US005751956A

United States Patent [19]

Kirsch

[11] Patent Number: 5,751,956

[45] Date of Patent: May 12, 1998

[54] **METHOD AND APPARATUS FOR REDIRECTION OF SERVER EXTERNAL HYPER-LINK REFERENCES**

[75] Inventor: **Steven T. Kirsch**, Los Altos, Calif.

[73] Assignee: **Infoseek Corporation**, Sunnyvale, Calif.

[21] Appl. No.: **604,468**

[22] Filed: **Feb. 21, 1996**

[51] **Int. Cl.**[6] **H04Q 9/00**

[52] **U.S. Cl.** **395/200.33**; 395/329

[58] **Field of Search** 395/329, 335, 395/200.33, 200.47, 200.49, 762, 774, 615

[56] **References Cited**

U.S. PATENT DOCUMENTS

5,572,643	11/1996	Judson	395/793
5,640,193	6/1997	Wellner	348/7

Primary Examiner—Ellis B. Ramirez
Assistant Examiner—Thomas Peeso

Attorney, Agent, or Firm—Fliesler, Dubb, Meyer & Lovejoy, LLP

[57] **ABSTRACT**

A Web server computer system provides for server based controlled management over a client reference to a resource locator independently selected by a client computer system and referencing a server external Web server. The Web server system provides a client system with a predetermined URL reference to the Web server system encoded with predetermined redirection and accounting data including a reference to a second server system On receipt by the first Web server system of the predetermined URL reference from the client system, the predetermined redirection and accounting data is decoded from the predetermined URL and processed by the Web server system to provide the client system with a redirection message including the reference to the second server system. The accounting data is processed by the Web server system and resulting data is selectively stored by the Web server system.

18 Claims, 2 Drawing Sheets

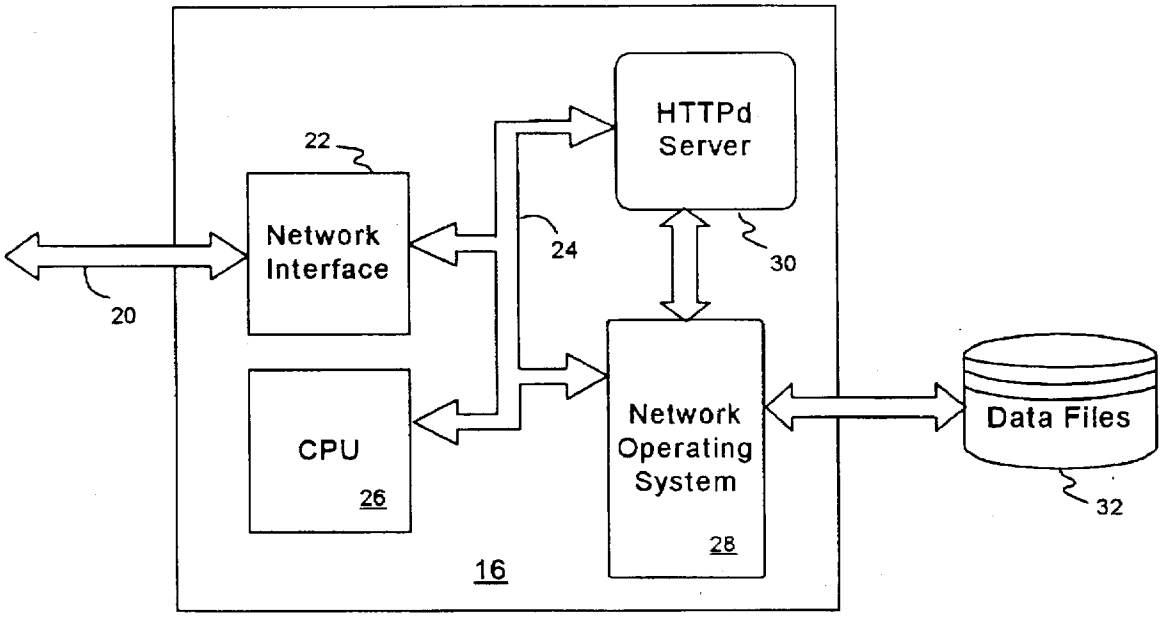
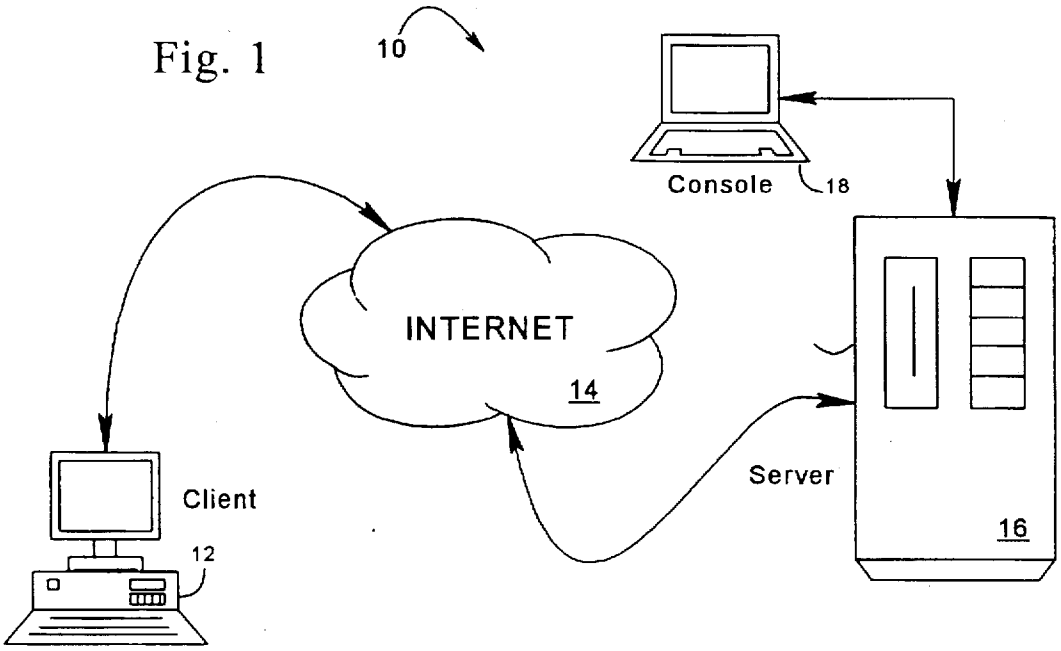


Fig. 1



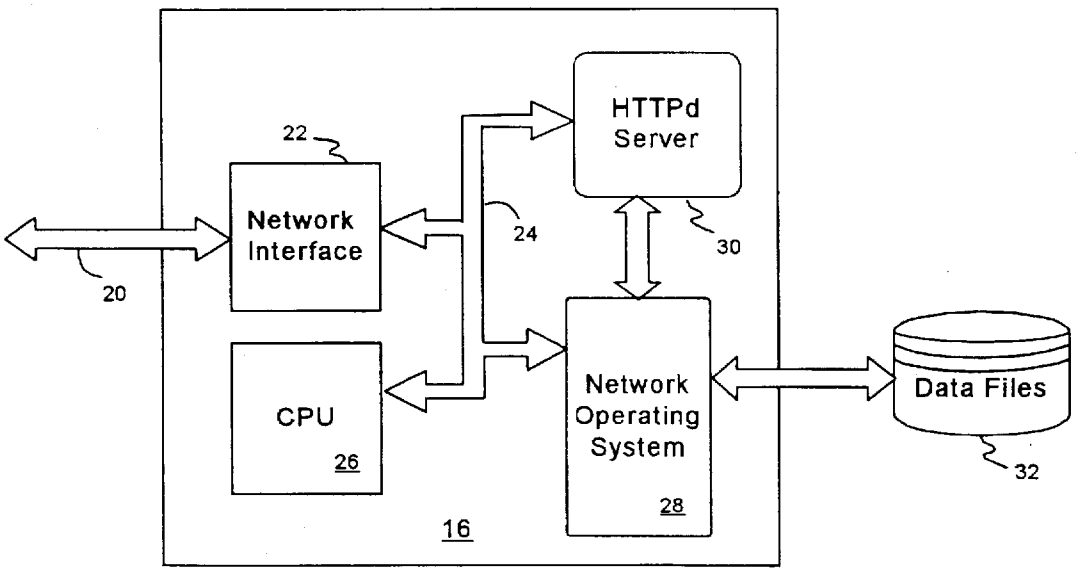
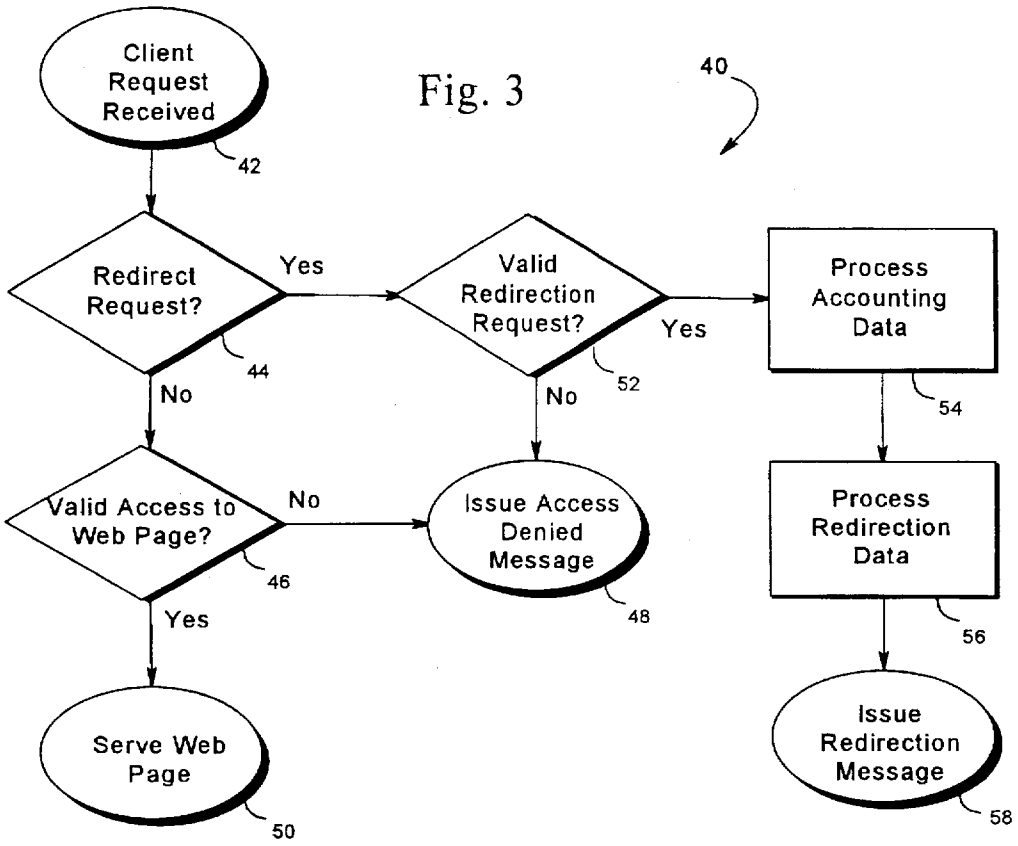


Fig. 2



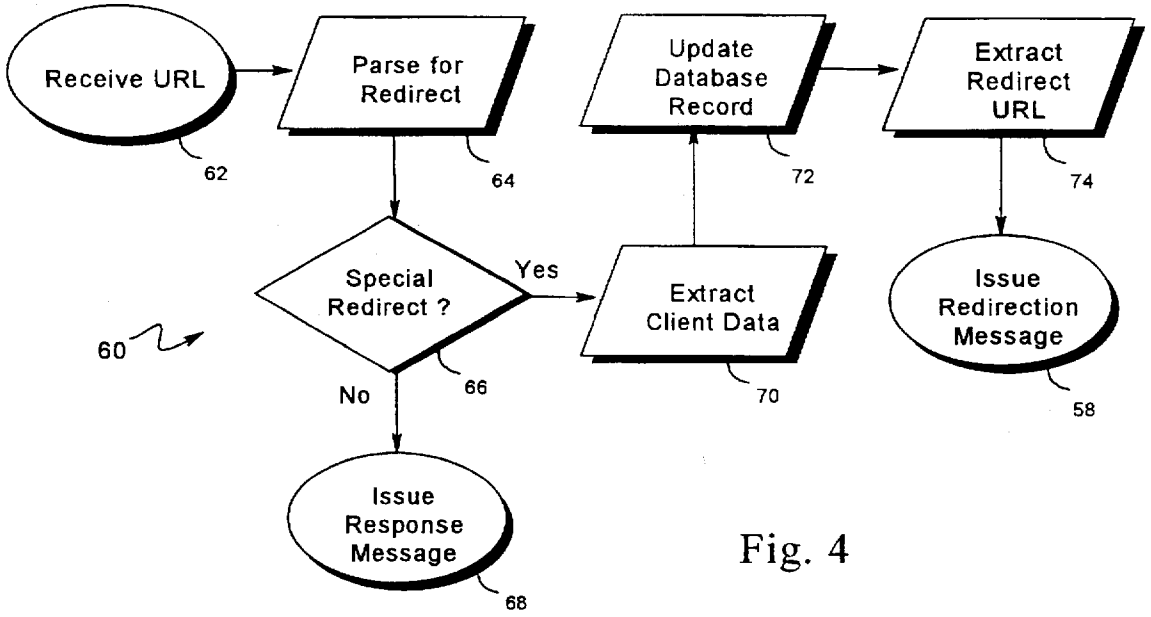


Fig. 4

METHOD AND APPARATUS FOR REDIRECTION OF SERVER EXTERNAL HYPER-LINK REFERENCES

CROSS-REFERENCE TO RELATED APPLICATIONS

The present application is related to the following Application, assigned to the Assignee of the present Application:

1) IMPROVED WEB SCAN PROCESS, invented by Kirsch, [Attorney Docket Number: INFS1003DEL/GBR], application Ser. No. 08/604,584, filed on Feb. 21, 1996 concurrently herewith and;

2) SECURE, CONVENIENT AND EFFICIENT SYSTEM AND METHOD OF PERFORMING TRANSINTERNET PURCHASE TRANSACTIONS invented by Kirsch, [Attorney Docket Number: INFS 1005DEL/GBR], application Ser. No. 08/604,506, filed concurrently herewith.

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention is generally related to the control of network information server systems supporting World Wide Web based data pages and, in particular, to a server system and process for efficiently redirecting external server hyper-link references for purposes of controlling, moderating, and accounting for such references.

2. Description of the Related Art

The recent substantial growth and use of the internationally connected network generally known as the Internet has largely been due to widespread support of the hypertext transfer protocol (HTTP). This protocol permits client systems connected through Internet Service Providers (ISPs) to access independent and geographically scattered server systems also connected to the Internet. Client side browsers, such as Netscape Mozilla and Navigator (Netscape Communications Corp.) and NCSA Mosaic, provide efficient graphical user interface based client applications that implement the client side portion of the HTTP protocol

Server side application programs, generically referred to as HTTPd servers, implement the server side portion of the HTTP protocol. HTTP server applications are available both commercially, from companies such as Netscape, and as copyrighted freeware available in source code form from NCSA.

The distributed system of communication and information transfer made possible by the HTTP protocol is commonly known as the World Wide Web (WWW or W3) or as simply "the Web." From a client side user interface perspective, a system of uniform resource locators (URLs) is used to direct the operation of a web browser in establishing atomic transactional communication sessions with designated web server computer systems. In general, each URL is of the basic form:

http://<server_name>.<sub-domain.top_level-domain>/<path>

The server_name is typically "www" and the sub_domain.top-level_domain is a standard Internet domain reference. The path is an optional additional URL qualifier.

Specification by user selection of a URL on the client side results in a transaction being established in which the client sends the server an HTTP message referencing a default or

explicitly named data file constructed in accordance with the hypertext mark up language (HTML). This data file or web page is returned in one or more response phase HTTP messages by the server, generally for display by the client browser. Additional embedded image references may be identified in the returned web page resulting in the client browser initiating subsequent HTML transactions to retrieve typically embedded graphics files. A fully reconstructed web page image is then presented by the browser through the browser's graphical user interface.

Due to the completely distributed client/server architecture of the Web, as made possible by the URL system further supported by the existing Internet name resolution services and routing conventions, HTTP servers can be independently established with little difficulty. Consequently, the Web has no centrally or even regionally enforced organization other than loosely by name of the top level domain. Searching for information or other resources provided by individual HTTP servers is therefore problematic almost by definition. Because of the time, cost and complexity of assembling comprehensive, yet efficiently searchable databases of web information and resources, commercial Internet Business Services (IBS) have been established to provide typically fee based or advertising revenue supported search engine services that operate against compilations of the information and resources available via the Web correlated to source URLs. Access to such search engines is usually provided through server local web pages served by the Internet Business Services. The results of a search are served in the form of local web pages with appropriate embedded remote or hyper-linked URLs dynamically constructed by the server of the Internet Business Service.

Because of the opportunity presented by the likely repeated client access and retrieval of search engine and search result web pages, providers of other Internet based services have begun to actively place advertisements on these web pages. As is typical in advertising mediums, the frequency of display of an advertisement generally defines the compensation paid to the advertisement publisher. Thus, the number of times that an advertisement is simply transferred to a client browser provides an indication of how effectively the advertisement is being published. A more direct measure of the effectiveness of a particular advertisement on a particular web page is the number of times a client web browser chooses to actively pursue the URL represented by the advertisement. Thus, there is a need to be able to track information obtainable from a client browser when a hyper-linked advertiser's URL is selected.

The difficulty in obtaining direct reference information arises from the fact that a web page with an embedded advertisement and corresponding remote URL is served in its entirety to the client browser upon first reference to the web page. The selection of a particular advertiser's URL is then by definition performed through an independent transaction directed to the HTTPd server associated with the advertiser. Since the advertiser publishing HTTPd server is not part of this subsequent transaction, the publishing server is conventionally incapable of tracking client browser hyperlinks actually executed to an advertiser's URL or any other URLs embedded in a web page previously served to the client browser.

Simple web page access counters are relatively well known and used throughout the Web. These access counters are based on a common gateway interface (CGI) facility supported by modern HTTPd server systems The CGI facility permits generally small programs, at least typically in terms of function, to be executed by a server in response

to a client URL request. That is, the HTML web page definition provides for the embedding of a specific HTML reference that will specify execution of a server side CGI program as part of the process of the web browser reconstructing an image of a served web page. Such a HTML reference is typically of the form:

<img src="http://www.target.com/cgi-bin/count.cgi">

Thus, a counter value incremented with each discrete execution of the CGI program (count.cgi) dynamically provides part of the displayable image of the reconstructed web page. The time, remote client requester, client domain, client browser type and other information that may be known through the operation of the HTTP protocol may be logged as part of the CGI program's function. Consequently, a reasonable manner of accounting and auditing for certain web page accesses exists.

Access counters, however, fundamentally log only server local web page accesses. The client browser to the CGI program is evaluated by the client in connection with the initial serving of the web page to the client browser. The initial serving of the web page to the client browser can be connected, but any subsequent selection of a URL that provides a hyper-link reference to an external server is not observed and therefore is not counted by a CGI program based access counter. Other limitations of access counters arise from the fact that the implementing CGI program is an independently loadable executable. The CGI program must be discretely loaded and executed by the server computer system in response to each URL reference to the CGI program The repeated program loading and execution overhead, though potentially small for each individual invocation of the CGI program, can represent a significant if not substantial load to the sever computer system. The frequent execution of CGI programs is commonly associated with a degradation of the effective average access time of the HTTPd server in responding to client URL requests. Since an Internet Business Service providing access to a search engine logs millions of requests each day, even small reductions in the efficiency of serving web pages can seriously degrade the cost efficiency of the Internet Business Service. As of December, 1995, InfoSeek Corporation, in particular, handles an average of five million retrievals a day.

The execution overhead associated with CGI programs is often rather significant. Many CGI programs are implemented at least in part through the use of an interpreted language such as Perl or TCL. Consequently, a substantial processing overhead is involved in multiple mass storage transfers to load both the interpreter and CGI program scripts, to process the scripts through the execution of the interpreter, and then actually log whatever useful data is generated, typically to persistent mass storage. Finally, the interpreter and/or CGI program may have to be unloaded.

In addition, external CGI programs present a significant problem in terms of maintenance, including initial and ongoing server configuration and control, and security in the context of a busy server system. Individual CGI programs will likely be needed for each independent web page in order to separately identify web page service counts. Alternatively, a CGI program can be made sufficiently by complex to be able to distinguish the precise manner in which the program is called so as to identify a particular web page and log an appropriately distinctive access count. Maintenance of such CGI programs on a server system where large numbers of page accesses are being separately counted is non trivial.

Further, the existence of external programs, particularly of scripts that are interpreted dynamically, represents a potential security problem. In particular, the access and execute permissions of interpreted scripts must be carefully managed and monitored to prevent any unauthorized script from being executed that could, in turn, compromise the integrity of the data being collected if not the fundamental integrity of the server computer system itself. Consequently, known access counters provide no solution directly in full or in part to the need to account or audit URL references to external servers based on hyper-links from previously served web pages.

The HTTP protocol itself provides for a basic server based system of URL redirection for servers and clients supporting the 1.5 or later versions of the HTTP protocol. A configuration file associated with an HTTP server (typically srm.conf) can specify a redirect directive that effectively maps a server local directory URL reference to an external URL reference through the use of a configuration directive of the form:

Redirect /dir1 http://newserver.widget.com/dir1

When a Version 1.5 or later HTTP server receives a URL reference to a local directory (/dir1) that is specified as above for redirection, a redirect message is returned to the client browser including a new location in the form of an URL (http://newserver.widget.com/dir1). This redirect URL is then used by the client browser as the basis for a conventional client URL request.

This existing server based redirection function is insufficient to support external server access tracking since, in its usual form, the redirection is of the entire directory hierarchy that shares a common redirected base directory. Even in the most restricted form, the redirection is performed on a per directory reference basis. Thus, every access to the directory, independent of the particular web page or graphics image or CGI program that is the specific object of an access request is nonetheless discretely redirected without distinction. Any potential use of the existing server redirect function is therefore exceedingly constrained if not practically prohibited by the HTTP protocol defined operation of the redirect directive.

Furthermore, the redirect directive capability of the HTTP protocol server does not provide for the execution of a CGI program or other executable coincident with the performance of the redirection thereby essentially precluding any action to capture information related to the redirect URL request. In addition, the complexity of the resource configuration file necessary to specify redirection down to a per directory configuration again raises significant configuration, maintenance and, to a lesser degree, security issues. Thus, server redirection does not possess even the basic capabilities necessary to support external URL hyper-link reference auditing or accounting.

Finally, a form of redirection might be accomplished though the utilization of a relatively complex CGI program. Such a redirection CGI program would likely need to perform some form of alternate resource identification as necessary to identify a redirection target URL. Assuming that a unique target URL can be identified, a redirection message can then be returned to a client from the CGI program through the HTTP server as necessary to provide a redirection URL to the client browser.

Unfortunately, any such CGI program would embody all of the disadvantages associated with even the simplest access counter programs Not only would problems of execution load and latency, as well as configuration, maintenance and security remain, but such an approach to providing redirection is inherently vulnerable to access spoofing.

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.