Wayback, WAT, and Flying Pigs

Brad Tofel
Internet Archive
WAGW Workshop Jun 17, 2011

Agenda

- Wayback Design, Options, and Installations
- Data formats(ARC,WARC,CDX,WAT)
- Data Mining with Hadoop and Pig
- Count 'em, 55 slides!!

Wayback Overview

- Open source project (sourceforge)
- Java + Spring IOC
- Webapp, hadoop jar, command line scripts
- Current version 1.6

Wayback Architecture

- Resource Store
- Resource Index
- Replay UI
- Query UI
- AccessPoints

ResourceStore

- A bunch of documents, html, css, gif, etc.
- Local W/ARC
- HTTP 1.1 exposed remote W/ARC
- Can support other formats
- Can support other storage layers (ex HDFS)

Resource Index

- A list of all the Resources in a ResourceStore, optimized for URL-Date lookup
- Local BDB/CDX
- Remote “ZipNum”
- Hbase planned

Wayback: Replay UI

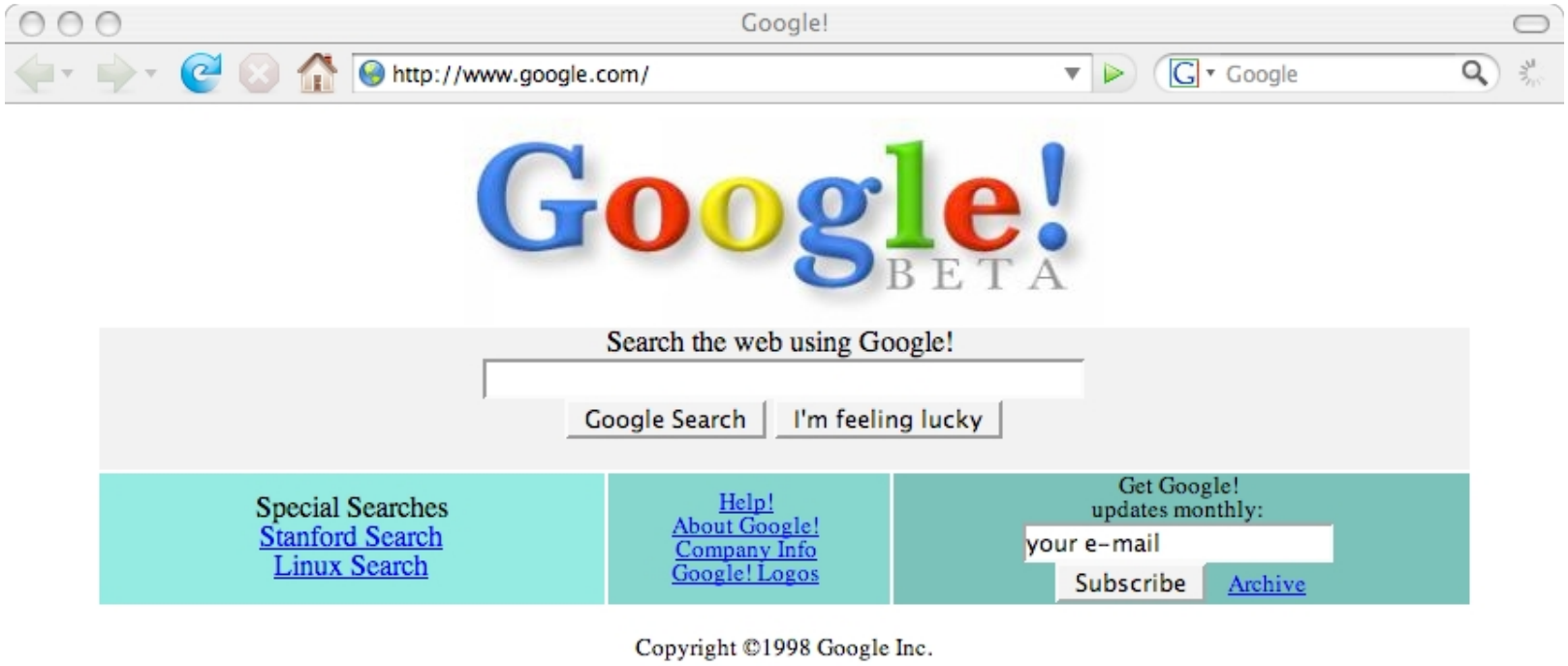
- Defines a method of re-contextualizing resources so they work in a modern browser
- ArchivalURL
  - http://HOST:PORT/COLLECTION/DATETIME/URL
- Proxy
- Domain Prefix (experimental)

Replay Mode: Archival URL

- http://WBHOST:PORT/COLLECTION/DATE/URL
- URL rewrite via client Javascript, or entirely on server
- Allows embedding of .jsp generated content in documents

Replay Mode: Proxy

- Users configure browser to proxy requests through Wayback
- no URL-rewriting needed: Javascript, Flash, PDF links “just work”



Replay Mode: Domain Prefix

- Requires wildcard DNS configuration
- Embeds host and date information in hostname
- Path information remains the same
- Highly experimental
- Simplifies server-side URL rewriting (HTML only)
- Example:
  - http://yahoo.com/b.gif @ 20070101
  - http://yahoo.com.20070101.wb.archive.org/b.gif

.jsp Extension Points: Replay Insertion

```
<property name="replay">
  <bean class="org.archive.wayback.archivalurl.ArchivalUrlReplayDispatcher">
    <property name="jspInserts">
      <list>
        <value>/replay/ArchiveComment.jsp</value>
      </list>
    </property>
  </bean>
</property>

(Contents of ArchiveComment.jsp)
<%
  UIQueryResults results = (UIQueryResults) UIResults.getFromRequest(request);
  StringFormatter fmt = results.getFormatter();
  Date exactDate = results.getExactRequestedTimestamp().getDate();
  Date now = new Date();
  String prettyDateFormat = "{0,date,H:mm:ss MMM d, yyyy}";
  String prettyArchiveString = fmt.format(prettyDateFormat,exactDate);
  String prettyRequestString = fmt.format(prettyDateFormat,now);
%>
<!--
     FILE ARCHIVED ON <%= prettyArchiveString %> AND RETRIEVED FROM THE
     INTERNET ARCHIVE ON <%= prettyRequestString %>.
     JAVASCRIPT APPENDED BY WAYBACK MACHINE, COPYRIGHT INTERNET ARCHIVE.

     ALL OTHER CONTENT MAY ALSO BE PROTECTED BY COPYRIGHT (17 U.S.C.
     SECTION 108(a)(3)).
-->

(HTML output of replayed Pages)
<!--
     FILE ARCHIVED ON 4:40:00 Apr 7, 2000 AND RETRIEVED FROM THE
     INTERNET ARCHIVE ON 14:18:05 Oct 23, 2007.
     JAVASCRIPT APPENDED BY WAYBACK MACHINE, COPYRIGHT INTERNET ARCHIVE.

     ALL OTHER CONTENT MAY ALSO BE PROTECTED BY COPYRIGHT (17 U.S.C.
     SECTION 108(a)(3)).
-->
```

Wayback: Replay Toolbars 1

Wayback: Replay Toolbars 2

Wayback Query

- Wayback software parses request, and extracts matches from the Resource Index
- The Query UI is responsible for rendering the matching results for users
- Current implementation just “calls out” to .jsp to draw the results.

Query: Classic

Query: Bubbles

Query: RSS

Query UI: .jsp example: XML

```
<%
UIQueryResults uiResults = (UIQueryResults) UIResults.getFromRequest(request);
SearchResults results = uiResults.getResults();
Iterator itr = uiResults.resultsIterator();
%>
<wayback>
  <request>
<%
  Properties p = results.getFilters();
  for (Enumeration e = p.keys(); e.hasMoreElements();) {
    String key = UIQueryResults.encodeXMLEntity((String) e.nextElement());
    String value = UIQueryResults.encodeXMLContent((String) p.get(key));
%>
    <<%= key %>><%= value %></<%= key %>>
<%
  }
  String type = uiResults.isUrlResults() ? WaybackConstants.RESULTS_TYPE_URL :
    WaybackConstants.RESULTS_TYPE_CAPTURE;
%>
    <<%= WaybackConstants.RESULTS_TYPE %>><%= type %></<%= WaybackConstants.RESULTS_TYPE %>>
  </request>
  <results>
<%
  while(itr.hasNext()) {
%>
    <result>
<%
    SearchResult result = (SearchResult) itr.next();
    Properties p2 = result.getData();
    for (Enumeration e = p2.keys(); e.hasMoreElements();) {
      String key = UIQueryResults.encodeXMLEntity((String) e.nextElement());
%>
      <<%= key %>><%= UIQueryResults.encodeXMLContent((String) p2.get(key)) %></<%= key %>>
<%
    }
%>
    </result>
<%
  }
%>
  </results>
</wayback>
```

Query UI: XML output

```
<wayback>
        <request>
                <url>adhesion.udf.org/</url>
                <firstreturned>0</firstreturned>
                <enddate>20071023210351</enddate>
                <exactdate>20071023210351</exactdate>
                <resultsrequested>1000</resultsrequested>
                <numresults>3</numresults>
                <type>urlquery</type>
                <startdate>20050101000000</startdate>
                <numreturned>3</numreturned>
                <resultstype>resultstypecapture</resultstype>
        </request>
        <results>
                <result>
                        <url>adhesion.udf.org/</url>
                        <originalurl>http://adhesion.udf.org</originalurl>
                        <arcfile>foo-20070131135818-00001-crawling09.us.archive.org.arc.gz</arcfile>
                        <httpresponsecode>302</httpresponsecode>
                        <md5digest>dc31dafe8068a7689978d03fcaf24057</md5digest>
                        <capturedate>20070131135829</capturedate>
                        <compressedoffset>944959</compressedoffset>
                        <urlkey>adhesion.udf.org/</urlkey>
                        <mimetype>text/html</mimetype>
                        <redirecturl>http://adhesion.udf.org/main/_adherents/</redirecturl>
                </result>
                <result>
                        <url>adhesion.udf.org/</url>
                        <originalhost>adhesion.udf.org</originalhost>
                        ...
```

Wayback Query

- It's all in the wrist... er, .jsp

Wayback: Access Points

- Glues together a Store, Index, Query and Replay at a specific URL
- Allows configuration of Authentication, Access Control, and much more

Configuration Options

- Uses Spring IOC
- Spring enables simple sharing of common object instances between AccessPoints
-

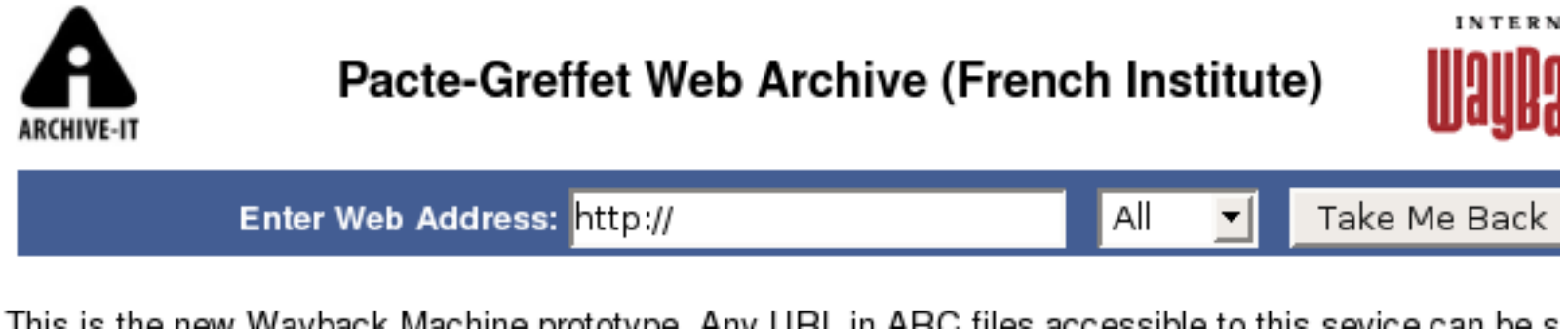
Access Point Spring Config

Generic text configuration allows AccessPoint-specific customizations with common .jsp files.

```
(AccessPoint Spring configuration)
<property name="configs">
  <props>
    <prop key="inst">French Institute</prop>
    <prop key="coll">Pacte-Greffet</prop>
    <prop key="logo">archive-it.gif</prop>
  </props>
</property>
```

```
(common .jsp code)
<td width="70%" align="center" style="font-size:18px;">
  <strong>
    <%= results.getContextConfig("coll") + " Web Archive (" + results.getContextConfig("inst") + ")" %>
  </strong>
</td>
```

(Resulting AccessPoint page)



Production Installations

- LOC installation
  - >15 collections
  - > 200 TB
- ArchiveIt installation
  - partner institutions: about 180
  - collections: 1578 public collections
  - Total URLs: 2,874,990,426
  - Seed URLs: 63,694
  - TBs: 150 TB of data
  - Runs on a 2Ghz AMD dual core, almost always idle
  - uses ~1.7GB RAM

Production Installations

- web.archive.org
  - Runs on ~12 Vms
  - > 6 PB
  - > 150B pages (yes, that's billion!)
  - > 1500 requests/sec max
  - 5 distinct tiers
  - All on open source software stack (almost.. see ZipNum)

Production Installations

- Dozens around the world, at national libraries, and memory instituations near you.

</Wayback>

Any questions on Wayback?

Web Archive Formats

- ARC
- WARC
- CDX

ARC Format

URL IP Datespec Mime Length<nl>
CONTENT(Length bytes)<nl>

URL IP Datespec Mime Length<nl>
CONTENT(Length bytes)<nl>
URL IP Datespec Mime Length<nl>
CONTENT(Length bytes)<nl>
URL IP Datespec Mime Length<nl>
CONTENT(Length bytes)<nl>
URL IP Datespec Mime Length<nl>
CONTENT(Length bytes)<nl>
URL IP Datespec Mime Length<nl>
CONTENT(Length bytes)<nl>

…(for 100 Mbytes)

WARC Format

- Just like ARC, except instead of a single, fixed metadata line, there are HTTP headers
- Records are typed:
  - Request
  - Response
  - Resource
  - Metadata
  - ...

CDX Format

- Sorted Flat File
- Contiguously Partitioned(in global WM)
- Fields
  - URL(canonicalized)
  - Datespec
  - Original URL
  - Mime
  - HTTP Response Code
  - Digest
  - Redirect URL (or '-')
  - ARC offset
  - ARC filename

```
danish.com/ 20031026205948 www.danish.com text/html 200 d6 - 84818496 DT_slash_crawl8.20031026204456
danish.com/ 20031120012854 danish.com text/html 200 d6 - 75287053 DU_slash_crawl8.20031120011713
danish.com/ 20031218050822 www.danish.com text/html 200 8e - 93588444 DU_slash_crawl8.20031218044741
danish.com/ 20031226040022 www.danish.com text/html 200 8e - 1827107 DU_slash_crawl8.20031226040004
```

WAT


```
{
 "Container": {
  "Filename": "TENN-000001.warc.gz",
  "Offset": "677"
 },
 "Envelope": {
  "Format": "WARC",
  "Payload-Metadata": {
   "Actual-Content-Type": "application/http; msgtype=response",
   "HTTP-Response-Metadata": {
    "Headers": {
     "Accept-Ranges": "bytes",
     "Connection": "close",
     "Content-Length": "22",
     "Content-Type": "text/plain",
```

“Data Mining”

- Extraction of useful information from a potentially heterogenous corpus.
- Corpus: A bunch of data. Web data is a good example.
- Heterogenous: Data not in a particular format: “some HTML, some images, some PDFs..”
- Useful information: counts, tables, sometimes non-obvious transformations based on complex analysis. Probably not heterogenous.

How valuable is your data?

- What do you know about it?
  - “We have 55 Terabytes. That's a lot.”
  - “It's mostly websites about X.”
  - “I can give you a list of the seeds.”
- How accessible is it?
  - To users? To researchers?

Data mining can help make your data more valuable

- Some metadata can enable better UIs.
  - Geo-location maps
  - Tag clouds
  - Classification
  - Facets
  - Rate of change
  - Related information
- Better UIs increases value to end users.

Researcher success stories can help demonstrate value

- To do many kinds of web analysis, researchers have to first crawl the web.
- We've already done that! We can lower the barrier for their work, allowing them to focus on their subject matter.
- We can further lower the barrier by providing frameworks and tools.

Providing access can be hard

- Logins
- Limiting access
- Managing resource contention
  - Network
  - CPUs
  - Disks getting full
- Interference with core services
- High SW engineering costs

Robust data mining tools can make this a reality

- Implementing a framework and infrastructure which lowers the cost of this type of analysis and research enables:
  - internal and external experimentation
  - Implementing more robust, scalable automation
    - Wayback indexing
    - FT search indexing
    - Enhanced UIs

The Logical Stack

Researchers

Engineers

Tools

Hardware

DATA!!! : D

Researchers

- Folks who want to do analysis of some kind on your data.
- Some will have technical expertise, and know exactly what they want, some will have a vague notion..

Engineers

- Individuals familiar with:
  - The data
  - The tools
  - The infrastructure
- Translate researcher questions into code.

Tools

- This is the major component we'll be focusing on today:
  - Hadoop DFS
  - Hadoop Map Reduce
  - IgPay AtinLay
  - Web Archive specific tools

Hardware

- How much you need depends on what you will use it for, and how much data you need to process
- Can grow over time
- IA configuration:
  - 40 x 2 Core, 4 GB, 4x700GB disks
  - 40 x 4 Core, 16 GB, 4x1TB disks

Data

- ARCs and WARCs are “heavy”
- WAT – Web Archive Transformation
  - Uses WARC format as a generic meta data container
  - Extract everything you're likely to want from ARCs/WARCs once
  - Transferable?!?
  - Feedback requested!
- Store into HDFS?
- Part of standard ingest process?

Tools: Redux

- HDFS
- Map Reduce
- Pig Latin
- Web archive code
- Other extraction layers: Tika, Jhove(2), etc

HDFS

- Distributed
- Durable
- Reliable
- Scalable
- Data stored in blocks on the same nodes that run processes
- Single “head” node stores filename, permissions, and knows where a files blocks are located – right now.

Map Reduce

- 2 parts:
  - Programming model, “map()” and “reduce()”
  - Execution framework which distributes jobs and manages the 10s, 100s, or 1000s of nodes where those jobs run
- Fault tolerant
- Robust
- Scalable

Map Reduce programming

Submit job

Run Job

Hadoop cluster

Job tracker

client

Client sends map-reduce job to Job Tracker, inside a “Jar” file
Job Tracker distributes Jar file to cluster, assigns tasks, monitors
Hadoop cluster runs job, stores result
Job Tracker informs client of success

Pig Latin

- Map-Reduce programs are not too hard to write, but can get verbose for multi-stage jobs, and have low code reuse
- Many interesting operations require more than one job

Pig Latin:

- High level data flow language
- Allows concise expressing of complex transformations
- Sends multiple Map-Reduce jobs to a cluster
- Manages intermediate data, cleanup
- Handles messy Joins!

Example: Count MIME Types in CDX

```
--Load CDX lines
CDXLines = LOAD '/tmp/example.cdx' USING PigStorage(' ') AS (url, timestamp,
orig_url,mime,response_code,checksum,redirect_url,offset,filename);

--Group CDX lines by MIME Type
GrpdMimes = GROUP CDXLines BY mime;

--Count number of occurrences of each MIME type
CountMimes = FOREACH GrpdMimes GENERATE group, COUNT(CDXLines) AS cnt;

--Order the counts in descending order
SortedCountMimes = ORDER CountMimes BY cnt DESC;

--Dump the output to screen
DUMP SortedCountMimes;
```

Web archive code

- Currently just a bit of glue code around an ARC/WARC reader
- Does HTML metadata extraction
- Includes example “UDF” code
  - simplifies and speeds up injecting external libraries into Pig Latin, to expand Pig Latin's capabilities
- Will integrate with Jhove(2), Tiki, etc

Trivial Example: URL + Title

```
-- Load IA magic sauce:
REGISTER /home/brad/archive-meta-extractor-20110413.jar;

-- load data from INPUT_DIR:
Orig = LOAD '/tmp/sample-wats/' USING
org.archive.hadoop.ArchiveJSONViewLoader('Envelope.WARC-Header-Metadata.WARC-
Target-URI','Envelope.Payload-Metadata.HTTP-Response-Metadata.HTML-
Metadata.Head.Title') AS (src,title);

-- discard lines without titles
PagesWithTitles = FILTER Orig BY title != '';

-- remove duplicates
Result = DISTINCT PagesWithTitles;

STORE Result INTO '/tmp/sample-output/' USING PigStorage();
```

Interoperability, Sharing UDFs

- Common formats enables sharing of data, code, experience across institutions
- As various institutions develop more UDFs and Pig Latin scripts, the whole community grows more capable

Did we make it?!

- More info from the full day IIPC 2011 workshop at:
  - http://home.us.archive.org/~vinay/iipc-2011/
- brad@archive.org