An Integrated System For Weblog Study

by

Makanaokalani Oshiro

Submitted to the Department of Electrical Engineering and Computer Science

in Partial Fulfillment of the Requirements for the Degrees of

Bachelor of Science in Computer Science and Engineering

and Master of Engineering in Electrical Engineering and Computer Science

at the Massachusetts Institute of Technology

May 20, 2004 [June 2004]

Author ____
Department of Electrical Engineering and Computer Science
May 20, 2004

Certified by ____
James E. Short
Thesis Supervisor

Accepted by ____
Arthur C. Smith
Chairman, Department Committee on Graduate Theses

An Integrated System For Weblog Study
by
Makanaokalani Oshiro

Submitted to the
Department of Electrical Engineering and Computer Science

May 20, 2004

in Partial Fulfillment of the Requirements for the Degrees of
Bachelor of Science in Computer Science and Engineering
and Master of Engineering in Electrical Engineering and Computer Science

ABSTRACT

Weblogs, personal online journals, are a new medium for expression and communication. Studies to determine their effectiveness of information transmission are interesting, but difficult to perform without access to underlying data. We propose an independent weblog server that records all relevant data and a procedure to analyze the content and access patterns. This complete system is useful for weblog research.

Thesis Supervisor: James E. Short
Title: Visiting Scientist, Center for eBusiness at MIT

Acknowledgements

I would first like to thank Nortel Networks, the Center for eBusiness, and MIT Sloan School for helping sponsor my thesis research. Without their assistance, I would not have been able to pursue my research at all, much less complete this thesis. Particular thanks go to Deborah Stokes and Todd Etchinson at Nortel for getting excited about weblog research and putting us in touch with Project Acumen. They opened my eyes to uses for weblogs that I had not expected or considered.

I would also like to thank my thesis advisor James Short. I have worked with Jim for two years, and he has consistently served as my research guide. His knowledge and input have added greatly to the content analysis portions of my thesis.

Finally, I would like to thank my parents and sister, who have supported me through this odyssey that is MIT. When each trial or challenge appeared, they were there to keep me going forward.

Contents

Introduction

The expansion of the Internet has allowed the development of an increasingly online society. In addition to downloading information from conventional sources, the submission of content by ordinary individuals has sharply increased in the past few years. Personal web pages have been available for several years, but they are mostly static and rarely updated. Web forums, where large numbers of users contribute to discussions, have become popular, but the principal method of individual rapid content generation is the weblog.

"Web logs" are best described as personal online journals. In a reversal of the privacy of a conventional journal, the weblog is available to the Internet at large. Users typically post mundane details of their everyday lives, highly opinionated "rants" about current topics, or hypertext links to interesting material. The indirect references and direct links form a network of information connectivity in the format of the World Wide Web, but with a drastically increased formation rate. Weblogs are a goldmine for researching online communication and discourse.

Weblogs

The burgeoning popularity of weblogs can be traced to tools that allow users to post content frequently and easily. Without these tools, users were forced to format and date additional content by hand, whereas these mundane tasks are completely automated now, making them accessible to users who would not have taken the time and effort. Another cause for the expanding weblog user base is the availability of mobile weblogs, or “moblogs,” weblogs primarily updated from a mobile device such as a PDA or camera phone. With a camera phone, a user can snap a picture, add a short caption, and upload it directly to their moblog. Standard blogs can be made to support these features as well, but moblogs are characterized by shorter posts and an emphasis on media.

Weblogs can also be categorized by their hosting. There are major sites (Xanga, TextAmerica) that provide free hosting and for-pay extended services for the weblog user. These can be differentiated by the size of the site, the number of users, and the amount of daily traffic. Site structure is fairly standard among the large sites, with each weblog having a list of "friends" and "blog-rings" (groups) on its front page. Submission can be done through email or the web interface. The site front pages generally have selections of interesting posts, images, and entire weblogs and links to help pages. There are also a number of free software packages (LiveJournal, Movable Type) that a technically savvy user can run on his own web hosting. Specialized submission software, such as KABLOG for cellular phones, works only on individually hosted weblogs.

The weblog form factor can be applied to uses besides personal journals. The format can just as easily cover a project log, accessible to the group members, who can post comments about each status update. A feedback mechanism can be built in as well, so that test user comments are distinguished from developers, but still clearly in reference to a new particular revision. Although robust against drifting off-topic, the weblog comment format does not allow comments to be easily made in response to other comments.

Other researchers are studying weblogs for a number of features. One is Cameron Marlow at the MIT Media Lab, who is researching information propagation between weblogs with his blogdex site, itself a weblog. Many of these groups run a weblog to report on their project progress.

Textual Content Analysis

Our group has previously done analysis of disclosure data. We saved thousands of articles in text format, ran them through a parser, and fed them to General Inquirer, a dictionary-based content analysis tool. With it, we can generate a weight matrix measuring different features that can be analyzed statistically. The copy of General Inquirer was modified to support batch processing of many small files. An aggregation program for combining General Inquirer results for a large number of small files was also written.

The same approach should work on weblog content. The weblog posts are exported to text, and then processed into properly formatted text for the parser. The parser screens the text for keywords that are supplied in its dictionary. The comparative changes in General Inquirer weights between unparsed and parsed sets can be used to measure the effectiveness of the keywords.

Advantages of a Complete Server

Unmodified textual content analysis can be run on text data only, without a context. It would be possible, given permission, to download all the data posted on a popular weblog site and run an analysis. The site structure would be maintained and posting dates are typically displayed below each post, but the data of access patterns would not be downloaded. The context surrounding a user's post, such what they were reading immediately prior or how many edits were used, would be unavailable.

A server under complete control of the research group would have none of the disadvantages of the downloading method, and would have complete knowledge of a number of interesting features. In particular, web server specific data, such as click-through and referrer data can be logged, tracking where users come from and where they go. By creating indirect links to offsite sites, users can be tracked as they go offsite. Correspondence can also be drawn between readers and writers. We can see a user read another user's weblog, go to a referenced site, and then write his own post.

Design Considerations and Assumptions

The first consideration of the proposed server is that it must run the Windows operating system. All of our text analysis is also run on the machine, so it must be user-friendly to anyone who needs to process or retrieve any data. An OS that users are typically familiar with makes it easier to perform basic tasks. However, there is a tradeoff, as open-source software does not tend to immediately run on Windows. The system is being run headless in a server room, with limited physical access, so it must run reliably with zero local access except in exigent circumstances. Since text-only support is lacking, we need solid remote GUI access, TightVNC, to properly control the Windows system.

At the same time that we allowing access, we must make sure that the system is not vulnerable to compromise. All legitimate administrative functions (TightVNC remote control and file transfer) are done through secure ssh tunnels provided by Cygwin. A software firewall, typically ZoneAlarm, limits world-access to the intended web server only.

The web server itself needs scripting support and a backend database to support the weblog. EasyPHP bundles together an Apache web server, a MySQL database, and PHP scripting support in a package pre-configured to work together.

The choice of weblog software package has a number of limiting requirements. The selected system must run reliably on Windows, must be extensible, should support weblog standards, and will hopefully be popular and in common use. A number of the available packages, such as LiveJournal, do not run on Windows. As LiveJournal fits all the other requirements well, this was quite disappointing. The ability to add our enhancements through a plugin system is a key requirement. Compatibility with weblog interface specifications allows greater flexibility in submission options, as the user is free to use third-party software without problems. A popular system will be more likely to be familiar to users that used it previously, and would be likely to be easy to learn for new users. Movable Type satisfies all of the requirements.

The analysis software used includes a basic parser to select "paragraphs" based on a dictionary. The text exported from the weblog content database must be formatted to meet the parser's input specifications, but has no restrictions otherwise.

Prototype System Specifications

Our server is a Dell workstation pressed into service as a server. It runs an Intel Xeon processor 2.40GHz with 1GB of physical RAM, plenty for a server with a limited number of users. The operating system is Microsoft Windows XP Professional Version 2002, SP1. ZoneLabs ZoneAlarm is configured as a loose firewall shield, allowing all of MIT to access available services (web server and ssh port), but disallowing IP addresses outside of MIT.

Support System Software

Cygwin provides basic sshd services that allow a client to connect securely with ssh and then send data through the established secure tunnel. Both TightVNC and file transfer are conducted through these secure channels. The administrator must login using a standard ssh client supporting tunneling, then run the desired program to use. Putty is a free ssh client that also provides a scp (secure copy) program for file transfers. The ssh connections can be compressed, allowing better performance in addition to security. Cygwin can also be used as a limited text-only shell if VNC fails, and can restart the computer remotely with the shutdown command if absolutely necessary.

A TightVNC server program is run on the server on the VNC default port 5900, and the client is run on the administrator's local machine. The server program is run as a Windows service, so that it starts automatically on boot and is always available. The preferred settings are optimized for speed by enabling image compression and color reduction.

Weblog Software

The web server is the EasyPHP 1.7 bundle of Apache, MySQL, PHP, PHPmyadmin, and other utilities. It comes configured with working defaults. The support scripts, including registration, are written in PHP. Movable Type requires Perl to run, so ActiveState Perl for Windows was used. Movable Type is installed with all the required and optional Perl modules for maximum functionality.

On the web server, the registration scripts, the Movable Type installation, and the published blogs are each stored in their own directory within the web-accessible directory. The directories help with site organization and with individualized file permissions. To protect data, a number of permissions were customized, including making include files non-readable and cgi files execute-only, preventing unauthorized access to source code.

Within the Movable Type installation, the first user is named Administrator and given complete access to all weblogs. Each new weblog is initialized with both the new user and the Administrator as owners. The user cannot remove the Administrator as a co-owner, and some features are hidden from the users, such as local file paths and user renaming.

Analysis Software

The basic parser was written from scratch in Java. It requires specific text formatting to break up paragraphs. If a word from the dictionary is present in a given paragraph, the paragraph is kept in the output file; if no words are present, the paragraph is removed. General Inquirer requires no special formatting.

Demonstration of Use

Registration

To register, a user goes to http://disclosure.mit.edu/register.html. Accessing the site requires an MIT IP address, and successful registration requires an MIT Athena username. The user submits their Athena username and a desired weblog username. This flexibility allows non-matching usernames to preserve privacy if desired. The script (source code included in Appendix) checks if the Athena username is well-formed, if either username has been already registered, and finally verifies the Athena username with People Search from the MIT homepage. If all succeeds, a confirmation email is sent to the user's Athena email address and a temporary record is saved. A confirmation email is common web practice for verifying that a given user has proper access to the account in question.

The email contains a short description of its purpose and a confirmation link with a randomly generated key. When the user enters the link, the system verifies that the registration details and key match an existing temporary record. Once confirmed, the new user's information is saved in the permanent registration database, their weblog is generated, and the temporary record is purged. The user is forwarded to a "registration successful" page with a link to their new weblog.

It is also possible for the administrator to add usernames if necessary, using a private script that ignores the verification steps. The username should not match Athena username, but there is no restriction. Privacy must be maintained if the data ever leaves the study group, so the registration database is kept separate from the weblog data structure. If the data is ever provided to an outside party, either only the analyzed conclusion data will be provided, or the registration database will be used to excise personal information from the weblog contents.

Typical Post

The user logs in via the Movable Type login page, which sends him to a listing of his weblog. Movable Type allows for multiple weblogs per user, but the standard structure of our implementation limits this to one. He selects his weblog and then new post from the sidebar. The user enters a title and post content, and then publishes the post. After a quick generation process, the weblog is updated and can be previewed.

The discerning user has a choice of saving the post as a draft for finishing later or publishing immediately, as well as an option to have a summary of the post on the front page with a link to a complete copy. There is also the option of submitting posts from a third-party application, such as KABLOG, or emailing it to a special submission address.

Remote Administration

The administrator ssh's to the server using with Putty with tunneling enabled on port 5900. The client vncviewer program is run, which pops up a window of the server's screen. The administrator is now remote controlling the server from his computer.

Most of the administrative tasks of interest can be accessed from the EasyPHP system tray icon. There are entries for changing configuration settings, restarting the server processes, or opening PHPmyadmin for MySQL or other maintenance. The web server logs can be read in plain text in the apache directory.

Analysis

A weblog's content is analyzed through the following process: the MySQL database entry is exported to text, formatted for the parser, parsed, and then run through General Inquirer, resulting in a output weight matrix. The Apache web server logs are imported into MySQL, where they can be easily arranged by IP address, date, or other access feature.

Conclusions and future work

The server and analysis tools, as constructed, form a complete toolkit for our approach to weblog research. Further research can be easily done using this project as a "turnkey" solution and a stepping-stone.

Difficulties

We experienced pronounced difficulties with generating data from our preliminary user set. The attempted trial period, one week of heavy, forced activity was far too short. A month-long trial would have yielded more data, but our feeling is that it may be too short as well. Our limited set of users was also too small, but the underlying problem is assembling more than a few related users. Either the overall number of users is too low or the expected yield per user is reduced due to the users not knowing each other well. Weblog content seems to be event driven, that is, an interesting event yields interesting weblog post. A potential approach would be to study a student group for the month before a major event, but preparations could have the reverse effect of decreasing postings.

Purely volunteer weblog users do not generate enough content on their own. An incentive structure of some sort may increase useful postings or may attract users more likely to post. A weblog site must provide a compelling reason to be used, such as being large or better than its alternatives.

Future work

Adding security and user attribution to the server would be an interesting improvement that is quite easily implemented. A server certificate can be requested from MIT to certify the server, allowing it to conduct secure https connections. The functionality is already present in the default configuration of EasyPHP, so it is just a matter of enabling it for the proper sections. Secure login and post submission, or even a completely secure weblog site, are all applications. Additionally, secure web-based administrator access, which was previously too insecure, could be enabled as another fallback option.

Requiring users to login for read access would allow tighter tracking of users. While some tracking can be implied using IP addresses, logging in to read as well as to post is a powerful tool. The cookies that Movable Type uses to track submission logins can be reused for this application. A limited "preview" mode could be made available so that potential users can read weblogs, but a disincentive should be added to prevent users from using this exclusively and invalidating the new login.

Click-through tracking for offsite links is another source of tracking information. Whenever a user reads a page on our server, the access is logged, but no information is recorded if they follow a link to another site. However, if the offsite link is replaced with a local page that then forwards the user, this access is recorded. This layer is easily implemented with a database holding the local/offsite link relationships and is in common use on many sites. Click-through can also check the effectiveness of advertising, either in the form of banner ads or inline advertisements.

Movable Type does not natively support the feature of user groups. Adding the groups or "blogrings" structures would allow group formation to be studied. There are implicit groups created if users consistently link to each other, but these structures make the relationship explicit. If users can create their own groups, the formation patterns are clear as new members join.

It may be possible to replace the Movable Type weblog system with LiveJournal, which has a better design than Movable Type for this application, but does not run on

Windows. Movable Type is intended for a small number of weblogs (up to 15 in common usage) with each user assumed to be a system administrator. LiveJournal supports large amounts of users and has appropriate security measures. The principal difficulty with installing LiveJournal is that there are a number of dependencies on Perl modules that are simply not implemented in Windows. It is possible to edit the header files to remove the dependencies and force LiveJournal to install, but operation is unreliable. Compiling the required modules for Windows would be a sizeable task, but would have good returns.

Citations

[1] *ActiveState - ActiveState ActivePerl free Perl open source binary language distribution*. http://www.activestate.com/Products/ActivePerl/?_x=1.

[2] *blogdex - the weblog diffusion index*. http://blogdex.net.

[3] *Camera Phone Moblog Community | Textamerica.com*. http://www.textamerica.com/.

[4] *Cygwin Information and Installation*. http://www.cygwin.com.

[5] *[EasyPHP] - Apache | MySQL | PHP | PhpMyAdmin*. http://www.easyphp.org/index.php3?lang=en.

[6] *ImageMagick - Convert, Edit, and Compose Images*. http://www.imagemagick.org.

[7] *Inquirer Home Page*. (General Inquirer) http://www.wjh.harvard.edu/~inquirer/.

[8] *KABLOG*. http://www.rawthought.com/projects/kablog/.

[9] *LiveJournal.com*. http://www.livejournal.com.

[10] *Movable Type Publishing Platform*. http://www.movabletype.org.

[11] *Seb's Open Research*. http://radio.weblogs.com/0110772/2003/10/16.html#a1172.

[12] *Xanga.com*. http://www.xanga.com/Default.aspx?.

[13] *Zone Labs: Zone Labs, Internet security products, online safety, software, protection*. http://www.zonelabs.com/store/content/home.jsp.

Appendices

Prospective User Email

```
Subject: Registering to use a weblog
From: Kalani Oshiro <koshiro@MIT.EDU>

The registration page is here:

    http://disclosure.mit.edu/register.html

"Blog name" is your desired username on the blog server. For this trial,
it can match your Athena username if you like. Your blog's overall name
can be changed later as desired (it is initialized to your username).

For example, my Athena username is koshiro, my blog username is kalani,
and my blog's name is "Kalani's Blog".

You should receive a confirmation email to your MIT address with a
link.

The blog management page is here:

    http://disclosure.mit.edu/movabletype/mt.cgi

The blogs are stored here:

    http://disclosure.mit.edu/blogs/blogusername

Two things are required to use the system automatically:

  MIT email address (username) - evident by "Invalid Athena Username"
  error while registering; I can add users manually if necessary

  MIT (18.*) IP address - evident by "connection refused"; other static
  addresses can be allowed

Thanks for your help,
Kalani
```

Registration Scripts

register.html

```
<html>

  <head>
    <title>Initial Registration Screen</title>
    </head>

<body>
<h3>PHP User Registration</h3>
<form method="post" action="register.php">
<table>
<tr><td>Athena username:</td><td><input type="text" name="athena" size="8" maxlength="8"
/></td></tr>
<tr><td>Blog name:</td><td><input type="text" name="blog" size="25" maxlength="25"
/></td></tr>
```

```
</table>
<input type="submit" value="Register" />
</form>
</body>
</html>
```

register.php

```
<?php

// register.php
// Registers new users
// User Fields: $athena, $blog
//   $athena = Athena username
//   $blog   = Desired blog username

include 'reg-globals.inc';

// accept POST submission only, no GET
$athena = $_POST['athena'];
$blog = $_POST['blog'];

if (($athena == '') || ($blog == '')) {
  // fields left blank
  $error = 'Please complete both username fields.';
  include $ERROR_PAGE;
  exit;
}

if (strlen($athena) > 8) {
  // Athena username is too long
  $error = 'Athena username more than 8 characters, invalid.';
  include $ERROR_PAGE;
  exit;
}

if (strlen($blog) > 25) {
  // Blog username is too long
  $error = 'Chosen blog username more than 25 characters. Please choose a shorter one.';
  include $ERROR_PAGE;
  exit;
}

$athena = strtolower($athena);
$blog = strtolower($blog);

if (ereg("[^\-_\.0-9a-z]", $athena)) {
  // Athena username contains invalid characters
  $error = 'Athena username contains invalid characters.';
  include $ERROR_PAGE;
  exit;
}

if (ereg("[^\-_\.0-9a-z]", $blog)) {
  // Blog username contains invalid characters
  $error = 'Chosen blog username contains invalid characters. Please choose another
one.';
  include $ERROR_PAGE;
  exit;
}

$encodedAthena = urlencode($athena);
$fingerpage = fopen ("http://web.mit.edu/bin/cgicso?query=username%3D{$encodedAthena}",
"r");
if (!$fingerpage) {
  $error = 'Unable to load Athena finger page.';
  include $ERROR_PAGE;
  exit;
```

```
}
while (!feof ($fingerpage)) {
   $line = fgets ($fingerpage, 1024);
   if (ereg("     email: <A HREF=\"mailto:(.*)\"", $line, $out)) {
     $email = $out[1];
     break;
   }
}
fclose($fingerpage);


if ($email == '') {
  $error = 'Athena username not found.';
  include $ERROR_PAGE;
  exit;
}

$db_conn = mysql_pconnect ("localhost", $DB_USER, $DB_PASS);
mysql_select_db($DB_NAME_REG);

$mq = mysql_query("SELECT athena " .
                  " FROM `queue`" .
                  " WHERE `athena` = '{$athena}'");
if (mysql_num_rows($mq) != 0) {
  $error = 'Athena username present in queue pending confirmation.';
  include $ERROR_PAGE;
  exit;
}

$mq = mysql_query("SELECT id " .
                  " FROM `registered`" .
                  " WHERE `athena` = '{$athena}'");
if (mysql_num_rows($mq) != 0) {
  $error = 'Athena username present in registered users.';
  include $ERROR_PAGE;
  exit;
}

$mq = mysql_query("SELECT athena " .
                  " FROM `queue`" .
                  " WHERE `blog` = '{$blog}'");
if (mysql_num_rows($mq) != 0) {
  $error = 'Blog username present in queue pending confirmation.';
  include $ERROR_PAGE;
  exit;
}

$mq = mysql_query("SELECT id " .
                  " FROM `registered`" .
                  " WHERE `blog` = '{$blog}'");
if (mysql_num_rows($mq) != 0) {
  $error = 'Blog username present in registered users.';
  include $ERROR_PAGE;
  exit;
}

// alphanumeric
function randomstring($len) {
  while($i<$len) {
    $select = rand(0,2);
    switch($select) {
    case 0:
      // numbers
      $str .= chr(rand(48,57));
      break;
    case 1:
      // uppercase letters
    $str .= chr(rand(65,90));
      break;
    case 2:
      // lowercase letters
      $str .= chr(rand(97,122));
    }
    $i++;
  }
  return $str;
}

$key = randomstring(25);

$mq = mysql_query("INSERT INTO `queue` ( `athena` , `blog` , `key` ) " .
                  " VALUES ( '$athena' , '$blog' , '$key' );");
mysql_close();

// send email

$content = "Thank you for registering.\n\n" .
"Please go to the URL below to confirm your registration.\n\n" .
"http://{$_SERVER['SERVER_NAME']}/confirm.php?athena={$athena}&blog={$blog}&key={$key}";

mail($email,
     'Blog Registration Confirmation',
     $content);


// please check your email to confirm
include('register-success.inc');

?>
```

Installation Instructions

Installing programs

Install Cygwin, the only additional package is sshd. Once the package is installed, set sshd to run as a Windows Service. When sshd is first run, it will ask to build a password file from the Windows Users database. Allow this to copy usernames and passwords to its file. Otherwise, sshd will not be aware of any users or their passwords will be initialized to random characters, and login will not be allowed.

Install TightVNC and set to run as service. The server will be enabled on port 5900, the VNC default.

Install EasyPHP. Set the language to English. Set to run as service if the site is meant to be up all the time, but this is optional. If it is not set as a service, it must be manually started after boot. Set correct hostname and contact information in httpd.conf.

Install ActiveState Perl. It is best to install to \Perl, because all the Perl scripts in Movable Type must be edited to point to this path.

Install Movable Type, following the provided installation manual. Make sure to edit each of the Perl scripts to point to the correct Perl path. Install the required Perl packages using ppm in the Perl directory. ImageMagick is an optional but useful module.

Copy script files into standard web directory, generally EasyPHP/www.

Installation complete.

Possible problems

Be sure to verify that you can access the server remotely before removing local access. The easiest way is to ssh out and back in. If ZoneAlarm is misconfigured, the connection will be refused. If the connection is accepted, but the username or password is invalid, it is an sshd configuration problem.

When tunneling VNC with Putty, make a local tunnel from L5900 to 127.0.0.1:5900. The remote address is relative to the **remote host**, so 127.0.0.1 specifies the server itself.

LiveJournal on Windows

LiveJournal server code is readily available (http://www.livejournal.com/code/), as well as documentation for installing and running on Unix-like systems. However, there are missing Perl modules on Windows that need to be either dereferenced or compiled.

ppm automatically downloads and installs Perl modules if they are available in binary format. CPAN has access to both binary and source distributions, but needs to be configured with a compiler.

Unicode::MapUTF8 is the module that prevents LiveJournal from installing, but there are other, optional, modules with problems. The suggested workaround is to edit MapUTF8.pm to remove the reference to Unicode::Map8. In theory, there are two conflicting implementations of this function, so removing one does not cause any problems. LiveJournal was completely installed, but would not load properly after this fix.