Constituencies for Users: How to Develop them by Interpreting Logs of Web Site Access

Michael J. Wright

KMI-TR-75

January, 1999

AAAI Spring Symposium on Intelligent Agents in Cyberspace
March 22-24, 1999, Stanford University, California

Constituencies for Users:
How to Develop them by Interpreting Logs of Web Site Access

Michael J. Wright

Knowledge Media Institute
The Open University
Walton Hall
Milton Keynes, MK7 6AA, U.K.
m.j.wright@open.ac.uk

Abstract

The number of electronic journals is growing as rapidly as the World Wide Web on which many are published. Readership of an electronic journal is important to quantify, just as it is for a printed journal. In maintaining the Journal of Interactive Media in Education (JIME), a scholarly electronic journal open to all, we require readership statistics more meaningful than the variations on the theme of "number of hits" given by many log analysis packages. Important aspects of JIME's open access are the decisions to not require subscription, nor the use of hidden tracking aids such as cookies. Readership information is acquired from the server log of client requests. We are investigating the identification of user sessions from such logs, and the development of classification of sessions into constituencies of readership. In this paper, we present the result of manual analysis from which we are developing automatic analysis mechanisms.

Keywords: Web Log Analysis; Classification; Electronic Publishing; Scholarly Publishing; Recommender Systems

1.0 Introduction

As the World Wide Web develops, the number of quality resources is increasing; for example digital libraries are growing in size and numbers. In addition to libraries, specific journals are appearing on-line. One example is the electronic Journal of Interactive Media in Education (JIME http://www-jime.open.ac.uk/), a scholarly journal hosted by KMi at The Open University and only available on-line. The journal has the same basic publication model as other scholarly journals; submission, peer review, debate and acceptance for final publication based on the review process.

As part of maintaining JIME, there is a need to be able to quantify the readership of the journal. To characterize the use of the journal, a number of approaches are available; from simple statistics gathered from the Web log to sophisticated mechanisms utilizing one or more of information, content-based and collaborative filtering. These mechanisms, often referred to as recommender systems (Resnick and Varian, 1997), usually maintain profiles of the interests of registered users in order to recommend content to them.

As part of the design rationale of JIME, there are no impositions placed on those accessing the journal; anyone may visit and read free from giving any information or receiving hidden tokens (e.g. cookies.) As mentioned earlier, JIME is but one of many journals appearing on the Web. As a user of JIME is likely to be a user of many other such resources, I believe they will not be very interested in directly maintaining a profile of themselves on every resource site they visit. They will find the emerging 3rd party recommender, or broker, systems of more use for this.

This presents a problem to those developing and maintaining a Web site such as JIME. We would like to be able to say that of x visitors to JIME, y of them looked at the abstract of a particular article, A, and z of y went on to read large parts of the article. We also want to see what collections of articles they were interested in, and to what extent they were interested in specialized parts of the journal. To collect such information, we must use whatever records are available; e.g. access logs and queries used in any site based search mechanism. In the future, protocols to enable the exchange of profile information at the time of a user visit will be an additional benefit by bringing more information; they should not be viewed as a complete replacement for the local analysis.

The JIME site access log is a local record that can be analyzed for the information sought. Present log analysis tools, with their statistical output of variations of hits per page, do not give the information on readership patterns required. However, by analyzing the logs in more detail, I believe we can gain insight into the users of JIME and how they read it's contents. Such analysis only gives results for a percentage of the actual visitors due to browser caching and proxy use, but should be an improvement over the present problem of estimating the true readership of a printed journal where estimates of readership in libraries must be made. This information can then be combined with other knowledge of the site (e.g. search queries used on a local search mechanism, or an ontology describing the site) to help understand the use of JIME and potentially give the opportunity for the site to recommend content to visitors.

The work in progress begins the process of looking to identify reader sessions from log files and to be able to classify sessions into one or more constituencies where a constituency has a description of some actions, e.g.

readers who like to view demonstrations. The initial work has been manual analysis of the JIME logs to see that such simple constituencies can be identified. More complex constituency descriptions require the automation of this task. The results will be available to improve the development and maintenance of the JIME site. Further development of methods to automatically correlate new visitors to constituencies will lead to being able to recommend content to visitors.

After a discussion of related work, I will review the basic web log file and examine what types of information can be interpreted from such a log. A user session is defined the implications of a user's browser settings with respect to caching and proxy use are discussed. We then present an initial set of constituencies that can be seen from an inspection of the JIME access log. Examples are taken from that log.

2.0 Related Work

The previous section introduced the project to analyze web logs to develop constituencies of users, with a goal of being able to recommend resources on the JIME site to future users based on their match to one or more constituencies. This relates to research in areas of Web log analysis and recommender systems (Resnick and Varian, 1997).

The need of understanding more about those visiting an information archive (e.g. a Web site) is ubiquitous. Much of the early Web based work has been driven by requirements to improve navigation and caching on the web (Tauscher and Greenberg 1997, Recker and Pitkow, 1996). How to get the information has generated many ideas and implementations beyond just web log analysis as used in the previous work. Two of the more common methods are subscription or registration (information in exchange for access) and the use of cookies. With these methods, a database of access patterns of individuals to a site can be kept. I believe the use of such tracking mechanisms has resulted from them being a simple method to track individuals, bypassing the more complex work of more fully interpreting a log file. Further, their use will probably decline at individual resource sites (except for obvious use at specific sites e.g. E-commerce) as the number of potential resource sites increases to where individuals will find it impossible to maintain current profiles at all these sites.

As the volume of information resources increases, research has evolved into helping people find information by developing systems based on maintaining current user profiles. Recommender systems describes an area devoted to developing systems that help people by recommending possible resources based on profiles of their interests. These systems are seen as an improvement over basic query based search engines such as AltaVista and Yahoo. If we view the continuum from the client interface (e.g. Web browser) to a resource location (e.g. a Web site,) there are systems that work at various stages in this continuum. While three areas are described and examples given, it is inferred that some examples may well bridge areas hence the use of “continuum”.

An example of a system that works at the client interface is Letizia (Lieberman, 1997). Letizia maintains a profile of the user to help present possible alternative Web resources. We can consider the user profiles to be “private” in that they are held at the local client only. Referral is based on interests in the profile matched to resources seen subject to application constraints. Another example of a client system is SiteSeer (Rucker and Polanco, 1997), which builds a profile from a user's bookmarks in a Web browser.

Other systems work in the area between the client and resource location acting as recommender (or brokering) services. Examples of these are Firefly (http://www.firefly.com,) PHOAKS (Terveen et al., 1997), Fab (Balabanovic and Shoham 1997), ReferralWeb (Kautz, Selman and Shah 1997), GroupLens (Sarwar et al. 1998), and SOaP (Guo, Kreifelts and Voss 1997). This intermediate area is a popular area of research at present with each using some form of collaborative (social) or content-based filtering, or a hybrid of the two. They all maintain profiles of many users and have large document spaces from which to recommend, e.g. Fab, ReferralWeb and SOaP draw from the Web, PHOAKS and GroupLens from Usenet News. As users make use of the services on a frequent basis, their profiles are kept up to date. The user profiles are “public” in that the user has agreed to be profiled by signing up. The level of privacy of a profile is on the broker’s terms.

At the individual site (or group of sites) are systems such as ArquiTec (Ferreira, Borbinha and Delgado, 1997) and Planet-Onto (Domingue and Motta, 1999). ArquiTec keeps a user profile database of registered users in conjunction with a digital library in Portugal. Planet-Onto is a site that allows researchers to post articles that are then linked to an underlying ontology. Readers can register to be supplied updates from the Newsboy agent (Newsboy could also be set in the mid-area of the continuum), updates based on a profile specified in unambiguous declarative specifications. This is different from many of the other systems mentioned as those often build up profiles from watching how users deal with recommended material. Again, the profiles are “public”.

The process of developing user constituencies from log analysis will fit the individual site part of this continuum where it can work in conjunction with other site-based information. In the future, as mechanisms develop to allow profile information to be shared with the site from systems in the other parts of the continuum, this will augment the local information.

3.0 Web Site Statistics - Present Practice

3.1 The Log File Format

Visitors leave a trace in a log file kept by the server. Each record in a log is for an individual client request received and dealt with by the server. For example, the request for http://www-jime.open.ac.uk/ will result in the document associated with the root of the server (the file index.html in the root directory of the server document tree) being returned to the client and an entry placed in the server log (single line):

```
dhcp.open.ac.uk - -[20/Apr/1998:10:43:05+0100] GET
                    /index.html HTTP/1.0 200 3347
```

The record has the following syntax (also known as the Common Log Format, CLF, the basic syntax of a log record for most Web servers,) and contains the minimum information around which a request can be completed plus the date:

[Host] [Ident] [AuthUser] [Date:Time] [Method] [Request Path] [HTTP version] [Status] [Bytes]

Host	IP number/domain name of host requesting
Ident	Not used now
AuthUser	User name used in any user/password access
Date:Time	Date and time the request received
Method	HTTP request method (GET, HEAD, POST, PUT, etc.)
Request Path	Path to resource
HTTP version	HTTP protocol version used in request
Status	Result code of the handling of the request
Bytes	Bytes transferred back to the client

Some Web servers can be configured to log more information, but further information is usually based on extra information the client's browser may give, e.g. the URL of the page from which the current request came from, the "referrer page". This information is optionally given by the client's browser, and may in time be blocked by users (following recent concern over privacy and Web access.) The current basis of analysis is to only use the basic minimal set of information used by the server to complete a request. Once done, the analysis can be augmented with extra volunteered information such as "referrer page." Entries in the log file accumulate over time with entries from different clients interleaving if more than one user happens to be browsing the server at a given time.

As we go through the discussion, we will show examples taken from the log of JIME covering the 20th and 21st of April, 1998. The log file for this 48-hour period had 6,952 records (requests for files). Results have been verified in later periods. For clarity, some fields in records may be left out. We should also note that the home page file, index.html, may not appear in the log file explicitly, often appearing as just /. This is because the JIME server is set to automatically assume that a request to http://www-jime.open.ac.uk/ (or any sub-directory of the server) is a request for the "index.html" file that will be found at that location (server configuration of default HTML files in document directories.)

3.2 Conventional Web Statistics

In the example period chosen, 6,952 records, or "hits", were logged. The records show 142 distinct computers were used (IP/host names logged) to make these requests. Many of the web log statistics packages (both commercial and free) give simplistic breakdowns of the numbers: "Web site JIME has received 6952 hits from 142 unique sites, each downloading an average 15.7 pages." There will be further quotes of the number of errors, total bytes transferred, and in some cases, the packages will sort the host names and give a list of the files they requested most frequently (occasionally chronologically.) These type of statistics give an indication of the "loading" of the server, and help identify missing pages or problems (from error reports.) In no way do they give more useful information such as how many are repeat users of the site, or what clusters of information were users looking at during visit sessions.

In some cases, web site managers now put their log results into databases from which they can query for these types of statistics. In such a case, it is possible to query about how often a given computer (via the IP number or host name) has come to the site. However, this type of statistic may not indicate a true repeat user due to the common use of dynamic IP allocation, or shared computer use. This is discussed in more detail later in the section about identifying a user session. To move beyond these simple statistics requires more interpretation of the log file for a web site, interpretation that takes into account the structure of the web site layout.

4.0 Log File Interpretation

In the previous section, the common types of statistics generated from Web log files were described. These statistics do not give information about how a visitor is moving through the site, nor the areas of the site they visit in a meaningful manner; e.g. the visitor came to the home page then went to an article, read the first few sections and downloaded the print version. This type of information comes from closer inspection of the log records in conjunction with knowledge of how the web site is organized. To do this manually beyond looking at a few examples would be impractical; however, performing a manual analysis leads to an understanding of the problem for an automated system.

In this section, aspects of interpretation of the log file are presented: document clusters, user session, the impact of caching and proxies, identifying whether the visitor has been before. A description of a canonicalization process based on the design of the site is discussed, a process that can be used for faster interpretation of visitor actions. This is all going towards making a judgement of what the visitor is doing.

In the next main section, the analysis discussion will be broadened to identify classifications for constituencies of users.

4.1 Document Clusters

The idea of viewing web documents as clusters is not new, e.g. in work that has been directed at ways to improve a user's browser (Pirolli, 1997, Pitkow and Pirolli 1997). However, in this case, the documents are located at a particular location, JIME. Such clusters are site specific. A request for the JIME home page returns a document (the file, index.html) which defines a frame set. This frame set means further documents and resources would normally be requested by the browser in order to fill the frame set (exceptions are seen such as those from certain Web crawlers which may just request a single document, or the "HEAD" of the document for indexing purposes.) This aspect of retrieving several documents and resources is part of the user's browser function to construct the web page for a user. The documents and resources needed by the browser are defined by the web site development. As a result, an interaction with a web site can be viewed as collections of log entries based on what file was initially requested.

The following list shows the set of documents (HTML files) and resources (image files) that would be requested by a conventional browser that requests the JIME home page. These 16 files (the initial request of index.html followed by 15 requests for further documents and resources) would be requested by the browser in order to complete the home page view. As the browser can choose to assemble the page in whatever way it wants after the initial request, the order of the subsequent requests is not set.

```
/index.html
            /jimelogosmall.html
            /contents.html
            /comments.html
            /banner.html
            /current.html
            /resources/icons/jime-small-no-text.gif
            /resources/icons/kmi-button-small.gif
            /resources/icons/help.gif
            /resources/icons/new.gif
            /resources/icons/published.gif
            /resources/icons/pre-print.gif
            /resources/icons/3-frame-icon.gif
            /resources/icons/2-frame-icon.gif
            /resources/icons/charlesworth.gif
            /resources/icons/d3e-icon.gif
```

This can be viewed as a single document "cluster". The cluster is composed of two parts; a set of unique elements only associated with the document initially requested, and a set of shared elements that can be used in a number of documents. While any individual document or resource can be requested as an individual unit at any time, a cluster is usually requested. The visitor will proceed through a web site using the links in the view presented by the browser leaving log traces of requests for the components making up a cluster. Sub-sets of this cluster can be viewed as clusters too, e.g. requesting the document "jimelogosmall.html" file would result in two further files being requested (a cluster of 3 all together):

```
/jimelogosmall.html
            /resources/icons/jime-small-no-text.gif
            /resources/icons/kmi-button-small.gif
```

This cluster is not interesting unto itself as it is a web document holding two graphics. The document is used in completing the JIME home page.

Observing document clusters in the server log file can help interpret what a visitor was doing: looking at the home page, then moving to another page via a link from the home page. Some files are members of one or more clusters (shared elements.) This is especially true of resource files such as the small GIF images used in many web documents. The use of shared elements can help in identifying return visitors via the status codes associated with them (discussed later.)

Such clusters can be condensed to save space and allow for easier interpretation. However, as discussed in the next few subsections, the cluster cannot be compressed into a single entry without first inspecting the cluster entries for information that would be lost.

4.2 User Session

As described above, the request to the JIME home page will result in a maximum of 16 records in the log file. Fewer records may occur due to browser cache settings, and browsers set to not load images. The order of records may change from request to request dependent on the order the browser decides to request them. However, the time between the first and last resource will usually be a few seconds (unless the link is very slow.) If the request is not followed by requests for the remaining documents in a cluster, further information about the visitor may be concluded (e.g. web crawler activity.)

Ascertaining a user session is the first requirement to then go on to make any judgment of what is happening. In the early days of the Web when users were few, and the vast majority of computers on the Internet had their own IP number and host name, following an individual in a log was reasonably easy. Today, that ease has diminished and in future may well disappear. Two things directly influence this; the use of IP allocation on the fly (dynamic allocation), and sharing machines (computer laboratories, cybercafes etc.) We need to distinguish individual visitors from the log bearing in mind these problems.

While a client's machine is on-line, it will usually have a specific IP number (and hostname perhaps.) This is a unique identifier for the visitor for that present moment. With the proliferation of dynamic IP, we cannot guarantee that an IP (or host name) is from the same machine if the time between records using that IP is more than a few tens of minutes. Thus, we can only conclude that a record is part of an ongoing visitor session if the IP's of subsequent records occur within a certain time of the last record. If someone left their machine for a while (coffee break, lunch,) then return to carry on viewing, then this would be interpreted as a new session.

A given IP may be for a computer used in a common area (library, shared office, computer lab, café etc.). In this case it must be assumed that the user conducts a session over a period of time and that subsequent records occur within a certain time of the last record from that IP before having to conclude that there may be a new user. The same conclusion can be drawn if a hidden cookie is used – it only indicates the computer has been used before to connect to the site.

Proxy machines are another problem for the idea of distinct IP's per user. A proxy is a computer that acts as a go-between between a client's browser (machine) and the actual server on which the web page resides. Proxies are becoming more common where "firewall" style security is in place, and in large networks (schools, colleges, business) where caching is often performed by the Proxies.

In past work (Manley and Seltzer, 1997), session time limits have been set to a few tens of seconds as the investigations were looking into server performance and caching. One log analysis package (WebTrends http://www.webtrends.com/) now gives a count of user sessions based on a time set by the user of the analysis package (the results are still defined in terms of the conventional statistics described earlier.) The time set must be determined from the type of site in use; a large search site such as a Yahoo or Alta Vista may consider a time of one or two minutes as a user session. In our journal, 20 minutes appears a reasonable starting point, a time ascertained from manual examination of random sections of the JIME log.

4.3 Browser Caching and Proxies

Most web browsers now in use have the ability to keep copies of pages recently visited locally in a "cache". If a browser is using caching, it will first look to see if the page the user requested is in it's own cache. If it is, it looks at the date the page was stored there, then sends a request to the server including the date of it's cached version. If the copy of the page on the server has not been modified (or updated) since the date of the cached version, the server will tell the client to use it's local copy (the status code 304 appears in the web log.) If a newer version is on the server, the server copy is sent to the client (status code 200 in the web log.) Thus, the occurrence of a status code 304 can indicate that the client browser has been used before to visit this site.

However, there are caveats. Browser caching can be set to check once per session (cached material will only be compared once in the current application session – quit the application and start again to get another check,) checked each time, or turned off. No caching means all records show for every user action with a 200 status code. Once per session means 304 codes may appear, and some actions won't appear at all making it appear the visitor jumps around the site (as they may go back to a page already checked once, so no further check occurs.) The browser is usually set to have a cache size, so less frequently visited pages will eventually be removed from the user's cache. The time to removal depends on how big the cache allocation is, and the amount of use the browser gets. Once removed from the cache, the interaction with the server will look like a new request. This will be a problem for infrequent visitors.

The use of the "back" button, history lists, and the "Reload" button on browsers also has similar effects. The back button and history list usually leaves no record (even if caching is set to "every time" in Netscape.) The use of these can result in what appears to be a jump in the log record for a given visit session. Adding various "buttons" to the web page may only help (if they're used) in the no cache set, or cache set to "every time" states.

The use of a Proxy server by a user can also present problems in that the record may only show the IP or host name of the proxy server. This could result in confusing session records if two or more local users of the proxy server were viewing the web site at the same time. The proxy server will pass along "last modified" requests to see if it has to update it's own version and if it's set to check at each local request. They can also be set to only look after certain periods of time, though this is not usually the case.

The example log trace shown in figures 1, 2 and 3 illustrates how the use of the cache in the browser affects the records put into the log file (commas are used instead of white space as the field separator in this example.) The browser is set to check once per session. In this trace, the visitor works from the home page to visit two articles (noted "murray" and "fischer") and download the PDF file of each article. This is an example of a user interested in hardcopies. In figure 1, the 16 records associated with the home page request are shown.

In figure 2, the visitor moves to the "murray" article. The user has their cache set to "once per session" as a number of the files, e.g. "jimelogosmall.html", do not appear in the set of files requested even though it's required to form the article page. Thus the files are now held in the visitor's browser cache. After receiving the 11 records associated with the "murray" article page, the visitor moves to the PDF page, a cluster of 3 documents.

In figure 3, the visitor now does the same process to get the "fischer" article PDF file. The jump from "murray.pdf" (end of figure 2) to "fischer.html" (start of figure 3) is a result of the user going back to the home page via the history list in the browser. Notice again the GIF files that came with the "murray" set of downloads are not present in the "fischer" set even though they are used in rendering the various "fischer" pages; they are in the cache. After downloading this cluster, the visitor goes on to the PDF page resulting in another cluster of 4 entries (note the htmlicon.gif file appears again – sometimes the browsers don't get it right!)

```
IP,-,-,[21/Apr/1998:21:45:11+0100],GET,/jime/,HTTP/1.0,200,3347
IP,-,-,[21/Apr/1998:21:45:12+0100],GET,/jime/banner.html,HTTP/1.0,200,311
IP,-,-,[21/Apr/1998:21:45:12+0100],GET,/jime/comments.html,HTTP/1.0,200,256
IP,-,-,[21/Apr/1998:21:45:12+0100],GET,/jime/contents.html,HTTP/1.0,200,1093
IP,-,-,[21/Apr/1998:21:45:12+0100],GET,/jime/current.html,HTTP/1.0,200,16391
IP,-,-,[21/Apr/1998:21:45:12+0100],GET,/jime/jimelogosmall.html,HTTP/1.0,200,513
IP,-,-,[21/Apr/1998:21:45:12+0100],GET,/jime/resources/icons/3-frame-icon.gif,HTTP/1.0,200,303
IP,-,-,[21/Apr/1998:21:45:12+0100],GET,/jime/resources/icons/help.gif,HTTP/1.0,200,686
IP,-,-,[21/Apr/1998:21:45:12+0100],GET,/jime/resources/icons/jime-small-no-text.gif,HTTP/1.0,200,2193
IP,-,-,[21/Apr/1998:21:45:12+0100],GET,/jime/resources/icons/kmi-button-small.gif,HTTP/1.0,200,1825
IP,-,-,[21/Apr/1998:21:45:12+0100],GET,/jime/resources/icons/new.gif,HTTP/1.0,200,72
IP,-,-,[21/Apr/1998:21:45:12+0100],GET,/jime/resources/icons/pre-print.gif,HTTP/1.0,200,571
IP,-,-,[21/Apr/1998:21:45:13+0100],GET,/jime/resources/icons/d3e-icon.gif,HTTP/1.0,200,1341
IP,-,-,[21/Apr/1998:21:45:13+0100],GET,/jime/resources/icons/published.gif,HTTP/1.0,200,543
IP,-,-,[21/Apr/1998:21:45:15+0100],GET,/jime/resources/icons/2-frame-icon.gif,HTTP/1.0,200,327
IP,-,-,[21/Apr/1998:21:45:16+0100],GET,/jime/resources/icons/charlesworth.gif,HTTP/1.0,200,10736
```

Figure 1: The 16 records resulting from the Home Page request

```
IP,-,-,[21/Apr/1998:21:45:25+0100],GET,/jime/98/murray/banner.html,HTTP/1.0,200,501
IP,-,-,[21/Apr/1998:21:45:25+0100],GET,/jime/98/murray/bubble.html,HTTP/1.0,200,687
IP,-,-,[21/Apr/1998:21:45:25+0100],GET,/jime/98/murray/contents.html,HTTP/1.0,200,2448
IP,-,-,[21/Apr/1998:21:45:25+0100],GET,/jime/98/murray/murray-01.html,HTTP/1.0,200,1816
IP,-,-,[21/Apr/1998:21:45:25+0100],GET,/jime/98/murray/murray.html,HTTP/1.0,200,1230
IP,-,-,[21/Apr/1998:21:45:25+0100],GET,/jime/98/murray/references.html,HTTP/1.0,200,6330
IP,-,-,[21/Apr/1998:21:45:26+0100],GET,/jime/resources/icons/large-bubble.gif,HTTP/1.0,200,1616
IP,-,-,[21/Apr/1998:21:45:26+0100],GET,/jime/resources/icons/next.gif,HTTP/1.0,200,415
IP,-,-,[21/Apr/1998:21:45:26+0100],GET,/jime/resources/icons/pdficon.gif,HTTP/1.0,200,224
IP,-,-,[21/Apr/1998:21:45:26+0100],GET,/jime/resources/icons/printer.gif,HTTP/1.0,200,2211
IP,-,-,[21/Apr/1998:21:45:27+0100],GET,/jime/resources/icons/feedback.gif,HTTP/1.0,200,1227
…..
…..
…..
IP,-,-,[21/Apr/1998:21:51:03+0100],GET,/jime/98/murray/murray-pdf.html,HTTP/1.0,200,705
IP,-,-,[21/Apr/1998:21:51:18+0100],GET,/jime/resources/icons/htmlicon.gif,HTTP/1.0,200,187
IP,-,-,[21/Apr/1998:21:55:44+0100],GET,/jime/98/murray/murray.pdf,HTTP/1.0,200,56693
```

Figure 2: A request for “murray” follows, then the request for the PDF.

```
IP,-,-,[21/Apr/1998:22:01:08+0100],GET,/jime/98/fischer/fischer.html,HTTP/1.0,200,1205
IP,-,-,[21/Apr/1998:22:01:09+0100],GET,/jime/98/fischer/banner.html,HTTP/1.0,200,521
IP,-,-,[21/Apr/1998:22:01:09+0100],GET,/jime/98/fischer/bubble.html,HTTP/1.0,200,690
IP,-,-,[21/Apr/1998:22:01:09+0100],GET,/jime/98/fischer/contents.html,HTTP/1.0,200,3587
IP,-,-,[21/Apr/1998:22:01:09+0100],GET,/jime/98/fischer/fischer-01.html,HTTP/1.0,200,2341
IP,-,-,[21/Apr/1998:22:01:09+0100],GET,/jime/98/fischer/references.html,HTTP/1.0,200,13965
……
……
……
IP,-,-,[21/Apr/1998:22:01:16+0100],GET,/jime/98/fischer/fischer-pdf.html,HTTP/1.0,200,834
IP,-,-,[21/Apr/1998:22:01:16+0100],GET,/jime/resources/icons/colour.gif,HTTP/1.0,200,1306
IP,-,-,[21/Apr/1998:22:01:16+0100],GET,/jime/resources/icons/htmlicon.gif,HTTP/1.0,200,187
IP,-,-,[21/Apr/1998:22:01:32+0100],GET,/jime/98/fischer/fischer.pdf,HTTP/1.0,200,280414
```

Figure 3: Next, the request for “fischer”, then the associated PDF.

4.4 Status Codes – What Do They Indicate

In the earlier description of the syntax of a record in the log file, the status code was described as the "result code from the server on completing the request." There are a number of code numbers returned which are specified in the HTTP protocol. Some denote errors, others redirects. They will not be discussed here. For the particular need to trace visitors, two codes are of interest: 200 meaning the request was accepted and the file returned, and 304 which means the request was accepted, but the browser already has a copy of the file to use (see the earlier section on caching.)

As noted in the previous section, if a browser uses caching, and a copy of the requested file is in the cache, the browser will send an "if-modified-since date" request to the server. If the server copy of the file is newer, the file is sent back and a status 200 logged. If the server copy is as old as the cached version, a 304 is returned so the browser can use the local copy. One problem this highlights is the fact the use of the "if-modified-since" request results in a 200, web servers do not automatically log the fact the "if-modified-since" request was made. This is unfortunate due to the fact that if the request was made, it is an indication that the user has visited the web site before.

There are a number of conclusions that can be deduced from the log entries and their status codes for a given cluster:

1. A 304 status code within a cluster can indicate if the visitor has been to the site before. Some resources are common to a number of web pages and even if the visitor appears to be new to the current session or request, the appearance of the 304 on a common file indicates that other parts of JIME have been visited in the past.

2. A 304 along with some files not appearing in the list (or 200 code with files missing as in the cache section example earlier) would indicate the browser has caching set to "once per session" and the pages/resources not requested are already cached. Thus a record of what proportion of all the files associated with a record set were requested is useful.

3. At each main transition (home page to article, article to home page etc.,) if all the records are present and the status codes are 304's, the browser is set to check the cache version against the server "every time." Thus if 304's are recorded and all the required files were downloaded at every main transition (the previous two checks above,) we have a cache set to "every time".

4. One browser setting that can affect such judgments is if "image loading" is turned off. If no GIF files are requested during a session where the related HTML pages are all status 200 (or a mix of 304 and 200), it can be concluded that image loading is off. (One odd case is the reload in Netscape – the first reload of a page will bring all HTML and GIF files. A second and any subsequent reloads produce a set of requests for just the HTML files.) So, in order to know if the "image loading" is off, we need to check if all the GIF files are requested for each web page cluster. If they're not requested over the whole session, then image loading is probably off.

4.5 Canonicalization

In the discussion on document clusters above, it was stated that a set of log file entries could be condensed to an entry representing a document cluster. However, as the discussions on user sessions, proxies and status codes illustrate, there is further information that can be extracted from interpretation of the log entries for a document cluster, information that should be retained in any canonicalization process. This process is a form of protocol analysis (Anderson et. al. 1984). Thus, an automated canonicalization process needs to be able to interpret document clusters for a given web site.

The canonicalization is unique to a web site as it is determined by the clustering of documents based on knowledge of the site layout. An automated agent performing the canonicalization process in an autonomous way would need initial cluster information, and input about new clusters not seen before (when new pages area added to the site.) It will need an interface such that the web site manager can impart such knowledge. If a knowledge base exists that describes the main features of the web site, then the canonicalization agent may well be able to derive the information it needs on how to cluster based on that knowledge.

While we could argue that we don't need a canonicalization step if we fully automate log file interpretation, it does give us an intermediate form that is understandable to the human reader and against which fully automated interpretation can be compared. It also provides a human readable view of navigation through the site that can be used in identifying problem areas in the site; e.g. highlighting where navigation options on a page are not being used.

5.0 Constituencies for Users

Discussions in previous sections have covered the interpretation of a web site log file with specific examples from the JIME log. While the discussion has been limited to the JIME log, the approach would apply to any information repository at which a trace of interaction (a log) is available. The next step is to identify higher packets of information associated with the trace of interactions. This is the identification of constituencies to which users can be associated during their visit. The following examples show some constituencies identified from the JIME log. As these come from a manual analysis, they are simple to start off with.

Based on these examples, one can go through the web log for JIME and classify many of the user sessions to one, or more, of these constituencies. Note the "or more" as a visit may appear in one constituency at first, but as the session progresses, the visitor's actions may well place them into a new constituency. Constituencies themselves can overlap, or even be sub-sets of others. While the examples indicate constituencies of actions, it is hoped that the integration of identification with knowledge of the content of the site will produce further constituencies. Once identified, the goal is to develop mechanisms that will allow new visitors to be matched to a constituency profile by their navigation of the site, and site features or content recommended to them as a result, effectively producing a recommender system.

5.1 Androids (Robot Check)

The web has many crawlers scouring the web for documents to index into search databases, e.g. AltaVista, Yahoo. They often have crawlers checking for updated materials. Some of these crawlers can be recognized as automated robots thus coming under this heading, but others may appear more "user" like and be classified in another way e.g., as "Anything New?".

The examples below from JIME are derived based on the interpretation discussed earlier. Some examples of robot action are requests for the robots.txt file held on the server:

```
superewe.searchuk.com,-,-,[21/Apr/1998:12:04:17+0100],
   GET,/robots.txt,HTTP/1.0,200,94
```

These "robots" may also download other files to update their databases. The robots.txt file contains a list of directories the robot should not bother with (helps search engines be guided on a site,) but some people (a very small number) do use them to then check what they can get to. There are also those automated systems that use the "HEAD" HTTP request method. This asks the server to send back just the HEAD section of an HTML file where keywords may be stored for indexing. This also allows the requester to get the date of the file to see if it needs to update it's local version (automated proxy system.)

What is common to these entries (both HEAD and requests to robots.txt) is that they are usually single line entries in the log (at least single line with long time periods before another appearance.)

Others can be recognized from the way requests are made. The following (scooter3.av.pa-x.dec.com) is one of the crawlers for the AltaVista search engine. Here, we can see that it usually requests just one, or perhaps two files at a time with quite long periods of time between these short requests. The requested files will not always be in the same document cluster, and connected files are not requested soon after. These crawlers act in this manner to keep their impact on web sites to a minimum.

```
scooter3.av.pa-x.dec.com,-,-,[20/Apr/1998:20:32:23+0100],
   GET,/jime/03/demo/slides-simple.html,HTTP/1.0,200,8020
scooter3.av.pa-x.dec.com,-,-,[20/Apr/1998:20:32:46+0100],
   GET,/jime/02/java/isvl-1.html,HTTP/1.0,200,4613
scooter3.av.pa-x.dec.com,-,-,[20/Apr/1998:22:01:53+0100],
   GET,/jime/me-sim/me-sim-16.html,HTTP/1.0,200,3151
```

5.2 Anything New?

Visitors who come to the site home page only and download the document cluster associated with the home page and do nothing further. The JIME home page contains the "current.html" file that is a list of the articles available. Thus a visit to just the home page only may indicate someone checking for new articles, or a request from a broker service to see if their clients need notification of an update to the page. The occasional request for the HEAD of the current.html is a way to check the date of the file for it's last update and to collect any metadata that may be in the header of the file.

5.3 What's JIME then? (New visitor)

The first-time visitor to JIME will often look at the pages about JIME itself; the aims page, background, designers, etc. They may go to look at some abstracts just before or after looking at these pages. This reader often looks like an "abstract reader" except for the jumps into these JIME description pages.

5.4 Abstracts Only

These readers will often visit one or two articles usually just requesting the main (abstract) page of the article (reached from the home page.) They may move to the first section of an article, the introduction in most, but will then move back to the home page or to another article, perhaps spending only seconds on some abstracts and longer on others. Having come to JIME out of general interest, they are now searching for more specific interest items.

5.5 On-Line Reader

In figure 4, an example of an on-line reader is given. These visitors start out like an abstract reader, but will then move on to further sections of an article, and may move between non-adjacent sections (using the table of contents rather than the next-previous buttons.) They usually are around for quite a while overall, and may jump back and forth between two or three articles.

In the record shown in figure 4, some of the lines associated with document clusters have been removed to make it easier to see (the start of canonicalization.) Initially, the reader jumps between a few articles only going to one or two sections (like an "abstract reader".) Later on, they visit the East/West special issue editorial, and then start to move between two articles ("repenning" and "fischer") plus the editorial over a period of about an hour. Thus the reader is spending time on sections of the articles comparing perhaps, along with jumps back to the editorial. The total time of the session is about 1.5 hours.

5.6 Demo Fiend

A visitor to the JIME site who seems to have a particular interest in looking at the on-line demonstrations. Visits may come via the home page, or direct to an article with a demonstration, and sometimes directly to a demonstration. Coming directly to the demonstration would probably be the result of using a bookmark, a recommendation. It is interesting to observe visitors fitting this grouping, perhaps the embedding of interactive components into a journal article intrigues them.

```
IP,-,-,[21/Apr/1998:19:30:05+0100],GET,/jime/,HTTP/1.0,200,3347
…. Requested home page…
IP,-,-,[21/Apr/1998:19:32:02+0100],GET,/jime/98/fischer/fischer-t.html,HTTP/1.0,200,1369
…. Moves to article by “Fischer”…
IP,-,-,[21/Apr/1998:19:32:33+0100],GET,/jime/98/fischer/fischer-03.html,HTTP/1.0,200,7406
…. Moves to section 3…
IP,-,-,[21/Apr/1998:19:33:21+0100],GET,/jime/me-sim/me-sim-t.html,HTTP/1.0,200,1179
….Moves to article “me-sim”… (and views sections 3 and 4 – omitted)
IP,-,-,[21/Apr/1998:19:34:56+0100],GET,/jime/02/jime-02-t.html,HTTP/1.0,200,1392
….Move to article 02 “isvl” … (and views sections 2 and 7 – omitted)
IP,-,-,[21/Apr/1998:19:35:59+0100],GET,/jime/www-structure/www-structure.html,HTTP/1.0,200,1026
….Moves to “www-structure” article.. (and views sections 2 and 3 – omitted)
IP,-,-,[21/Apr/1998:19:37:10+0100],GET,/jime/98/repenning/repenning.html,HTTP/1.0,200,1249
….Moves to “repenning” article..
IP,-,-,[21/Apr/1998:19:38:46+0100],GET,/jime/98/repenning/repenning-02.html,HTTP/1.0,200,10399
….section 2..
IP,-,-,[21/Apr/1998:19:40:36+0100],GET,/jime/98/repenning/repenning-03.html,HTTP/1.0,200,7313
….section 3..
IP,-,-,[21/Apr/1998:19:43:57+0100],GET,/,HTTP/1.0,200,3347
….back to the home page..
IP,-,-,[21/Apr/1998:19:46:02+0100],GET,/98/bondaryk/bondaryk.html,HTTP/1.0,200,1237
….to the “bondaryk” article..
IP,-,-,[21/Apr/1998:19:46:44+0100],GET,/conferences/icde97/icde97.html,HTTP/1.0,200,11342
….conference report..
IP,-,-,[21/Apr/1998:19:48:33+0100],GET,/jime/98/repenning/repenning-02.html,HTTP/1.0,200,10399
….back to “repenning”, section 2..
IP,-,-,[21/Apr/1998:19:56:40+0100],GET,/jime/98/ew-editorial/ew-editorial.html,HTTP/1.0,200,1260
….and to the East/West editorial..
IP,-,-,[21/Apr/1998:19:59:01+0100],GET,/Reviews/get/repenning-reviews/26/2.html,HTTP/1.0,200,8550
….Moves to a review comment in “repenning” (so went back to “repennning” again)..
IP,-,-,[21/Apr/1998:19:59:08+0100],GET,/Reviews/get/bondaryk-reviews/3/1.html,HTTP/1.0,200,8907
….And then to the review debate for “bondaryk”…
IP,-,-,[21/Apr/1998:20:00:26+0100],GET,/jime/98/ew-editorial/ew-editorial-10.html,HTTP/1.0,200,3457
…. returning to the editorial
IP,-,-,[21/Apr/1998:20:00:43+0100],GET,/Reviews/get/ritter-reviews/8.html,HTTP/1.0,200,8156
….then a look at “ritter” reviews (suggested via the editorial)..
IP,-,-,[21/Apr/1998:20:01:08+0100],GET,/jime/98/fischer/fischer.html,HTTP/1.0,200,1205
….then returning to “fischer”...
IP,-,-,[21/Apr/1998:20:02:53+0100],GET,/jime/98/fischer/fischer-02.html,HTTP/1.0,200,3714
….section 2..
IP,-,-,[21/Apr/1998:20:11:44+0100],GET,/jime/98/fischer/fischer-11.html,HTTP/1.0,200,17841
….section 11..
IP,-,-,[21/Apr/1998:20:12:03+0100],GET,/jime/98/fischer/fischer-12.html,HTTP/1.0,200,3655
….section 12..
IP,-,-,[21/Apr/1998:20:13:51+0100],GET,/jime/98/repenning/imgs/paper506.gif,HTTP/1.0,304,-
….now a check of “repenning” again (via a single image request)..
IP,-,-,[21/Apr/1998:20:16:14+0100],GET,/Reviews/get/ritter-reviews/8.html,HTTP/1.0,200,8156
….check that “ritter” review again..
IP,-,-,[21/Apr/1998:20:16:35+0100],GET,/98/repenning/repenning-11.html,HTTP/1.0,200,6641
….and back to “repenning”...
IP,-,-,[21/Apr/1998:20:16:47+0100],GET,/jime/98/ew-editorial/ew-editorial-06.html,HTTP/1.0,200,9631
….then back to the editorial…(and so on for another 40 minutes.)
```

Figure 4: The movement of an on-line reader.

5.7 Traditional Paper Reader

This visitor comes to the site, finds an article of interest, and then downloads the PDF file of the article and leaves. The example in section 4 shows how a visitor went to the PDF download area for two papers, “murray” and “fischer”. This is important to observe in JIME as it gives an indication of someone wanting to do more reading. Of course, we now loose the information related to how they read the articles.

6.0 Summary

The present methods of web log analysis produce very limited information about the use of a web site. In many cases, more information about the type of use is desired, yet to extract such information requires much work in log interpretation. Little appears to have been done in this area as efforts have gone into work helping individuals use the Web more effectively. For those designing, organizing and maintaining a web site, there is little to help beyond getting various "hits" style analyses.

In this paper, I have presented a pathway to analyzing a web log that can give more useful information for those maintaining a site. The canonicalization process described is used to observe navigation of the archive. It can also help highlight design flaws, problems with the interface where parts are not used.

Automation of the canonicalization process requires knowledge of the site layout being available to the automatic process. An autonomous system doing canonicalization needs information to determine the clustering needed. The information can come from the administrator or from other site systems, e.g. from a knowledge base about the site. An agent that works on canonicalization with interfaces to qualitative and quantitative outputs can help in administering a site. Part of the automation would be to link it to knowledge about the site to reduce the amount of time needed to work directly with a site administrator.

At a higher level is the identification of constituencies of users from the observation of use of the site. Again, this is tied into knowledge of the layout of the site, and the content of documents. Once in place, the goal is to have a system that can observe a visitor, make a judgement to what constituency they may belong (at any given moment,) and present other items on the site that may be of interest to a user fitting the constituency profile. This can be viewed as a recommender system based on constituency profiles and knowledge of the site layout and content.

In the introduction it was noted that JIME does not maintain individual user profiles. With the increase in broker services, the exchange of user profile information will probably become an important area for users of the Web. At the individual site where profiles are not kept, receiving information with a request would augment the processes outlined in this paper.

References

Anderson, J.R.,; Farrell, R.; and Sauers, R. (1984). Learning to Program in LISP. In *Cognitive Science* 8:87-129

Balabanovic, M.; and Shoham, Y. (1997). Fab: Content-based, Collaborative Recommendation. In *Communications of the ACM*, 40(3), 66-72, ACM Press.

Domingue, J.; and Motta, E. (1999). A Knowledge-Based News Server Supporting Ontology-Driven Story Enrichment and Knowledge Retrieval. Submitted to *11th European Workshop on Knowledge Acquisition, Modelling, and Management* (EKAW ’99).

Ferreira, J.; Borbinha, J.L.; and Delgado, J. (1997). Using LDAP in a Filtering Service for a Digital Library. In *ECRIM Workshop Proceedings No. 98/W001 of the 5th DELOS Workshop on Filtering and Collaborative Filtering.* The European Research Consortium for Inormatics and Mathematics.

Guo, H.; Kreifelts, T.; and Voss, A. (1997). SOaP: Social Filtering through Social Agents. In *ECRIM Workshop Proceedings No. 98/W001 of the 5th DELOS Workshop on Filtering and Collaborative Filtering.* The European Research Consortium for Inormatics and Mathematics.

Kautz, H.; Selman, B.; and Shah, M. (1997). ReferralWeb: Combining Social Networks and Collaborative Filtering. In *Communications of the ACM*, 40(3), 63-65, ACM Press.

Lieberman, H. (1997). Autonomous Interface Agents. In *Proceedings of CHI 1997*, 67-74, ACM Press

Manley, S.; and Seltzer, M. (1997). Web Facts and Fantasy. *In Proceedings of the 1997 USENIX Symposium on Internet Technologies and Systems*, Monterey, CA.

Pirolli, P. (1997). Computational Models of Information Scent-Following in a very large browsable text collection. In *Proceedings of CHI 1997*, 3-10, ACM Press

Pitkow, J.; and Pirolli, P. (1997). Life, Death, and Lawfulness on the Electronic Frontier. In *Proceedings of CHI 1997*. 383-390, ACM Press

Recker, M.; and Pitkow, J. (1996). Predicting Document Access in Large Multimedia Repositories. In *ACM Transactions on CHI*, 3(4):352-375, ACM Press

Resnick, P; and Varian, H.R. (1997). Recommender Systems. In *Communications of the ACM*, 40(3), 56-58, ACM Press.

Rucker, J.; and Polanco, M.J. (1997). SiteSeer: Personalized Navigation for the Web. In *Communications of the ACM*, 40(3), 73-75, ACM Press.

Sarwar, B.M.; Konstan, J.A.; Borchers, A.; Herlocker, J.; Miller, B.; and Riedl, J. (1998). Using Filtering Agents to Improve Prediction Quality in the GroupLens Research Collaborative Filtering System. In *Proceedings of Computer Supported Cooperative Work (CSCW98*), 345-354, ACM Press

Tauscher, L.; and Greenberg, S. (1997). Revisitation Patterns in World Wide Web Navigation. In *Proceedings of CHI 1997*, 399-406, ACM Press.

Terveen, L.; Hill, W.; Amento, B.; McDonald, D.; and Creter, J. (1997). PHOAKS: A System for Sharing Recommendations. In *Communications of the ACM,* 40(3), 59-62, ACM Press.