb
- American National Standards Organization. (1995). *Information retrieval (Z39.50), application service definition and protocol specifications. ANSI/NISO Z39.50-1995*. Bethesda, MD: NISO Press.
- Aronson, L. (1995). *HTML 3 manual of style*. Emeryville, CA: Ziff-Davis Press.
- Ayre, R. & Mace, T. (1996). Internet access: Just browsing. *PC Magazine*, 15(5), 100–147.
- Barnes, M. F. (1979). *Measurement and modeling methods for computer systems performance studies*. Langton Information Systems series. Input Two-Nine Ltd. Studentlitteratur. An ECSS European Computer Science Series publication. Surrey, Eng.: Langton Information Systems.
- Berghal, H. (1996). The client's side of the World-Wide Web. *Communications of the ACM*, 39(1), 30–40.
- Borgman, C. L., Hirsh, S. G., & Hiller, J. (1996). Rethinking online monitoring methods for information retrieval systems: From search product to search process. *Journal of the American Society for Information Science*, 47, 568–583.
- Bowman, J. S., Emerson, S. L., & Darnovsky, M. (1996). *The practical SQL handbook: Using structured query language* (3rd ed.). Reading, MA: Addison-Wesley Developers Press.
- Carmonia, A., Domingo, L., Macai, R., Puigjaner, R., & Rojo, F. (1994). Performance experiences of the Barcelona Olympic Games computer system. In G. Haring & G. Kotsis (Eds.), *Computer performance evaluation: Modelling techniques and tools*. Proceedings of the 7th International Conference, Vienna, Austria, May 3–6, 1994. Lecture notes in computer science No. 794 (pp. 52–75). Berlin: Springer-Verlag.
- Computer Emergency Response Team (CERT). (1992). *Keystroke logging banner. CERT Advisory. CA-92:19, December 7, 1992*. Available: [ftp://ftp.uwsq.indiana.edu/pub/security/cert/cert_advisories/CA-92:19.Keystroke.Logging.Banner.Notice](http://ftp.uwsq.indiana.edu/pub/security/cert/cert_advisories/CA-92:19.Keystroke.Logging.Banner.Notice)
- Computer Professionals for Social Responsibility. (1996, September 9). *Electronic privacy guidelines (1996)*. Cyber Rights Working Group of Computer Professionals for Social Responsibility. Available: <http://www.netaction.org/privacy/guidelines.html>
- Cooper, M. D. (1996). *Design of library automation systems: File structures, data structures, and tools*. New York: Wiley.
- Dragan, R. V., Mace, T. (1996). Inside the Netscape navigator platform. *PC Magazine*, 15(17), 283–289.
- Elmasri, R., & Navathe, S. B. (1994). *Fundamentals of database systems* (2nd ed.). Redwood City, CA: Benjamin/Cummings.
- Ferrari, D. (1978). *Computer systems performance evaluation*. Englewood Cliffs, NJ: Prentice-Hall.
- Flaherty, P. (1993). Transaction logging systems: A descriptive summary. *Library Hi Tech*, Issue 42, 11(2), 67–78.
- Gomez, L. (1996, February 13). Leading Web browsers may violate privacy of users' computers, activities. *San Jose [California] Mercury News*.
- Grant, R. A., & Higgins, C. A. (1996). Computerized performance monitors as multidimensional systems: Derivation and application. *ACM Transactions on Information Systems*, 14(2), 212–235.
- Gray, J., & Reuter, A. (1993). *Transaction processing: Concepts and techniques*. San Mateo, CA: Morgan Kaufmann.
- Groff, J. R., & Weinberg, P. N. (1990). *Using SQL*. Berkeley, CA: Osborne McGraw-Hill.
- Gundavaram, S. (1996). *CGI programming on the World Wide Web*. Sebastopol, CA: O'Reilly & Associates.
- Hunt, C. (1994). *TCP/IP network administration*. Sebastopol, CA: O'Reilly & Associates.
- Johnson, D. G., & Mulvey, J. M. (1995). Accountability and computer decision systems. *Communications of the ACM*, 38(12), 58–64.
- Kaske, N. K. (1993). Research methodologies and transaction log analysis: Issues, questions, and a proposed model. *Library Hi Tech*, Issue 42, 11(2), 79–86.
- Kruse, R. L., Leung, B. P., & Tondo, C. L. (1991). *Data structures and program design in C*. Englewood Cliffs, NJ: Prentice-Hall.
- Kurth, M. (1993). The limits and limitations of transaction log analysis. *Library Hi Tech*, Issue 42, 11(2), 98–104.
- Libes, D. (1995). *Exploring expect: A Tcl-based toolkit for automating interactive programs*. Sebastopol, CA: O'Reilly & Associates.
- Lipschutz, R. P. (1996). Web servers. *PC Magazine*, 15(15), 167–204.
- Liu, C., Peek, J., Jones, R., Buss, B., & Nye, A. (1994). *Managing Internet information services*. Sebastopol, CA: O'Reilly & Associates.
- McKerrow, P. (1988). *Performance measurement of computer systems*. International computer science series. Sydney: Addison-Wesley.
- Microsoft Corporation. (1995). *Administrator's guide*. Microsoft systems manager server for windows NT, Document No. 66254. Redmond, WA: Microsoft.
- Peters, T. A. (1993). The history and development of transaction log analysis. *Library Hi Tech*, Issue 42, 11(2), 41–66.
- Russell, S., & Norvig, P. (1995). *Artificial intelligence: A modern approach*. Prentice-Hall series on artificial intelligence. Upper Saddle River, NJ: Prentice-Hall.

- Stevens, W. R. (1990). *UNIX network programming*. Englewood Cliffs, NJ: Prentice-Hall.
- SunSoft. (1994a). *Security, performance, and accounting administration* (Pt. No. 801-6629-10). Mountain View, CA: SunSoft.
- SunSoft. (1994b). *TCP/IP network administration guide* (Pt. No. 801-6632-10). Mountain View, CA: SunSoft.
- Svobodova, L. (1976). *Computer performance measurement and evaluation methods: Analysis and applications*. Computer design and architecture series: 2. New York: American Elsevier.
- Trivedi, K., Haverkort, B. R., Rindos, A., & Mainkar, V. (1994). Techniques and tools for reliability and performance evaluation: Problems and Perspectives. In G. Haring & G. Kotsis (Eds.), *Computer performance evaluation: Modelling techniques and tools*. Proceedings of the 7th International Conference, Vienna, Austria, May 3–6, 1994 (pp. 1–25). Lecture notes in computer science No. 794. Berlin: Springer-Verlag.
- Washburn, K., & Evans, J. T. (1993). *TCP/IP: Running a successful network*. Data communications and network series. Wokingham, UK: Addison-Wesley.
- We know you're reading this. (1996, February 10). *The Economist*, pp. 27–28.
- Williams, M. (1996, April 8). Getting to know you: Internet spins its Web. *International Herald Tribune* (Paris ed.), pp. 11, 13.
- Wilson, S. (1995). *World Wide Web design guide*. Indianapolis, IN: Hayden Books.
- Wood-Harper, A. T., Corder, S., Wood, J. R. G., & Watson, H. (1996). How we Profess: The ethical systems analyst. *Communications of the ACM*, 39(3), 69–77.