Efficient Ingest of Datasets in a Two-Stage Archival Process: Easy-Store

Rutger Kramer and Laurents Sesink; Data Archiving and Networked Services; The Hague, The Netherlands

Abstract

The recently founded organization Data Archiving and Networked Services, based in The Hague, the Netherlands, has been given two basic responsibilities: storing datasets resulting from humanities and social sciences research, and improving the data infrastructure for these two fields. From the start it was decided that for DANS to be able to take on these responsibilities, a new approach to data archiving should be developed. This paper outlines this new approach, which is based on a two stage archival process, and will highlight one component of this approach which we nicknamed Easy-Store.

Introduction

In September 2005 a new data archiving institute has been founded in the Netherlands called Data Archiving and Networked Services (DANS)[1]. DANS is *the* national organization responsible for storing and providing permanent access to research data from the humanities and social sciences. DANS is a joint initiative of the Royal Netherlands Academy of Arts and Sciences (KNAW)[2] and the Netherlands Organization for Scientific Research (NWO)[3]. By 2004, these organisations had recognized the impending backlog in the storage and dissemination of digital research data from the humanities and social sciences, despite the rapid progress offered by fast-growing electronic facilities. Such progress is vital as it facilitates both the *reuse* of data and the possibility to *verify* publications based upon these datasets.

Reliability

The main goal of digital archives is to underwrite the access to the digital information stored in the archive. Another important role of the archive is to give users an indication of the quality of the stored data. Although there are thousands of digital archives in all kinds of settings, there is until the present no common strategy for the long term preservation of digital data. Thereby is it impossible for archivists to benchmark the quality of the digital data in the archive. DANS will fulfil its remit in a way that satisfies certain criteria of quality and permanent accessibility to the data. For this purpose DANS is developing a seal of approval. The DANS seal of approval sets minimum requirements, which guarantees that data sets are: of a reliable quality, permanently traceable, accessable and useable. These requirements are consistent with international standards and guidelines for digital archiving, such as OAIS (the Open Archival Information System), and the standards for Trusted Digital Repositories of the RLG and NARA (Research Libraries Group and National Archives and Records Administration) in the United States, and Germany's NESTOR (Network of Expertise in Long-term STOrage of Digital Resources - A Digital Preservation Initiative for Germany).

Challenge

It is expected that DANS will have to ingest and manage an increasing number of datasets. It is thereby of particular importance to ensure that deposited data meets a maximum standard of quality, traceability, accessibility and usability.

If DANS were to hold on to the traditional process of data archiving, i.e. having archivists enter extensive metadata for each deposited dataset, it is likely that the throughput of data from ingest to dissemination will clog due to the strain on personnel entering the metadata.

Thus, it was clear that in order to enhance the efficiency of the process, archivists would have to save time on acquiring and entering metadata, checking the file format and structure and converting it to a durable format. What also became clear was that if you were to spend less time on these basic responsibilities, the overall quality of the archive – metadata, retrievability, usability of datasets – could be jeopardized.

DANS developed a suitable compromise that can serve as a solution to the data-flood problem and can also eliminate the quality vs. quantity dilemma.

Two-Stage Archiving

The archival process DANS is implementing consists of two separate processes we nicknamed Easy-Store and Deep-Store.

Easy-Store

Every dataset that comes in will be handled according to the steps defined by the Easy-Store process, which effectively means that:

1. the depositor provides basic metadata according to an application profile based on Qualified Dublin Core[4],
2. he or she uploads the metadata and the dataset files to DANS,
3. an archivist checks the metadata and the files using a predefined workflow,
4. finally, if everything checks out, the dataset is published on the DANS website.

Shifting the responsibility of the creation of metadata from the archivist to the researcher is the key difference between the old process and the new. Researchers used to be contacted whenever the archivist needed help creating the metadata; in the new situation the researchers themselves will be able to describe their work. For 'self archiving' their data sets researchers must not be confronted with a twenty page metadata entry form whenever they want to upload their material. Entering some basic but very necessary metadata must not take a lot of time. If researchers have to spend too much time filling out a form, they are not likely to upload more material in the near future. To make the 'self archiving' process as smooth as possible this information has to be

retrieved automatically. Most of the contextual information which can give an opinion of the quality of the datasets is stored in administrative systems. Research information can be exchanged for example by means of the Cerif standard (Common European Research Information Format)[5] and information about publications based upon the data sets can be retrieved by means of the Open Access Initiative - Protocol for Metadata Harvesting (OAI-PMH)[6].

The dataset files will be stored in their original format. Periodically migrations will have to be performed, but this does not guarantee easy access to the files in general. Although we are not able to address this issue for each and every deposited dataset, some datasets can be marked as exceptionally useful or important. These datasets will go through the Deep-Store process.

Deep-Store

Design of the Deep-Store process is currently in a conceptual phase. Although no explicit implementation plans have yet been made, the following things are clear about the objectives of Deep-Store:

- it should make datasets as easily accessible as possible,
- datasets should be accessible through the internet,
- it should be able to link datasets together or integrate with existing dissemination systems.

As an example, we're currently looking at the NESSTAR[7] system as a possible Deep-Store solution for social science datasets. NESSTAR enables end users to view metadata, perform basic statistical analyses, and download the dataset in a number of popular file formats.

Additionally, new systems can be built within externally funded projects within the Thematic Development Programs of DANS, in which we will cooperate with researchers to design and/or improve a data-infrastructure within their field. One of the components of such a data infrastructure can be elaborate dissemination of relevant datasets.

Implementation

The basic functionality that is needed to support the core activities of DANS is not new or innovative. Document Management Systems already offer a good basis for ingest, storage and retrieval of document like objects, which are exactly some of the basic requirements of the Easy-Store system. The question was, however, if there is a DMS solution available that can be implemented according to all of the requirements set by DANS, and that would fit into the two-stage archival strategy. To answer this question, we first set out to get an overview of the actual requirements our archivists will have when it comes to Document Management Systems.

Apart from basic search and retrieval requirements, some requirements turned out to be key characteristics that would determine whether a proposed DMS would suit the needs of DANS. Some of these requirements are listed below:

- the possibility for researchers to enter the metadata for a dataset themselves, and upload the dataset through the Internet,
- the possibility for (anonymous) users to download datasets for reuse, and determine which users can download which dataset,
- the possibility to implement a workflow process that will be applied by the archivists,
- support for persistent identifiers,
- support for authorization and authentication,
- linking and integration with other (3rd party) systems in the future.

Especially the last requirement is of paramount importance; integration with future implementations of Deep Store solutions has to be possible.

Acquisition vs. In-House Development

A number of commercial and open source implementations have been considered during the analysis. Unfortunately, and not unexpectedly, none of them gave a 100% coverage of the requirements set by DANS. This is not due to structural shortcomings of the respective products, but rather due to the wide range of implementation specific requirements set by DANS. We estimated that if we were to choose one of the off-the-shelf solutions and customize it to suit our needs, a substantial amount of development would be needed to make the software fit into our organization.

In order to get a better understanding of the implementation issues posed by the requirements, we decided to start development of a small proof-of-concept application. During the two weeks that were reserved for this proof-of-concept, we came to realize that instead of acquiring an off-the-shelf application and customizing it, in-house development of a complete system could also be a viable solution. Developing the application ourselves would cost a lot of additional development – more that customization would cost – but on the other hand could guarantee that the requirements would be covered. Moreover, we would have complete control over the source-code which would enable us to make integration with, for now, unknown applications possible.

Proof of Concept

The proof-of-concept application that was developed consisted of two separate components: a storage component and a web-access component. The Open Archival Information System (OAIS)[8] is used as a reference model for the archiving process.

The storage component should ensure reliable storage of deposited datasets and its metadata. In order to do this, a distributed storage system will be implemented that will keep redundant copies of every dataset dispersed over two or more servers. The idea for this redundant storage is based on the Lots of Copies Keeps Stuff Safe (LOCKSS)[9] concept. Every data object that comes in will be stored on two or more autonomous storage servers. If, at some time, one of the servers is not available, the other server should still be able to deliver the data object when requested. Moreover, as soon as the 'back-up' server detects that another server holding a copy of one of its data-objects is off-line, it will try to mirror its copy at another available server.

Data-objects are stored on the server along with their metadata. If the data-object consists of multiple individual files, each file can have specific metadata associated with it, as well as inherit general metadata that is descriptive for the entire data-object. It doesn't matter which metadata format is used to describe the objects: although the server stores the metadata and generates

full-text indices, it doesn't try to interpret the metadata in any way. The only requirement to the metadata is that it is formatted as XML.

The storage servers can be contacted by using a simple, but specific API. The API contains all of the commands that can be issued for the server, e.g.

- CreateAip, Create a new storage container for a data-object
- AddCategory, Create a new category that can contain data-objects
- AddAipToCategory, Place a reference to a data-object into a category
- GetCategoryContents, Retrieve a list of data-objects contained in a category

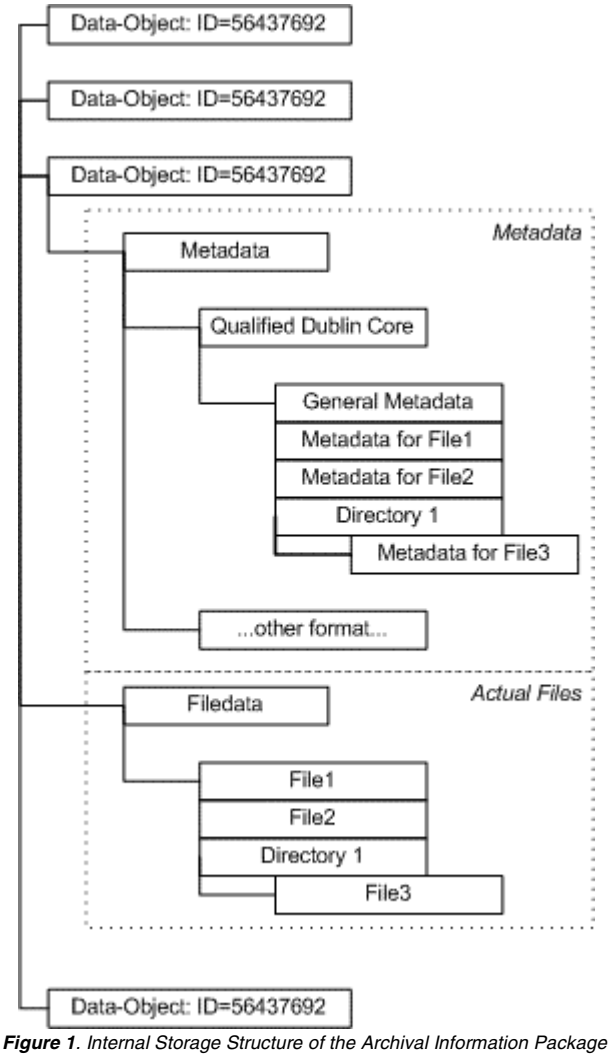

***Figure 1**. Internal Storage Structure of the Archival Information Package (AIP)*

- GetMetadata, Retrieve the metadata of a data-object, or of an individual file inside a data-object
- QueryMetadata, Full-text search the available metadata and return a list of data-objects that correspond to the query.

These are just a number of commands that can be issued and by no means represent the full functionality of the system.

The RPC mechanism for issuing commands is based on simple TCP-IP socket communication. Every command is atomic in nature, in that it performs exactly one operation at a time.

Applying a distributed architecture to this storage layer offers more advantages than redundant storage. Since the servers are implemented as autonomous entities, adding new servers is as simple as starting up a new instance, and telling that instance where it can find other servers. Furthermore, queries over large amounts of metadata can be forked over all the servers, which can significantly speed up search result retrieval time.

Internal Storage

Every data-object will be stored as-is on the server's underlying file-system. For each data-object, a folder is created named after the data-objects identifier. This folder will contain three subfolders (see also fig. 1):

- metadata, containing the XML metadata documents,
- filedata, containing all of the individual files contained in the data-object.
- Mgmdata, containing managementdata used by the server system itself.

The metadata folder can be subdivided into folders for each metadataformat used by the client. Underneath each format-folder, one general metadata document is maintained describing the contents of the entire data-object. Apart from that, a metadata document is maintained for each individual file in the data-object. The organization of the metadata document follows the internal organization of the data-object.

The filedata folder contains all of the individual files and can be organized in a folder tree-structure, just like common file-system implementations. It is this internal organization that is mirrored in the metadata folder for each metadata format.

The managementdata holds information about the data-object itself, such as creation and modification dates, operations performed on the data-object, and membership of categories. It can also contain client-specific information, such as workflow information.

Categories can be used to categorize the data-objects. Every data-object is always part of the so-called 'root' category, and any data-object can be added to every category. This means that data-objects can be a member of several categories. The categories themselves are organized as a simple tree structure. One category can contain several subcategories, and subcategories will always have exactly one parent.

As was said before, the implementation of the storage system is to be as generic as possible. Interpretation of the metadata or any other kind of intelligence should be implemented by the client, following the data and services model.

User Interface: Web Application

The client we are developing at the moment will be the core application of DANS. It will contain some specific workflow and

management functionality that may or may not be applicable to other organizations. However, we wanted the webapplication to be available for reuse as well. In order to address this, we decided to implement the webapplication as Open Source Software and based on a plug-in framework; in our case the Eclipse Plug-in Framework.

Although the Eclipse Plug-in Framework[10] is generally used as a basis for stand-alone application development, e.g. the Eclipse IDE, it can also be used for web-applications provided that you follow a few simple guidelines. The plug-in framework is based on OSGI Bundles, which defines bundles as individual functional components. A bundle-developer can define which parts of the implementation can be 'enhanced' by others by providing an extensionpoint. This extensionpoint provides, among other things, an Interface that the enhancing developer should implement. When the extension has been written, it can be hooked into the plug-in registry, which will make it immediately available to underlying plug-ins and applications.

Every part of the webapplication that can be enhanced, or could require an organization- or process-dependent implementation will be built as an extensionpoint. When a third party adopts the webapplication for its own use, customization can be achieved by providing suitable extensions; this means implementing actual classes in Java. Although programming knowledge is required to customize the webapplication, this approach offers a wide range of flexibility to anyone interested in using our application.

Of course, anyone could implement an entirely different (web)application that communicates with the storage-system. The API will make it possible for anyone to use the functionality of the storage-system.

Conclusion

The amount of time an archivist has to spend to prepare a deposit for archiving can cause clogging of the ingest processes which will either stall or eliminate the possibility of reuse of datasets. In order to guarantee that deposited datasets will be disseminated even though the amount of datasets that will be deposited exceeds the normal processing capacity of DANS, some fundamental changes are needed in the archival process.

The Two-Stage archival process can prove to be a good compromise:

- datasets can be archived and republished relatively quickly
- a subset of datasets will be archived extensively

We are working on the implementation of an information system that will enable researchers to deposit and download research data and enable archivists to work more efficiently.

Although there are still some uncertainties, such as the implementation of DeepStore solutions, we feel that we have made a good start with the implementation of the EasyStore system. It is based on some of the best practices from the archiving field, and will be scalable to future needs.

Future Work

We have scheduled our first release of the system for August 2006. After the first release, DANS will begin using it for its daily work. We are anticipating additional development work to implement additional wishes and requirements that will arise when archivists and researchers will actually start using it.

The first release will be made available under an Open Source license, and we would like to encourage other organizations to try it, provide us with feedback, and possibly join us in future enhancement of the software.

The DeepStore concept will become more concrete during the following months. Issues like the selection process for datasets that are eligible for DeepStore dissemination must be tackled, and projects will be started to actually disseminate selected datasets for instance through the NESSTAR system.

References

[1] Data Archiving and Networked Services Website, http://www.dans.knaw.nl/nl/ (2006-03-09).

[2] Royal Netherlands Academy of Arts and Sciences Website, http://www.knaw.nl/ (2006-03-09).

[3] Netherlands Organization for Scientific Research Website, http://www.nwo.nl/nwohome.nsf/pages/SPPD_5R2QE7_Eng (2006-03-09).

[4] Dublin Core Qualifiers, http://dublincore.org/documents/2000/07/11/dcmes-qualifiers/, 2006-03-09

[5] euroCRIS, http://www.eurocris.org/en/taskgroups/cerif/

[6] Open Archives Initiative, http://www.openarchives.org/

[7] NESSTAR Ltd. Website, http://www.nesstar.com (2006-03-09).

[8] ISO Archiving Standards, http://nost.gsfc.nasa.gov/isoas/

[9] LOCKSS Program Home, http://www.lockss.org/index.html (2006-03-09).

[10] Eclipse, http://www.eclipse.org/

Author Biography

Rutger Kramer received his MSc in Information Technology from the Technical University of Delft (2004). At that time, he was employed as a Technical Scientific Researcher at the Netherlands Institute for Scientific Information Services. He is currently working for Data Archiving and Networked Services as an Information Scientist. In this capacity he is responsible for the development of the EasyStore system, as well as carrying out several R&D related projects.

Laurents Sesink studied history at the University of Utrecht and historical information science at the University of Leiden. He worked during 1995-2002 at the Netherlands Institute for Scientific Information Services on different large scale Research & Development projects as senior specialist digitisation services, technical scientific programmer and coordinator of a software development group. During the period 2003-2006 he was worked as a senior policy advisor at the Dutch Academy of Sciences within the subject of scientific and administrative information.
He is currently employed as an Information Scientist at DANS and his focus is on fundamental issues regarding accessibility to digital scientific data.