Web Archiving within the KB and some preliminary results with JHove and DROID.

Koninklijke Bibliotheek

September 2007

TABLE OF CONTENTS

1 KB and web archiving

In 2006, the National library of the Netherlands, the Koninklijke Bibliotheek (henceforth: KB), started archiving a selection of Dutch websites. As the country's national library, the KB is responsible for the permanent storage of both printed and electronic publications. Because more and more publications are appearing in electronic form, storing them permanently and keeping them accessible has become a very important task.

Whereas most international initiatives began concentrating on harvesting websites at an early stage and are still following this approach as a general rule, the KB emphasizes the permanent storage and future presentation of archived websites. This means that not only are websites are not just harvested but also a strategy for long-term access is being developed.

The complexity of this task is the reason why the KB did not start web archiving until 2006. From the very beginning, the KB has acknowledged the importance of digitally expanding its national depository function and has also taken practical steps in that direction. In 1995 it began investing in research on the development and furnishing of an electronic deposit library. Since 2003, this e-Depot system has provided the KB with an infrastructure that makes it possible not only to store articles from periodicals electronically but also to guarantee the archiving of websites.

2 First- and second phase of the project

During the first phase of the KB project (January 2006 - June 2007) the goal was to acquire as much knowledge and experience of website archiving as possible. Consequently only a limited number of the approximately 100 websites have been archived so far. This provided enough information to make a fair estimate of the resources and infrastructure that will be needed. During this first phase, the 100 selected sites were crawled, involving more than 360 GB of uncompressed data. These harvested sites consist of more than 16 million files with 200 different file formats.

The second phase of the project is concentrated on embedding Web Archiving in the existing organization using the Web Curator Tool[1], and increasing the selection to around 3,000 unique sites by the end of 2008. The intention is that this number will grow by the year and the selected sites will have to be archived a number of times per year. Given the amount of data and unique files that will have to be stored, as well as the abundance of file formats, it will take quite some work to develop a strategy for permanent access.

3 Digital Preservation

Not until the websites are gathered, indexed and made properly accessible for the user does the problem really begin. How can we make sure that these websites will still be accessible to the user in 50 years or so? We won't be using the browsers and platforms that we're now accustomed to using, and it may be that the concept of the web will have changed altogether by then. Even so, we

[1] http://webcurator.sourceforge.net/

must make sure that scientists 50 years from now can do their research, gather their data and put it to use. It is therefore realistic to assume that a great deal of that research data will come from web archives. The fact that our present websites are stored in the e-Depot is very reassuring, but it's not enough. We will have to do more. Active research will have to be conducted on how we can keep these sites accessible. Preserving the correct metadata so that people later on will be able to figure out what it is and how it should be presented is essential. Also, the presentation of a website depends to a great extent on the browser as well as the plug-ins needed for the presentation of specific aspects of a website (such as Flash, video and audio). Because of that, the KB is actively researching, and developing, techniques and methods that will be able to migrate or emulate[2] (old) digital objects so they can still be viewed on modern day computers. Of course, this will also mean that older browsers/viewers/plug-ins needs to be stored in some sort of software repository.

4 JHove and DROID tests

As a part of the ingest procedure, all digital objects should be validated before being stored in the e-Depot system. Although we might not be able (or willing) to correct possible errors found in the file, it is important to store as much of that metadata as possible. Because there are various digital projects being developed at the KB, each with their own specific file format(s) as output, the KB is currently working on a generic file validation tool/procedure where JHove and DROID are most probably going to be a part of.

To be able to see what metadata these tools can provide us with, we took 10 small- to medium sized websites and extracted all files from their ARC containers and had them identified/validated by DROID and JHove. These 10 websites were approximately 2.2 Gigabytes, consisted of 40.000 unique objects and were divided over about 110 different file types. Of those 110 different file types, only ten made up the majority of the 2.2 Gigabytes of data.

Below, in table 1, are the results of DROID:

			DROID		
% of total	**extension**	**identified as**	***Positive***	***Tentative***	***Not identified***
47%	html	htm / html	18408		77
25%	php	htm / html / php	9637	14	223
13%	jpg	jpg	4921		5
4%	gif		1563	4	5
3%	jsp	html / xhtml	1187		
2%	doc		664	1	
2%	xml		639		1
1%	pdf		313	47	
1%	png		335		
1%	txt		33	139	

Table 1

[2] http://dioscuri.sourceforge.net/

And in table 2, JHove's results are presented:

			JHove			
% of total	**extension**	**identified as**	***consistent***	***well-formed***	***valid***	***not well formed***
47%	html	htm / html				18485
25%	php	htm / html / php				9874
13%	jpg			1	4818	107
4%	gif				1549	23
3%	jsp	html / xhtml				1187
2%	doc		not supported by JHove			
2%	xml			3		637
1%	pdf			25	321	14
1%	png		not supported by JHove			
1%	txt				169	3

Table 2

The test with JHove is performed using its Audit Output Handler which causes JHove not to load a specific hul, or module, but will keep going through its available huls to try and validate a given file. When the file could not be validated as a specific format, JHove will label it as being a valid byte stream, which every digital object is, of course. So in table 2, where 18485 files are *not well formed*, it means they were *not well formed* as being html files.

The *Audit Output Handler* cannot tell anything about the (possible) errors in the file which we will need in order to take preservation actions, or do so at a later time. To get that metadata, an individual hul needs to be invoked against the file(s). Table 3 is a result of such a test whereby for example all files with an html, xhtml, php and jsp extension were validated with the HTML hul. The number in the first column is the number of occurrences of a specific error. The error itself is represented in the second column. The complete error messages have been reduced so they fit in the table below; originally, they contain details like at which location in the file the error occurred. A complete example of an error message from an html file is this:
ErrorMessage: TokenMgrError: Lexical error at line 57, column 36. Encountered: ")" (41), after : ""

HTML hul → *.html, *.xhtml, *.php, *.jsp	
361126	Construction with "/>" is incorrect except in XHTML
302369	Unknown tag
71460	Close tag without matching open tag
41585	Parsing error
20515	Tag illegal in context
18369	Undefined attribute for element
7319	Unrecognized or missing DOCTYPE declaration; validation continuing as HTML 3.2
4371	TokenMgrError
767	The reference to entity "task" must end with the ';' delimiter.
388	The processing instruction target matching "[xX][mM][lL]" is not allowed.
148	Document contains no html, head, body or title tags
92	PCData illegal in context
72	Attribute "type" is required and must be specified for element type "script".
52	Attribute "language" must be declared for element type "script".
20	The content of element type "html" must match "(head,body)".
12	Attribute "target" must be declared for element type "a".

8	Element type "script" must be declared.
6	The reference to entity "Itemid" must end with the ';' delimiter.
4	Attribute "border" must be declared for element type "img".
4	Attribute "name" must be declared for element type "img".
4	Parse error
4	The reference to entity "lang" must end with the ';' delimiter.
4	The reference to entity "mosform" must end with the ';' delimiter.
3	The reference to entity "act" must end with the ';' delimiter.
2	Attribute "vspace" must be declared for element type "img".
2	Attribute "width" must be declared for element type "td".
2	Document is empty
2	The content of element type "body" must match "(h1|h2|h3|h4|h5|h6|ul|ol|dl|p|div|rddl
2	The content of element type "head" must match "((meta|link|object)*,((title,(meta|link|object)*, (base,(meta|link|object)*)?)|(base,(meta|link|object)*,(title,(meta|link|object)*)))".
1	Attribute "height" must be declared for element type "td".
1	Attribute value "titlebar-west" of type ID must be unique within the document.
1	Document must have implicit or explicit HEAD element
1	The element type "img" must be terminated by the matching end-tag "</img>".
PDF hul → *.pdf	
152	Invalid destination object
16	Improperly formed date
12	No PDF header
6	Outline dictionary missing required entry
1	Invalid ID in trailer
1	Lexical error
1	Invalid character in hex string
XML hul → xml	
634	The element type "link" must be terminated by the matching end-tag "</link>".
40	White spaces are required between publicId and systemId.
3	File not found
GIF hul → gif	
24	Invalid GIF header
2	End of file reached without encountering Trailer block
JPEG hul → *.jpg, *.jpeg	
1	File does not begin with SPIFF, Exif or JFIF segment

Table 3

When looking at table 3, it is apparent that a lot of files from the web, especially html-like files, contain many errors. In order to develop a good preservation strategy, we will have to categorize the errors from the digital objects and assess the impact of it.

5 Future work / Questions

As a part of our *File Characterizing* project, we are making plans to develop a PDF JHove error/validation database which will tell us something about the impact of specific errors we encounter while validating digital objects. We would like to extend this work to HTML and are looking for collaboration: possibly within the IIPC preservation working group?

Below are listed a couple of questions related to digital preservation that arise from our work on archiving websites, and are curious if other organizations encountered them, and perhaps even have solutions for:

How to perform quality assurance on a harvested website?

At the moment, we manually check a couple of pages of the crawled websites to see if the representation has not change (too much) of the original website. Doing this for the entire website, especially for large ones, is very hard to do: it would take too much time. Another thing to do is examine log- and report files from Heritrix: check to see if there are a lot of 404 response codes, for example. Of course, a 404 could mean the webmaster has made a mistake somewhere, but it could also mean that the web server the file resided on was (temporarily) down but the document was indeed present at an earlier time.

What could be automated in question 1?

Obviously, the examination of Heritrix' log- and report files can be done automatically, but what about the first QA-tactic?

When do we reject a harvested website?

Or will we be archiving it no matter what the representation of the harvested material looks like? Or is there perhaps a turning point: if X number of files could not be validated, then ...

What preservation action to focus on (in the case of websites)?

Currently our main focus is on emulating websites once they cannot be viewed in conventional browsers. Migration could play a role in our digital preservation process, but only for a selection of file types, most probably not for entire websites.

Running times DROID JHove tests

Koninklijke Bibliotheek

In addition to the document: ***IIPC-PWG-Webarchiving-JHove-DROID-test.doc***

Online: https://wiki.nla.gov.au/download/attachments/15551/IIPC-PWG-Webarchiving-JHove-DROID-test.doc (19-Nov-2007)

September 2007

Bart Kiers bart.kiers@kb.nl

Some technical background on the machine these tests were performed on and the software used:

- Architecture: 64 bit Dual Core Intel Xeon, 3.0 GHz;
- OS: 64 bit RedHat Linux 4 ES;
- RAM: 4 GB;
- JRE: Sun Microsystem, version 1.6;
- JHove: Version 1.0;
- DROID: Version 1.1, signature file 12.

All times are measured by the operating system's **time** command instead of JHove or DROID's built in reports and are rounded to the nearest integer value.
All test were performed twice, right after each other and the measured time of the second test was taken into account. This was done to eliminate (possible) time differences the OS might have when starting up the Java Runtime Environment.
There were no specific modules invoked while running JHove's tests: it's *Audit Output Handler* was used.

website	*# files*	*# MB*	**JHove** *sec*	**DROID** *sec*
cbg.nl	4367	74	19	8
debibliotheken.nl	650	125	11	5
den.nl	1359	221	80	14
deverdiepingvannederland.nl	221	751	6	30
edusite.nl	17233	488	81	32
geheugenvanoost.nl	3988	109	224	9
huygensinstituut.nl	7057	115	125	10
museumboerhaave.nl	1239	63	30	6
tweedekamer.nl	2820	154	34	12
wsf.nl	238	19	13	3

There are two times that are notable:

- in case of the website *deverdiepingvannederland.nl*, DROID is about 5 times slower than JHove. After a close inspection it turned out that *deverdiepingvannederland.nl* had 18 large TIF files present. Running a test with only those TIF's through DROID, it seemed that these files were the cause of DROID being so "slow": it took DROID ~25 seconds to identify only those 18 files.
- in the case of *geheugenvanoost.nl* JHove was considerably slower: around ~25 times. This was caused by the great number of JPG's (2397) present in that website. Checking only those JPG's took JHove around 3 minutes.