A Framework for Describing Web Repositories

Frank McCown
Department of Computer Science
Harding University
Searcy, AR, 72149
fmccown@harding.edu

Michael L. Nelson
Department of Computer Science
Old Dominion University
Norfolk, VA, 23529
mln@cs.odu.edu

ABSTRACT

In prior work we have demonstrated that search engine caches and archiving projects like the Internet Archive's Wayback Machine can be used to "lazily preserve" websites and reconstruct them when they are lost. We use the term "web repositories" for collections of automatically refreshed and migrated content, and collectively we refer to these repositories as the "web infrastructure". In this paper we present a framework for describing web repositories and the status of web resources in them. This includes an abstract API for web repository interaction, the concepts of deep vs. flat and light/dark/grey repositories and terminology for describing the recoverability of a web resource. Our API may serve as a foundation for future web repository interfaces.

Categories and Subject Descriptors

H.3.7 [**Information Storage and Retrieval**]: [Digital Libraries]

General Terms

Standardization, Design

Keywords

Preservation, Web Repositories, Web Resources

1. INTRODUCTION

As more of the activities of our daily lives are represented on the Web, the associated curatorial and preservation challenges continue to increase. Conventional web preservation projects and techniques require a significant investment of time, money and effort and are thus applicable only to collections of known value. The limited scope of such projects may leave a number of potentially important web collections unprotected. For these unprotected collections, *lazy preservation* can provide a broad preservation service with no cost to individual content producers [8]. Lazy preservation makes use of the Web Infrastructure (WI). The WI refreshes and migrates web content in bulk as side-effects of its user-services, and these results can be mined as a useful, but passive preservation service. The WI includes search engine companies (e.g., Google, Yahoo, Live Search), nonprofit companies (e.g., Internet Archive's "Wayback Machine"), personal archiving services (e.g., Hanzo:Web, Furl) and large-scale academic projects (e.g., CiteSeer, NSDL Persistent Archive). These repositories of web resources (or **web repositories**) form the backbone of the WI. In prior work we have investigated the nature of lost websites and how they can be reconstructed from the WI [7, 10].

Despite the importance of these web repositories, existing frameworks such as OAIS [4] lack the terminology for describing these repositories, their functionality and how web resources move through the WI. To this end we present: an abstract API for describing interaction with the WI; the concepts of deep vs. flat and light, dark and grey web repositories; and states of recoverability (vulnerable, replicated, endangered) of a web resource in the WI.

JCDL'09, June 15–19, 2009, Austin, Texas, USA.

2. WI LIMITATIONS

The WI relies primarily on web crawling to refresh and migrate web resources. But web crawling is limited to the surface web, the portion of the Web that is connected with hyperlinks. Therefore pages that are returned in response to a query, hidden behind JavaScript, Flash and CAPTCHAs, and pages that are not connected to others are usually inaccessible to the WI. Additionally, search engines and web archives respect the robots exclusion protocol and will not store web pages that use `noarchive` meta tags.

The search engine's goal of providing information to users may sometimes conflict with preservation goals. Search engines often avoid crawling or indexing duplicate content [13] or content determined to be spam [6]. They may not crawl deeply into a website since such pages may not be useful to search engine users [2]. Furthermore, their cache replacement policies are proprietary and subject to change without notice. In previous work, we have measured resources staying in search engine caches for 10 to 103 days [11].

Web archives face many of the same technical challenges as search engines when crawling the Web, but they are less likely to avoid duplicate content, spam or any particular resource format. Because the archive's goal is to preserve the Web as it was found, web archives are useful when reconstructing lost websites. Unfortunately, web archives may not have the same amount of resources (human and technical) available as commercial search engines, and their coverage of the Web may be somewhat limited.

3. WEB REPOSITORY TYPES

Web repositories are members of the WI that crawl and store a sizeable portion of the Web and provide URI granularity to access their stored resources. Web repositories are generally accessible to automated queries through the HTTP protocol, using direct GET queries or an API, web service, or any other method that can be automated.

Web repositories may be characterized by the depth of their holdings. Search engine caches like Google's are repositories that store only the latest resource crawled; when a resource is re-crawled, the new resource replaces the older version in the repository. The depth d of such repositories is one since only one copy of a resource is maintained. Search engine caches are thus examples of **flat repositories** ($d = 1$). A **deep repository** ($d > 1$) maintains older versions of resources; it stores a datestamp along with each version of the resource with the same URI. The Internet Archive (IA) (with $d = \infty$) is an example of a deep repository.

Web repositories may also be categorized by the access granted to their holdings [14]. Borrowing from conventional archiving terminology, **dark repositories** do not provide public access to their holdings. Some repositories are made dark due to legal restrictions (e.g., national web archives) or because they are merely serving as a fail-safe in case the original resource is no longer accessible. **Light repositories**, however, place minimal access controls on their holdings. Search engine caches and IA are examples of light repositories. In some cases IA holdings may be considered dark since IA will not allow public access to an archived resource when the website from which the resource was obtained contains a robots.txt entry blocking access to the item. **Grey repositories** are limited to a small number of individuals, like Cornell's Yesternet [1] which is limited to researchers.

Recent overlap studies have shown that Google, Live Search (previously MSN Search) and Yahoo have indexed significantly different portions of the Web [3]. All three search engines make their indexed pages available from the "Cached" link (or "View as HTML" for other document types) provided next to each search result. These cached resources can be used for recovering lost pages if caught in time.

The Internet Archive is currently the world's largest publicly accessible web archive. Although there are several large national web archives in existence [5], none are focused on saving the entire Web. An overlap analysis of the Internet Archive's holdings indicates there are numerous resources found in search engines caches that are not found in the Internet Archive [7].

The Internet Archive strives to maintain an accurate snapshot of the Web as it existed when crawled. Therefore they archive each resource in the same format in which it was crawled. Search engines have traditionally been HTML-centric, but as the amount of non-HTML resources has grown, so has their ability to index and cache these types of resources. When adding PDF, PostScript and Microsoft Office resources to their cache, the search engines create HTML versions of the resources which are stripped of all images. In most cases it is not possible to recreate the canonical version of the document from the HTML version.

The search engines have separate search interfaces for their images, and they keep only a thumbnail version of the images they cache due to copyright law [12]. As shown in Table 1, most resources are not stored in their canonical format in the search engine caches. In some cases like Flash, text from the resource may be indexed, but the binary resource is not accessible from the repository. Only HTML appears to be stored in its canonical format across all four repositories.

Table 1: Web repository-supported data types as of July 10, 2007.

Type	Google	Yahoo	Live	IA
HTML	C	C	C	C
Plain text	M	M	M	C
GIF, PNG, JPG	M	M	M	C
JavaScript	M		M	C
MS Excel	M	∼S	M	C
MS PowerPoint	M	M	M	C
MS Word	M	M	M	C
PDF	M	M	M	C
PostScript	M	∼S		C
Flash	∼S			C
XML	C	∼S		C

C = Canonical version is stored
M = Modified version is stored (thumbnails or conversions)
∼S = Indexed but stored version is not accessible

There are many other resource types not mentioned in Table 1 that may be found on the Web: binary programs, video, audio and archive files. While the IA attempts to preserve these resource types, search engines are generally not interested in them since there is little or no textual content that can be indexed.

4. ABSTRACT API

In this section we introduce an abstract API for accessing cached and archived resources in the WI and describe which methods are supported by the major web repositories. A web repository must support, at a minimum, the ability to handle the query, "What resource r do you have stored for the URI u?" where u is the resource's URI when it was obtained by the repository on the Web:

$$r \leftarrow \texttt{getResource}(u) \tag{1}$$

The repository will respond to this query with the resource in the same format (canonical format) in which it was crawled from the Web or in some altered format, such as a thumbnail version of an image. If the resource has not been stored, the repository will respond with some negative response.

The format of the resource may be supplied as metadata with the resource, or it may be inferred from the URI, its usage in the source HTML page or the repository from which the resource is obtained. For example, if Google were queried for the resource at `http://foo.org/hello.gif`, it could be inferred that the resource is an image because of the URI's .gif ending or because it was referenced in an <img> tag. Since Google Images is known only to store thumbnail images, it can additionally be inferred that the resource is not in its canonical format. Had the resource been obtained from IA, it could be assumed the image was in its canonical format.

Deep repositories should allow resources to be obtained

Table 2: Implementation summary of web-repository interfaces.

Queries	IA	Google	Live	Yahoo
getResource	✓	✓	✓	✓
getResourceList	✓	N/A	N/A	N/A
getCrawlDate	✓	Limited to HTML	Images not supported	Last modification date, images not supported
getAllUris	✓	First 1000 results only	First 1000 results only	First 1000 results only

using a URI u and the datestamp ds, the day on which the resource was crawled, to distinguish among multiple versions of the same resource:

$$r \leftarrow \texttt{getResource}(u, ds) \quad (2)$$

The repository will only respond with the resource at URI u that was crawled on date ds. To obtain a list of available datestamps that are acceptable, the repository should ideally support a query which returns the stored resources for the given URI u:

$$D \leftarrow \texttt{getResourceList}(u) \quad (3)$$

where D is the set of all datestamps stored in the repository for the resource.

Flat repositories should ideally provide the date the returned resource was crawled, perhaps as metadata in r from getResource, or by supporting such a query:

$$d \leftarrow \texttt{getCrawlDate}(u) \quad (4)$$

Having a datestamp for each resource allows a web-repository crawler to chose between multiple resources from multiple repositories that have been crawled at different times, like, for example, when wanting to select the most up-to-date resource. It also allows the crawler to reject resources that are not from a particular time frame.

An additional query type which allows for more efficient crawling is, "What resources R do you have stored from the site s?":

$$R \leftarrow \texttt{getAllUris}(s) \quad (5)$$

The returned value R is a set of all URIs and datestamps stored in the repository $\{(u_1, d_1), (u_2, d_2), ..., (u_n, d_n)\}$. This type of query, called a **lister query**, can greatly decrease the number of queries that a repository must handle since it can be asked for only those resources it is known to contain. Additionally, deep repositories may provide an enhanced query to limit the returned resources to a particular date range dr:

$$U \leftarrow \texttt{getAllUris}(s, dr) \quad (6)$$

5. IMPLEMENTING THE API

Each of the repositories implement the abstract API in different ways and with varying completeness. A summary of the web-repository interfaces supported by the four web repositories is given in Table 2.

The three search engines implement the repository interfaces in a similar manner with a few significant differences. All three search engines support `getAllUris`, `getCrawlDate` and `getResource` queries. To perform `getAllUris` (lister queries), the query parameter "site:" is used. For example, to retrieve all URIs for the website `www.cs.odu.edu` from Google, the query `site:www.cs.odu.edu` is used. All three search engines will return only the first 1000 results, so lister queries are of limited use for large websites.

The `getCrawlDate` query is not supported directly by Google and Live, but it is indirectly supported by examining the metadata returned from the `getResource` query. Unfortunately, Google does not make the cached date available for non-HTML resources, so `getCrawlDate` is only partially supported by Google.

Yahoo does not place the crawled date in their cached page's heading; it must be retrieved using their API. The date returned by Yahoo's API is not the date the resource was crawled but instead the date they last noticed it had changed (the ModificationDate). Yahoo does not specify how they measure change, but it is likely based on the resource's Last Modified HTTP header.

In order to access images from the search engines, a different but similar procedure is involved. The `getAllUris` query is invoked against Google Images (or images API for Live and Yahoo). Again, only the first 1000 results can be obtained. The `getResource` query is implemented by finding the URI of the thumbnail image and accessing it directly. Note that no datestamp is available for images from any of the search engines.

The interface queries `getAllUris`, `getResourceList` and `getResource` are all supported by IA. To perform `getAllUris`, an HTTP request is made in the form: `http://web.archive.org/web/*sr_0nr_20/http://www.cs.odu.edu/*`. IA will respond to this query with a listing of all URIs it has stored that match the given URI prefix. IA does not limit the number of results returned, so paging through all resources stored is possible. The `getAllUris` query with date range is not completely supported by IA, but a date range of one year (e.g., limiting to 2006 is `http://web.archive.org/web/2006*/http://www.cs.odu.edu/`) or one month (e.g., limiting to January 2006 is `http://web.archive.org/web/200601*/http://www.cs.odu.edu/`) is supported. IA also supports the `getResourceList` query by returning a page listing all the stored versions of a page.

The `getResource` query is implemented by directly accessing any of the stored resources. For example, the page stored on January 24, 2005, can be accessed at `http://web.archive.org/web/20050124084644/http://www.cs.odu.edu/`. Note the date (ds) in YYYYMMDD format (20050124) embedded in the URI.

6. RESOURCE RECOVERABILITY

If a resource is discovered and archived by a web archive (deep repository), it will likely remain accessible from the WI even when the resource disappears from the Web. How-

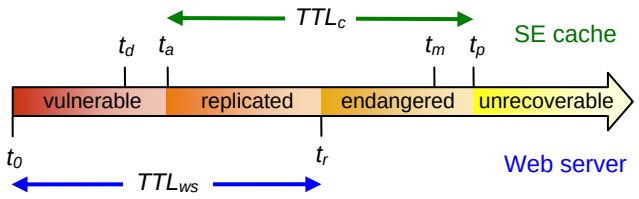

Figure 1: Timeline of search engine resource acquisition and release.

ever, if a resource is discovered and cached by a search engine (flat repository) and then disappears from the Web, it will only have a limited time of protection before it is purged from the search engine's cache.

Figure 1 illustrates the life span of a web resource from when it is first made available on a web server to when it leaves a search engine cache. A web resource's time-to-live on the web server (TTL_{ws}) is defined as the number of days from when the resource is first made accessible on the server (t_0) to when it is removed (t_r). The period beginning when the resource is accessible from a search engine's cache (t_a) to when it is finally purged (t_p) defines a resource's time-to-live in the search engine cache (TTL_c). The following classifications are used to describe the resource's availability:

- **Vulnerable** - A new resource which has not yet been discovered by a search engine (t_d) and made available in the search engine cache (t_0 to t_a).
- **Replicated** - A resource which remains accessible on the web server and has also been cached by a search engine (t_a to t_r).
- **Endangered** - A resource which is no longer accessible on the web server but still remains cached (t_r to t_p).
- **Unrecoverable** - A resource which has been discovered to be no longer accessible on the web server (t_m) and has been evicted from cache (t_p).

A resource is *recoverable* if it is currently cached (i.e., is replicated or endangered). Ideally, a cached resource will always be accessible from the cache at any given time but this is not always true [9]. Therefore, a recoverable resource can only be recovered during the TTL_c period with a probability of P_r, the observed number of days that a resource is retrievable from the cache divided by TTL_c.

It should be noted that the TTL_{ws} and TTL_c values of a resource may not overlap. A search engine that is slow in updating its cache, perhaps because it obtains crawling data from a third party, may experience late caching where $t_r < t_a$. This is particularly true for a web archive like IA that makes resources available from their archive months after they have been crawled.

7. CONCLUSIONS

We have presented a framework for discussing web repositories in the web infrastructure. We distinguish between deep repositories (those that store multiple timestamped versions of a resource) and those that are flat (storing only the most recent version). We also borrow from conventional archiving the terms of dark and light repositories, and introduce the term grey repository for those that are partially accessible. We have introduced the concepts of recoverable web resources being vulnerable (accessible but not cached), replicated (accessible and cached) and endangered (not accessible but still cached). Finally, we have presented an abstract API for discovering resources in web repositories and have discussed to what extent four repositories implement that API. We hope this abstract API can be a foundation for future work in building a standardized API for web repositories.

8. REFERENCES

[1] W. Y. Arms, S. Aya, P. Dmitriev, B. J. Kot, R. Mitchell, and L. Walle. Building a research library for the history of the web. In *Proceedings of JCDL 2006*, pages 95–102, 2006.

[2] R. Baeza-Yates and C. Castillo. Crawling the infinite web: five levels are enough. In *Proceedings WAW 2004*, pages 156–167, 2004.

[3] Z. Bar-Yossef and M. Gurevich. Random sampling from a search engine's index. In *Proceedings of WWW '06*, pages 367–376, 2006.

[4] Consultative Committee for Space Data Systems. Reference model for an open archival information system (OAIS). Technical report, 2002.

[5] M. Day. Preserving the fabric of our lives: A survey of web preservation initiatives. *Research Advanced Technol. Digital Libraries*, pages 461–472, 2003.

[6] D. Fetterly, M. Manasse, and M. Najork. Spam, damn spam, and statistics: using statistical analysis to locate spam web pages. In *Proceedings of WebDB '04*, pages 1–6, 2004.

[7] C. Marshall, F. McCown, and M. L. Nelson. Evaluating personal archiving strategies for Internet-based information. In *Proceedings of IS&T Archiving 2007*, pages 151–156, May 2007.

[8] F. McCown. *Lazy Preservation: Reconstructing Websites from the Web Infrastructure*. PhD thesis, Old Dominion University Department of Computer Science, 2007.

[9] F. McCown and M. L. Nelson. Characterization of search engine caches. In *Proceedings of IS&T Archiving 2007*, pages 48–52, May 2007.

[10] F. McCown and M. L. Nelson. Usage analysis of a public website reconstruction tool. In *Proceedings of JCDL '08*, pages 371–374, 2008.

[11] F. McCown, J. A. Smith, M. L. Nelson, and J. Bollen. Lazy preservation: Reconstructing websites by crawling the crawlers. In *Proceedings of WIDM '06*, pages 67–74, 2006.

[12] S. Olsen. Court backs thumbnail image linking. *CNET News.com*, July 2003. `http://news.com.com/2100-1025_3-1023629.html`.

[13] N. Shivakumar and H. Garcia-Molina. Finding near-replicas of documents and servers on the web. In *Proceedings of WebDB '98*, pages 204–212, 1999.

[14] S. E. Thomas and C. A. Kroch. Project Harvest: A report of the planning grant for the design of a subject-based electronic journal repository, Sep 2002. `http://diglib.org/preserve/cornellfinal.html`.