

IBM
long

KB
term

preservation

study

010010

archiving
web
publications

00010





010010 archiving 00010
web
publications

Ir. Hans Verhoeven

01001011

01000010

Design: Steven L. Stijger
Published by: IBM Netherlands, Amsterdam
IBM / KB Long-Term Preservation Study
Report Series Editor: Dr. Raymond J. van Diessen

Available from:
IBM Global Services Netherlands
PO Box 9999
1000 CE Amsterdam
The Netherlands

Koninklijke Bibliotheek
PO Box 90407
2509 LK The Hague
The Netherlands

Title: **Archiving Web Publications**

ISBN: 90-6259-159-0
Author: Ir. Hans Verhoeven
Date: December 2002
Copyright: IBM / Koninklijke Bibliotheek

*This study was commissioned by the Koninklijke Bibliotheek,
National Library of the Netherlands*

01 IBM / KB

01000010

long-term preservation study

The National Library of the Netherlands (Koninklijke Bibliotheek, KB) is faced with the problem of preserving large amounts of digital documents for the long term. These documents come from two sources: from media published directly in digital form and from digitizing paper documents. In 2000, the KB and IBM started building an electronic deposit system ("Digital Information Archiving System or DIAS"), the technical core of the infrastructure for KB's e-Deposit for the Netherlands.

From the beginning it was clear that this project could not rely on out-of-the-box solutions alone because up to that time no solution readily addressed both the aspects of large volume and durable storage as well as the long-term preservation requirements. So an IBM / KB Long-Term Preservation Study (LTP Study) was initiated as part of the overall project of developing an electronic deposit system.

The primary objective of the LTP Study was to investigate the functionality required for the long-term preservation (hundreds of years) of the digital information stored in DIAS. This study has resulted in 6 reports: one overview report and five specific reports, each one addressing an important aspect of long-term preservation in its own right.

Participants in the LTP Study:

IBM

Raymond J. van Diessen
Raymond Lorie
Sidney Huiskamp
Hans Verhoeven

Koninklijke Bibliotheek

Johan F. Steenbakkens
Titia van der Werf-Davelaar
Patricia Alkhoven
Adriaan Lemmen

RAND Corporation

Jeff Rothenberg

British Library

Deborah Woodyard

I would like to thank all the participants for their input and enthusiasm. The results make an important contribution to the development and implementation of dedicated functionality for the long-term preservation of digital information and for guaranteeing long-term access.

Report Series Editor,
Raymond J. van Diessen

Titles of the Reports Series

Number 1: The Long-Term Preservation Study of the DNEP Project - an Overview of the Results

This report explains the reasons and objectives behind defining the LTP Study as part of the overall project to implement an electronic deposit system. It also provides a quick and general overview of all the study results, which are then elaborated on in the other published reports.

Number 2: Authenticity in a Digital Environment

Authenticity acquires a new meaning in a digital context. Normally objects are physical and their physical characteristics are the main source for defining authenticity. Moreover, authenticity is not a single concept, but involves different aspects that can be associated with an object:

- € A traceable path from the object's origin to its current ownership.
- € Measures and techniques for safeguarding against and/or recognizing modifications.
- € Techniques for establishing the use of original materials.

The problem of digital objects is that in fact they are just conceptual objects. A digital object is a conceptual object to be interpreted (rendered) by executing the digital object in a specific IT infrastructure (hardware & software). This report focuses on defining a framework in which we can define what is actually meant when one speaks of an authentic digital object.

Number 3: Preservation Requirements in a Deposit System

The initial DIAS release only provides basic functionality for preserving and rendering the stored digital objects for the long term. One of the primary responsibilities of the LTP Study is to define the functional requirements of the Preservation Subsystem, which is scheduled for development later. This report identifies requirements of the DIAS Preservation Subsystem so as to provide the services and functions for monitoring the technical environment associated with the digital objects stored in DIAS.

The Preservation Subsystem can be summarized by the following three objectives:

- € Identifying digital objects that are in danger of becoming inaccessible because of changes in technology.
- € Implementing the activities associated with technical preservation.
- € Supplying the requisite technical metadata in order to generate / validate the environments needed during digital object delivery.

Number 4: The UVC: a Method for Preserving Digital Documents - Proof of Concept

Within IBM Research in Almaden, Raymond Lorie was already working on a combined emulation / migration approach to preserve a certain class of digital objects with an approach called the Universal Virtual Computer (UVC).

The main idea consists of archiving a program P along with the data file that decodes the data and returns the information to a future client based on a logical view. The logical view of the data is simple and self-contained enough to be interpreted without any specific software or hardware. Program P is written for the Universal Virtual Computer (UVC) that is general, yet basic enough to continue to be relevant in the future. Given the simplicity of the UVC, it will be relatively easy to write an emulator of the UVC in the future on a real machine of that time. The emulated machine will run the program P and return all data in an easy to understand logical view of the data.

The LTP Study conducted a proof of concept with the KB to test the UVC approach in a library environment. The PDF format was selected because it is the primary data format for electronic publications to be stored in DIAS.

Number 5: Managing Media Migration in a Deposit System

Storage technology obsolescence makes media migration a necessity. Data has to be copied from one storage medium to another on a regular basis. However, the fact that storage technology becomes obsolete is not the only trigger for rewriting previously stored digital objects. All storage media degrade over time and have to be rewritten either on the same medium (refreshing) or on another medium (migration).

Ordinarily media refreshment / migration would be a straightforward process. However, the large amounts of storage associated with an electronic deposit system introduce certain volume-specific requirements. Most electronic deposit systems define their storage capacity needs in several TeraBytes (10^{12} Bytes). Take a deposit system with 100 TeraBytes of information stored on tape, for example. Let's assume that you want to migrate all this information to an optical storage medium. Current optical storage media have a capacity of around 5 GigaBytes and a write speed of around 4 MegaBytes/second. A quick calculation shows that a complete migration to optical storage would take at least 290 days (100 TeraBytes / 4 MegaBytes per second)!

This report describes the actions to be taken to manage media migration / refreshment effectively within an electronic deposit system, focussing specifically on the media migration issues within DIAS. Potential additional capacity required for media migration might be created by redundancy and parallelism.

Number 6: Archiving Web Publications

More and more Web publications are becoming a primary source of information and will thus be stored as digital objects in DIAS. Web publications have specific characteristics and requirements that DIAS must meet if they are to be archived successfully.

This report investigates the issues and requirements introduced by archiving Web publications and their potential impact on DIAS.

1/	Summary	1
2/	Introduction	3
	2.1 Identification	3
	2.2 Purpose	3
	2.3 Scope	4
	2.4 Assumptions	4
	2.5 Document Overview	4
3/	Web Concepts & Standards	5
	3.1 The Web	5
	3.2 Communication Protocols	7
	3.3 Uniform Resource Identifiers	7
	3.4 Domain Name versus Host IP Address	8
	3.5 Mark-Up Language	8
	3.6 Content Types	10
	3.7 Web Page Types	11
4/	Web Harvasting Tools	15
	4.1 NEDLIB Harvester	15
	4.2 HTTrack Web Site Copier	16
	4.3 IBM Web Crawler	17
	4.4 MetaProducts Offline Explorer	17
	4.5 Commonalities among Web Capturers	18
5/	Web Archiving in the Context of DIAS	19
	5.1 Creating Web Snapshots	19
	5.2 Web Archiving Business Processes	20
6/	Web Archiving Impact on DIAS	23
	6.1 Granularity of the Archiving	23
	6.2 Version Management	24
	6.3 Manipulation of the Original URIs	25
	6.4 Examples	26
7/	Web Preservation Layer Models	33
8/	Conclusions	37
9/	Recommendations	39
	Appendix A: References	41
	Appendix B: Glossary	43
	Appendix C: Issues	45

//

summary



Deposit libraries like the National Library of the Netherlands (Koninklijke Bibliotheek, KB) are increasingly confronted not only with electronic publications created by the conventional publishers but by publications that are published directly on the World Wide Web (the Internet) as well. These days with only a minimal investment anyone can be a publisher. The Web is increasingly becoming a place that contains a wealth of information, which is of interest to organizations like the KB who have the responsibility of preserving a nation's cultural heritage with regard to published information in all forms. These so-called Web publications impose additional requirements on an electronic deposit system.

The granularity used to archive the Internet content influences the way Web publications must be ingested into the electronic deposit system can be accessed. Web snapshots can be archived in a number of ways into an Archival Information Package (AIP):

1. Each Web snapshot can be archived into one AIP.
2. Each Web snapshot contains a number of domains with relations and each domain has its own AIP.
3. Each Web snapshot contains a number of Web sites that have mutual relationships and each Web site has its own AIP.
4. Each Web snapshot contains a number of URLs with mutual relationships and each URL has its own AIP.

Versioning is another potential problem area due to the fact that there is no uniform way to manage the references to outside controlled links. This lack of ability to trace changes with regard to external references makes it hard to assess whether or not the reference is still valid.

The Web archiving study focused on the impact of archiving Web pages in the KB's Digital Information Archiving System (DIAS) and the associated long-term preservation requirements. For this purpose a number of generally available Web Harvesting applications were studied for potential use in gathering the Web pages. Most of these harvesters are developed to provide personal off-line browsing facilities. The only exception is the NEDLIB Harvester that was specifically developed for large scale Web harvesting. However, none of the tools provides a full Web archiving solution. A full Web archiving solution would provide support for collecting, archiving, accessing and preserving Web assets.

This report identifies the major issues related to Web archiving. The initial conclusions have been augmented with some recommendations regarding the archiving of Web pages within DIAS. Once the first increment of the Preservation Subsystem has been implemented, more practical experiments will have to be conducted to decide on the best methods for archiving the Web. In particular, the archiving of "dynamic" Web pages produced on request by specialized content management systems requires further investigation. Some difficult challenges regarding archiving the Web are still unresolved and will require the aid of some innovative new technologies and techniques.

2/ 01 introduction 1000010

2.1 Identification

This document contains the results of the Long-Term Preservation (LTP) Web Archiving Study. This study is one of the areas investigated during the LTP Study, which is part of the DNEP project for implementing the deposit system called Digital Information Archival System (DIAS) for the Koninklijke Bibliotheek (KB) in The Netherlands. The study will address issues related to the archiving of static Web pages in general and specifically the potential impact on DIAS.

The following topics have been identified as relevant to the archiving of static Web pages:

- € *What impacts will archiving the Web have on DIAS?*
- € *How should version control be dealt with?*
- € *What Preservation Layer Models (PLM) are required for the Web archive?*
- € *What business processes are required to implement Web archiving?*
- € *What are the possible organizational consequences?*

2.2 Purpose

The following topics have been identified as relevant to the archiving of static Web pages:

- € What impacts will archiving the Web have on DIAS?
- € How should version control be dealt with?
- € What Preservation Layer Models (PLM) are required for the Web archive?
- € What business processes are required to implement Web archiving?
- € What are the possible organizational consequences?

2.3 Scope

The study will address issues related to the archiving of static Web pages in general and specifically the potential impact on DIAS. The study will not cover archiving dynamic Web pages nor will it address Web sites that require user intervention (interactive sites). However, many useful remarks will be made regarding this. The research area of Web archiving is largely uncharted. Therefore not all problems can be examined in complete detail. Nevertheless, we expect that the view of the problem offered will provide new impulses to additional areas of investigation that may be requested by the KB.

2.4 Assumptions

The following assumptions are made:

- ∄ A suitable Web capture tool exists that produces the original Web pages and their related content (resources) in a useful structure.
- ∄ Capacity and performance can be regarded as parameters that depend on time and budget and can be solved.
- ∄ Bibliographic metadata is stored in the catalogue (KB Catalogue system) system. The KB Catalogue system is the central search portal to Assets that are available at the KB. The KB Catalogue system has a Web interface.
- ∄ Interfacing with the KB Catalogue system works in the following way. When a consumer selects an asset from a search results page, a request for a National Bibliography Number (NBN) is sent to the deposit system (DIAS). The deposit system responds one of a number of ways:
 - ∄ A list of possible versions that are available for the NBN.
 - ∄ A redirection to a Reference Workstation where the asset will be made available.
 - ∄ The single asset if only one AIP version exists.
- ∄ The deposit system (DIAS) only contains limited technical metadata which is packaged in the Table of Contents (ToC) of the AIP [Diessen 2002].
- ∄ The KB Catalogue system is separated from the deposit system (DIAS).

2.5 Document Overview

The report starts with an introduction (a recap) of Web concepts and standards, which provides the foundation of the problem. This is followed by a brief overview of the most popular Web harvesting tools. The knowledge of Web concepts and Web harvesting tools is then applied within the DIAS context to assess the potential impact. This results in a number of possible do's and don'ts as well as possible solutions.

3/ Web concepts & standards

3.1 The Web

The World Wide Web (the Internet) is a distributed collection of Web sites that are accessible over the entire world. The Web sites contain collections of Web pages that are written in mark-up language (e.g. HTML) and that contain references (i.e. links) to other Web pages and references (i.e. also links) to content (resources) for the Web page.

A very simplified example of the Internet is shown in Figure 3.1 and explained below.

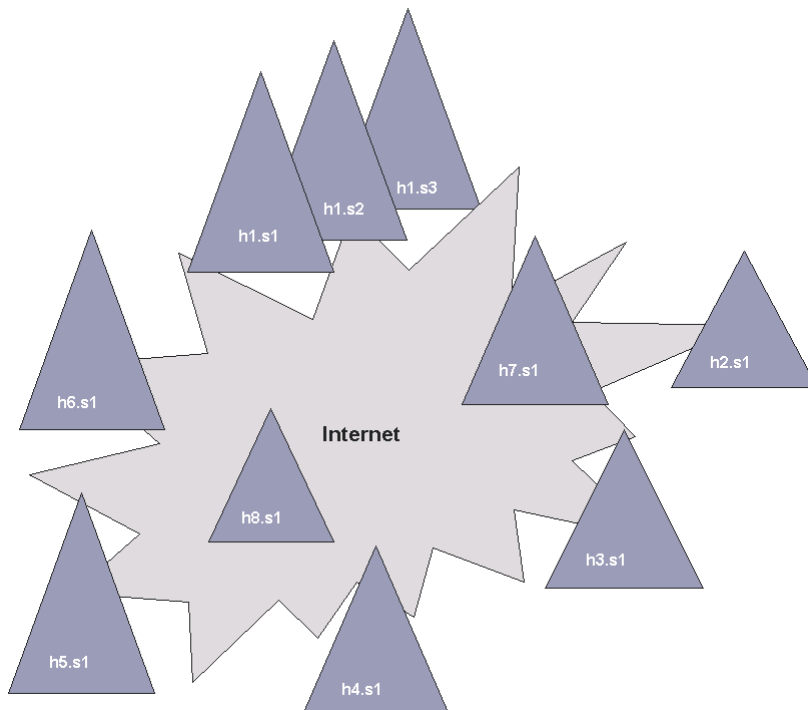


Figure 3.1 / The internet

The figure shows the Internet with a number of Web sites ($h\langle i \rangle.s\langle j \rangle$). A specific Web site is uniquely identified by a domain name that translates to a host system ($h\langle i \rangle$) and a Web site name ($s\langle j \rangle$). Quite often the Web site name is simply '/' which stands for the root directory but it can also have a name (i.e. www.kb.nl/NEDLIB). The Figure shows that host h1 has three Web sites, named h1/s1, h1/s2 and h1/s3. The other hosts only have a single Web site. A Web site can be depicted by a triangle representing the collection of directories and files (resources) that make up the site.

Note that Web pages can link to other Web pages and objects (resources) of other Web sites. This is illustrated in Figure 3.2.

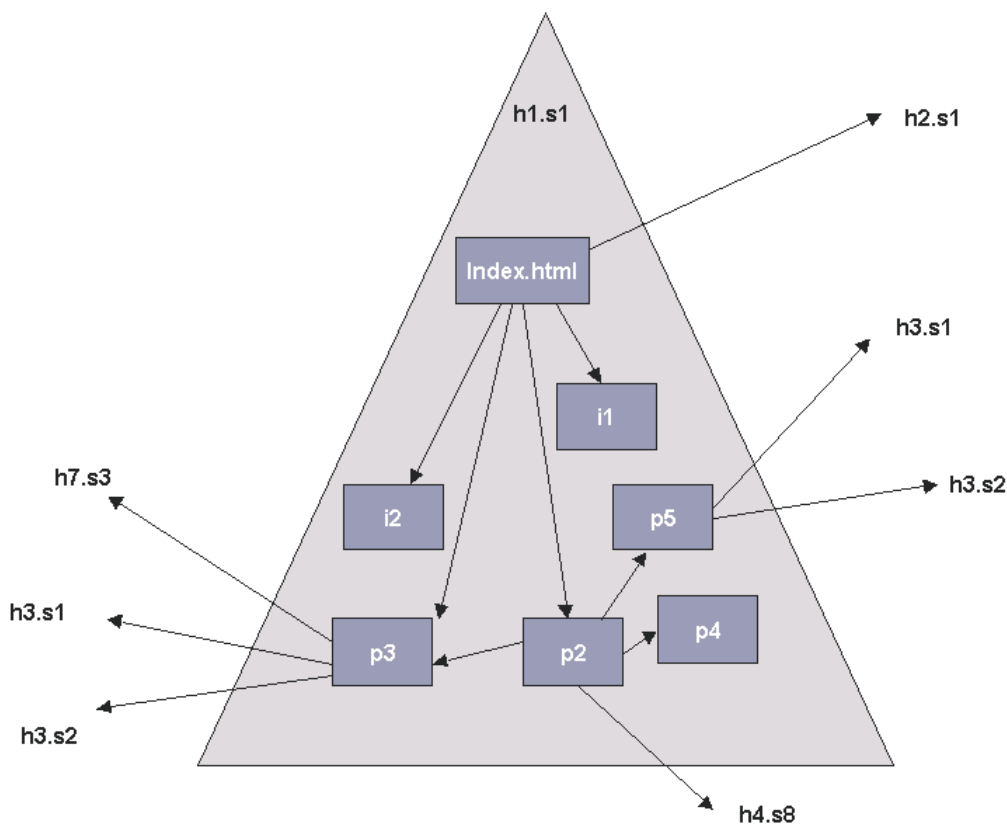


Figure 3.2 / Web site with Web pages with internal and external links

The Web pages can be accessed using an HTML Viewer (e.g. Netscape, Explorer). The Browser software runs on (client) computers. The Web sites are stored on (server) computers that are distributed throughout the world.

Web browsers on the client workstations connect to the HTTP Web server using the TCP/P communication protocol. In addition to the Internet, there are also intranets and extranets. These follow the same principles as the Internet but are dedicated for internal use (i.e. company-wide) and often have restricted access rights.

3.2 Communication Protocols

The Hypertext Transfer Protocol (HTTP) is the communication protocol that is most frequently used on the Internet [Internet Society 1999]. The HTTP protocol has existed since 1990. The first version was HTTP/0.9, which was later improved into HTTP/1.0. The current version is HTTP/1.1.

The HTTP protocol consists of request and response messages. A client requests a resource via an HTTP request message. The server returns the resource in an HTTP response message that includes a result status code.

File Transfer Protocol (FTP) is also frequently used. It has less overhead than the HTTP protocol and as such can be more efficient for transferring large objects.

3.3 Uniform Resource Identifiers

Uniform Resource Identifiers (URIs) are known under many names:

- ∉ World-Wide-Web (www) addresses.
- ∉ Universal Document Identifiers.
- ∉ Universal Resource Identifiers.
- ∉ Combination of Uniform Resource Locators (URL) and Uniform Resource Names (URN).

The generic syntax of Uniform Resource Identifiers is defined in RFC2396 (see <http://www.ietf.org>). A URI provides a simple and extensible means for identifying a resource. A resource can be anything that has an identity, for example an electronic document or image, a service or a collection of other resources.

For the HTTP protocol a URI is a formatted string that identifies a resource using a name, location or any other characteristic.

A request for a resource via the HTTP Protocol is defined in a `http_URL` (e.g. `http://www.ibm.com`) as follows:

```
http_URL ::= "http:" "/" <host> [:<port>] [<abs_path> [ "?" query ]]
```

If the port is not specified the address defaults to port 80. The identified resource is located at the host machine listening for TCP (Transmission Control Protocol) connections on the specified port on that host machine. The URI for the requested resource is `abs_path`, which is an absolute path name for the resource. If `abs_path` is not specified the address defaults to `/"` (the root directory).

The host machine is most often identified using the Domain Name System (DNS), which is able to translate a domain name (i.e. `kb.nl`) into a host system IP (Internet Protocol) address.

3.4 Domain Name versus Host IP Address

A Web site can have one or more Internet Domain Names associated with it. For example, the domain name for the Koninklijke Bibliotheek Web site is "kb.nl". The translation of domain names as to the Internet Protocol (IP) address is performed with the Domain Name System (DNS). The IP address identifies the host computer that provides the Web site. The IP address is currently 4 bytes but will change to 6 bytes in the future (IPv6). An example of an IP(v4) address is: "192.87.222.200". Note that the translation from domain name to host IP address offers flexibility. It is possible to move a site to a different computer system that has a different IP address without changing the domain name. Domain names must be registered (and are maintained) via Internet Standards Committees. For the Netherlands, this committee is "Stichting Internet Domeinregistratie Nederland" (<http://www.sidn.nl>).

Different domain names can translate to the same IP address. This technique is used frequently by Web hosting companies. They host Web sites for different companies on the same (virtual) host.

Domain names are divided into two major categories: country specific domains and other (generic) domains. The country- specific domains are the domains that end in .nl, .ge, .uk, .fr, etc. (ISO 3166). The other domains are the domains that end in .gov, .com, .org, .edu, .mil, etc. Domain names are resolved from right to left. This means that within the .nl domain there are a number of sub-domains, such as kb.nl, dnep.kb.nl, ns.nl, uni-maastricht.nl, each of which can have additional sub-domains. A detailed explanation of the domain name system and the resolution of domain names into IP host addresses can be found in Computer Networks [Tanenbaum 1990].

3.5 Mark-Up Language

Web pages are written in Hypertext Mark-up Language (HTML). The HTML defines how a Web page must be rendered by the Browser and displayed on the computer screen. HTML has a version number. Continued development of HTML has resulted in versions 2.0, 3.2 and 4.0. The latest standard is version 4.01 [Raggett and Le Hors 1999].

Unfortunately, not all Browsers render HTML the same way. There are differences between displayed pages in Microsoft Internet Explorer and Netscape Navigator Browsers, for instance. Moreover, browser manufacturers often define their own extensions to HTML specifically for their own browser which sometimes makes it impossible to access or view a Web site with a browser other than the one intended.

HTML contains tags. These tags are like <H1> and </H1>. Some of these tags contain a reference to content that is to be placed on that Web page.

For HTML 4.0 the following tags contain references to content or to other sites:

- € hypertextlink.
- € <BODY background="background-imagefile-pathname">...</BODY>.
- € imagelink.
- € <MAP ...><AREA ... href="link-reference" ...> </MAP>.

€ <FRAME src="Web-frame-to-display" >.
€ <FORM action="link-reference" method="post">...</FORM>.
€ <LINK href="link-reference" ...>.

The link references contain a URI in either absolute form or in a form relative to some base URI. An absolute URI always contains the scheme (protocol) name followed by a colon (i.e. http://). The relative URIs can be as simple as the name of a resource, which indicates that the resource is located in the same location as the container resource, or via directory path names. It is also possible to have URIs link to anchors within the current page (i.e. href="thisPage#PartTwo"). The linked text in the Web page must then be marked with the Anchor tag with the name attribute (i.e.).

In order for a Web page to be displayed completely all content linked to using page links must be collected. A Web page is dependent on these resources

A Web page consists of the page made up of HTML Mark-up constructs and can contain a number of URIs (page links) that are required in order to construct the contents of the page (for instance images). A Web page can also contain URIs that link to other Web pages (navigational links). Both types of links can link to internal resources (local to the current domain, within the Web site or to another Web site within the domain) or to external resources (on Web sites of other domains).

In order for a Web page to be displayed completely all content linked to using page links must be collected. A Web page is dependent on these resources. The navigational links are not required to display the page completely. The navigational links are only required for navigation, i.e. when the URI link is selected (clicked). This is illustrated in Figure 3.3. Resources 1, 2, 3 and the external resources are required to display Page 1 and are therefore shown with dotted lines indicating dependent relationships. The relationships with Page 2 and 3 are shown as associations. It is possible to navigate from Page 1 to Page 2, 3 and external pages.

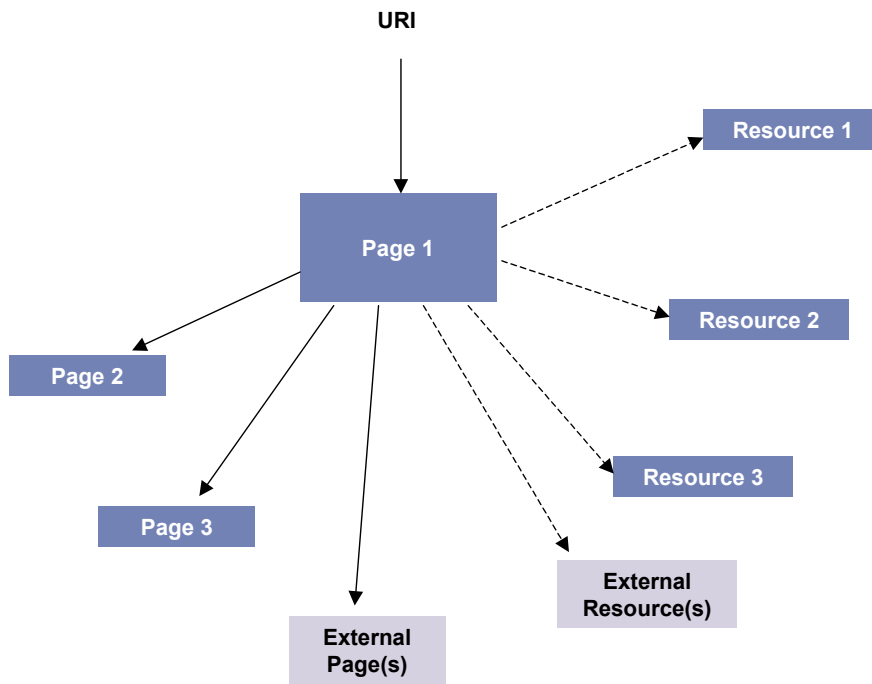


Figure 3.3 / URI dependencies vs. navigation

3.6 Content Types

There are a number of built-in content types that are supported by modern Browsers. These standard content types are:

- ∉ htm, html.
- ∉ jpeg, jpg.
- ∉ gif.
- ∉ text.

The number of content types allowed can be expanded using plug-ins. A plug-in is a program that supports a specific content type. Examples are Adobe Acrobat Reader for viewing documents in Portable Document Format (PDF), Macromedia Flash Player for playing Shockwave Flash Objects (SWF) and LizardTech MrSID Viewer for viewing MrSID Images (SID).

3.7 Web Page Types

Web pages can be divided into the following types:

∄ **Static**

Such Web pages are hard (HTML) coded. They do not contain any dynamically generated parts. They are stored on the Web server as is. These pages are typically files with an extension .htm or .HTML. The Web page cannot be changed via input parameters. If there are input parameters these will be ignored. The Web page has a single URI. When the system detects that such a Web page has changed, this indicates that a new version of the page has been published.

∄ **Dynamic**

These Web pages are generated, via some sort of server-side processing (for instance Common Gateway Interface (CGI) Scripts, Perl Scripts, Java Servlets, etc.). These generated Web pages may be dependent on the specified input parameters. These Web pages may also be (re)generated because of (server-side) data changes that must be distributed. If the resulting Web page only depends on the input parameters then these are similar to the static Web pages. It is fairly simple to detect when a new version of the Web page has been published. Web pages that are different on each invocation as the result of some unknown event are the most difficult. The criteria for deciding that a new version of the Web page has been published are hard to identify.

Static and dynamic Web pages appear in two forms:

- ∄ Automatic, i.e. Web pages that require no user interaction.
- ∄ Interactive, i.e. Web pages that do require or even expect user input.

Automatic Web pages are predictable in the sense that they do not depend on unknown user input. They contain links to other Web pages. Sometimes these links are hard to figure out because they are dynamically constructed as the result of client-side processing (i.e. a JavaScript running within the Browser and constructing menu items).

Interactive Web pages can be unpredictable. They comprise complete Web applications. A specific Web page that requires user input can be captured but it may be hard to find out what other Web pages are referred to and how (i.e. with what parameters) these will be invoked. The Web page sequences that are the result of navigating through the Web pages may be dependent on data inputs and selections made by the user. It is almost impossible to sort out all possible inputs for the URIs referenced in an interactive Web page.

All of this results in a number of Web page types that are discussed in the following paragraphs.

3.7.1 Static Web Pages

Static Web pages are the easiest form of Web pages. The Web page is requested by its URI. The result of requesting the URI is always the same. The result only changes if a new version of the Web page is published.

Note that these Web pages can contain embedded resources that give the impression that the page is dynamic, because the page contains specific resources like animated gif images or shockwave flash objects.

Interactive Web pages can be unpredictable. They comprise complete Web applications. A specific Web page that requires user input can be captured but it may be hard to find out what other Web pages are referred to and how (i.e. with what parameters) these will be invoked. The Web page sequences that are the result of navigating through the Web pages may be dependent on data inputs and selections made by the user.

These Web pages may request user interaction via HTML forms, Java Applets, ActiveX and other client-side logic. The user should provide input for some entry fields. The HTML form is posted to the HTTP server with the click of a submit button. It can be difficult to capture the Web page that will be returned after a successful submit action.

3.7.2 Dynamic Web Pages

Dynamic Web pages are generated and therefore more complex. The generated content depends on the input parameters, the result of two different invocations may be the same. However, if the content depends on internal data, the output may be different every time.

3.7.3 Web Pages and Parameters

Web pages can depend on the value of parameters that are specified in URI's query string. The following URL shows an example of this:

```
http://www.example.bl/aWebpage?arg1=val1&arg2=val2
```

The resulting Web (HTML) page (the document) can be different for individual values (val1 and val2) of the arguments (arg1 and arg2).

It is important to note that the Web page that is returned based on specific argument values will be the same unless the content for the page is dynamically generated.

One URI results in one document (Web page). The drawback to parameters is that each URI with specific parameter values must be harvested separately.

3.7.4 Protected Web Pages

Some Web sites (or Web pages) require user identification. Before the system returns the result (the Web page) of the requested URI, the user must enter his (or her) user name and password. The Web page for the requested URI will only be returned if the login is successful.

The Web page that is returned for a URI will be the same for each user that can successfully log on. Another Web page will be returned if the logon is unsuccessful.

3.7.5 Personalized Web Pages

Personalized Web pages are an extension of protected Web pages. Personalization requires user identification. Some user preferences, which are associated with the user identity, can be stored. When the user requests a URI and has successfully identified himself (or herself) a specific (home) Web page is constructed based on these user preferences.

The effect of personalization is that each user can have a different Web page for the same URI.

3.7.6 Cookie-Dependent Web Pages

Some Web pages use 'cookie' mechanisms. A cookie is a small piece of information that can be persistent (retain its value between subsequent invocations). Persistent Cookies are stored on the client system. Cookies have a number of applications:

- ∄ It is used to 'remember' the path through a Web application that a user has followed.
- ∄ It retains selections that a user has made or stores user preferences.

The requested Web page may depend on the value of the cookie.

Cookies can cause unpredictable variations in the Web page for the same URI. However, a request based on a given set of cookies will always return the same Web page.

4/ Web harvesting tools

This chapter briefly describes the most popular Web archiving tools available today. It is not intended to be a complete investigation into the functionality of these tools, nor is it a selection of the most appropriate tool to handle the job. The purpose is to understand how these tools function because the resulting output is input for Web archiving in DIAS.

4.1 NEDLIB Harvester

The NEDLIB Harvester robot is a freeware application built to collect Web documents and store them as archive files. The harvester has been developed as part of the NEDLIB project funded by the European Union. The harvester is implemented in the C programming language. The Center for Scientific Computing (CSC) Finland is author of this tool, which is still being improved. The latest version is version 1.2 of October 2001.

The harvester stores the collected Web pages as flat files with a unique name based on the MD5 checksum. Metadata is stored in another file (a file with extension .meta). Yet another file (access<NO>.data) is created to provide access. The files are initially stored in a directory structure. The harvester creates a new directory each day and creates numbered sub-directories for every 10,000 files. The captured files for a day can be packed into a UNIX tar archive with the packer script. The harvester uses a database (MySQL) to store schedules, lists of interesting URLs, etc.

A number of National Libraries, including the KB, have evaluated the NEDLIB Harvester in a number of harvesting experiments. These experiments have discovered some interesting statistical results. These results are documented in the NEDLIB Demonstrator Report [Sijtsma 2002].

Some interesting figures:

- € The size of most Web sites is limited (more than 60% of all sites have less than 10 URLs of content).
- € Most Web pages are small (i.e. 10 KiloBytes on average).
- € Each Web page contains an average of 3 embedded resources.
- € The Dutch domain (.nl) is estimated to currently contain 20 million Web pages.
- € The Dutch domain (.nl) covers 90% of the Dutch part of the Internet, 10% consists of other domains.
- € A single full snapshot of the Dutch part of the Internet may contain as many as 10,000 Web sites.

- € Most content types (i.e. 98%) are types that are supported by standards (HTML, gif, jpg, text).
- € The KB's NL Menu service (The Dutch Web Index since 1992) has 30,000 registered Dutch companies (i.e. registered with the Chamber of Commerce ("Kamers van Koophandel") that can be used as a starting point for harvesting the Dutch part of the Internet.

Note that these figures will change in future as the Internet becomes more popular. Before making any real conclusions the figures must be checked.

However, some conclusions are interesting. A single full Web snapshot (20 million pages) would consume (only) 200 GigaBytes when we assume the average page size to be 10 KiloBytes (embedded resources are included in the average page size).

The NEDLIB manual [NEDLIB 2001] indicates that the harvester collects 10 URLs/sec on average. For the full Web snapshot containing 20 million pages with an average of 3 embedded resources per page it would take 69 days to complete.

The Nordic National Libraries are conducting a project (Nordic Web Archive) to develop an access module on the archive built by the NEDLIB harvester (<http://nwa.nb.no>).

4.2 HTTrack Web Site Copier

The HTTrack Web Site Copier, Offline Browser and Mirroring utility is a free product. The current version is 3.08. It is a program written in the C programming language. It is available for Windows (95/98/NT/2000) and Unix platforms. The Windows version (WinHTTrack) has a Graphical User Interface (GUI). This GUI makes it very easy to define URLs that have to be collected and to configure all kind of settings that influence the collection process.

The following list contains the most important features of the product:

- € Download (mirror) Web sites for offline viewing.
- € Update mirrored sites.
- € Resume interrupted downloads.
- € Powerful Scan Rules for including/excluding URLs, links or file types (with advanced wildcards).
- € Set structure depth, file size, site size.
- € Various options for local directory structure; default structure follows the site structure.
- € Set the start time for downloading.
- € Respects robots.txt rules, but can override to ignore.
- € Can use stored cookies.
- € Multiple connections for maximum download speed.
- € Proxy support with optional authentication.
- € Timeout and minimum transfer rate manager.
- € Clear error logging.

HTTrack changes the URL links in the original pages to relative links in a local directory structure. The advantage is that the mirrored sites directory structure may be copied to other places and can still be browsed.

4.3 IBM Web Crawler

The IBM Web Crawler is included in the Information Mining feature of version 7.1 of IBM Enterprise Information Portal (EIP). The EIP Information Mining feature is only available for the Windows (NT/2000) platform. The Web Crawler is a robot that monitors a particular area on the Web, specified by a user-defined set of addresses (the Web Space). The Web Crawler keeps a current copy of every object in the Web space on disk, ready to be viewed or indexed with the IBM Text Search Engine, which is a feature of IBM Content Manager.

The IBM Web Crawler has the following features:

- ⊘ Extensive customization capabilities is comprehensive and easy.
- ⊘ Runs multiple Web Crawlers in parallel.
- ⊘ Retrieves objects of any content type and language (these include HTML, text, image, audio and video).
- ⊘ Contains a tool kit for individual development purposes.
- ⊘ Can restrict retrieval to specific domains.
- ⊘ Stores Metadata in DB2 Relational Database Management System (RDBMS);
- ⊘ Respects robots.txt rules.
- ⊘ Monitors changes to Web pages.
- ⊘ Notifies external Subsystems of the creation of new objects and changes to existing objects.
- ⊘ Places collected objects in a region on the disk in a structure similar to the sites.
- ⊘ Reviews rapidly changing objects more frequently.
- ⊘ Logs unusual events.

The crawled documents can be imported into the EIP Information Mining application. New documents are imported and made searchable; previously imported documents are updated.

4.4 MetaProducts Offline Explorer

MetaProducts Offline Explorer is a Windows 9x/NT/2000/ME tool that is able to download an unlimited number of Web, FTP and HTTPS sites to local disk for later offline viewing, editing or browsing.

Offline Explorer (version 1.9) is available in three editions: Offline Explorer, Offline Explorer Pro and Offline Explorer Enterprise. The latter has OLE automation support and the ability to perform huge Web downloads. This offers the ability to build custom browsing/downloading solutions.

Some of the features Offline Explorer offers are:

- ⊘ Allows the user to selectively include (or exclude) individual servers, directories, and files using only keywords.
- ⊘ Has an excellent user interface.
- ⊘ Is one of the fastest known Web site download tools.
- ⊘ Supports industry-standard technologies, such as FTP, different proxy servers, Macromedia Flash, Cookies.

- ⊄ Has a built-in internal HTTP server that allows to share downloaded files over an Intranet;
- ⊄ Has an export feature that allows to users to copy downloaded sites in different formats to other locations.

4.5 Commonalities among Web Capturers

This paragraph summarizes the findings on the functionality of the Web harvesting tools from the previous paragraphs and concludes with a number of common categories.

Web capturers are developed to provide off-line browsing facilities. The only exception is the NEDLIB Harvester that is specifically developed for Web harvesting. None of the tools provides a full Web archiving solution. A full Web archiving solution would provide support for: collecting, archiving, accessing and preserving of Web assets.

No capturer has functionality for specifying the capture schedule per URL.

Almost all capturers have problems with pages containing complex java scripts or applets. Dynamically generated pages (.cgi, .asp, .php3) also cause significant problems.

Common functionalities:

- ⊄ Defining URLs to include/exclude.
- ⊄ Defining pattern matching rules for scanning (what links to follow or skip).
- ⊄ Defining number of links to follow (width and depth).
- ⊄ Respecting robots.txt rules.

Other useful functionalities could include:

- ⊄ Defining the number of parallel connections.
- ⊄ Defining the maximum object size (page, file, site).
- ⊄ Defining the time to run.
- ⊄ Defining the structure of the collected objects on disk.
- ⊄ Supporting FTP protocol via FTP Proxy.
- ⊄ Notifying listeners (client applications that have subscribed) of any changed URIs.

Web capturers are developed to provide off-line browsing facilities. The only exception to this is the NEDLIB Harvester that is specifically developed for Web harvesting. None of the tools provides a full Web archiving solution. A full Web archiving solution would provide support for: collecting, archiving, providing access and preserving of Web assets.

5/ Web archiving in the context of DIAS

5.1 Creating Web Snapshots

Once a suitable tool for Web harvesting has been selected the moments in time at which a Web snapshot can be assumed to be complete must be defined. Since it may take a long time to harvest all the selected sites, the snapshot will always be incomplete and inconsistent.

If it were really possible to take snapshots, then the result of three snapshots would be as depicted in Figure 5.1.

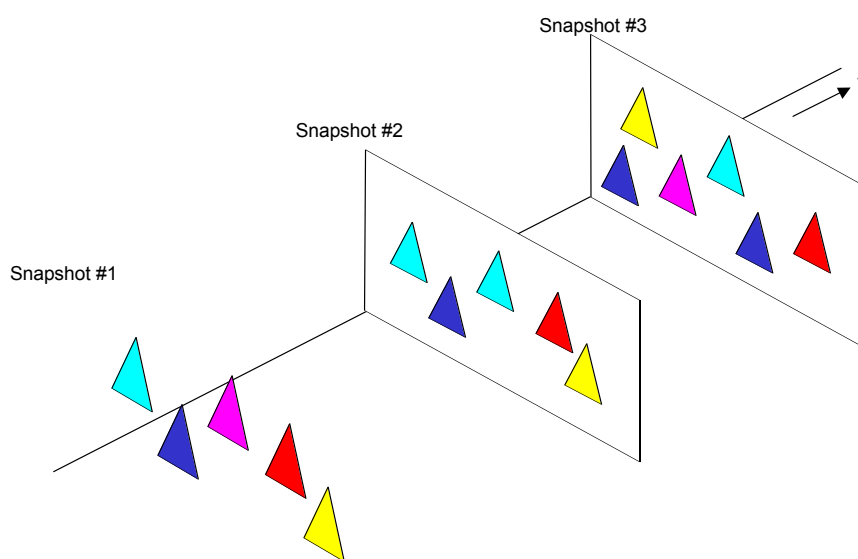


Figure 5.1 / Web snapshots in time

These snapshots may have relations with one other (shown via colors). From such relations it is possible to derive the development of a Web site included in snapshot#1 from its instance in snapshot#2, for example.

Relations may also exist because of the mechanism of taking snapshots. The snapshots can be full snapshots or incremental snapshots. Full snapshots contain all objects. Incremental snapshots only contain those objects that have changed since the previous snapshot.

If incremental snapshots are taken it will be more difficult and time consuming to restore the requested Web snapshots. Of course, the Web snapshot will require less archive space (disk, optical or tape).

5.2 Web Archiving Business Processes

The Web archiving of business processes will be described in the context of the Reference Model for an Open Archival Information System (OAIS) [CCSDS 2001]. The OAIS Reference Model with the DIAS extensions for Delivery&Capture and Packaging&Delivery is shown in Figure 5.2.

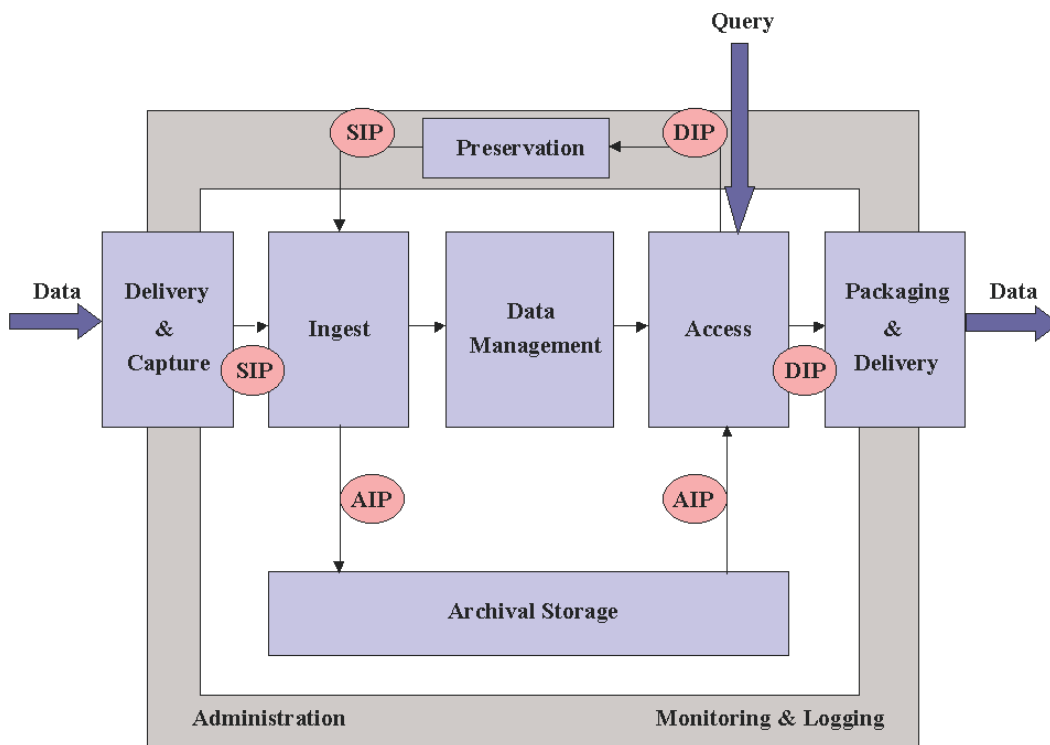


Figure 5.2 / The DIAS extended OAIS reference model

5.2.1 Delivery&Capture

This OAIS functional entity is responsible for collecting the required Web pages and related objects (resources). The collection process should be as automatic as possible. The pages and objects to be collected follows from lists of included and excluded Uniform Resource Identifiers (URI). These lists have to be administered by Topic Experts. These Experts are able to perform content selection. The KB's Dutch Electronic Search Services (DuchESS) perform similar services. How often each URI should be checked for changes and how deep the internal links must be followed should be defined for each URI. The Topic Experts should also check URIs that the harvester failed to collect.

After collection there must be a differentiator that determines what URIs have changed and prepares those changed objects for submission. The result of the preparation will be a Submission Information Package (SIP). The question of how to update the KB Catalogue System with the correct bibliographic information is related to the creation of the SIP that goes into DIAS. Another question is whether the new pages have to be included in a text search index and how this index can be searched.

Preparation has to deal with possibly changing the original URIs in such a way that, when requested, they will be retrieved from the archive system instead of the Internet. Possible manipulations of original URIs will be addressed in section 6.3 Manipulation of the Original URIs.

Another important question involves the number of objects that will be combined within a single SIP. This will be investigated in section 6.1 Granularity of the Archiving.

5.2.2 Ingest

The OAIS Ingest functional entity extracts Submission Information Packages (SIP) and creates Archival Information Packages (AIP) and associated Metadata for each package. If an SIP has been ingested by the Preservation functional entity, Ingest will maintain the relationship between the AIP containing the original asset and the AIP containing the converted Asset. An SIP containing a converted asset includes a reference to the AIP with the original Asset.

If all Assets that have to be archived for a Web snapshot are ingested as a single entity, no changes to the Ingest functionality are required. Ingest simply creates a single AIP with Metadata for all information (the whole Web snapshot) in the SIP.

However, it is more likely that Web snapshots will be archived as a number of related AIPs. In that case Ingest will have to be changed in such a way that it checks, creates and updates these relations. Whether or not this is required depends on the granularity with which the Web objects will be stored.

5.2.3 Data Management

Data Management stores and is able to retrieve metadata for the Web snapshots. It must be able to store relationships between Web snapshots, domains (and hosts), Web sites, Web pages and any other objects (resources).

In addition, the version of each object must be stored. The data model depends on the interface with the catalogue system and requires further investigation.

5.2.4 Archival Storage

Archival Storage stores and is able to retrieve Archival Information Packages (AIPs). No functional changes are required. However, the AIPs may be bigger and thus require more storage space. The AIPs may be stored on magnetic disk, optical disk or tape. Storage on magnetic disk is the most expensive but offers quick access. Storage on tape is the cheapest, but requires more time to access. Optical disks may be an affordable alternative because Web snapshots are collections of relatively small objects that must be located relatively quickly.

The optimal storage medium depends on the granularity of how snapshots will be archived. If a complete snapshot is stored and retrieved as an entity, it is obvious that it must be stored on tape because of its size. The granularity issue is discussed in section 6.1 Granularity of the Archiving.

5.2.5 Access

Access requests metadata from Data Management and retrieves AIPs from Archival Storage. It creates Dissemination Information Packages (DIPs) from the collection of requested metadata and/or AIPs.

The AIPs retrieved may have to be modified to make them available (accessible) through the Consumer Browsers. The problem is that the original URIs point to places on the (past) Internet but should point to the archived content that must be retrieved from the archive.

5.2.6 Packing&Delivery

Packing&Delivery pack DIPs and deliver them to consumers. Since Access has already transformed all URIs in such a way that they can be accessed, Packing&Delivery only has to provide Web server functionality. Packing&Delivery should also perform authentication and authorization on the requests.

5.2.7 Administration

Administration must be able to maintain the lists of included and excluded URIs for harvesting as well as the definitions of domains, Web sites and Web resources.

5.2.8 Preservation

Preservation must be able to determine the AIPs that contain URIs having a file type that may become obsolete. These AIPs must be retrieved from the archive and the resources (URIs) with the obsolete types must be converted to new file types. The updated AIPs should be put in the archive with a reference to the AIP containing the original Asset. Another requirement for Preservation is to modify the view paths of URIs to reflect changes in URI applications, viewers and emulators.

Preservation queries Data Management via Access for AIPs that require preservation. The selected AIPs are retrieved via Access from Archival Storage. In order to keep this process manageable it should be possible to retrieve the AIPs separately.

6/

01 Web archiving 000010

impact on DIAS

6.1 Granularity of the Archiving

Archiving has to do with the creation of an Archival Information Package (AIP) for the directories and files that constitute a Web snapshot. Web snapshots can be archived in a number of ways:

1. Each Web snapshot can be archived into one AIP that has one NBN.
2. Each Web snapshot contains a number of domains with relationships and each domain has its own NBN.
3. Each Web snapshot contains a number of Web sites with relationships and each Web site has its own NBN.
4. Each Web snapshot contains a number of URIs with relationships and each URI has its own NBN.

It is important to remember that DIAS has a one-to-one relationship between NBN and AIP.

The OAIS reference model has a solution for specifying relationships between AIPs. An AIP can either be an Archival Information Collection (AIC) or an Archival Information Unit (AIU). An AIC references a number of AIUs. DIAS only stores AIPs. It does not involve the concepts of AIU and AIC. Thus it is impossible to define relationships between bits and pieces of a Web snapshot.

The possible variations will be discussed in the following sections.

6.1.1 Each Web Snapshot Archived into one AIP

A Web snapshot contains lots of directories and files that have to be archived as a whole. All these pieces must be included in one Submission Information Package (SIP) that results in one Archival Information Package (AIP) that can be found via its associated (technical) metadata. When someone wants to access the Web snapshot it is retrieved as a whole, a Dissemination Information Package (DIP) is created and the directories and files will be made available. After that, the Web snapshot is fully available.

The main disadvantage of this approach has to do with the size of the Web snapshot. The SIP, AIP and DIP will all be huge files. These files can exceed the 2GB limit of some of today's file systems. Other disadvantages are:

- ⊄ Ingesting takes a long time and could be more error prone. How does one resume after communication errors?
- ⊄ Possible packaging problems in the information packages.
- ⊄ Possible (network) transport problems.
- ⊄ Retrieval takes a long time and could be more error prone.
- ⊄ There is a long response time before the Web snapshot can be accessed.

The clear advantage is that a single Web snapshot can be handled as a single entity. There are no relationships between AIPs. Once the AIP is retrieved, it can be browsed quickly.

6.1.2 Each Domain in a Web Snapshot Archived into one AIP

This is a more manageable approach than one AIP for the whole snapshot. Each domain can be harvested separately if only internal links are followed. Of course there are also external links and a decision must be made regarding how to deal with these links.

The complication here is that the current DIAS data model is not equipped to maintain structural metadata (i.e. AICs that relate AIUs). DIAS only utilizes relationships between AIPs for preservation purposes.

6.1.3 Each Web Site in a Web Snapshot into one AIP

This approach is an extension of the domain approach. There will be many domains that only have one site. However, some domains can have a number of Web sites. This approach makes these possibly huge sites manageable as well.

Another problem with this approach is that the location where the relationships between the AIPs are maintained is not known.

6.1.4 Each URI in a Web Snapshot Archived into one AIP

If each URI is archived separately, it makes defining a Web snapshot easy. Each URI becomes one AIP. This creates a problem for Retrieval since each URI must be retrieved separately. This would be a problem if each AIP was stored on tape. But since many URIs will be a small file, it is also recommended that these be stored on magnetic disk.

Another problem with this approach is that it is not known where the relationship between AIPs are maintained.

6.2 Version Management

Version management depends on the archive granularity selected and the assignment of the National Bibliography Number (NBN) to an AIP.

DIAS is familiar with the concept of AIPs that contain an original asset and AIPs that contain converted or installed versions of the Asset. AIPs containing the converted or installed version contain a reference to the original AIP. This concept can also be used to

preserve Web pages. For example, an AIP that contains files that have a file type no longer be supported by the browser with plug-ins has to be converted. The whole AIP must be retrieved, the problematic files must be converted and the new files must be inserted via the Builder Application. A new AIP will be created that contains the converted version and that will link to the original version.

If Web snapshots are stored as one AIP there is only one version of a URI in each Web snapshot. Another version of the same URI can be in another Web snapshot. Note that a URI includes the query string (the parameters).

If the Web sites part of a Web snapshot are archived separately (within their own AIPs), it would be nice to store more than one version of such a site with the same NBN. The IBM Content Manager product used for the DIAS solution can store a maximum of 32767 versions of an AIP. A mapping is needed that defines which version of a NBN is associated with which snapshot.

This version mechanism of IBM Content Manager is also a possible solution if each URI is archived separately as one AIP.

6.3 Manipulation of the Original URIs

Whether or not the original URIs need to be changed depends on:

- ∄ Whether one wants to store the authentic Web pages and objects.
- ∄ How the archived snapshots are to be made available for browsing.

Clearly URIs have to be transformed to make the archived Web sites available. Otherwise all URIs would still link to the original sites. The question is when to transform the URIs and how to transform them.

6.3.1 When to Transform Original URIs

The URIs can be transformed before archiving or at the time they have to be made available. The problem of modifying the URIs before archiving is that this depends on how the Web snapshot will be accessed later. Since this must be a deposit solution for the long term, it is likely that access will change over time. If the URIs are modified before archiving and the URIs have to be transformed, the AIP must undergo a preservation conversion.

Changing the URIs when the Web snapshot is requested is one possible approach. The drawback involved here is that transforming the URIs requires some processing time. The benefit is that this offers a lot of flexibility in terms of making the requested Web snapshots available concurrently.

6.3.2 How to Transform Original URIs

The URIs must to be transformed in such a way that all HTTP requests will be given to the DIAS server. Possible options for resolving this problem are:

- € Transform each URI into a relative URI and give the base location of the site to the Consumer Browser.
- € Transform each URI via client side processing.
- € Transform each URI via an intelligent proxy.
- € Transform each URI via server side processing (i.e. a Java Servlet).
- € Transform each URI into a NBN request.

An additional complication is that the solution depends on the granularity of archiving the Web snapshots. If each URI is stored separately, the NBN for a specific URI must be found via the catalogue.

If the snapshot is accessed in pieces (per URI or per Domain) there is a dependency on when to retrieve AIPs and how to access the resulting Web snapshot.

6.4 Examples

This paragraph takes a small Internet and illustrates how it would be archived based on the different alternatives for AIPs for Web snapshots.

6.4.1 The Sample Internet

The sample Internet has a limited number of domains (d). Each domain has pages (d<i>p<j>) and each page has resources (d<i>r<j>). Each URI can either be internal (within the domain) or external (to other domains). The details regarding each domain are shown in the following Figures 6.1-6.2-6.3.

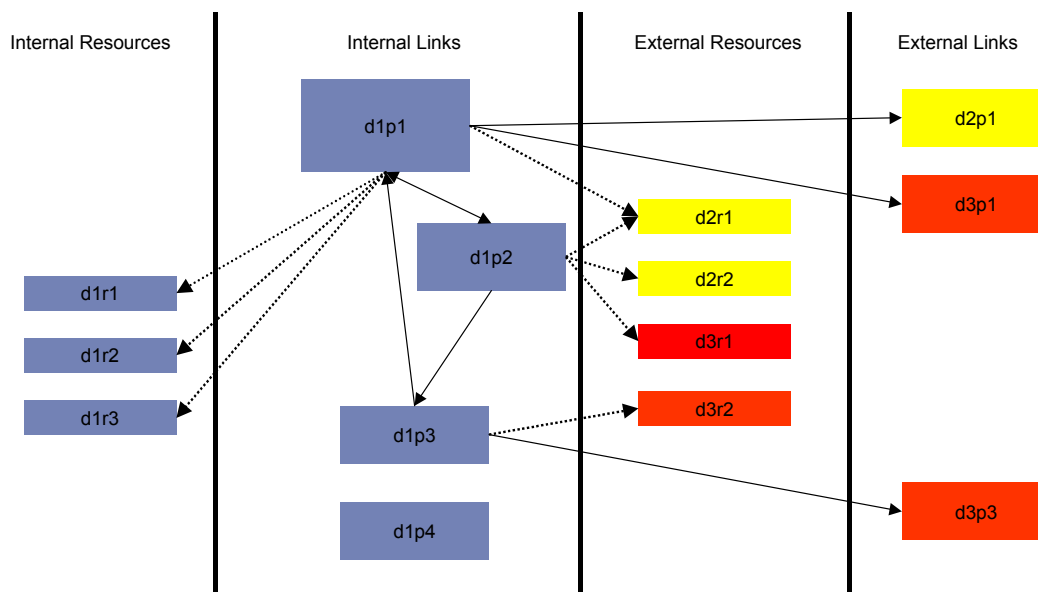


Figure 6.1 / Sample domain 1

Each sample domain is divided into four regions: Internal Resources (for embedded images, etc.), Internal Links (for links to HTML documents within the domain), External Resources (for embedded images that come from external sites) and External Links (for links to HTML documents outside the domain).

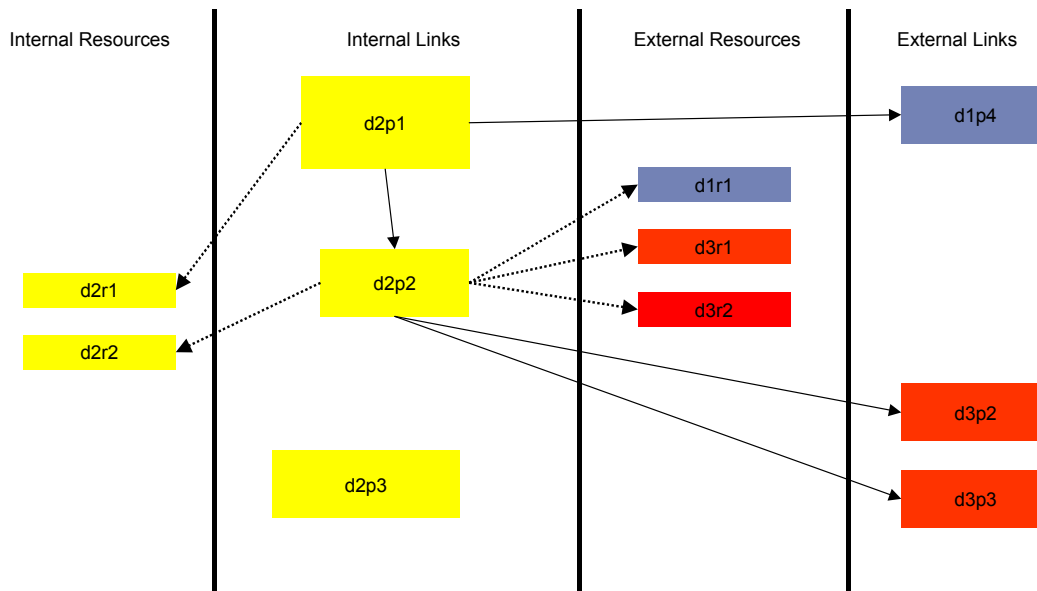


Figure 6.2 / Sample domain 2

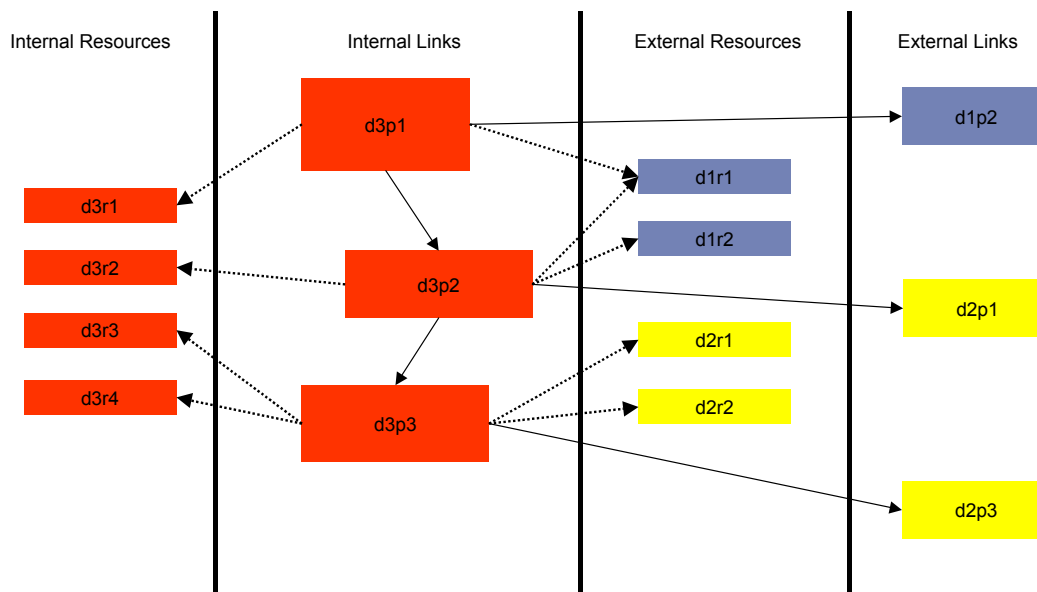


Figure 6.3 / Sample domain 3

6.4.2 One AIP per URI Example

In case each URI is archived as one AIP the following archive will be created:

NBN of AIP	URI	URI Type
1	d1p1	HTML Document
2	d1p2	HTML Document
3	d1p3	HTML Document
4	d1p4	HTML Document
5	d2p1	HTML Document
6	d2p2	HTML Document
7	d2p3	HTML Document
8	d3p1	HTML Document
9	d3p2	HTML Document
10	d3p3	HTML Document
11	d1r1	Resource
12	d1r2	Resource
13	d1r3	Resource
14	d2r1	Resource
15	d2r2	Resource
16	d3r1	Resource
17	d3r2	Resource
18	d3r3	Resource
19	d3r4	Resource

Table 6.1 / One AIP per URI

Here each AIP is only a single file. Retrieval can be fast if all these AIPs are stored on fast media. It is also possible to store the AIPs that contain the HTML documents on fast media and other resources on slower media. However, this organization is expected to involve too much overhead in interfacing with the catalogue system. It also requires a lot of overhead for assigning the NBN. Each file is assigned its own NBN.

A separate AIP can be used to keep track of which AIPs are contained in the full Web snapshot.

6.4.3 One AIP per HTML Document Example

In case each HTML document is archived along with its required resources as one AIP the following archive will be created:

NBN of AIP	HTML Document	Resources
1	d1p1	d1r1
		d1r2
		d1r3
		d2r1
2	d1p2	d2r1
		d2r2
		d3r1
3	d1p3	d3r2
4	d1p4	None
5	d2p1	d2r1
6	d2p2	d2r2
		d1r1
		d3r1
		d3r2
7	d2p3	None
8	d3p1	d3r1
		d1r1
9	d3p2	d3r2
		d1r1
		d1r2
10	d3p3	d3r3
		d3r4
		d2r1
		d2r2

Table 6.2 / One AIP per HTML document

Retrieval of one HTML page with all its required resources is very fast. Some duplication will occur for resources that appear in more than one HTML document. A variation of this approach is to save internal resources with the HTML document and leave external resources in their own AIP. These AIPs would have to be retrieved as well. However, the delay caused by retrieving the external resources might be acceptable. This approach is followed for the Internet Archive. The Internet Archive has developed the WayBack Machine (see <http://www.archive.org>).

Many offline explorer tools also use this approach.

Versions of each AIP can be created via relationships or via IBM Content Manager's versioning mechanism.

A separate AIP can be used to keep track of which AIPs are contained in the full Web snapshot.

6.4.4 One AIP per Domain Example

In cases where each domain is archived as one AIP the following archive will be created:

NBN of AIP	URI	URI Type
1	d1p1	HTML Document
	d1p2	HTML Document
	d1p3	HTML Document
	d1p4	HTML Document
	d1r1	Resource
	d1r2	Resource
	d1r3	Resource
2	d2p1	HTML Document
	d2p2	HTML Document
	d2p3	HTML Document
	d2r1	Resource
	d2r2	Resource
3	d3p1	HTML Document
	d3p2	HTML Document
	d3p3	HTML Document
	d3r1	Resource
	d3r2	Resource
	d3r3	Resource
	d3r4	Resource

Table 6.3 / One AIP per domain

Note that each AIP has all its internal resources available. External resources have to be retrieved separately. The domain approach is very manageable. The SIP, AIP and DIP objects do not become too big and it is much easier to determine whether a domain is up-to-date or not. The problem here is that the external resources have to be retrieved separately from other AIPs.

If an external link is selected another AIP must be retrieved from the archive after which browsing continues. Versions of a domain can be arranged via AIP relations or via IBM Content Manager's versioning mechanism.

A separate AIP can be used to keep track of which AIPs are contained in the full Web snapshot.

6.4.5 One AIP per Web Snapshot Example

If a full Web snapshot is archived as one AIP the following archive will be created:

NBN of AIP	URI	URI Type
1	d1p1	HTML Document
	d1p2	HTML Document
	d1p3	HTML Document
	d1p4	HTML Document
	d2p1	HTML Document
	d2p2	HTML Document
	d2p3	HTML Document
	d3p1	HTML Document
	d3p2	HTML Document
	d3p3	HTML Document
	d1r1	Resource
	d1r2	Resource
	d1r3	Resource
	d2r1	Resource
	d2r2	Resource
	d3r1	Resource
	d3r2	Resource
	d3r3	Resource
	d3r4	Resource

Table 6.4 / One AIP for one snapshot

All resources are contained within one AIP. This results in large AIPs that will present problems in intermediate work areas.

Each new snapshot can be stored in its own AIP or IBM Content Manager's versioning mechanism could be used. The latter depends on the interface with the catalogue system, which requires further investigation.

7/ Web

preservation layer models

This chapter discusses the Preservation Layer Models (PLMs) that play an important role in Internet technology. An example of a PLM is shown in Figure 7.1. For details regarding PLM models the user is referred to the LTP Study report on Preservation Requirements in a deposit system [Diessen 2002].

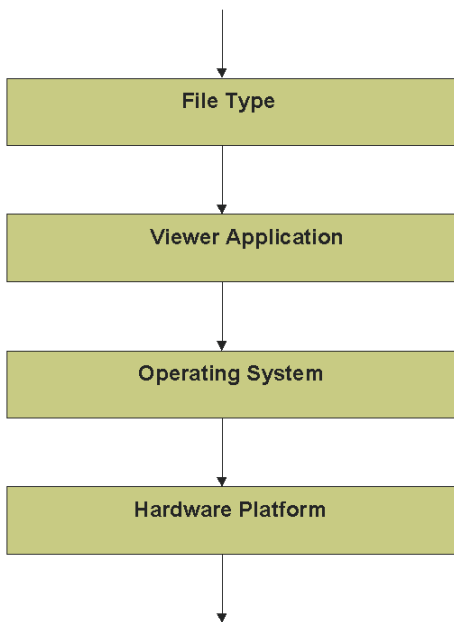


Figure 7.1 / Sample Preservation Layer Model

7.1.1 Basic PLM

The Internet works with Web pages that are written in Hypertext Mark-up Language (see paragraph 3.5), that are navigable and that contain (embed) other resources. A specific

HTML version is supported by specific versions of HTML Viewer Applications (i.e. Browsers).

HTML Viewers contain default support for viewing files with the file types .htm, .HTML, .gif, .jpg, .text (see section 3.6).

There is one important aspect that differs from many other PLMs. The file content for the Web PLM is stored (or created) on a computer system (server) other than the system on which the viewer runs (client). The Web content is always accessed via the network using the HTTP, FTP or other protocols (see 3.2). The network services that enable this communication can be thought of as being provided in a separate layer. Other PLMs access remote files via services provided by other mechanisms like file sharing, or via mapped network drives provided via the Operating System Layer. The basic PLM is depicted in Figure 7.2.

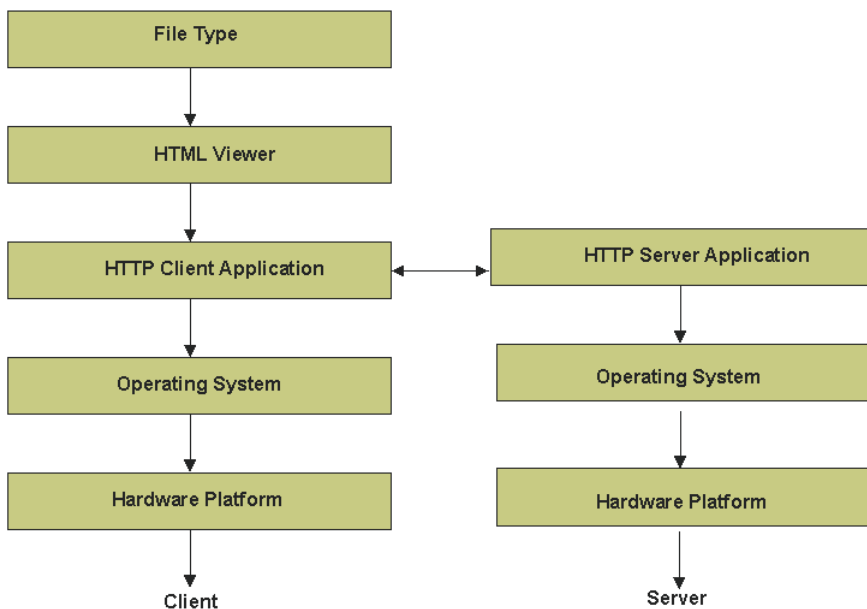


Figure 7.2 / Basic PLM for default HTML file types

This PLM shows that the Operating System and Hardware System (i.e. Platform) for the HTML Viewer are important for viewing the file types that are supported as standard features by a modern HTML Viewer (Web browser).

Note that the version of the HTTP protocol can also be important.

7.1.2 Extensions of the Basic PLM

The Basic PLM can be extended in several ways. At a minimum the following variations are possible:

€ **Basic PLM with Plug-Ins**

This PLM enables the browser to support additional file types using the mechanism of plug-ins. One example of this is support of the PDF file type via the Acrobat Viewer, MS Viewer, Quicktime Viewer, or MrSID Viewer plug-ins.

€ **PLM with processing on the Client side** (Client Side Logic)

Java enabled HTML Viewer that runs Java Applets and Java Script.

€ **PLM with processing on the Server side** (Server Side Logic)

This PLM can be used for content dynamically generated by the Server. The Server Side Logic can be CGI scripts, Perl Scripts, Java Servlets, etc.

€ **PLM with processing on both the Client and Server Sides** (Client and Server Side Logic)

Client Side and Server Side Logic that are independent of one another.

€ **PLM with client/server processing** (Client/Server Logic)

Client and Server Side Logic that are dependent of one another (Cooperates).

All these variations can be placed into one generic Web PLM that contains all these features. This generic Web PLM is shown in Figure 7.3 below:

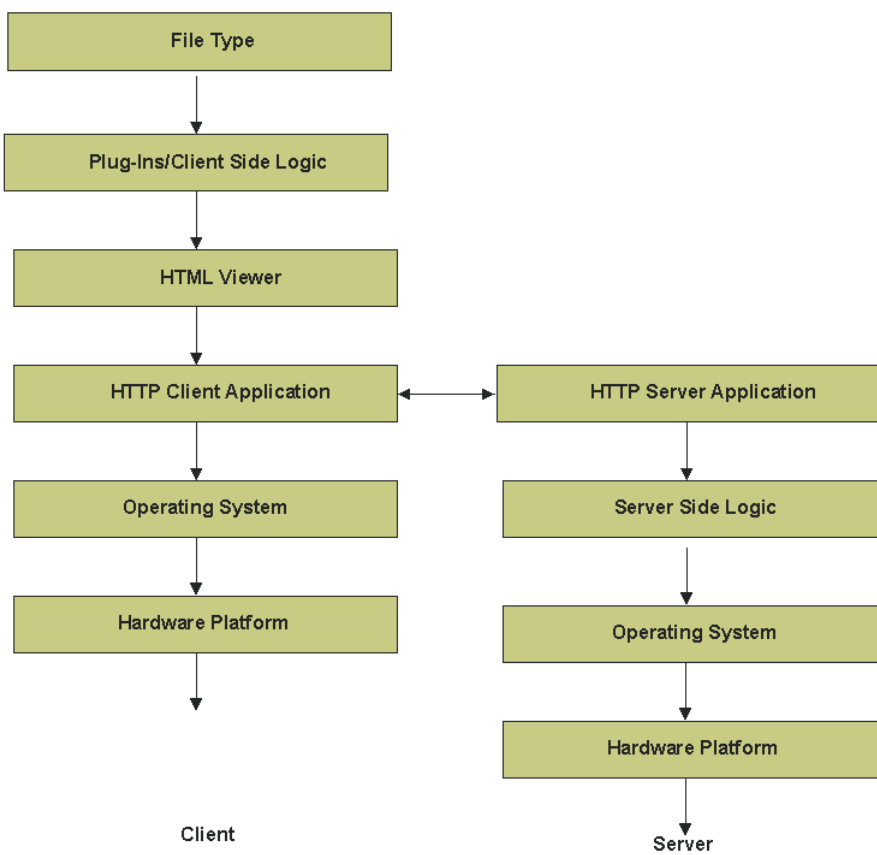


Figure 7.3 / Generic Web PLM

8/

01 conclusions 010000010

A number of conclusions can be drawn based on our evaluation of the Web harvesting tools in the DIAS environment.

Web harvesters must not change (localize) links in downloaded Web pages. They must leave the pages in their original form. Harvesters should run HTTP GET commands only when a local copy of a resource is not available; otherwise they should run HTTP HEAD commands. Sufficient metadata to make this possible must be stored per downloaded resource.

Harvesters should be able to check Web resources for changes based on user-defined settings (that specify how often to check).

Harvesters should store downloaded URIs in a Web space. This Web space is used as input by the process that takes Web snapshots. The process that takes the snapshots selects the changed resources from the Web space and stores them in a structure ready for archiving (SIP). This structure depends heavily on the requirements that must be met for accessing the snapshot.

The harvester could notify the process that takes the snapshot of changed resources.

If the harvesters run per domain, the process of searching the Web and preparing the downloaded resources for archiving becomes manageable. A single domain can be checked for changes in an acceptable time frame. A decision must be made regarding how embedded resources (i.e. images) appearing on the pages within the domain have to be collected, stored in the Web space and archived. The external URIs must be stored as external resources (dependencies) for the domain. The solution to this problem also depends on the harvester used and the disk structure produced.

New user roles are identified for defining the URIs to include and exclude items. This role should also investigate URIs that result in failures during harvesting and URIs that seem to change very frequently (i.e. weird programs with timestamps as parameters).

Also, we foresee new tasks for system administrators who deal with the harvester processes and disk space management.

If archiving takes place at the domain level the resulting SIP, AIP and DIP packages will have an acceptable size and will contain an acceptable number of files. Processing time also becomes acceptable. Whether or not archiving per domain results in an acceptable response time must be investigated. Improvements may be possible (i.e. pre-fetching).

It must be decided whether versions of downloaded resources of domains will be assigned their own NBN or whether all versions will be stored under a single NBN. This also has to do with the interface with the catalogue system.

Text searches on the contents of the documents in the snapshot must be further investigated in relation to the interface with the catalogue.

Harvesters should store downloaded URIs in a Web space. This Web space is used as input by the process that takes Web snapshots. The process that takes the snapshots selects the changed resources from the Web space and stores them in a structure ready for archiving (SIP). This structure depends heavily on the requirements that must be met for accessing the snapshot.

recommendations

Below are some recommendations regarding Web page archiving:

- ⊘ Each URI should be given a short, simple ID to speed up searches and comparisons. The ID can be some sort of hash value. (MD5 is a possible solution).
- ⊘ The SIP interface should include bibliographic Metadata. The OAIS system can extract Bibliographic, Technical and Preservation Metadata and store this in the appropriate system (i.e. KB Catalogue system, DIAS, or Preservation System).
- ⊘ Synchronization between Metadata Systems and Deposit Systems must be more strictly enforced. There should be referential integrity among all the systems.
- ⊘ Text Search on the contents of the archived Web pages should be a service provided by the Deposit System. Otherwise the index must be made available to the catalogue system and an interface must be defined that includes synchronizing index creation.
- ⊘ A Web Snapshot must not be stored and retrieved as one single entity (one AIP) with its NBN.
- ⊘ The catalogue system must be updated with bibliographic information about each Web snapshot that is available.
- ⊘ The catalogue system must know which Web sites/domains are included in a Web snapshot.
- ⊘ The catalogue system must know the relationship between snapshots of a specific Web site/domain.
- ⊘ A request for the AIP of a Web site/domain should have a time period (from-date to to-date) as well as an NBN.
- ⊘ Navigating to an external Web site/domain from a Web page could require a pop-up window that requests the user to enter the time period or lists the available time periods (hit list) for the requested link and requests the user to select one.
- ⊘ To make browsing a restored Web site/domain easier the HTML pages must be retrieved as fast as economically possible (RAM, Magnetical Disk, Optical Disk, Tape). The other resources (images etc.) may arrive later on the Web page (and may come from slower media).

The requirements for accessing the Web snapshots have to be formulated. These requirements have to be defined in terms of Use Cases. The Use Cases can be used to design a suitable Web archiving solution.

One possible (high level) use case is:

- ⊘ First browse available Web snapshots for bibliographic metadata. Then browse a selected Web snapshot for the available Web sites/domains for bibliographic metadata. Then request a selected Web site/domain from the deposit system. The Web site/domain is retrieved from the deposit system (DIAS), the URI links are transformed to allow local browsing and the result is stored in the download area. The Web server and Web application server provide access to the restored site. The consumer can browse the Web site as if it were the Internet. If an URI link is requested that falls outside the already restored Web site, this new URI will be retrieved from the deposit

system. Note that for this to work each URI must be assigned its own NBN or each URI must be assigned parameters that uniquely identify a URI belonging to a specific NBN. The interaction with the catalogue system is important here as well.

01 appendix a: 01000010 references

[CCSDS 2001]

Management Council of the Consultative Committee for Space Data Systems, *CCSDS650.0-R-2: Reference Model for an Open Archival Information System (OAIS)*, Red Book, Washington, DC, July 2001, http://ssdoo.gsfc.nasa.gov/nost/isoas/ref_model.html.

[Diessen 2002]

Diessen, R.J. van, *Preservation Requirements in a Deposit System*, IBM / KB Long-Term Preservation Study Report Series Number 3, December, 2002.

[Internet Society 1999]

The Internet Society, *Hypertext Transfer Protocol - - HTTP/1.1*, RFC2616, June 1999

[NEDLIB 2001]

Networked European Deposit Library, *Manual for Installation and Usage of the NEDLIB Harvester, Version 1.2*, October 2001, Helsinki University Library and Center for Scientific Computing (CSC) Finland.

[Raggett and Le Hors 1999]

D. Raggett, A. Le Hors, I. Jacobs, *HTML 4.01 Specification*, World Wide Web Consortium, December 1999

[Sijtsma 2002]

Sijtsma, L., *NEDLIB LB-5648/A D5.1 - Demonstrator Report*, Issue 1.0, Ref. 6048/D5.1, December, 2000

[Tanenbaum 1990]

Tanenbaum, A.S., *Computer Networks*, Prentice-Hall, Englewood Cliffs, N.J., 1996

01 appendix b: 01000010 glossary

Archival Information Collection (AIC): An Archival Information Package whose Content Information is an aggregate of other Archival Information Packages.

Archival Information Package (AIP): Content Information and the associated Preservation Description Information required to preserve the Content Information over the long term. This information includes the related Packaging Information.

Archival Information Unit (AIU): An Archival Information Package whose Content Information is not further broken down into other Content Information components, each of which has its own complete Preservation Description Information. It can be viewed as an 'atomic' AIP. An example of an AIU would be a table of numbers representing temperatures in a certain region with all the associated documentation describing how and where the temperatures were measured, what instruments were used to take the measurements, who made the measurements, why they were made, what processing has been performed on the measurements and who has had custody of these measurements since they were first created, how the measurements relate to other information, how the measurements can be uniquely referenced by others, etc.

Digital Information Archiving System (DIAS) is the core of the KB's electronic deposit system. Version 1 has been developed by IBM and was released in October 2002.

Dissemination Information Package (DIP): An Information Package that contains part or all of one or more AIPs and that is distributed to the consumer as requested.

DNEP: In September 2000 the KB and IBM Netherlands signed the final contract which initiated the project "Depot voor Nederlandse Electronische Publicaties" (DNEP) [Deposit for Dutch Electronic Publications] to design and implement DIAS with a Long-Term Preservation Study as an integral part of the total effort.

Domain Name System (DNS): This is the way that Internet domain names are located and translated into IP addresses. A domain name is a meaningful and easy-to-remember "handle" for an Internet address.

File Transfer Protocol (FTP): This is an application protocol on top of the TCP protocol.

Hypertext Markup Language (HTML): This is the set of markup symbols or codes inserted in a file intended for display on a World Wide Web browser. The markup tells the Web browser how to display a Web page's words and images to the user. Each individual markup code is referred to as an element (but many people also refer to it as a tag). Some elements come in pairs that indicate when a particular display effect is to begin and when it is to end.

Hypertext Transfer Protocol (HTTP): This is an application-level protocol for distributed, collaborative, hypermedia information systems. It is a generic, stateless, object-oriented protocol which can be used for many tasks, such as name servers and distributed object management systems, through extension of its request methods. One feature of HTTP is the typing and negotiation of data representation, allowing systems to be built independently of the data being transferred.

Internet Protocol (IP): This is the protocol that is used to exchange data packets over an IP network. When you send or receive data, the message is divided into little chunks called packets. Each packet can follow a different route across the network. IP is a connectionless protocol, which means that there is no continuing connection between the end points that are communicating. Each packet that travels through the network is treated as an independent unit of data without any relation to any other unit of data.

KB: The National Library of the Netherlands (Koninklijke Bibliotheek, KB).

National Bibliography Number (NBN): This is a generic name referring to a group of identifier systems utilized by the national libraries and only by them to identify deposited publications which do not have an identifier, or to identify descriptive metadata (cataloging) that describes the resources.

Open Archival Information System (OAIS): OAIS is a functional reference model. An OAIS is an archive consisting of an organization of people and systems that has accepted the responsibility to preserve information and make it available for a designated community. It specifies a specific set of responsibilities and it allows an OAIS archive to be distinguished from other uses of the term archive. The term 'Open' in OAIS is used to imply that this recommendation, as well as future related recommendations and standards are developed in open forums. It does not imply that access to the archive is unrestricted.

Submission Information Package (SIP): The Information Package identified by the producer in the submission agreement with the OAIS.

Transmission Control Protocol (TCP): This protocol manages the disassembling and assembling of messages into and from smaller data packets that are transmitted over a network.

Uniform Resource Identifier (URI): These form the generic superclass of resource identifiers. They are simply formatted strings that identify a resource by name, location, or any other characteristic.

Uniform Resource Locator (URL): This is a concrete resource identifier that gives the address of a resource. The URL contains the name of the protocol required to access the resource, a domain that identifies a specific computer on the Internet, and a hierarchical description of a file location on the computer.

Uniform Resource Name (URN): A class of resource identifiers that is persistent and location-independent. The resource identified by an URN may reside in one or more locations at any given time, may move, or may not be available at all.

01 appendix c: 01000010

issues

A number of issues have been identified that will shape the way Web publications / pages will be archived in an electronic deposit system. The list below summarizes the most important ones within the context of the KB:

- € How to select the Dutch part of the Internet, i.e. those Web sites that have to be captured and preserved.. The Dutch companies that are registered in the KB NL-Menu can be used as a starting point.
- € What Character Set will be used to store captured Web pages, Western (ISO-8859-1), Unicode (UTF-8) or some other character set? Some Web sites have different pages for different languages.
- € Is each version of a Web site assigned its own NBN or do all versions of a Web site receive the same NBN?
- € Must the original copy of a Web site always be stored? Such a copy will contain URL links that make it unusable.
- € What granularity will be used for indexing Web snapshots? One NBN for one snapshot, one NBN for each domain, one NBN for each Web resource?
- € AIP granularity versus Web snapshot. One AIP per snapshot, domain, resource?
- € How do we interface with the catalogue (KB Catalogue system) for text search on the content of the archived Web documents?

To all these issues there are currently no easy answers. Most of them have to be further explored now that the first version of DIAS will become operational. DIAS will provide the environment in which the KB can gain additional hands-on experience to search for the right answers to be included in the next versions of the DIAS system.

IBM
long

KB
term

preservation
study

