

Server-side Tracking of New Documents

Fred Douglass
douglass@research.att.com
AT&T Labs—Research
Florham Park, NJ, USA

September 10, 1999

Abstract

AIDE/WN is a tool for automating the “What’s New?” page found on many websites, to customize the display of new pages for each user. It uses cookies, something that is already common for other purposes, to identify a particular user across visits and to track the versions of each resource that the user sees. It uses the *HtmlDiff* tool to display the specific changes on request, and at the option of the web administrator, permits users to view the version history of a page.

1 Introduction

Pages on the Web change in unpredictable ways. For this reason, many sites include pointers to a “What’s New on this Site” page. (I abbreviate this as **WN** henceforth.) These are maintained by webmasters or others with direct access to the content, and are updated based on their notion of what might be interesting.

In some ways, manually maintained WN pages are desirable. They use a human’s judgment to distinguish between small, uninteresting changes and things that should really grab one’s attention. Furthermore, they permit a person to summarize the changes in a simple fashion, add emphasis when appropriate, age less interesting changes off the WN page faster than the most important changes, and so on.

Indeed, Chen and Koutsofios’s *Website News* system [3], which tracks changes recursively from a site’s home page, has demonstrated that the number of pages that change is much greater than the number highlighted in most site’s WN pages. It stands to reason that if every changed page appeared in every WN list, the list itself would be relatively useless.

At the same time, these generic WN pages are quite limited. If the same page is presented to every user, then after a user sees the WN page, she is likely to see the

same changes presented as “new” again in the near future. The basic problem is that what’s *new* to one user is *old* to another. A second problem is that determination of what’s interesting is left to the webmaster. A user who has seen a particular page on the site and would like to know when that page is updated must rely on the judgment of the webmaster to include an update to that page in the WN list. Finally, a third (and somewhat less important) problem is that the WN list might change before a user ever sees it: a new page shows up in the list and then drops off again. What’s *old* to most of the visitors to the site is *new* to the infrequent visitor.

The *AT&T Internet Difference Engine* (AIDE) has been developed over the past several years as one way to address these problems [4]. It allows users to specify pages of interest, be notified when the pages change, and most importantly, see *how* the pages have changed since the user last viewed them. A page that changes several times between visits by a particular user will be highlighted to show *all* the changes during that interval.

Here I describe a derivative of AIDE, which integrates AIDE’s page tracking and version management with the HTTP server, rather than relying on individual users to explicitly access AIDE to request notifications. In fact, the WN page can look like any other web page (e.g., `http://foo.bar/whatsnew.html`), except that its contents will reflect exactly what is new to the user accessing it. I refer to the derivative as **AIDE/WN**.

2 Customization

Customizing the WN page to each user requires that the browser store a *cookie*, which it passes to the server on each request. The use of cookies is somewhat controversial because of privacy considerations [6]. Therefore, the AIDE/WN system permits the content provider to present the user with an invitation to see customized WN information through the use of cookies, and requires the

Rewrite Exec	/tracked/whatsnew.html	/home/aide/cgi-bin/whatsnew
Rewrite Exec	/tracked/?	/home/aide/cgi-bin/record_visit

Table 1: Directives to rewrite URLs to invoke CGI scripts.

user to confirm that the use of cookies for this purpose is acceptable.

2.1 CGI Redirection

Cookies are useful only if a server-side program, such as a Common Gateway Interface (CGI) script, intercepts and interprets the cookie. Sending a cookie to access a static HTML file would be useful if the HTTP server did something with the cookie directly, but thus far HTTP servers appear to use cookies just to pass data to scripts. In the future I plan to modify the *Apache* server to handle cookies directly, but for now I use David Kristol's *htd* server with path redirection that turns all URLs into invocations of a CGI script. Currently, this is done by making accesses to

`http://aide.research.att.com/tracked/`
just like accesses to

`http://aide.research.att.com/`,

but redirected through a script. That script uses a back-end database to record accesses with a particular cookie. The redirection is accomplished via the directives in Table 1. The *whatsnew* script invokes a database query to list all pages viewed by the user, with the modification timestamp as of each access. Then it uses a brute-force technique to determine what has changed: it invokes the UNIX *stat* call to check each file at the time the WN page is created. (The list could be computed on a coarser granularity and cached, if and when performance becomes a bigger issue.)

The WN list can therefore be updated in real time to show exactly those pages that have changed since the user last saw them. In addition, other pages can refer to the WN list if and only if new pages exist. When one views the root of the *tracked* hierarchy, a comment `<!-- NEW_PAGES -->` is converted into an anchor for the WN list whenever new pages exist, with the text "There are *N* new pages." When no new pages exist, the comment has no effect. Figure 1 gives an example of the AIDE/WN "What's New" page.

The *record_visit* script updates the database each time a tracked page is accessed. This has the disadvantage of turning accesses to static files into CGI invocations, but (as discussed elsewhere) one could modify the

HTTP server to handle this data directly. If the overhead of individual database updates is too high, the server could log accesses and update the database periodically to amortize the costs.

3 Historical Data

Having a list of which pages have changed is useful, but experience with AIDE suggests that in many cases it would be hard to determine *a priori* what the changes are if the only information is a single bit ("look over there!"). AIDE/WN uses the Revision Control System (RCS) to access past versions of each page and to make sure that the version a user is viewing is available in the future. It does the latter by checking in a new version if the current version is more recent than the most recent checked-in version. On the other hand, if the most recent version has already been checked in, AIDE/WN uses the comment provided at the last check-in time as a hint to the user about how the page may have changed.

Given the RCS history, AIDE/WN can present the user with a view of how a particular page has changed since it was last seen, or in fact how it has changed at any two points in time. AIDE/WN does this by checking each page for a comment

```
<!-- SHOW_DIFFS -->
```

and replacing this text with special HTML markup that inserts an anchor to permit one to invoke *Htmldiff* to see how the page has changed. An example appears in Figure 2(a), and the source to this example appears in Figure 2(b).

The revision history is modified slightly if the current version is not new. In that case, AIDE/WN displays a message about seeing how the page has changed since the previous version, but otherwise functions the same.

4 Issues

The current prototype is functional, but has some open issues.

Internet Difference Engine

What's New

The following pages on this site have changed since you last visited them, or have been modified or created recently and never seen by you. Note that you may need to explicitly reload pages if they are already in your browser's cache.

Some of these pages may permit you to see how they have changed by clicking on a ball at the bottom of the page.

Page	Last Seen	Last Modified	Comment
/setvet.html	Tue May 26 17:20:29 1998	Tue May 26 17:21:15 1998	Added information about suppressing versions

Figure 1: Customized list of what's new to a particular user, with RCS log information about the most recent change.

Contents of page....

- *You last saw this page at:* Fri, 01 May 1998 13:03:31 GMT.
To see how this page changed since then, click on the ball.

(a) What the user sees.

```
<table> <tr> <td rowspan=2 align=center>
<FORM METHOD="POST" ACTION="/cgi-bin/rcsdiff" ENCTYPE=application/x-www-form-urlencoded>
<INPUT TYPE="hidden" NAME="diff" VALUE="diff">
<INPUT TYPE="hidden" NAME="file" VALUE="/examples/index.html">
<INPUT TYPE="hidden" NAME="current" VALUE="1">
<INPUT TYPE="hidden" NAME="1.23" VALUE="1">
<INPUT TYPE=image NAME="Submit" alt="submit!" VALUE="Submit"
src="http://www.research.att.com/icons/RedBall.gif" border=0>
</FORM> </td> <td>
<i> You last saw this page at:</i> Fri, 01 May 1998 13:03:31 GMT.
</td> </tr> <tr> <td> <i>To see how this page changed since then, click on the ball.
</td> </tr> </table>
```

(b) HTML source to the example, somewhat compressed for space.

Figure 2: An example of the information presented to the user when differences are offered.

4.1 Performance

Computation of the differences is time-consuming using the current implementation of *Htmldiff*.¹ The system caches *Htmldiff* output so that subsequent accesses are faster, but the first access can take many seconds or even minutes. Automatic computation of all possible differences (the current version compared against each old version that any user has seen most recently) is possible but not scalable.

The database operations to update the information about which user has seen which version, and to determine which pages have changed, are reasonably efficient but still not extremely fast. This means that dynamic computation of the number of new pages on each access to the root page might be undesirable, and instead that information could be calculated less frequently and cached. There are some systems (e.g., [5]) that focus on caching dynamic data, invalidating it only when appropriate, which could apply here.

4.2 User Interface

The user interface is perhaps AIDE/WN's weakest component. Right now, the WN list consists of an HTML table with one line per changed file. Each line reports when the user last saw the page, when it was modified, and any available RCS comment. There is no notion of prioritization, where the most important recently changed pages could be listed first. In contrast, the regular AIDE system allows users to specify an arbitrary integer priority for each page, which is then used to sort the WN list, and a manually generated WN list would have arbitrary order and text as decided by a webmaster.

4.3 Caching Effects

By default, in HTTP/1.0 [2], dynamic data is generated with no `Last-Modified` header and is not cached. Since AIDE/WN is taking static files such as `index.html` and wrapping a CGI script around them to be able to track cookies, it could have the undesirable effect of making all pages on a site uncacheable. Instead, the CGI script explicitly generates a `Last-Modified` header based on the actual modification timestamp in the file system, just as the HTTP server normally does.

However, there is a catch. When a page actually changes, and appears in the WN list, visiting that page should ideally show the current version of the page, offer to display changes from the previous version, and

remove the page from the WN list. In practice, since the page is considered cacheable, a browser that revalidates pages only when it first starts will display the older, cached version when one follows the anchor referring to the changed page. The user must then explicitly reload the page to see the latest version. It does not appear that HTTP/1.1 [1] is any less susceptible to these cache consistency issues.

4.4 Suppressing Versions

One catch to the automatic generation of WN lists is that they may catch exceedingly trivial changes, such as typographical errors. The regular (user-driven) version of AIDE suffers from the same problem and AIDE/WN could benefit from any techniques to ignore uninteresting changes.

5 Conclusions

AIDE/WN as it currently stands is a proof-of-concept. It demonstrates that the "What's New on this Site" page, which so many sites support, can be customized on a per-user basis in exchange for added computation and an imposition on users' privacy.

References

- [1] T. Berners-Lee, R. Fielding, H. Frystyk, J. Gettys, P. Leach, L. Masinter, and J. Mogul. RFC 2616: Hypertext transfer protocol — HTTP/1.1, June 1999.
- [2] T. Berners-Lee, R. Fielding, and H. Nielsen. RFC 1945: Hypertext transfer protocol — HTTP/1.0, May 1996.
- [3] Yih-Farn Chen and Eleftherios Koutsofios. Website news: A website tracking and visualization service,. In *Poster Proceedings of WWW8*, Toronto, Ontario, Canada, May 1999.
- [4] Fred Douglass, Thomas Ball, Yih-Farn Chen, and Eleftherios Koutsofios. The AT&T Internet Difference Engine: Tracking and viewing changes on the web. *World Wide Web*, pages 27–44, January 1998.
- [5] Arun Iyengar and Jim Challenger. Improving web server performance by caching dynamic data. In *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, pages 49–60, December 1997.
- [6] D. Kristol and L. Montulli. RFC 2109: Http state management mechanism, February 1997.

¹A new version of *Htmldiff*, using a much more efficient comparison algorithm, is under development.