Constructing a Records Archiving System Using Off-the-shelf Tools - A Lightweight Approach

Jan Askhoj
Graduate School of Library, Information and Media Studies
University of Tsukuba, Japan
s0621343@ipe.tsukuba.ac.jp

Mitsuharu Nagamori
Graduate School of Library, Information and Media Studies
University of Tsukuba, Japan
nagamori@slis.tsukuba.ac.jp

Shigeo Sugimoto
Graduate School of Library, Information and Media Studies
University of Tsukuba, Japan
sugimoto@slis.tsukuba.ac.jp

ABSTRACT
Content Management Systems (CMS) are widely used for organizations to publish information, to keep transactions and records, and so on. By this wide acceptance of electronic documents and records, organizations are facing demands for the safe archiving of electronic records in their repositories. However, in general, CMS in use today do not offer the required level of functionality for organizations which have a responsibility to maintain their records. It therefore becomes necessary to transfer the records to be retained to a Records Management System (RMS).
CMS and RMS are seldom interoperable out of the box, making archiving of retained records difficult. Reference models designed for dedicated archives are difficult to apply effectively in private organizations or companies. These models tend to be large scale, and contain a lot more functionality than is needed by an organization looking for a way to improve their internal archiving processes. So up to now, the solution has either been to transfer records to the archive by hand, or to use custom programs for records transfer that are made to match the software and hardware profile of the organization. Neither of the above solutions is optimal, so in this paper, we propose a lightweight approach to the problem to organize an archiving system that integrates Content Management and Records Management software. Our approach is based on a three layered model for the organization of an corporate records management system. The model allows the connection of one or more off–the-shelf CMS to a RMS by making it possible to transfer and ingest retained records for archival automatically. We plan to use the model to build a system named ATLAS (Automated Transfer Lightweight Archive Solution). ATLAS makes use of existing technologies and offers built in metadata schema conversion.
With ATLAS, we aim to produce a simple solution for the automated transport of records and metadata between Content Management and Record Management Systems.

Categories and Subject Descriptors
D.2.12 [**Software Engineering**]: Interoperability – *Interface definition languages.*

IWAW'07, June 23, 2007, Vancouver, B.C.

General Terms
Management, Standardization

Keywords
Archives, Records Management, Data Transfer

1. INTRODUCTION
In companies and organizations, there is a need to archive business critical content, regardless of form or media, in a secure consistent manner. There are many reasons for this, such as safeguarding vital information, improving efficiency and productivity by helping search and retrieval. Recently, following the Sarbanes Oxley Act and other legal measures in the US, having a proper records management program in use is vital for ensuring regulatory compliance. To ensure such compliance, most larger companies/organizations need records management systems with specialized functionality [1].

While some systems exist that can manage content and records throughout the entire content lifecycle, this is far from the norm. (We define a record as a piece of content that has reached its final form, and that is deemed archive-worthy according to the organizations retention schedule) Implementing a system with enough functionality to handle different types of content throughout all stages of its lifecycle is a costly endeavor. Organizations that want to avoid this and still need specialized records management functionality therefore need to transfer records from their place of creation to an RMS. Because of the large amount of records in organizations today, as well as the complexity of the many systems involved, the process of getting records in to the RMS can be a problem [2].

To make matters worse, many organizations today use a number of different tools to help with the content creation and management process, depending on the task. Examples of this are Document Management, Web content management, Digital Asset Management etc. Common for all these systems is that they contain content that needs to be retained in a secure manner, according to archiving policy. This means a situation where organizations are left with important records in a number of different systems that are not designed with archiving in mind, and that have insufficient functionality to guarantee the safe, long term preservation of records.

Therefore, a model that integrates both content creation from multiple sources as well as archiving functionality is desirable, as

it would allow organizations to manage their retained records from creation to destruction, using different off-the-shelf CMS and RMS solutions, which is less costly than buying or building specialized systems, or modifying existing systems on an ad-hoc basis.

Since our objective to integrate two types of existing systems to create a complete archiving solution, we will not go into detail in specifying the functionality and design of the CMS and RMS. The important point is to create a lightweight, working framework for the transfer of data. Such a framework will have the advantages of having fewer functional entities and thus being easier work with than a more comprehensive model.

In this research, we propose a three layered model consisting of Content, Metadata and Transport (meta-metadata) layers. This approach is useful in cases where conversion between different data formats is necessary. The model we propose will serve as a base for ATLAS (Automated Transfer Lightweight Archive Solution). ATLAS is a system that will implement the three-layered model and will be designed to integrate content from multiple CMS systems with different metadata schemes, by using a metadata schema database. This database provides metadata scheme conversion tables, enabling dynamic metadata exchange between the CMS and RMS.

Another key feature of ATLAS is the use of RSS as a transport protocol. RSS is a lightweight and flexible protocol used to aggregate content on the Web. RSS easily extensible, and is supported by most CMS systems, which makes it ideal for use in our system.

In this paper, we define a "content item" as an intellectual unit, rather than a technical unit. In other words, content can be a collection of different technical units (text blocks, images, attached files), but it has been designated as one coherent piece of information (e.g. a report) by an organization. Although content can be defined in many ways, we have chosen this approach, because we believe that it fits in with traditional corporate archiving methods [3].

2. BACKGROUND

There already exist a number of models for transferring electronic records to a repository or archive. Models such as the OAIS are generally applicable, very detailed, and offer a high degree of functionality. However, these kinds of system are aimed at archival institutions or large digital archive systems, where the scope is much broader than in a typical private organization. In smaller organizations, using a lightweight approach to the problem would be preferable to adopting such a comprehensive model. Therefore the solution has often been to transfer records deemed archive-worthy by hand, or to use custom systems for records transfer that are designed to match the software and hardware profile of the organization.

Both of these methods are problematic. Transferring records from one system to another by hand is both time consuming and error-prone. The larger the amount of records and metadata, the more costly the process becomes. Building specialized programs for transferring records, such as export-import scripts also has drawbacks: Since the programs have to be custom-built to fit the systems in use, in-depth knowledge about these system is required. Furthermore, custom built programs or specialized adaptations of existing programs can be expensive. And finally, in case of a system migration or changes to the APIs used on either the content creation side or records management side, the programs might have to be rewritten.

Getting digital content from multiple sources into a common repository is in some ways similar to existing Web Archiving projects, where the object is the ingest of content in order to ensure long-term preservation (for example The Internet Archive) [14]. One of the benefits of this approach is that it eliminates the cost of manually submitting content and filling in metadata, by reading and importing selected content automatically. On the other hand, a problem with crawler-based web archiving is that dynamically generated content (such as personalized pages) and metadata can be hard to capture.

In corporate records management, it is also desirable to eliminate the submission process. After all, the time spent submitting content manually costs money. But unlike web archives, it is absolutely essential that all published information to be archived is captured completely and in a format true to the original. Luckily, in a private company or organization, content to be archived is captured from known sources. In other words, the CMS used as source for the ingest, while they may be different, are all within the control of the organization in question. This makes it possible to make a model that where all the entities (CMS, RMS and the connecting data transfer) can be adapted to fit the organizations needs.

Another problem of conventional records transfer methods is metadata. Because of the differences between Content and Records Management the metadata schemes in use in both kinds of system will likely differ as well. This problem only gets worse when transferring content from multiple sources, in which case you have to deal several different metadata schemes. The need to design a model of interaction between the CMS and RMS comes from the fact that these two system types are unable to exchange data necessary for records management out of the box. In a CMS, managed content can be made up by any combination of elements from a number of sources, formatted according to display preferences. As an example, a piece of content can be a list of inventory in stock, composed of data from two different back-end databases, presented using a specific template. Related to this piece of content will be a certain amount of metadata.

These facts cause a number of problems when wanting to add the content to a generic RMS.

- First of all, not all content in a CMS should be archived, so a records declaration process needs to take place.
- The content may not be in a format that can be added directly to the RMS. In order to ensure long-term preservation, some organizations may wish to convert content into different file formats in order to ensure long-term usability, e.g. from Microsoft Word to PDF.
- Because of the inherent differences between content management and records management, metadata selection and conversion from one scheme to another will most likely be necessary. Where scheme conversion is concerned, three scenarios can be imagined. (1) The necessary metadata in the CMS fits the RMS metadata. (2) The necessary metadata exists in the CMS, but does not fit the RMS metadata scheme. (3) The metadata needed by the RMS does not exist in the CMS. Of these three scenarios, 2 and 3 will need to be addressed for successful metadata transfer to take place. If more than one CMS is used in the model, the conversion between schemes becomes $N \rightarrow 1$, as for example a CMS for Digital Rights Management may have a totally different set of metadata compared to a CMS for Web Publishing.
- While it is true that most current CMS and Records Management Systems support web protocols such as HTTP

and FTP, they are not designed to exchange data in a way that allows easy transfer and ingest of records.

An ideal solution to the above problems would be totally invisible to users and archivists. When a piece of content is finalized, they should not have to concern themselves with its archiving.

3. A LAYERED MODEL FOR CONNECTING CMS AND RMS

To address the problems listed under Background, we have developed a new model for integrating CMS and RMS functionality to create a complete archiving solution. Our model differs from existing models in that it works with many existing of off-the-shelf systems, while still being lightweight. It also solves the problems of content transfer and metadata conversion, and offers an automated approach to creating and managing content in native systems, while allowing them to remain compliant with records management regulations.

3.1 Three Layered Model

Figure 1 gives an overview of our three layered model. The left side represents the CMS and the right the RMS.

The first layer is the content layer. Content is produced or imported into the CMS, where it will most likely undergo a number of revisions. When the revised content is eventually published in one form or another, it will be transferred to the RMS, according to rules defined by layer three.

The second layer represents the metadata about the content. Metadata will almost certainly be different in the CMS and RMS. Generally speaking, in basic CMS, the focus is on managing and producing content. The CMS support this process by offering functionality such as versioning, keywords etc. All of these functions can be described as different types of metadata, such as descriptive metadata, administrative metadata and structural metadata. This metadata is all related to the content which it describes and is particular to the type of CMS in question. In an RMS, the focus is on long-term storage of records, and the metadata in use in such a system is tailored to these needs, so metadata is used that supports functionality such as retention and disposal management, record series management etc.

The third layer is the meta-metadata Layer which contains the data necessary for communication between the CMS and the RMS. This layer has several functions.

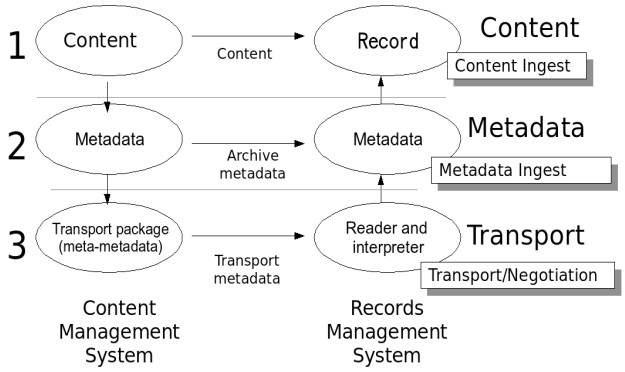

Figure 1. The three layered model

- The first function is to ensure that all relevant data is made available for transfer.
- The content and metadata should be transferred using suitable formats which can be read by the RMS.
- Finally, the third layer must be able to ensure safe transfer of data from one system to another

We have designed the thee layered model to be as lightweight as possible, while still ensuring that all archive-worthy content and metadata is transferred safely to the RMS. By lightweight we mean:

- It has only a few entities.
- It works with off-the-shelf tools.
- It uses simple protocols.

These factors form the basis of our approach, and will help facilitate the implementation of a system designed using the three layered model in a corporate setting.

3.2 ARCHIVE SYSTEM USE CASE

Unlike traditional archives and public electronic repositories, the main purpose of corporate records management is to support business processes and policy, enabling organizations to handle their own records effectively and in accordance with law. This lends some unique properties. Among these are: Shorter retention periods, the use of original document formats rather than conversion, automated workflows, and finally a high degree of integration between systems, procedures and tools. This integration is made possible by the aforementioned handling of the entire records lifecycle within a single organization.

Any model to be used in an corporate setting must take these properties into account, as they have an impact on the records management process. To illustrate how our model would work in practice, we have prepared a hypothetical use-case.

The dataflow is explained in the use case below:

3.2.1 Content Management System

The left part of our model represents the CMS, split in three layers. Our model can operate with more than one CMS, so the left side can be thought of as several entities, all represented in the three layered model.

(1) CMS Layer 1 - Content

In the CMS, new content is created or imported into the CMS from an external source. The content is revised and updated, using content management functionality in the system. At some point, it is considered final, and if it is marked as a retained record in the organizational retention schedule, it needs to be archived.

(2) CMS Layer 2 - Metadata

At the time of creation, some metadata will be created automatically (publishing status, creator name, time of creation etc), and some metadata may be added manually (keywords, subject etc). CMS functions such as versioning, check in/out, also add information to the content, and as such generate metadata.

All of this metadata is managed according to the metadata scheme used in the CMS. Some of this metadata may be relevant when archiving the content, while some may not.

Apart from the metadata used by the CMS to manage content, most content types themselves also contain metadata of some sort. For example, Microsoft Word documents have built in property fields for keywords and comments. This metadata will be transferred along with the originating file, and not be converted or otherwise managed by an ATLAS system. The reason for this is

that because of the relative short retention schedules in use in corporate RMS, documents or spreadsheets are mostly stored in their original format. This extends to any document-native metadata.

(3) CMS Layer 3 - Transport

Level three is responsible for selecting the content and metadata, and transporting it to the RMS for ingestion. Content selection is initiated according to parameters from the company retention schedule. For example, all content tagged "contract" must be archived. In the case of metadata, all available metadata from the originating CMS post is included in the transport package, and will be read and converted by the RMS Level 3 Reader and interpreter.

A transport package contains three elements: Content, Metadata, Transport Metadata. The elements of the package are made up of:

- Content: Content is actually an electronic copy of the post in the CMS. In the system, all the data that makes up the competent package will be stored in one or more databases. Content can be stored as plain text or HTML, code such as Java Script or PHP embedded in the page and mime-type file attachments such as Word or PDF.
- Metadata: Metadata is similarly stored as fields in a database, and the format and values it takes will depend on the metadata scheme used for the CMS.

  In order to advertise what metadata schema is used for the metadata in the package, a URI linking to a Metadata Authority is included in the package. The Metadata Authority is a database containing a list of used CMS metadata schemes and conversion tables (see 4.2 for more details).
- Transport package: On the CMS side, The transport package collects and packages the data to be sent to the RMS, namely the Content, Metadata and transport metadata. Transport metadata is generated by layer 3 to ensure safe and successful data transfer, and to help with metadata conversion. When a package is made available, it includes a metadata-schema ID for the metadata it contains.

3.2.2 Records Management System

The right part of our model represents the RMS part of the system, also split in three layers.

(1) RMS Layer 3 – Transport

On the RMS side as well, the transport layer has several functions:

- It regularly receives transport packages from the CMS and saves them to a temporary location.
- Using the Metadata Authority URI, the Reader and interpreter can acquire a conversion table for the metadata schema used in the received Transport Package. Metadata that is not needed by the RMS is ignored. Metadata fields that are needed by the RMS but not provided by the CMS are left empty.
- The transport layer is also responsible for logging the receipt of a transport package, and sending an acknowledgment back to the CMS to show the package has been received.

Layer three has an additional important function: To decide if content should be archived or not. Most corporate archives use Retention Schedules (also known as File Plans) to determine what content should be archived. These schedules are divided by record type, and contain detailed instructions for archiving. In our model, archiving is also decided by record type. In other words, all finalized content of a certain type must be archived. When a CMS metadata schema is registered in the Metadata Authority, whatever field in it that defines the Content Type will also be registered, and mapped to the corresponding Archive Metadata field for Record Type. If the value in the Record Type field matches a Record Type to be archived according to the Retention Schedule, the Content will be archived. If the Content type is not mentioned in the Retention Schedule, the content in question, along with its metadata will not be added to the RMS.

(2) RMS Layer 2 - Metadata

Before the converted metadata from the transport package is added to the RMS together with the corresponding content, it needs to be reviewed by an archivist. This is to ensure that no incorrect metadata is added to the RMS, but also to manually fill in any fields with no metadata (where there is no corresponding metadata in the CMS).

Depending on what metadata scheme is in use in the RMS, the transport package itself may also hold information that can be used as metadata in the RMS, an example of this is the date-time the transfer package was added to the RMS.

The metadata will be imported using the import tools built into the Archiving software.

(3) RMS Layer 1 - Content

The content and metadata from the data package will be ingested after the archivist review. In our model, content will be added in the same format as it was in when it existed in the CMS.

The content will be ingested using the import tools built into the Archiving software.

4. TECHNICAL OVERVIEW

The use case described in section 3.2 gives an overview of the flow of data in our three layer model. This model will be implemented in the ATLAS system, create a working archiving system. Since our system will be designed to enable transfer of data between different system types, we have chosen to rely on open technologies such as RSS, XML, SSL etc.

4.1 Protocol

The protocol we will use to transfer content from the CMS to the RMS is RSS 2.0 (Really Simple Syndication). RSS is an XML based lightweight format used to publish frequently updated digital content on the internet. It works by an information publisher providing a content "feed" (a machine readable list of published content) on a web site. This feed contains both an amount of content (sometimes only a preview, such as the first 10 lines of an article is included) and basic metadata. The feed can be downloaded at set intervals by a feed reader/aggregator and read online or offline without the user having to visit the originating site. There are a number of different more or less compatible versions of RSS in use on the net today. We have chosen RSS 2.0 because it is widely supported and can be extended via namespaces [4].

We will use RSS to create transport packages by extending the existing RSS in the Content Management System in the following ways:

- Create tags for the content metadata in the <*item*> fields (the fields in the feed for metadata for each content item) of the RSS Feed. For example

```
<content_metadata:unique_id>
    2007-01-01-0001
</content_metadata:unique_id>
```

- Adding a Metadata Authority URI in the <*channel*> field (the field in the feed for metadata about the feed) of the RSS Feed to help with metadata conversion. For example

```
<content_metadata:metadata_authority>
    http://www.example.com/schema_id/1
</content_metadata:metadata_authority>
```

- Adding "Ping" support for immediate content aggregation. Ping is a RSS service designed to send a notification to predetermined server(s), to let them know that the originating site has been updated with new content. This enables RSS aggregators to harvest new content immediately after publication, instead of at regular intervals. In the ATLAS model, the reader to be notified would be the Reader and interpreter part of the RMS server [5].
- Adding "TrackBack" support in order to verify whether the records transfer has completed successfully. TrackBack is a RSS service for peer-to-peer communication and notifications between two web sites. TrackBack uses a TrackBack ping, which shows, "resource A and resource B are linked". A TrackBack "resource" is represented by a TrackBack Ping URL, which is just a standard URI. In order for TrackBack Ping to work, it must be supported by both CMS and RMS. In the ATLAS model, the implementation of TrackBack Ping would take the following form [6]:
    1. The CMS publishes a page with archive worthy content using RSS. Embedded in the page is TrackBack Ping metadata, allowing auto-discovery of the TrackBack Ping URL. For example: `trackback:ping="http://www.example.com/tb.cgi/1"`, where 1 is the TrackBack ID
    2. The content is harvested by a customized RSS aggregation script.
    3. The RSS aggregation script client sends an HTTP POST request to the TrackBack Ping URL. Example:
    ```
    POST http://www.example.com/tb.cgi/1
    Content-Type: application/x-www-form-urlencoded; charset=utf-8
    url=http://www.rms_url.com/
    ```
    4. The CMS receives the HTTP POST as proof that the published page has been harvested successfully. It then returns a confirmation that the TrackBack Ping has been received.
- Adding support for SSL and HTTP Authorization. See 5.3 Security for more details

4.2 Metadata

It is difficult to generalize about what metadata needs to be exchanged between a CMS and a RMS. This depends on a wide number of factors such as system type, organization type, organizational policy etc. What can be expected, however, is that the CMS and RMS will be using different metadata schemes. As an example to illustrate this, we have chosen two general metadata schemes: The "CMS metadata elements and guidelines" from Monash University to represent a CMS metadata schema, and "Requirements for Electronic Records Management Systems 2: Metadata Standard" from the UK National Archives. Table 1 shows an overview of the two metadata schemes [7].

Table 1. Metadata Schemes

CMS metadata elements and guidelines	Requirements for Electronic Records Management Systems
- Identifier - Title - Subject - Description - Creator - Date - Language - Type - Audience - Publisher - Rights	- Identifier.System ID - Title - Creator - Date.Created - Date.Declared - Type.Record type - Relation.Copy - Relation.Parent object - Relation.Redaction/Extract - Relation.Reason for redaction/extract - Relation.Rendition - Aggregation - Rights.Protective marking - Rights.Descriptor - Rights.Individual user access list - Rights.Group access list - Rights.FoI disclosability indicator - Rights.FoI release details - Rights.FOI release date - Disposal.Schedule identifier - Disposal.Disposal action - Disposal.Disposal time period - Disposal.Disposal event - Disposal.disposal external event occurrence - Disposal.Disposal (due/effective) date - Disposal.Disposal authorised by - Disposal.Export destinationt

Since the metadata schemes used here serve no other purpose than to act as examples and to be used as schema representations when building a prototype of the ATLAS system, we have chosen to use only "Required/Required where applicable" and not "Optional" fields. As it can be seen, some fields of the CMS metadata elements and guidelines are identical to Requirements for Electronic Records Management Systems (for example: Title). In those cases, no conversion is necessary. However, in some cases there may be differences in the data formats used between the two systems, such as the formats used to express Date or Identifier. In these cases, the values will have to be converted, using a conversion table.

As mentioned previously, metadata coming from the CMS in transport packages will most likely have different field meanings and be in a different format than the metadata in use in the RMS. So before import, conversion is necessary. To preform this, our model performs metadata schema conversion using a so-called Metadata Authority. This is a web accessible database containing a list of conversion tables, with its own unique URI. Each table is a RDF schema containing information about the CMS metadata terms and their data formats along with their alternative RMS values. In other words, for metadata conversion to work, it is necessary to register all CMS metadata schemes with the Metadata Authority in advance. The metadata in the conversion tables are stored as XML tags.[8]

Up to now, we have mainly been dealing with metadata relating to individual content. However, where this content belongs in the RMS classification scheme (also referred to as fileplan) is also important. For example, the UK National Archives Requirements for Electronic Records Management Systems specifies 3 levels of metadata: Class, Folder, Record. Many CMSs also support some

form of Hierarchical structure, and there is nothing that prevents structural metadata to be converted and used, providing this metadata is explicitly expressed at the content level. Metadata that is inexplicitly inherited from a higher level can not be transferred, as it is not present at the content level.

4.3 Security

In order to ensure that the transfer of content and metadata between CMS and RMS takes place in a secure manner, the model we propose enables the use of a number of security functions [9]:

- Authentication: HTTP Authentication is a basic authentication scheme in which a user or application is required to input his credentials in the form of a user name and a password in order to gain access to a site. In HTTP Authentication, the user credentials are sent over the net in plain text, which makes interception by a third party possible. But if used together with HTTPS/SSL, the transmission becomes encrypted and thus secure.
- Confidentiality: HTTPS/SSL is used to ensure data confidentiality by encrypting the data stream between two communicating applications, and to authenticate the server, and client (optional). It can be used with the RSS protocol.
- Transfer verification: Using the TrackBack Ping functionality described previously, it becomes possible to verify that the Transport Package has been sent and received successfully. It should be noted that this does not guarantee that the content has not been altered or become garbled during transfer. The TrackBack Ping functionality does not guarantees the integrity of the contents of the Transport Package. For this, checksum or a similar kind of data verification is necessary.

5. Related Works

Setting aside the number of ad-hoc solutions in existence, there are already a number models that deal with records transfer and ingest. A well known standard is the OAIS (Open Archival Information System) [10].

According to the Reference Model for an OAIS paper, The purpose "Is to define the ISO Reference Model for an Open Archival Information System. 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.". The OAIS model may be applicable to any archive, and contains recommendations for a number of archival information preservation functions including ingest, archival storage, data management, access, and dissemination.

There are a number of differences between the OAIS and the model we are proposing. First of all, it must be said that OAIS and ATLAS differ in their approach. The object of OAIS is to describe an entire archive system, whereas ATLAS deals with integrating generic CMS and RMS, to make an archiving system without defining these entities in detail.

One of key points of the OAIS model is that it is focused on long time preservation. This is handled by the Preservation Planning entity, which functions include evaluating the contents of the archive and periodically recommending updates to the archival information and guaranteeing data accessibility even if the original computing environment becomes obsolete. Another characteristic of OAIS is the large number of functional entities and subsequent high number of functions provided. An example of this is the Ingest entity, which has 5 sub-functions, each responsible for carrying out a number of actions. This makes OAIS a very complex model its entirety. In an corporate setting, where records management is just one of many business functions that need to be carried out, only a subset of the OAIS would be needed to build a workable archiving system. Also, the high level of functionality to be got from the different OAIS Entities and the interaction between these, make the OAIS unsuited for implementation with the simple systems and procedures in use in organizations not specializing in archiving.

An example of this is the OAIS use of Information Packages (IP) for packaging content and metadata. An IP is a conceptual container consisting Content Information and Preservation Description Information. Content Information is the original target of preservation, and Preservation Description Information is the information needed for preservation purposes to ensure the Content Data Object is clearly identified and to understand the environment in which it was created. In a corporate RMS Preservation Description Information dealing with provenance, history and fixity (e.g. checksum data), will most likely be overkill, since long-term preservation is not as important. Many corporate records have a retention period of only 3-5 years (depending on country and record type).

Figure 2 shows an overview of the OAIS model and its main entities.

In such a setting, where the focus is on integrating off-the-shelf systems, the ATLAS model has an advantage in the fact that it makes little demand on the CMS and RMS and their functionality. It can also be kept comparatively simple by not using Information Packages that explicitly define Preservation Description Information. Finally, unlike OAIS, records do not necessarily need to be independently understandable (understandable without the assistance of the content producers). It is not uncommon for records in a RMS to be of no use to anyone but the producers, but they may still need to be archived because of legal, contractual or other obligations.

Another model that shares similarities to ours is the The Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH). This protocol is also built on existing tools, such as HTTP and XML, and harvests metadata of any format specified by a community, based on unqualified Dublin Core [11][12].

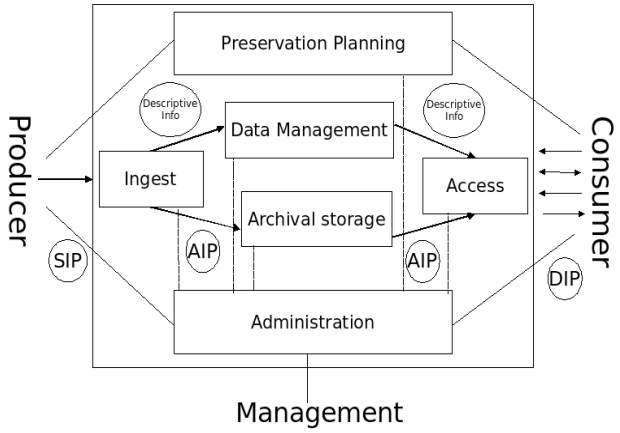

Figure 2. The OAIS Model [10]

OAI-PMH is useful as a protocol, and could with some modifications be used in a system built on the ATLAS model, but as it is designed for metadata harvesting, it cannot transfer content. Furthermore, as a harvesting protocol, it does not in itself contain functionality for reporting back to the data provider whether the data transfer has completed successfully. Finally, as OAI-PMH stands today, specialized metadata schemes is supported, but the protocol requires that servers do use unqualified Dublin Core metadata in XML as a basis. ATLAS makes no such requirement.

To sum up, ATLAS differs from other archival reference models, in that it is designed exclusively to ensure that archive worthy content and metadata can be transferred from CMS to a RMS, irrespective of type and functionality. Similar to OAI-PMH, ATLAS uses a harvesting approach, saving the users the task of going through a submit-process. Additionally, it offers built in conversion of harvested metadata which means that organizations can use their own custom metadata schemes for both CMS and Archive. On the other hand, since there are no metadata schema requirements, manual additions are necessary in those cases where required archival metadata is not present in any form in CMS.

6. IMPLEMENTING ATLAS

Three major components are needed to implement an ATLAS archive:

(1) One or more CMS with support for RSS and the RSS extensions described under "Technical Overview". The CMS could in principle be of any type, as long as the content can be embedded in an RSS feed. Since the object of ATLAS is to store business critical records, all embedded content must be transferred to the RMS to guarantee the completeness of the record. This means that at the time of publishing, the CMS must simultaneously publish (make available for the export module) all the content that is needed to make a complete record (intellectual unit).

(2) An RMS with import functionality supporting metadata in XML.
As mentioned previously, metadata coming from the Metadata Authority is formatted as XML, and the RMS needs to be able to read this during the import process.

(3) The ATLAS Add-ons, consisting of Export Module, customized Feed Reader, Metadata Authority and Import Module.

Once ATLAS has been set up, the archiving will take place automatically, freeing the content owner from having to manually add records to the archive. A piece of content will be archived depending on settings in the CMS itself.
In ATLAS, all content is published in a Transport Package using RSS as soon as it is finalized. The Transport Package is read by a customized Feed Reader, which also sends metadata to the Metadata Authority for conversion. After the metadata has been converted, it will be transferred to the Import Module, provided the content type matches the types included in Retention Schedule. After a Trackback package has been sent back to the originating CMS, the content needs to be approved by an archivist to correct metadata and add missing fields.

At its current stage, ATLAS has a some issues that need to be taken into account when implementing it.

The first of these has to do with the way metadata is handled. If the metadata provided by the CMS is very limited, or if it is very different from the one in use in the RMS, the archivist will have to fill out the missing data. This can be a very resource intensive task, if the number of transferred records is large. Furthermore, unlike the creator of the content, the archivist may not know enough about the content to add the correct metadata. One way to solve this problem is to extend the metadata schema used in the CMSs to contain the metadata fields used in the RMS.

The second point is that content is transferred to the archive "as is". While retention periods tend to be short in corporate archives, some provisions may need to be made for ensuring future readability. In systems where long term data preservation is an absolute necessity, it is possible to imagine a conversion process taking place at the RMS Content layer before import (similar to the National Archives of Australias XENA software) [12].

The third problem is more technical. We have tried to make our model flexible enough to work with any type of CMS and RMS, without having to any of these systems. However, some systems may not support the basic functions mentioned previously in this chapter. In those cases, this functionality will need to be added by hand, making implementation more costly.

These issues aside, implementing ATLAS should be relatively straightforward for most corporations. This simplicity has been achieved by a number of different means. First of all, ATLAS has an advantage in the fact that its elements all lie within the control of the organization, making it possible to customize. It can work with off-the-shelf tools, which also means that is doesn't require changing or extending of the systems in use in the organization.

ATLAS also doesn't specify provisions for ensuring long-time readability, leaving this up to the organization. And finally, by being lightweight ATLAS is easy to understand, adapt and implement.

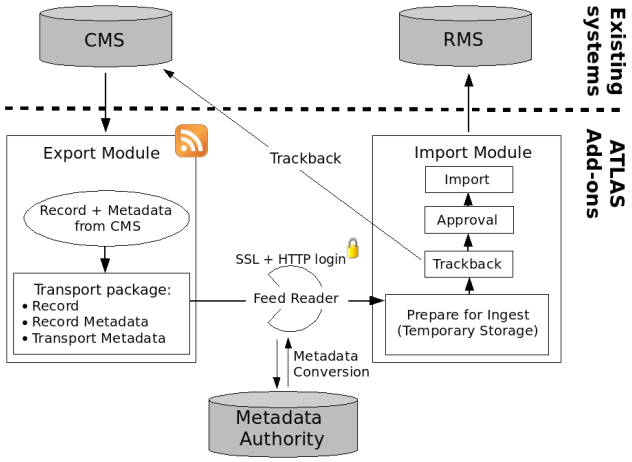

Figure 3. The ATLAS workflow

7. CONCLUDING REMARKS

In this paper, we have presented a new model for building an archiving system. By using a lightweight three layered model, we have reduced the complexity of integrating content management and records management functionality compared to other models of digital archiving and harvesting of web content. There is of course always a trade off between simplicity and functionality, and in some cases, our model may turn out to be too limited. But we believe that the openness of our model makes it possible to extend the basic functionality by adding extra functions such as content conversion. The fact that our model doesn't require

reprogramming of the organizations CMS or RMS, coupled with the fact that it is not tied to any one software solution makes it even more generally applicable.

The current ATLAS uses a Metadata Authority for storing and providing schema conversion tables. We believe that this solution is preferable to doing ad-hoc metadata conversion based on conversion tables hard-coded into the Reader and Interpreter, because it enables the Reader and Interpreter to perform metadata conversion without knowing what schemes are in use where. This would for example be convenient if a new CMS is added to the system. We would like to expand on this topic in future research.

At this point, our model is only a theoretical model. In order to test the solutions we have presented in this paper, we have plans to implement them by creating a real archiving solution using ATLAS.

8. REFERENCES

[1] Public Record Office. *Requirements for Electronic Records Management Systems, 2002 revision: final version.* Crown copyright 2002

[2] Mybrugh, Susan "Knowledge Management and Records Management: Is There a Difference?", RIMR, Vol 14, No. 7/September 1998.

[3] Boiko, Bob. Content Management Bible, 2nd Edition. Wiley Publishing, Inc., Indianapolis, Indiana, 2005

[4] RSS 2.0 Specification, RSS Advisory Board, 12 August, 2006, http://www.rssboard.org/rss-specification

[5] The Blog Herald, Understanding Blog and Ping, 2005, http://www.blogherald.com/2005/08/16/understanding-blog-and-ping/

[6] Sixapart, TrackBack Technical Specification. http://www.sixapart.com/pronet/docs/trackback_spec

[7] CMS metadata elements and guidelines, Monash University, http://www.lib.monash.edu.au/metadata/cms-metadata.html

[8] RDF Vocabulary Description Language 1.0: RDF Schema,W3C Recommendation, 10 February 2004, http://www.w3.org/TR/rdf-schema/

[9] Garrity, Steven. Private RSS Feeds: Support for security in aggregators. Silverorange, 2003

[10] Reference Model for an Open Archival Information System (OAIS), Consultative Committee for space Data Systems, Blue Book, Issue 1, 2002

[11] Open Archives Initiative Protocol for Metadata Harvesting - Version 1.1. 2001, http://www.openarchives.org/OAI_protocol/openarchivesprotocol.html

[12] Lagoze, Carl; Van de Sompel, Herbert. The Open Archives Initiative: Building a low-barrier interoperability framework, 2001, http://www.openarchives.org/documents/oai.pdf

[13] Digital preservation software applications, National Archives of Australia, http://www.naa.gov.au/recordkeeping/preservation/digital/applications.html

[14] The Internet Archive, http://www.archive.org/index.php