Issues in Information Systems
Volume 16, Issue IV, pp. 64-69, 2015

THE MANAGEMENT OF A WEBSITE'S HISTORICAL LINKS AND DOCUMENTS

David Chao, San Francisco State University, dchao@sfsu.edu

ABSTRACT

An organization's websites change constantly to reflect the dynamic nature of its activities and its environment. Consequently, historical links and documents are generated that include outdated URLs, the old versions of web pages and, the deleted web pages. These old versions are snapshots of a document as of a specific point in time. Studying a document's snapshots may help users identify trends and patterns of changes in a document, and as a consequence, the evolution of meaning from a historical perspective. Managing historical links and documents enable a website to satisfy users' needs for complete and correct historical information. This paper presents a logging and archiving scheme to track a document's history of changes, and a temporal URL scheme that is capable of retrieving web page instances that meet user's temporal requirements.

Keywords: Website Archive, Webpage Snapshot, Version, Historical Link

INTRODUCTION

A web site is a system of integrated web documents embedded with links to reference related documents. At any given time, a Uniform Resource Locator (URL) uniquely identifies a document on the Internet. URLs are also known as the addresses, or paths, of documents on the Internet. An organization's websites change constantly to reflect the dynamic nature of its activities and its environment. Consequently URLs are temporal and exist only for a limited period of time.

When a website changes its structure, folders may be renamed, relocated and deleted. Web pages in those folders may either be removed from the website or given a new URL. The old URLs associated with the web pages in those folders will either become out-of-date or be used to point to different web pages. When a website changes its contents, some web pages may be changed or deleted. Hence, the old versions of an updated web page become historical; so are the deleted web pages. A web page is a text file that may be defined by many types of code including not exclusively HTML, client-side JavaScript and server-side script such as PHP. A web page is considered *changed* if the code that defines the web page is changed.

These historical data are valuable. The old versions of a document are snapshots of the document as of a specific point in time. Studying a document's snapshots may help users identify trends and patterns of changes in a document, and as a consequence, the evolution of meaning from a historical perspective. Old URLs will be useful in forming audit trail for tracking website changes. One common frustration of surfing the Internet is receiving a HTTP 404, file-not-found error after submitting a URL to a web server, a problem known as "link rot". Users may get the invalid URLs from stale sources such as browser's bookmarks, published articles and books. And sometimes a modified document or an unrelated document is retrieved, a problem known as "content drift". Such problems and the pervasiveness of these problems are reported in a recent article published in New Yorker by Jill Lepore [7]. The author cited a 2014 study conducted at Harvard Law School [13] that "more than 70% of the URLs within the Harvard Law Review and other journals, and 50% of the URLs within United States Supreme Court opinions, do not link to the originally cited information."

There are many ongoing Internet archiving projects undertaken by government agencies and private organizations such as Bentley Historical Library [1], Library of Congress Web Archive [8] and Internet Archive and its WayBack Machine [6]. Other web archiving initiatives can be found in [5, 12]. These organizations are committed to collecting portions of the World Wide Web to ensure the information is preserved in an archive for future users. They typically uses crawling technology [3] to take snapshots of websites. Many commercial website archiving vendors, such as PageFreezer [10], also use crawling technology. The website archives thus created exist as a date-time stamped copy of the presentation of the website and only represents the website's state at the time of crawling. They are unable to satisfy a user's request for a snapshot at a different time. Crawling may cause other problems.

Since crawling a website takes long time to complete and crawling may take place while the website is undergoing changes. Thus causing temporal incoherence among the archived documents as explained in Niu [9]. There are web services available to address the link rot problem. One such service is Perma.cc developed by the Harvard Law School Library [11, 13]. When citing a web document in a legal article, the author can go to the Perma.cc website and input the URL of the referenced document, then Perma.cc archives a copy of the referenced web document and generates a new URL called "Perma.cc link" that can be inserted in the article. This is an external service and does not track changes to the referenced documents.

A website needs to preserve and provide its own historical data to satisfy users' needs. This paper presents a logging and archiving scheme for a website to track a document's history of changes including the changes in its URL and content, save versions of a modified documents and deleted documents in an archive, and designs a Temporal URL scheme that is capable of retrieving web page versions that meet user's temporal requirements. There has been a substantial amount of research on extending database technologies to manage web page versions. A good review is provided in Dyreson, Lin, and Wang [4], and it proposes an extension to the Apache web server to manage web page versions and designs a query language to retrieve desired versions. The scheme we proposed in this paper is independent of web server used and can be developed into an application to manage a website's historical data.

This paper is organized as follows. Section 2 presents an analysis of the problem in managing historical links. Section 3 develops a logging and archiving scheme for tracking changes. Section 4 presents the design of temporal URL to retrieve historical documents. Section 5 concludes the paper.

PROBLEM ANALYSIS

A web site is a system of integrated web pages embedded with links to reference related pages. At any given time, a Uniform Resource Locator (URL) uniquely identifies a web page on the Internet. URLs are also known as the addresses, or paths, of web pages on the Internet. The relationship between URL and web page is n:m where each URL may be associated with m web pages and each web page may be associated with n URLs over time. A URL may point to a web page that is different from the one it pointed to earlier. This may happen because of the renaming of web pages and directories.

The URLs of all web pages currently published by a web site are web site's *current links*. A web page is considered *current* since the time it is published until the time it is changed when the old version becomes the page's snapshot between the published time and the update time, and the update time becomes the new version's published time. In a sense, the current version of a web page is the page's snapshot since its published time to the time the page is modified or deleted. A web page may be changed repetitively. If snapshots are kept in an archive, the combined value of a page's URL and published time can be used as a *snapshot - ID* to uniquely identify a snapshot.

Table 1 summarizes the impact of a web site reorganization and change to a web page in generating historical resources; it assumes the existence of an archive where snapshots of web pages are stored. When a web site reorganizes its directory structure, all impacted web pages will have a new URL. Similarly, when a web page is renamed or relocated to other directory, it will have a new URL as well. When a web page is modified, its URL will not change and the old version becomes a snapshot and is archived. A deleted document is archived as well for later retrieval.

The URLs invalidated by a web site due to reorganization, web page removal, renaming, or relocation, plus all *snapshot -ID* s are a web site's *historical links*. Maintaining historical links enables a web site to retrieve the web page associated with an outdated URL submitted by users, deleted web pages, and web page snapshots. This will improve a web site's ability in searching documents and provide needed historical information for users.

Table 1. Impact of Website Reorganization and Web Page Changes

Web Site Actions	Impact on Current and Historical Links
Adding a new web page	Adding a new URL to current links
Modifying a web page	No change to URL; the old web page becomes a snapshot and archived

Deleting a web page	Deleting a current link; adding a historical link and web page is archived
Renaming a web page	Adding a new URL to current links; the old URL becomes historical link
Relocating a web page	Adding a new URL to current links; the old URL becomes historical link
Reorganization	Adding all affected web pages' URLs to current links; adding all affected web pages' old URLs become historical

DESIGN OF THE LOGGING AND ARCHIVING SCHEME FOR TRACKING WEBSITE CHANGES

A web page and all internal resources it references are *files* to the website. Changes to a file including adding, deleting, and modifying a file are detectable by the operating system. Many application development environments, such as Microsoft .Net framework, offer directory monitoring to detect such changes. A web page's URL consists of host name, the path to the directory where it resides on the host, and the document's name. The host name typically represents the website's root directory, and the path is the folders to the web page. We will use the path and the file name to identify a web page and all the internal resources. In the following discussions, the term *web document* represents either a web page or an internal resource, and URL is the path + file name of a web document.

We present a logging and archiving scheme for tracking historical links and snapshots associated with the historical links. This scheme extends the scheme published in [6] to include tracking changes to internal resources. The log, named TemporalURLLog, is designed to keep the history of changes to web documents. It has five fields: URL, PublishDate, DocExpireDate, URLExpireDate, and NewURL. The URL field records a document's path; the PublishDate records the time the document is published; the DocExpireDate field records the time the document is modified or deleted; the URLExpireDate records the time document's URL expires; the NewURL field records the document's new URL should a change to the document creates a new URL for the document. This log has a composite key of URL + PublishDate. The value of URL + PublishDate uniquely identifies a document in a web site's history.

The Archive is a directory that stores deleted documents and document snapshots. An ArchiveDirectory table is created to keep track of documents in the Archive. It has this simple structure: ArchiveFileID + URL + PublishDate. The ArchiveFileID is a generated unique ID. When an archived documents is saved in the Archive it is saved with a unique ArchiveFileID as the file name and an entry is entered into the ArchiveDirectory table with the ArchiveFileID + URL + PulishDate. Note that he ArchiveFileID can be identified uniquely by: URL + PublishDate in the ArchiveDirectory table.

The log is maintained according to the log maintenance algorithm described below:

TemporalURLLog and Archive maintenance algorithm

Changes to web documents are recorded in the log and saved in the Archive by the following rules:

New document: When a new document is added to the web site, a new entry is entered with its URL and the time the document is published. The DocExpireDate, URLExpireDate and the NewURL are set to null. Hence, log entries with a null DocExpireDate and null URLExpireDate are current document entries. Initially, all current documents have an entry in the log with null DocExpireDate, URLExpireDate and null NewURL.

Deleted document: The log entry associated with the document is the entry of which the URL equals the document's URL with a null DocExpireDate and null URLExpireDate. First, it locates the entry and changes its DocExpireDate and URLExpireDate to the time the document is deleted. Then, it saves the deleted document in the Archive that can be identified uniquely by: URL + PublishDate.

Modified document: When a document is modified, its old version becomes a snapshot and its new version is treated as a new document. The log entry associated with the document is the entry of which the URL equals the document's URL with a null DocExpireDate and null URLExpireDate. First, it locates the entry and changes its

DocExpireDate to the time the document is modified and leave the URLExpireDate be null. When a document is modified its old version is expired but the URL is unchanged. So it saves the old version in the Archive that can be identified uniquely by: URL + PublishDate. Then, it adds a new entry with the same URL and the PublishDate is set to the time the document is modified; the DocExpireDate, URLExpireDate and NewURL are set to null.

Consequently log entries with a non-null DocExpireDate and a null NewURL may be related to snapshots or deleted documents. For such an entry, if there exists another entry with the same URL and its PublishDate equals to this entry's DocExpireDate then this entry is a snapshot entry and the snapshot it associates with can be retrieved from the Archive using URL + PublishDate to identify the file. The snapshot is valid from the PublishDate to the DocExpireDate. Otherwise, the entry is associated with a deleted document which can be retrieved from the Archive using URL + PublishDate to identify the file.

Relocated or renamed page: When a document is relocated or renamed, its old URL is expired and a new URL is created. The log entry associated with the document is an entry with the URL equal to the document's URL and with a null DocExpireDate and null URLExpireDate. First, it locates the entry and changes the URLExpireDate to the relocation or renaming time and its NewURL field to the document's new URL. Note that the DocExpireDate field remains null unless the document is modified. Then, it adds a new entry with the new URL and the PublishDate is set to the time the document is relocated or renamed. If at the same time the document is also modified, the DocExpireDate of the original entry is changed to the time the document is modified and the old version is saved in the Archive can be identified uniquely by: URL + PublishDate.

Hence, a log entry with non-null NewURL field may have a null DocExpireDate indicating the document is not modified, or a non-null DocExpireDate indicating the document is modified.

Example: Table 2 illustrates the progression of a web site. T_0 is the time when the web site is initiated, and T_i denotes a point in time when its contents change. The second column shows the changes at each time period. Note that changes such as P1at P_0, and P4 and P5 at T_4 represent documents that are renamed or relocated. In this case, only the document's URL has changed but the document has not changed. Therefore, there is no need to archive any documents, but it is necessary to chain the old path to the new path in order to track the change. Changes such as P3 at P_2, and P3, P4, and P5 at T_3 represent documents that are modified. In this case, the URL is still valid but the document before the change needs to be archived as the document's snapshot. Changes such as P2 at T_2 and P3 at T_4 represent documents that have been deleted. Those documents must be archived.

Table 2. Impacts of Website Changes to Historical Links and Archive

Time	Website Changes	Current Web Pages	Historical Links Generated	Snapshots in Archive
T_0		P1, P2, P3	None	None
T_1	P1 renamed to P4 P5 is added	P2, P3, P4, P5	P1+ T_0	
T_2	P2 is deleted P3 is modified	P3, P4, P5	P2+ T_0, P3+ T_0	P2+ T_0, P3+ T_0
T_3	P3, P4, P5 is modified P1, P6 are added	P1, P3, P4, P5, P6	P3+ T_2, P4+ T_1 P5+ T_1	P3+ T_2, P4+ T_1, P5+ T_1
T_4	P3 is deleted P4 is renamed to P8 P5 is renamed to P7 A new page P3 is added	P1, P3, P6, P7, P8	P3+ T_3, P4+ T_3 P5+T_3	P3+ T_3

Using the log maintenance algorithm for the changes described in Table 2, the contents of the TemporalURLLog are shown in Table 3, and there are six files in the Archive. With the TemporalURLLog and the Archive, a web server is capable of performing the following tasks as illustrated by the examples.

a. Retrieve a snapshot of a current web page:

A current page P7's snapshot at T_2 can found in the Archive with file name P5+ T_1.

A current page P6 itself is its snapshot since T_3.

b. Retrieve a deleted document:

The web page associated with an out-dated URL P2 can be found in the Archive with file name P2+T_0.

c. Retrieve the snapshot of a deleted web page:

The snapshot of the historical link P3 at T_2 can be found in the Archive with file name P3+ T_2.

d. Retrieve the current web page of an out-dated URL:

An old URL P5 is now renamed to P7. If users submit a request for P5, it can be traced to P7.

e. Retrieve the web page previously associated with a current link:

A historical link P1 is now renamed to P8, and a current link P1 points to a new web page. If the current web page associated with P1 is not what the users need, it can be redirected to P8.

f. Determine if an invalid URL ever exists

A URL P12 has never existed in the web site.

Table 3. The Contents of TemporalURLLog Based on Changes Described in Table 2

URL	PublishDate	DocExpireDate	URLExpireDate	NewURL
P1	T_0	Null	T1	P4
P2	T_0	T_2	T2	Null
P3	T_0	T_2	Null	Null
P4	T_1	T_3	Null	Null
P4	T3	Null	T4	P8
P5	T_1	T_3	Null	Null
P3	T_2	T_3	Null	Null
P3	T_3	T_4	T4	Null
P5	T_3	Null	T4	P7
P6	T_3	Null	Null	Null
P1	T_3	Null	Null	Null
P8	T_4	Null	Null	Null
P7	T_4	Null	Null	Null
P3	T_4	Null	Null	Null

Using the log maintenance algorithm for the changes described in Table 2, the contents of the ArchiveDirectory table are shown in Table 4, and there are six files in the Archive.

Table 4. The Contents of ArchiveDirectory Table Based on Changes Described in Table 2

ArchiveFileID	URL	PublishDate
AF1	P2	T_0
AF2	P3	T_0
AF3	P3	T_2
AF4	P4	T_1
AF5	P5	T_1
AF6	P3	T_3

REQUESTING HISTORICAL DOCUMENTS WITH TEMPORAL URL

A temporal URL is a URL submitted with temporal requirements of which the documents associated with the URL must meet. A typical way to submit additional information with a URL is through query strings. A query string is a set of name=value pairs appended to a URL. It is created by adding a question mark (?) immediately after a URL followed by name=value pairs separated by ampersands (&). To retrieve a document from the Archive, the document's URL and the temporal requirement must be specified. The temporal URL has the following general format: Archive/URL/TemporalRequirement

The web seerver will search the ArchiveDirectory to find entries satisfying the requirement. Special keywords can be created for users to specify temporal requirements for snapshots. Some typical keywords are:

?SnapshotAsOf=*date*

?SnapshotsBefore=*date*

?SnapshotsAfter=*date*
?SnapshotsBetween=*date1*&And=*date2*

The ?SnapshotAsOf=*date* query string will retrieve a document's snapshot at the specified date. Note that if no entry satisfies the equality criteria, then the entry with the same URL but with a smaller PublishDate entry will be retrieved; others will retrieve all snapshots that meet the date criteria.

SUMMARY

This paper presents a logging and archiving scheme to track a document's history of changes including the changes in its URL and content, save versions of a modified documents and deleted documents in an archive, and designs a Temporal URL scheme that is capable of retrieving web page versions that meet user's temporal requirements. Unlike methods that using crawling technology, the proposed scheme preserves complete history of changes to a website's documents. A web page is considered modified if the code that defines the web page is modified including not exclusively HTML, client-side and server-side script code. The proposed scheme works well in preserving changes to the code that defines the web documents. In order to create the snapshot of a web page at a point in time with contents retrieved from a database, the database system supporting the web page must be able to create the database snapshot at the same point in time. The proposed scheme can be integrated into a framework for managing the snapshots of a database-driven website as proposed in Chao and Gill [2].

REFERENCES

1. Bentley Historical Library, University of Michigan, http://bentley.umich.edu/
2. Chao, D., Gill, S. (2008). A Framework for A Website Snapshot Management System. *Issues in Information Systems, Vol. IX, No. 2*, 279-285.
3. Crawling technology: http://en.wikipedia.org/wiki/Web_crawler
4. Dyreson, C., Lin, H., Wang, Y. (2004). Managing Versions of Web Documents in a Transaction-time Web Server. *The thirteenth World Wide Web conference*, New York, 422-432,
5. Goethals, A., Oury, C., Pearson, D., Sierman, B., Steinke, T., (2015). Facing the Challenge of Web Archives Preservation Collaboratively: The Role and Work of the IIPC Preservation Working Group, *D-Lib Magazine, May/June, 2015, Volume 21, Number 5/6* [online]. Available: http://www.dlib.org/dlib/may15/goethals/05goethals.html
6. Internet Archive WayBack Machine, http://www.archive.org/
7. Lepore, Jill. (2015). The Cobweb: Can the Internet be archived?. *The New Yorker*, January 26, 2015. Available: http://www.newyorker.com/magazine/2015/01/26/cobweb
8. Library of Congress Web Archive, http://lcweb2.loc.gov/diglib/lcwa/html/lcwa-home.html
9. Niu, J. (2012). An Overview of Web Archiving. *D-Lib Magazine. Volume 18, Number 3/4*. Available: http://doi.org/10.1045/march2012-niu1
10. PageFreezer: https://www.pagefreezer.com/products/website-archiving/
11. Perma.cc https://perma.cc/about
12. Web archiving initiatives: http://en.wikipedia.org/wiki/List_of_Web_archiving_initiatives
13. Zittrain, J., Albert, K., and Lessig, L. (2014). Perma: Scoping and Addressing the Problem of Link and Reference Rot in Legal Citations. *Legal Information Management*, 14, pp 88-99.