

Revised Server Log XML Format (PDF Tutorial)

XML-Tips Blog Post: Revised WSML Web Server Markup Language

In the [last post](#), I discussed what I'm calling WSML (Web Server Markup Language), which is in XML format. The WSML format is used for temporary XML files that transfer web server log data from the [Extended Log Format](#) into a PHP application (which I have yet to write or post to my PHP blog on).

The current WSML format is efficient. As I mentioned in the previous XML Tips post, a WSML file representing the same information as the source web server access log takes up almost twice as much file space. There are data redundancies we could eliminate. In database lingo, we need to "normalize" the WSML-formatted log data. Note that because the WSML format is used for temporary files, normalizing the XML document structure may not be of any benefit. However, I'll do so anyway, as the techniques in this post can be used to normalize any XML file that carries consistent/ predictable redundancies. The general principle is to "collapse" the XML document-tree nodes to accumulate common child nodes under a particular category. This will become clear with an example. The remainder of this post is in a [PDF file](#). This is an experiment to see whether a combination of Blog posts and PDF tutorials is an efficient way of discussing technical concepts. If you have any comments about this method, please drop me a line at rdash001-at-yahoo-dot-ca (email mangled to fool spambots).

PDF Tutorial – Normalizing XML Formats By Collapsing Redundant Nodes

What we will do in this tutorial is:

- (1) Determine which XML nodes or attributes are consistently redundant.
- (2) Collapse redundancies by creating a new sub-tree below each unique value of a redundant node.

For reference, here is an example WSML file:

```
<?xml version="1.0" ?>
<serverlog_partial domain="chameleonintegration.com">
<logentries>
<serverlogentry id="1" clientip="66.196.91.165" date="30/Aug/2005" time="03:38:31" tzzone="-0400">
  <method>GET</method><protocol>HTTP/1.0</protocol><status>304</status><bytes>-</bytes>
  <requiri></requiri>
  <referer>-</referer>
  <useragent>Mozilla/5.0 (compatible; Yahoo! Slurp;
http://help.yahoo.com/help/us/ysearch/slurp)</useragent>
</serverlogentry>
<serverlogentry id="2" clientip="207.46.98.33" date="30/Aug/2005" time="07:13:45" tzzone="-0400">
  <method>GET</method><protocol>HTTP/1.0</protocol><status>200</status><bytes>980</bytes>
  <requiri>/robots.txt</requiri>
  <referer>-</referer>
  <useragent>msnbot/1.0 (+http://search.msn.com/msnbot.htm)</useragent>
</serverlogentry>
<serverlogentry id="3" clientip="207.46.98.33" date="30/Aug/2005" time="07:13:45" tzzone="-0400">
  <method>GET</method><protocol>HTTP/1.0</protocol><status>200</status><bytes>0</bytes>
  <requiri>/itsmybizniz.html</requiri>
  <referer>-</referer>
  <useragent>msnbot/1.0 (+http://search.msn.com/msnbot.htm)</useragent>
</serverlogentry>
<serverlogentry id="3" clientip="207.46.98.33" date="30/Aug/2005" time="07:18:12" tzzone="-0400">
  <method>GET</method><protocol>HTTP/1.0</protocol><status>200</status><bytes>0</bytes>
  <requiri>/articles.html</requiri>
  <referer>-</referer>
  <useragent>msnbot/1.0 (+http://search.msn.com/msnbot.htm)</useragent>
</serverlogentry>
<serverlogentry id="4" clientip="207.46.98.33" date="30/Aug/2005" time="07:39:37" tzzone="-0400">
  <method>GET</method><protocol>HTTP/1.0</protocol><status>200</status><bytes>19939</bytes>
  <requiri>/index.html</requiri>
  <referer>-</referer>
  <useragent>msnbot/1.0 (+http://search.msn.com/msnbot.htm)</useragent>
</serverlogentry>
```

```

<serverlogentry id="5" clientip="83.42.70.129" date="30/Aug/2005" time="08:36:01" tzzone="-0400">
  <method>GET</method><protocol>HTTP/1.1</protocol><status>200</status><bytes>3282</bytes>
  <requiri>/blog/closeup-verysm.jpg</requiri>
  <referer>http://curryelviscooks.blogspot.com/2005/08/modular-sandwich-artichoke-
cucumber.html</referer>
  <useragent>Mozilla/5.0 (Macintosh; U; PPC Mac OS X; en) AppleWebKit/312.1 (KHTML, like Gecko)
Safari/312</useragent>
</serverlogentry>
</logentries>
</serverlog_partial>

```

If you've read the previous posts (linked earlier), then the values of the XML elements in the example above should be fairly obvious. If they are not obvious, then please re-read the other posts.

As you can see, starting from the second <serverlogentry> element, there are a collection of sequential requests coming from the same IP address, 207.46.98.33, which is an MSN bot (search engine robot crawler). The first thing we can do is collapse requests from this IP address into a mini-document hierarchy. Then we are not duplicating the IP address, nor the XML tagname. Notice, below, that the <serverlogentry> elements are now called <logentry> and are the children of a new element called <visitor>. Notice also that even for visitors than only request one web resource (page/ script/ image/ document) from the web server, we still have to use the <visitor> element as the parent. That means that if we have transient visitors to our website that only request one resource, collapsing the older WSML format to the one below may not be worthwhile. Unless we think that the visitor will return at a later date.

Here's what we have after the first step:

```

<?xml version="1.0" ?>
<serverlog_partial domain="chameleonintegration.com">
<logentries>

<visitor ip="66.196.91.165">
  <logentry id="1" date="30/Aug/2005" time="03:38:31" tzzone="-0400">
    <method>GET</method><protocol>HTTP/1.0</protocol><status>304</status><bytes>-</bytes>
    <requiri></requiri>
    <referer>-</referer>
    <useragent>Mozilla/5.0 (compatible; Yahoo! Slurp;
http://help.yahoo.com/help/us/ysearch/slurp)</useragent>
  </logentry>
</visitor>

<visitor ip="207.46.98.33">
  <logentry id="2" date="30/Aug/2005" time="07:13:45" tzzone="-0400">
    <method>GET</method><protocol>HTTP/1.0</protocol><status>200</status><bytes>980</bytes>
    <requiri>/robots.txt</requiri>
    <referer>-</referer>
    <useragent>msnbot/1.0 (+http://search.msn.com/msnbot.htm)</useragent>
  </logentry>
  <logentry id="3" date="30/Aug/2005" time="07:13:45" tzzone="-0400">
    <method>GET</method><protocol>HTTP/1.0</protocol><status>200</status><bytes>0</bytes>
    <requiri>/itsmybizniz.html</requiri>
    <referer>-</referer>
    <useragent>msnbot/1.0 (+http://search.msn.com/msnbot.htm)</useragent>
  </logentry>
  <logentry id="3" date="30/Aug/2005" time="07:18:12" tzzone="-0400">
    <method>GET</method><protocol>HTTP/1.0</protocol><status>200</status><bytes>0</bytes>
    <requiri>/articles.html</requiri>
    <referer>-</referer>
    <useragent>msnbot/1.0 (+http://search.msn.com/msnbot.htm)</useragent>
  </logentry>
  <logentry id="4" date="30/Aug/2005" time="07:39:37" tzzone="-0400">
    <method>GET</method><protocol>HTTP/1.0</protocol><status>200</status><bytes>19939</bytes>
    <requiri>/index.html</requiri>
    <referer>-</referer>
    <useragent>msnbot/1.0 (+http://search.msn.com/msnbot.htm)</useragent>
  </logentry>
  <logentry id="5" date="30/Aug/2005" time="08:36:01" tzzone="-0400">
    <method>GET</method><protocol>HTTP/1.1</protocol><status>200</status><bytes>3282</bytes>
    <requiri>/blog/closeup-verysm.jpg</requiri>
    <referer>http://curryelviscooks.blogspot.com/2005/08/modular-sandwich-artichoke-
cucumber.html</referer>
    <useragent>Mozilla/5.0 (Macintosh; U; PPC Mac OS X; en) AppleWebKit/312.1 (KHTML, like Gecko)

```

```

Safari/312</useragent>
  </logentry>
</visitor>

</logentries>
</serverlog_partial>

```

Have we really saved any significant space using this revised format? Not much, but in a large access log with a large number of web resource requests per visitor per day, it adds up. We can, however, collapse the above format even further by rearranging the time zone and date (but not time) information. The time zone goes into the <serverlog_partial> element as an attribute. We can do this because it represents the time zone of the actual computer running the web server in question. This information is unlikely to change. The date becomes an element on its own. Notice that because we want the IP address to be the predominant value, the date value gets repeated for each visitor. This is wasteful for a single day, but our future intent is to add subsequent days. We now have the following XML format:

```

<?xml version="1.0" ?>
<serverlog_partial domain="chameleonintegration.com" tzzone="-0400">
<logentries>

<visitor ip="66.196.91.165">
  <date value="30/Aug/2005">
    <logentry id="1" time="03:38:31">
      <method>GET</method><protocol>HTTP/1.0</protocol><status>304</status><bytes>-</bytes>
      <requiri></requiri>
      <referer>-</referer>
      <useragent>Mozilla/5.0 (compatible; Yahoo! Slurp;
http://help.yahoo.com/help/us/ysearch/slurp)</useragent>
    </logentry>
  </date>
</visitor>

<visitor ip="207.46.98.33">
  <date value="30/Aug/2005">
    <logentry id="2" time="07:13:45">
      <method>GET</method><protocol>HTTP/1.0</protocol><status>200</status><bytes>980</bytes>
      <requiri>/robots.txt</requiri>
      <referer>-</referer>
      <useragent>msnbot/1.0 (+http://search.msn.com/msnbot.htm)</useragent>
    </logentry>
    <logentry id="3" time="07:13:45">
      <method>GET</method><protocol>HTTP/1.0</protocol><status>200</status><bytes>0</bytes>
      <requiri>/itsmybizniz.html</requiri>
      <referer>-</referer>
      <useragent>msnbot/1.0 (+http://search.msn.com/msnbot.htm)</useragent>
    </logentry>
    <logentry id="3" time="07:18:12">
      <method>GET</method><protocol>HTTP/1.0</protocol><status>200</status><bytes>0</bytes>
      <requiri>/articles.html</requiri>
      <referer>-</referer>
      <useragent>msnbot/1.0 (+http://search.msn.com/msnbot.htm)</useragent>
    </logentry>
    <logentry id="4" time="07:39:37">
      <method>GET</method><protocol>HTTP/1.0</protocol><status>200</status><bytes>19939</bytes>
      <requiri>/index.html</requiri>
      <referer>-</referer>
      <useragent>msnbot/1.0 (+http://search.msn.com/msnbot.htm)</useragent>
    </logentry>
    <logentry id="5" time="08:36:01">
      <method>GET</method><protocol>HTTP/1.1</protocol><status>200</status><bytes>3282</bytes>
      <requiri>/blog/closeup-verysm.jpg</requiri>
      <referer>http://curryelviscooks.blogspot.com/2005/08/modular-sandwich-artichoke-
cucumber.html</referer>
      <useragent>Mozilla/5.0 (Macintosh; U; PPC Mac OS X; en) AppleWebKit/312.1 (KHTML, like
Gecko) Safari/312</useragent>
    </logentry>
  </date>
</visitor>

</logentries>
</serverlog_partial>

```

Now, we have a format in which, if we add web server access data for additional days, we will gain file space savings over the old WSML format. Since most web servers are configured for their log files to record one week or one month at a time, our exercise is not a waste of time.

Summary

Because my intent is to produce an XML format that I can visually glance at and see repeat visitors, I have collapsed the access log information first by IP address and then by date. If were interested in browser usage by country, I might ignore the placement of date information and collapse by <useragent> values. In fact, depending on our ultimate use of the access log data, there are many ways we can compress on the redundancy of ip address, date, requested page, success status, referrer, or user agent. If your intent is to be able to group access log data on any of these fields, as necessary, then we need to put the data into a database. Having the information in a database will allow us to conduct complex analysis. This is my ultimate goal with this miniseries. Future posts to my [Perl-Tips](#) blog will discuss the Perl code to parse the access log. After that, my PHP log (still in development) will discuss a web-based application that (1) transfers access data to a mySQL database; and (2) provides an interface for querying and viewing access data in a web browser. My [SQL-Tips](#) blog will contain posts relating to the mySQL database aspects of these applications.

(c) Copyright 2005-present, Raj Kumar Dash, <http://xml-tips.blogspot.com>