DRS Futures: Separation of functional areas

Stefano Cossu stefano_cossu@harvard.edu

05-15-2023

Contents

1 Scope

This document defines a conceptual design with the goal of separating DRS Futures functionality into two discrete areas fulfilling separate but inter-dependent tasks of a digital preservation ecosystem: one dedicated to long-term archiving, the other to frequently revolving digital resource management.

This conceptual design has informed several technical requirements that we can measure our software product choices in Phase 2 against. As such, this document does not define a software architecture; it rather describes an abstract set of functions that may be fulfilled by of one or many applications, either home-built, acquired, obtained by adapting existing applications, or a mix of the above.

2 Rationale

During Phase 1 of the DRS Futures project, many discussions aimed at gathering stakeholder needs revolved around digital resource management (deposit, update, access, reporting) rather than around digital preservation. Since some Harvard staff use DRS as the only permanent store for their data, this naturally doubles as both management and archival platform.

There are several reasons for having separate archival and management areas:

1. Users often need a place to store their work that is safer and more easily shareable than their workstations, without the need to preserve its full change history. By offering a management service with a separate option to commit lumps of changes to a long-term preservation service on demand, they may benefit both from long-term safeguarding and a more accessible platform.
2. From stakeholder input and early requirements drafts, it emerged that supporting content versioning is one of the requirements of DRS Futures. This means greatly increased complexity and volume of the archival store, especially in terms of number of resources to be handled and indexed. Keeping a non-versioned store for fast indexing and access separated from a fully versioned store for archival and historic research seems necessary to ensure access to DRS information in a way that is compatible with future growth.
3. OCFL, which is for the foreseeable future the bedrock of our preservation services, is specifically designed for archival purposes. Day-to-day management implies a different set of requirements. In order to keep the OCFL specification sustainable in the very far future, keeping its scope narrow is key to slowing down changes of the spec as much as possible.
4. In general, separate functional areas would allow for a better separation of concerns, user access, and the choice of dedicated products for each area.

3 Proposal overview

3.1 Key points

- An area for ongoing digital resource management (called "workspace" hereinafter) and an archival area are developed and maintained independently.
- Workspace and archival requirements may be fulfilled by different applications and data stores.
- The workspace is not a mandatory step for all data.
- The workspace presides over:
  - Frequent updates of contents;
  - Fast content access and discovery;
  - Content modeling and validation;
  - User access policies.
- The workspace is unaware of:
  - Versioning and historic data;
  - OCFL semantics.
- The archival area presides over:
  - Allocation and retrieval of resources in the OCFL store;
  - Long-term archival services;
  - Versioning information;
  - Complete audit trails and preservation metadata.
- The archival area is unaware of:
  - Access policies (only a special group has access to this area);
  - Content model;
  - The workspace as a whole.
- The workspace may act as a gateway for some archival services.
- Resources may be pushed to the archive on demand, which creates a new version in the archival store.
- Users have an option to "shelve" resources, i.e. delete them from the workspace after they are archived.
- Users may request previous versions of resources from the archive without mediation.
- The workspace store can be entirely rebuilt from the archive.

3.2 Topology

At the core of this approach are two functional areas: one dedicated to long-term storage and disaster recovery operations, the other one to mid-term storage and management of digital files and metadata. Each of the areas consist of a set of services and one or more data stores (Fig. 1).

Fig. 2 illustrates the relationships between various functional areas of a OAIS according to the OAIS specification.

The key difference between the OAIS model and our functional area design is that the workspace described here does not correspond to the "data management" area in the chart. The former is external to the preservation workflow, stores data in a format that is functional to fast-turnover management and has no concept of SIP.

The archival store is OCFL as we know it, with added services that preside over typical digital preservation functions: creating archival packages, disaster recovery, managing and retrieving versions, fixity checking, etc. The expectation for these services is that they function without any change to the underlying OCFL fabric.

The workspace consists of a set of services, exposed via API/s and Web UI/s, allowing users to manage contents that don't otherwise belong to any other CMS. This means that the workspace is only one of multiple source systems for the archival store, which should have no awareness of or dependency on it.

The workspace store consists of a metadata store, likely a DBMS, and a file store, e.g. an attached POSIX file system or S3 object store.

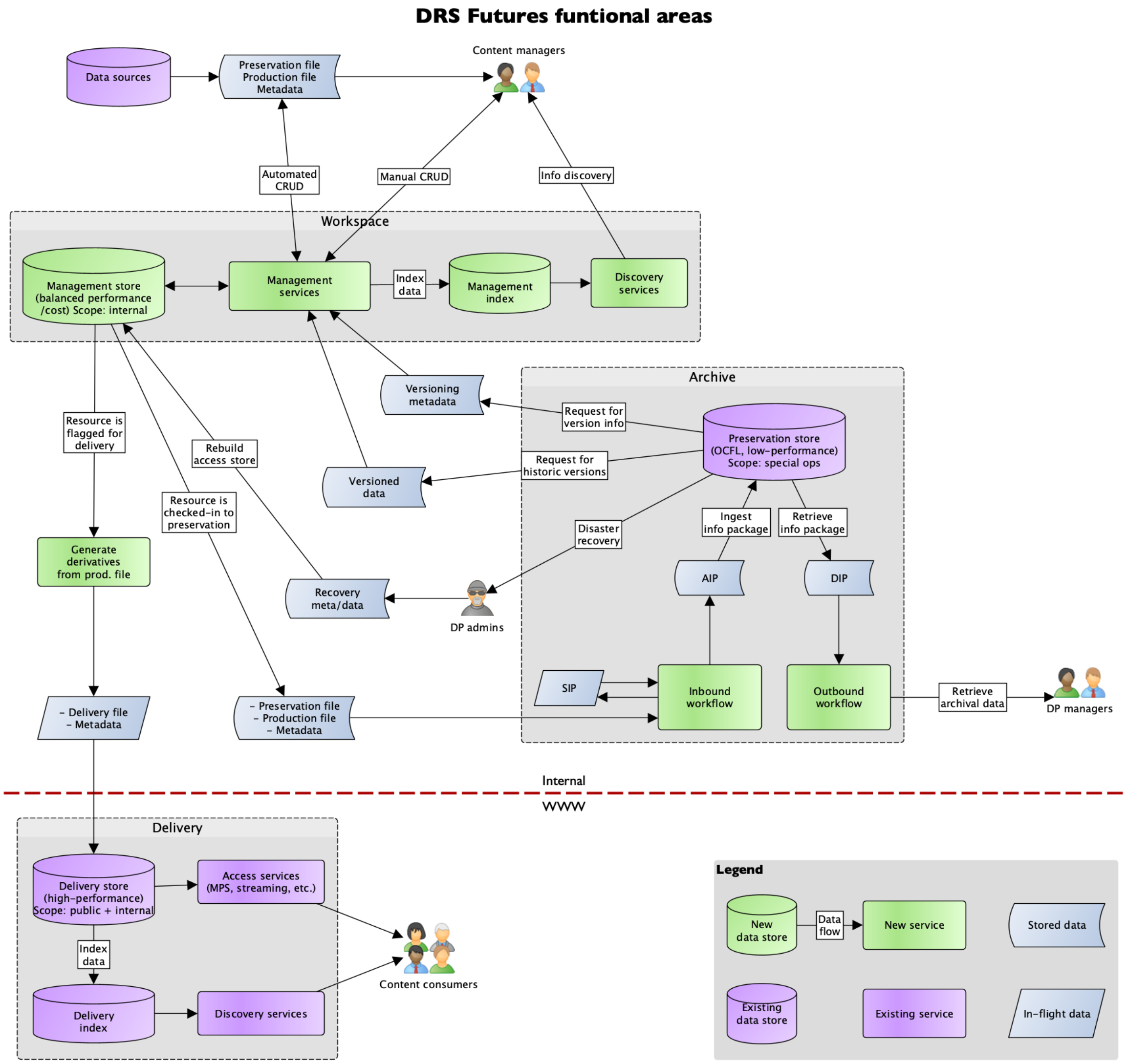

Figure 1: workspace + archival areas

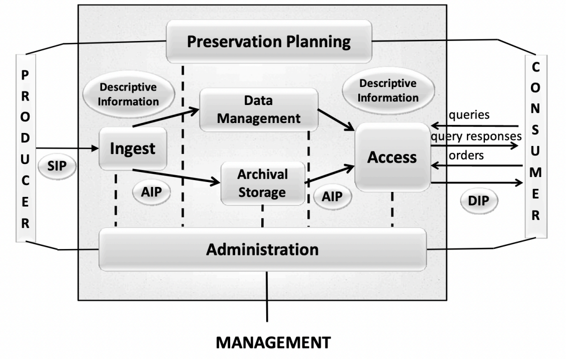

Figure 4-1: OAIS Functional Entities

Figure 2: OAIS functional entities

The metadata store provides key data for indexing and feeding internal search engines. It holds all the technical, descriptive, and admin metadata (including inter-resource relationships) to allow full editing capabilities. It also holds the mapping to file locations in both the workspace and archival stores, so that the two can be easily crosswalked.

The workspace file store may follow in part the OCFL layout. The main departure from OCFL, however, is the lack of a versioning structure. In fact, update of a file in the workspace store causes an overwrite of that file. Also, as explained above, metadata are not kept in the filesystem but in a database.

Both file and metadata workspace stores are considered "safe enough" for mid- term retention of data. They may be supported by fault-tolerant volumes, etc., but do not guarantee long-term preservation. It should be reasonably safe to leave data in the workspace for weeks or months, as long as at a certain point the data are pushed to archive.

The workspace store is considered disposable, in that all the data it contains can be rebuilt from the archival store. In case of a disastrous data loss in the workspace store, the only data lost would be updates that have not yet been pushed to archive.

The archival services preside over the maintenance and navigation of the OCFL fabric. They have no knowledge of content model, access policies, and such semantics. They should be accessed by DP specialists with special permissions. Some indirect access of preservation data through the workspace services may be mediated by the workspace services' access controls.

3.3 Process

Users can access the workspace to iteratively assemble, review, update, and publish both simple digital objects and complex hierarchies. They can search by arbitrary fields, view, edit, add or remove relationships, assign or unassign access policies, and anything that a generic CMS would do.

The workspace is also where content models are defined and validated, so the services presiding over it have some awareness of the semantics of the contents.

For complex digital objects or sets thereof that require long-running iterations of updates, all edits can be done in the workspace until the data set is considered complete or internally consistent by its author. This is a good moment to commit the state of the resources to the archive.

Committing to archive can be done with the simple click of a button or an API call. This adds the digital object(s) to the OCFL fabric as a new version. This action is initiated by the content manager at their discretion.

All pre-ingest operations, including gathering preservation metadata, characterization, validation, etc. are performed at this point. Before that, no preservation criteria would impose any restriction on data and metadata entry.

Once committed to archive, the resources remain in the workspace, but only the latest versions are retained there. This operation would be similar to committing staged edits into a Git repository: only the latest changes are visible in the editor, while the full history is available on Git.

A simple visual cue could be added to notify the user of whether the content they are viewing or editing has changes that have not been committed to archive, and since when.

In a normal scenario, the update of the archival store should be completely opaque to the content manager, unless an error occurs. Normally, a content manager would not have to interact directly with the archival store except in case of data loss; in which case, the intervention of a DP specialist may be required.

Some limited interaction with the archival services may be beneficial in some occasion: for example, a content manager may want to view a list of the versions for an object, retrieve contents from a previous version, revert to an older version due to a wrong update, or download the full audit trail or history of a digital object;

this information is not kept in the workspace, but it may be made available by the archival services via API back-channels.

3.4 Shelving resources

In order to save space on a relatively expensive medium (the workspace store) for resources that are not expecting much modification in the near future, users have the option to “shelve” such resources the moment they are committed to archive, or any time after that. Shelving means removing the resource contents from the workspace, only leaving minimal metadata in the index, so that they can still be found and possibly reinstated in case an edit or access to preservation copies is required.

Resources can be shelved and reinstated at will. The reinstatement uses the same procedure as a recovery action, however this changes a flag in the resource metadata in the OCFL store (this cannot be stored in the workspace only, as a recovery operation needs it in the archival store). This creates a new version for each shelving and reinstatement action, so shelving is advised in combination with a commit of other changes.

Policies can be set for resources to be automatically shelved after a defined amount of time, or upon any other detectable event.

3.5 Integration with existing CMSs

The use of the workspace services for a department or team is optional, but if used, they should be used consistently with whole collections of information managed by a team, in order to ensure consistency of workflows, metadata entry, and information access.

One case in which using the workspace may be avoided is for resources already managed by another Content Management System in a satisfactory way, e.g. ArchiveSpace or TMS resources. In this case, resource metadata may still be made available in the DRS workspace and transformed to fit in its content model, in order to establish further relationships between different CMSs and to promote the use of a central discovery and access platform. This is optional and achievable any time in the future if desired, with the same caveat that it should be applied consistently to entire collections.

4 Example workflow scenarios

4.1 Incremental build-up and archiving

Content Manager Alice is editing a collection of maps. This is a process that spans over several days. Alice enters and organizes the resources in the workspace, edits metadata, checks files for accuracy, and edits and replaces images with incorrect cropping or orientation. She saves her work in the content workspace as she goes.

Once Alice has completed her work, she notifies Bob so he can review it. Once Bob approves the work, Alice commits the whole collection to the archive. The resources will appear in the archive as Version 1, without the iterative edits that Alice performed in between.

4.2 Updating resources

Alice notes that a page of one book in the collection she previously committed to archive is rotated incorrectly. She uploads a new image to replace the incorrect one, and commits her changes to archive.

The new page will be stored in OCFL as Version 2 and the old one will remain at Version 1.

The version 1 of the page image will not be accessible in the workspace store.

4.3 Restoring versions

On a deeper look, Alice realizes that the page she replaced was indeed the correct orientation. She browses through the version history of the resource, finds Version 1 with the original, correct orientation, and restores it.

The workspace does not store the version information, but it makes it available to Alice in its UI by looking up the OCFL store behind the scenes.

The page will be at Version 3 in OCFL upon restoring.

4.4 Recovery (individual resource)

Content manager Charles notices that an image he previously uploaded to the workspace has become corrupted. He notifies DP specialist Dana who confirms that there is a checksum mismatch.

OR

An automated checksum verfication process detects a mismatch in Charles' image and notifies Charles and Dana.

Dana performs a routine recovery procedure on the corrupted image. The correct image is replaced in the workspace store, and since it is a published image, the delivery copy is also updated in the delivery store.

No change occurs in OCFL.

4.5 Disaster recovery

Sysop Emily notices that a flawed storage migration operation has corrupted many files in the data store. Since this was caused by a software bug, a hardware recovery is not possible. In addition, resources in the metadata store have become corrupted as well.

Emily notifies DRS users about the incident and takes the workspace service down for maintenance. Then, she runs a complete rebuild procedure that restores all the resources in the workspace from the latest version that had checked into archive.

All uncommitted edits in the workspace will be lost in this case.

No change occurs in OCFL. The archive services remain online all throughout the recovery process (ideally– need to deal with race conditions and load).

4.6 Deleting resources (with redirect)

Content manager Finnegan is going over a data cleanup in his collections. He notices that a duplicate image has been entered and the same content is showing under two different identifiers.

Finnegan deletes one of the images in the workspace. Upon deletion, he opts to create a redirect from the deleted image ID to the one that has been kept.

The image is removed from the workspace store and a redirect is issued, which will prevent a broken link.

In the OCFL store, a new version of the deleted image is created that marks the resource with a tombstone. Upon disaster recovery, this resource will not be rebuilt in the workspace store, but the redirect will be reinstated.

4.7 Deleting resources (without redirect)

Content manager Greta notices that she inadvertently uploaded a picture of her cat to the workspace, and that image has been committed to archive, even though her cat has no historical relevance.

Greta deletes the image from the workspace store, and opts out of a redirect. Now users of the workspace and delivery services will receive a `410 Gone` status code when they try to access the image.

A new version of the resource is created in OCFL with a tombstone. The image will not be restored to the workspace upon disaster recovery, but Greta's cat will persist forever in Harvard's memory.

4.8 Shelving resources

Content manager Hakeem has uploaded a large body of documents from a donation. He wants to preserve and publish these contents, but nobody will need to reference them for a while in the workspace after they have been archived.

Hakeem selects some resources and presses a "shelve" button. The resource contents are removed from the workspace store and free up space. The resources will still show in search results, but accessing them will display a "shelved" label and only minimal metadata will be displayed. Published resources will still have delivery copies online.

The OCFL store will save a new version of the resources with the "shelved" flag on. Upon disaster recovery, only the minimal metadata and the delivery derivatives will be restored.

4.9 Reinstating access to shelved resources

Content manager Inez detects some incorrect metadata in one of the records that Hakeem created and shelved.

Inez navigates to the resource in questions and sees that it shows as "shelved", and a "reinstate" button is available.

Upon pressing the "reinstate" button, the resource is reinstated using the same mechanism for recovering individual resources from archive, and is available for full access and editing as other resources. Inez has the option to shelve the resource again after she is done editing.

In the OCFL layer, one version is created for reinstating the resource (for removing the "shelved" flag), plus one version for each time that Inez commits changes to archive, and one more if the version is shelved again as a separate action.

4.10 Short-term storage

Content manager Jasmin is working on a promotional pamphlet. She uploads and assembles some images in the workspace because it's the most accessible content management system at her disposal.

Jasmin uses the resources but does not commit them to archive. She may or may not delete them at some point. In any case, the resources will be lost if they become corrupted in the workspace store.

No changes to the OCFL store will occur.