Bit Preservation Specifications

Version 2.7
10/26/05

Prepared by:
David Hafken*
Babak Hamidzadeh*
Justin Littman*
Elizabeth Madden*

Reviewed by:
Martha Anderson*
Elizabeth Dulabahn*
Michael Stelmach*

*Office of Strategic Initiatives, Library of Congress

Table of Contents

Introduction

Like other cultural heritage institutions, the Library is increasingly undertaking custodianship of collections of digital objects. These digital objects are extremely diverse: each digital object may be small in size or contain terabytes of data; the digital objects may be in innumerable formats, standardized to varying degrees; and the digital objects are grouped into collections which may be static or regularly growing and which may contain a small number of digital objects or millions of digital objects. In addition, depending on its source, each digital object may be accompanied with various quantities and forms of metadata, which itself may exhibit extreme diversity. The goal of a bit preservation system is to provide the Library with a robust yet simple mechanism for the archival storage of the data that represents and is associated with the digital objects over which it has custodianship. The purpose of these specifications is to describe such a system.

Scope

The OAIS Reference Model defines archival storage as "the services and functions for the storage, maintenance and retrieval" of digital data. Similarly, the Sustain process of the LC Digital Life Cycle Framework calls for "fault-tolerant storage, including backup and recovery" and best practices for maintaining the integrity of digital data. Unlike storage for access or "scratch" storage, bit preservation storage is intended to be permanent or for "the Long Term." Thus, to fulfill its function a bit preservation system must accept data for storage; ensure that the data has been correctly stored; periodically validate that the data remains correctly stored over time; export data upon request; and ensure that the data has been correctly exported. To protect against technological obsolescence, hardware failure, or disaster, multiple copies of the data must be stored in multiple locations using multiple technologies. While the scope of a bit preservation system may be similar to a standard enterprise backup system in some respects, the specifications exceed those of an enterprise backup system for availability; redundancy; and auditing/verification of ingest, storage, and export.

In scope

1 Bit preservation of files and directories. In particular:
  1.1 Accepting files and directories for storage.
  1.2 Ensuring that files and directories have been correctly stored.
  1.3 Periodically validating that the files and directories remain correctly stored over time.
  1.4 Exporting files and directories upon request.
  1.5 Ensuring that the files and directories have been correctly exported.
2 Commercially reasonable methods (e.g., firewalls, access control) to secure the bit preservation system and prevent access to the bit preservation system by unauthorized users.

Out of scope

1 All other functions of an archive. These include, but are not limited to:
  1.1 Access (So, for example, the bit preservation system is not intended to directly support a public website or portal.)

1.2 Ingest (So, for example, the bit preservation system does not validate file formats.)

2 Any function which requires:
   2.1 An understanding of the bits
   2.2 Any naming conventions used for the files or directories
   2.3 Any logic inherent in the directory structure
   2.4 Any associations between files
   2.5 Any metadata / data relationships
3 Version control.
4 Granular permissions control to the bit preservation system.
5 Extraordinary methods (e.g., encryption of data) to secure the bit preservation system and prevent access to the bit preservation system by unauthorized users.
6 Any business or contractual relationships required to interact with the bit preservation system or the organization maintaining the bit preservation system. This includes:
   6.1 Business or contractual agreements for establishing the bit preservation service.
   6.2 Business or contractual conditions for terminating the bit preservation service.
7 Scheduling or workflow of the addition or export of data.

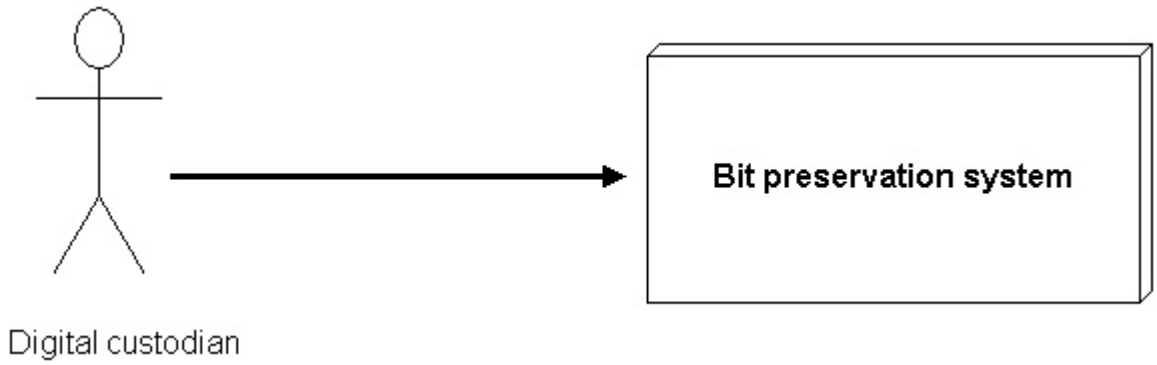
Conventions

These are intended to be general specifications. As such, numerous details are left unspecified. These details are identified by the notation "TBD." Presumably, based on discussions with a potential implementer the specification would be completed.

Additional considerations

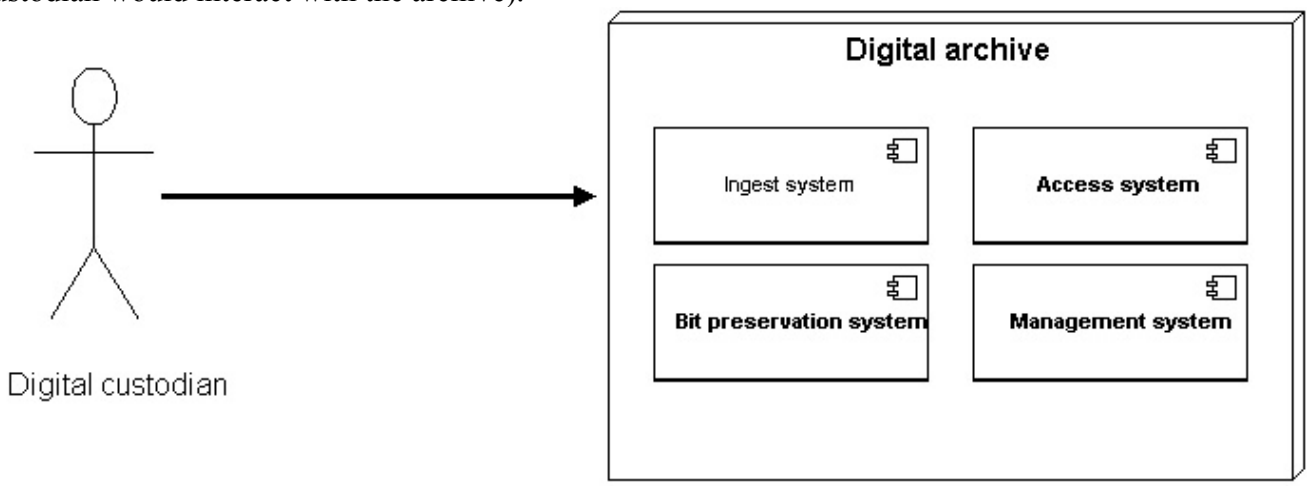
Context

The context in which the bit preservation system operates is not specified by these requirements. However, there are two likely contexts in which it would operate: (1) stand-alone and (2) as part of a digital archive. When the bit preservation system is standing-alone, a digital custodian (a person, not the actor specified later in these requirements) might interact directly with the bit preservation system (perhaps via a web interface) to manage data.



When the bit preservation system is part of a digital archive, other systems within the digital archive

(e.g., the ingest system) would interact with the bit preservation system to manage data. (The digital custodian would interact with the archive).



Of course, there are innumerable variations on the spectrum between direct interaction by a digital custodian and mediated interaction by a full-featured archive.

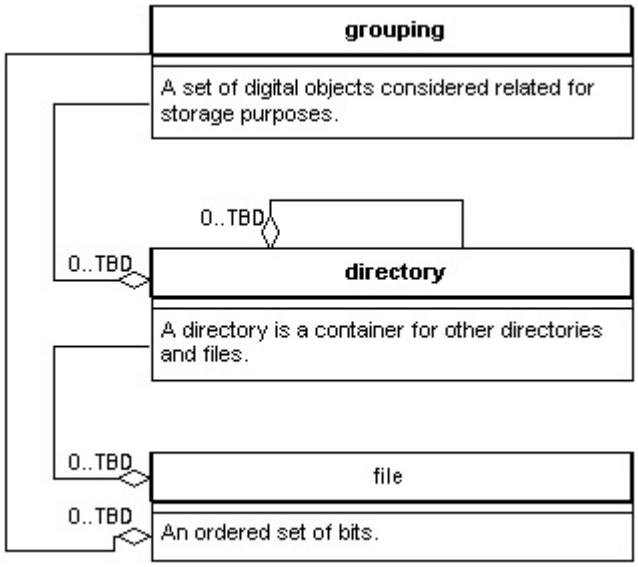
Explanation of system

As used in these requirements, "system" means the set of interrelated components and processes, both human and machine. Its use is not limited to a set of hardware/software components. Thus, it is expected that an implementation of a bit preservation system that satisfies these requirements will include both human and machine components.

Furthermore, though this element specifies aspects of the system, especially its interaction with the digital custodian, internal functions of the bit preservation system are not specified. These include, but are not limited to storage technologies, disaster recovery, refreshing media, and technology migration.

External interface to the system

For the purposes of the external interfaces to the bit preservation system, data will be organized by a grouping / directory / file approach. (The actual approach used to store the data inside the bit preservation system is outside the scope of these requirements.) That is, sets/collections of related data will be organized into groupings. (Defining what constitutes a grouping or how best to split data into groupings is outside the scope of these requirements.) In turn, data within a grouping will be organized using the file system metaphor of directories and files. Thus, the bit preservation system will be able to store data which is already stored in a file system or can be mapped to a file system. This is summarized below:



Trust

Using a bit preservation system inherently requires trust between the digital custodian and the bit preservation system. However, where possible the digital custodian will be allowed to place less trust in the bit preservation system or to verify that trust.

Frequency of usage

Most operations performed by the bit preservation system will be asynchronous. This is consistent with the role of the bit preservation system as an archival storage system, not an access system. Thus, it is assumed that additions to the bit preservation system will be grouped together into large batches (rather than small, frequent additions of individual files) of completed digital objects (as opposed to partially completed or converted digital objects) and that retrievals from the bit preservation system will be relatively infrequent. Note, that where appropriate, performance requirements for the bit preservation system will be provided.

Value of stored data

As will be evident, these requirements exceed those for standard enterprise data backup and restore. However, the bit preservation system is intended for high value data for which the additional effort and cost is justified. Determining what constitutes high value data is outside the scope of these requirements.

Status of data

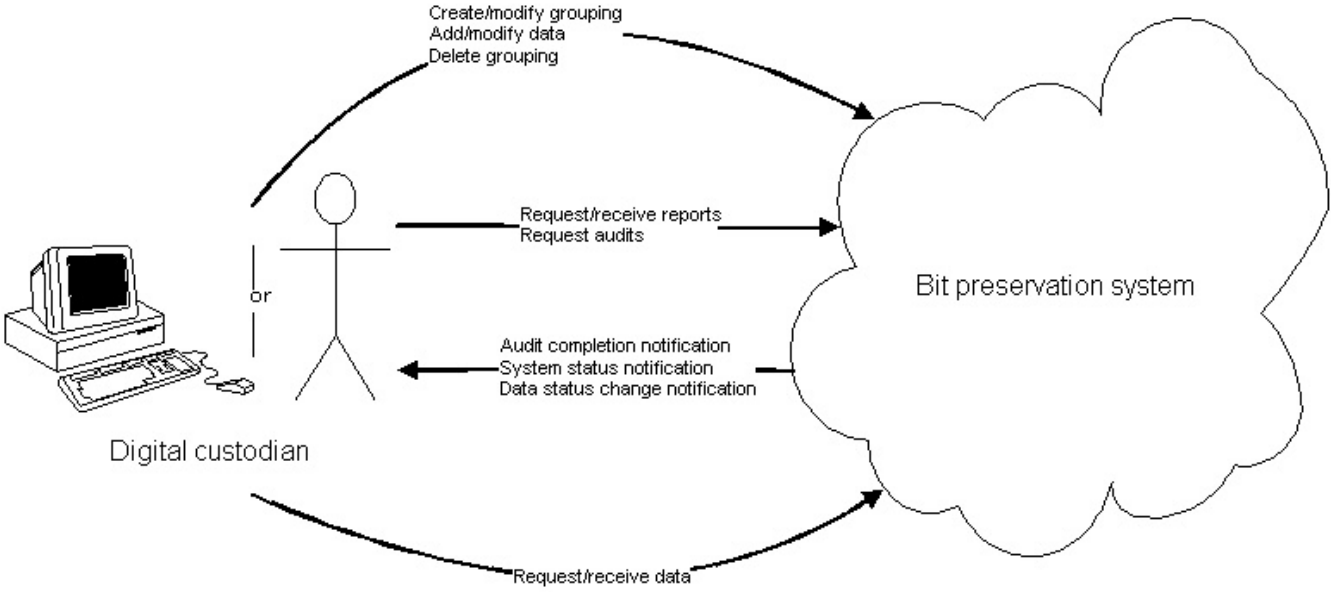
As will be described below, to protect the data stored by the bit preservation system, multiple copies of the data will need to be maintained. If the proper number of copies are maintained, the data is considered properly protected. If less than the proper number of copies are maintained, the data is considered at risk. If no copies are maintained, the data is considered lost. An important function of the bit preservation system will be to keep the digital custodian apprised of the status of the data.

Use Cases

Actors

In the use cases for the bit preservation system there is a single actor, the digital custodian. The digital custodian is the person or system that is responsible for the preservation of the digital objects and interacts with the bit preservation system. In terms of LC roles, a digital custodian might be a member of the digital conversion team, the web capture team, or the curator for a collection. Alternatively, if the bit storage system is part of a larger archive, the digital archive might be the digital custodian.

Context diagram



Create new grouping use case

Description:

The goal of this use case is for the bit preservation system to create a new grouping as requested by the digital custodian.

Preconditions:

1. There is an agreement (perhaps a contract) for the bit preservation system to store data for the digital custodian.
2. The bit preservation system has authenticated the digital custodian.
3. The grouping does not exist in the bit preservation system.

Note that as appropriate, the bit preservation system will be responsible for insuring that these preconditions have been satisfied.

Successful end condition:

The grouping has been created in the bit preservation system and the digital custodian has been notified.

Failed end condition:

The grouping has not been created in the bit preservation system and the digital custodian has been notified.

Primary actors:

Digital custodian

Trigger:

Digital custodian submits a request to the bit preservation system that a new grouping be created.

References:

Interface mechanisms, page 28
Authentication keys, page 27
Authenticity digital signatures, page 27
Requesting a new grouping, page 28

Note:

As part of this use case, a digital custodian may be issued an authentication key and/or an authenticity digital signature private key for the grouping. It is the digital custodians responsibility to insure that (1) the key is not lost and (2) the key is not distributed to unauthorized individuals.

The steps in this use case are synchronous.

Basic flow:

1. *Submit*: The digital custodian submits a request to the bit preservation system that a new grouping be created. The request includes the grouping name, security information, notification information, and auditing information.
2. *Validate*: The bit preservation system validates the request and that the grouping does not already exist.
3. *Create authentication key*: The bit preservation system creates an authentication key for the grouping.
4. *Create authenticity key*: The bit preservation system creates an authenticity digital signature public/private key for the grouping.
5. *Create grouping*: The bit preservation system creates the grouping.
6. *Confirm*: The bit preservation system informs the digital custodian that the grouping has been created and provides the grouping's authentication key, authentication key verification, and authenticity digital signature public/private key.

Digital custodian provides authentication key verification alternative flow:

If as part of *Submit*, the digital custodian provides a key verification, *Create authentication key* is omitted and a key is not returned to the digital custodian as part of *Confirm*. A digital custodian might

want to supply her own key verification for the grouping if (1) she wants to re-use existing keys, e.g., to share a common key between groupings or (2) she does not trust the bit preservation system to ever be in possession of the grouping's key.

Digital custodian provides authenticity digital signature public key alternative flow:

If as part of *Submit*, the digital custodian provides an authenticity digital signature public key, *Create authenticity key* is omitted and a public/private key is not returned to the digital custodian as part of *Confirm*. A digital custodian might want to supply her own key verification for the grouping for the reasons given above.

Validation fails alternative flow:

If validation fails at *Validate*:

1. The bit preservation system notifies the digital custodian that the grouping was not created and the reasons for validation failing.

Create grouping fails alternative flow:

If creating the grouping fails at *Create grouping*:

1. The bit preservation system removes any parts of the grouping that may have been successfully created.
2. The bit preservation system notifies the digital custodian that the grouping was not created and the reasons for the failure.

Modify grouping use case

Description:

The goal of this use case is for the bit preservation system to modify an existing grouping as requested by the digital custodian.

Preconditions:

1. The grouping exists in the bit preservation system.
2. The digital custodian has an authentication key to the grouping.

Successful end condition:

The grouping has been modified in the bit preservation system and the digital custodian has been notified.

Failed end condition:

The grouping has not been modified in the bit preservation system and the digital custodian has been notified.

Primary actors:

Digital custodian

Trigger:

Digital custodian submits a request to the bit preservation system that a grouping be modified.

References:

Interface mechanisms, page 28
Authentication keys, page 27

Notes:

The steps in this use case are synchronous.

Basic flow:

1. *Request*: The digital custodian submits an authentication key and a request to the bit preservation system that an existing grouping be modified. The request includes the grouping name and the grouping information to be modified.
2. *Validate*: The bit preservation system validates the authentication key and the request.
3. *Create authentication key*: The bit preservation system creates an authentication key for the grouping.
4. *Create authenticity key*: The bit preservation system creates an authenticity digital signature public/private key for the grouping.
5. *Modify grouping*: The bit preservation system modifies the grouping.
6. *Confirm*: The bit preservation system informs the digital custodian that the grouping has been modified and provides the grouping's authentication key, authentication key verification, and authenticity digital signature public/private key.

Digital custodian provides authentication key verification alternative flow:

If as part of *Submit*, the digital custodian provides a key verification, *Create authentication key* is omitted and a key is not returned to the digital custodian as part of *Confirm*.

Digital custodian provides authenticity digital signature public key alternative flow:

If as part of *Submit*, the digital custodian provides an authenticity digital signature public key, *Create authenticity key* is omitted and a public/private key is not returned to the digital custodian as part of *Confirm*.

Validation fails alternative flow:

If validation fails at *Validate*:

1. The bit preservation system notifies the digital custodian that the grouping was not modified and the reasons for validation failing.

Modify grouping fails alternative flow:

If modifying the grouping fails at *Modify grouping*:

1. The bit preservation system removes any modifications to the grouping that may have been successfully performed.
2. The bit preservation system notifies the digital custodian that the grouping was not modified and the reasons for the failure.

Delete grouping use case

Description:

The goal of this use case is for the digital custodian to delete/close a grouping from the bit preservation system. Once a grouping is deleted from the bit preservation system, the data and related records are no longer available to the digital custodian.

Preconditions:

1. The grouping exists in the bit preservation system.
2. The digital custodian has a key to the grouping.
3. Deleting a grouping may have contractual (legal or otherwise) aspects.

Successful end condition:

The grouping has been deleted from the bit preservation system and the digital custodian has been notified.

Failed end condition:

The grouping has not been deleted from the bit preservation system (though some of the data may have been deleted) and the digital custodian has been notified.

Primary actors:

Digital custodian

Trigger:

Digital custodian submits a signed delete grouping request to the bit preservation system.

References:

Interface mechanisms, page 28
Authentication keys, page 27
Authenticity digital signatures, page 27
Deletion of grouping, page 28

Note:

As described in the requirements, upon deleting a grouping, the bit preservation system is not obligated to maintain an audit trail for the grouping. If the digital custodian requires an audit trail for the grouping, she may export the grouping's related records by performing the Request report use case or opt to delete the grouping's data by performing the Add/modify/delete data for grouping use case, but not delete the grouping.

Also note that prior to initiating this use case, the digital custodian may wish to export the grouping's data by performing the Receive data for grouping use case.

The steps in this use case are synchronous, except for the *Delete* step, which is asynchronous. Performance requirement are specified in Deletion of grouping.

Basic flow:

1. Digital custodian submits an authentication key and a signed deletion request to the bit preservation system. The deletion request specifies the grouping to be deleted.
2. *Validate*: The bit preservation system validates the authentication key, the digital signature of the deletion request, and validates the deletion request.
3. The bit preservation system acknowledges that it has received and validated the deletion request.
4. *Delete*: The bit preservation system deletes the grouping.
5. The bit preservation system signs the deletion request.
6. The bit preservation system notifies the digital custodian that the grouping has been deleted and provides a deletion report. The deletion report includes the signed deletion request.

Validation error alternative flow:

If the bit preservation system encounters an error during *Validate*, then *Validate* is followed by:

1. The bit preservation system notifies the digital custodian that the grouping was not deleted and the reasons for the failure.

Deletion error alternative flow:

If the bit preservation system encounters an error during *Delete* and the method of deletion allows recovery of data (e.g., undeletion):

1. The bit preservation system recovers all data.
2. The bit preservation system provides a deletion report to the digital custodian.

If the bit preservation system encounters an error during *Delete* and the method of deletion does not allow recovery, e.g., because the bits have been destroyed:

1. The bit preservation system provides a signed deletion report to the digital custodian providing the reasons for the failure and listing each file and directory that have been successfully deleted.

Add/modify/delete data for grouping use case

Description:

The goal of this use case is for a digital custodian to add or modify data for an existing grouping that is stored by the bit preservation system.

Preconditions:

1. The grouping exists in the bit preservation system.
2. The digital custodian has an authentication key for the grouping.
3. The bit preservation system is ready to receive the data. So, for example, storage capacity within the bit preservation system has been requested and arranged for. The processes for this are outside the scope of these requirements.
4. The data is accessible to the bit preservation system via one of the supported submission mechanisms. This includes the bit preservation system having appropriate permissions.

Note that the bit preservation system will be responsible for insuring that all of these preconditions have been satisfied.

Successful end condition:

The data stored by the bit preservation system has been added/modified and a submission report has been provided to the digital custodian.

Failed end condition:

No data stored by the bit preservation system has been added/modified (except, possibly, data that has been deleted) and a submission report has been provided to the digital custodian.

Primary actors:

Digital custodian

Trigger:

Digital custodian submits a submission manifest to the bit preservation system.

References:

Interface mechanisms, page 28
Authentication keys, page 27
Authenticity digital signatures, page 27
Data submission, page 29

Notes:

The steps in this use case are synchronous, except for *Data validation* and *ingest*, which are asynchronous.

In cases in which add a file or directory fails, e.g., because a file or directory with the unique identifier already exists, it is the bit preservation system's responsibility to detect and report the problem, but the digital custodian's responsibility to resolve the problem, e.g., by changing the file or directory's name.

Basic flow:

1. *Submit*: Digital custodian submits an authentication key and a signed submission manifest to the bit preservation system. The submission manifest lists all of the files and directories that are to be added, replaced, deleted, renamed, and/or have fixity information changed.
2. *Request validation*: The bit preservation system validates the authentication key, the digital signature of the submission manifest, and the submission manifest.
3. *Acknowledgement*: The bit preservation system acknowledges that it has received the submission manifest.
4. *Data validation*: The bit preservation system validates the data (as specified in Validation).
3. *Ingest*: The bit preservation system ingests and stores the data validating that the stored data matches the submission manifest.
4. *Sign submission manifest*: The bit preservation system signs the submission manifest.
5. The bit preservation system stores the submission manifest.
6. The bit preservation system notifies the digital custodian that the data has been stored and provides a submission report. The submission report includes the signed submission manifest.

Request Validation problem alternative flow:

If the bit preservation system encounters an error during *Request validation*, then *Request* v*alidation* is followed by:

1. The bit preservation system notifies the digital custodian that the submission failed and the reasons for the failure.

Data Validation problem alternative flow:

If the bit preservation system is unable for any reason to complete *Data Validation* within the maximum time permitted or the bit preservation system encounters an error, then *Data Validation* is followed by:

1. The bit preservation system provides a submission report to the digital custodian providing the reasons for the failure.

Unrecoverable ingest error alternative flow:

If an unrecoverable error is encountered during *Ingest* or the bit preservation system is otherwise unable to perform *Ingest* within the maximum time permitted, then *Ingest* is followed by:

1. The bit preservation system removes all data that has been successfully stored for the grouping and/or removes any modifications of data. (Depending on the method of deletion, it may not be possible to perform recovery.)
2. The bit preservation system provides a submission report to the digital custodian providing all of the errors encountered. If data was deleted and recovery cannot be performed, the submission report should list each of the files and directories and be signed.

Note that the bit preservation system may attempt to recover from internal errors without entering the unrecoverable ingest error alternative flow. An error is considered unrecoverable if it prevents the bit preservation system from performing ingest within the maximum permitted ingest period.

Data error alternative flow:

If during *Ingest*, the submission manifest and the stored data are found to be inconsistent:

1. The bit preservation system removes all data that has been successfully stored for the grouping.
2. The bit preservation system provides a submission report to the digital custodian providing all of the errors discovered.

Request report use case

Description:

The goal of this use case is for a digital custodian to request and receive a report. Available reports include data reports, activity reports, component logs, submission manifests, historical lost data reports, and historical data storage reports.

Preconditions:

1. The grouping exists in the bit preservation system.
2. The digital custodian has an authentication key to the grouping.

Successful end condition:

The digital custodian has successfully received the data report.

Failed end condition:

The digital custodian has not successfully received the data report.

Primary actors:

Digital custodian

Trigger:

Digital custodian requests a list of reports.

References:

Interface mechanisms, page 28
Authentication keys, page 27
Component logs, page 31
Activity reports, page 31
Data reports, page 32

Note:

A data report lists all of the files and directories for a grouping stored in the bit preservation system.

An activity report includes any changes made to the grouping, the data of the grouping, any audits performed on the data of the grouping, or any exports performed for the grouping. Note that the activity that is reported on is an "external view" of the activity of the bit preservation system as a whole. It does not include the activity of individual components of the bit preservation system, except for auditing actions of individual storage mechanisms.

Component logs are the logs created by the components of the bit preservation system (e.g., the backup software). Viewing these logs allows the digital custodian to "peek at the books" of the bit preservation system so that the digital custodian can verify trust of the bit preservation system.

Submission manifests are the submission manifests submitted by the digital custodian that were successfully completed. They have been signed by the digital custodian and the bit preservation system to prove authenticity.

Historical lost data reports list every unrecoverable error for all groupings for a time period. Lost data is an instance of a file or directory being permanently lost by the bit preservation system.

Historical data storage reports lists for each month, the number of files and directories stored by the bit preservation system for all groupings.

The combination of the historical lost data reports and the historical data storage reports allows the digital custodian to determine the failure rate for the bit preservation system for a particular time period.

Except for preparing a report, the steps in this flow are synchronous. Preparing a report may be asynchronous.

Basic flow:

1. The digital custodian provides an authentication key and requests a list of reports.
2. *Select report*: The bit preservation system checks the authentication key and provides the list of reports. Available reports include data reports, activity reports, component logs, submission manifests, historical lost data reports, and historical data storage reports.
3. The bit preservation system acknowledges that it has received the request for the report.
4. *Prepare report*: The bit preservation system prepares the data report and provides to digital custodian.

Select date range alternative flow:

If at *Select report*, the activity report, historical lost data report, or historical data storage report is selected, *Select report* is followed by:

1. The bit preservation system provides the digital custodian with the means to select a date range.
2. The digital custodian selects a date range.

Component logs alternative flow:

If at *Select report*, component logs is selected, *Select report* is followed by:

1. The bit preservation system provides the digital custodian with a list of available component logs.
2. The digital custodian selects a component log.

Submission manifests alternative flow:

If at *Select report*, submission manifests is selected, *Select report* is followed by:

1. The bit preservation system provides the digital custodian with a list of submission manifests.
2. The digital custodian selects a system manifest.

Request audit use case

Description:

The goal of this use case is for a digital custodian to request an audit for a grouping. An audit involves verifying the correct storage of files and directories.

Preconditions:

1. The grouping exists in the bit preservation system.
2. The digital custodian has an authentication key to the grouping.

Note that the bit preservation system will be responsible for insuring that all of these preconditions have been satisfied.

Successful end condition:

The audit has been completed and the digital custodian has received an audit report.

Failed end condition:

The audit has not been completed and the digital custodian has been notified.

Primary actors:

Digital custodian

Trigger:

Digital custodian requests an audit.

References:

Interface mechanisms, page 28
Authentication keys, page 27

Note:

An audit request is in addition to any scheduled audits. While a scheduled audit may be performed at the convenience of the bit preservation system, a requested audit is to be performed as soon as possible.

The steps in this use case are synchronous, except for performing the audit, which is asynchronous.

Basic flow:

1. The digital custodian provides an authentication key and an audit request for a grouping.
2. *Validate*: The bit preservation system checks the authentication key and validates the request.
3. The bit preservation system acknowledges that it has received the request.
4. *Audit*: The bit preservation system performs the audit.
5. *Notification*: Completion of the audit triggers the Audit completion notification use case.

Validate fails alternative flow:

If at *Validate*, an error is encountered:

1. The bit preservation system notifies the digital custodian that the audit request failed and the reasons for the failure.

Incomplete audit alternative flow:

If at *Audit*, the bit preservation system cannot complete the audit within the time period permitted for the audit:

1. The bit preservation system notifies the digital custodian of the failure to complete the audit.

Audit completion notification use case

Description:

The goal of this use case is for the bit preservation system to notify the digital custodian that an audit has been completed for a grouping.

Preconditions:

1. Notification information, e.g., an email address, has been provided for each grouping.

2. An audit has been requested or an audit periodicity has been specified for a grouping.

Successful end condition:

The digital custodian has been notified of the completion of the audit.

Failed end condition:

The digital custodian has not been notified of the completion of the audit.

Primary actors:

Digital custodian

Trigger:

The bit preservation system has completed an audit of a grouping requested by a digital custodian or the audit period for a grouping has elapsed.

References:

Interface mechanisms, page 28
Authentication keys, page 27

Basic flow:

1. *Trigger*: The bit preservation system completes a requested audit.
2. The bit preservation system notifies the digital custodian of the audit results.

Audit period elapsed alternative flow:

Instead of *Trigger*:

1. *Elapse*: The audit period for a grouping elapses.

Incomplete audit alternative flow:

If at *Elapse* in audit period elapsed alternative flow, the bit preservation system has not completed the audit:

1. The bit preservation system notifies the digital custodian of the failure to complete the audit.

System availability notification use case

Description:

The goal of this use case is for a digital custodian to be notified of scheduled activities that will affect the availability of the bit preservation system or unscheduled changes in the availability of the bit preservation system.

Preconditions:

1. Notification information, e.g., an email address or a web page, has been provided for each

grouping.

Successful end condition:

The digital custodian has been notified of the present or future change in system availability.

Failed end condition:

The digital custodian has not been notified of the present or future change in system availability.

Primary actors:

Digital custodian

Trigger:

A change in system availability is scheduled or occurs.

Note:

A change in system availability is anything that affects a digital custodian's ability to use the bit preservation system, e.g., a system upgrade, component failure, backup windows, completion of system upgrade, or recovery from component failure.

References:

Interface mechanisms, page 28
Authentication keys, page 27
System availability, page 32
Data status reports, page 32

Basic flow:

1. *Trigger*: The bit preservation system becomes unavailable.
2. The bit preservation system notifies the digital custodian of the system availability change.
3. The bit preservation system becomes available.
4. The bit preservation system notifies the digital custodian of the system availability.

Scheduled outage alternative flow:

Instead of *Trigger*:

1. The bit preservation system schedules an activity that will make the bit preservation system unavailable.

Changed scheduled outage alternative flow:

Instead of *Trigger*:

1. The bit preservation system changes the schedule of a previously scheduled activity that will make the bit preservation system unavailable.

Data status change notification use case

Description:

The goal of this use case is for a digital custodian to be notified of any changes in the status of data. Changes in the status of data occur when data becomes more or less at risk.

Preconditions:

1. Notification information, e.g., an email address or a web page, has been provided for each grouping.

Successful end condition:

The digital custodian has been notified of changes in the status of data.

Failed end condition:

The digital custodian has not been notified of changes in the status of data.

Primary actors:

Digital custodian

Trigger:

This use case is initiated periodically every TBD.

Note:

Changes in status are reported as file or directory at risk discovery events, file or directory at risk recovery events, and file or directory lost events.

References:

Interface mechanisms, page 28
Authentication keys, page 27
Data status, page 25
Events, page 25

Basic flow:

1. *Trigger*: The period has elapsed.
2. The bit preservation system provides the digital custodian with a data status change report. The data status change report provides the files or directories whose status has changed since the last report.

Receive data for grouping use case

Description:

The goal of this use case is for a digital custodian to request and receive a copy of data for a grouping that is stored by the bit preservation system.

Preconditions:

1. The grouping exists in the bit preservation system.
2. The digital custodian has a key for the grouping.
3. There is a location to place the data accessible via one of the supported export mechanisms. The bit preservation system has appropriate permissions to write to the location. (This will need to be verified by the bit preservation system.)
4. The accessible location's file system is capable of handling the data. (For example, if the data contains file names with 500 characters, then an NTFS file system (which does not handle 500 character names) would not be an acceptable file system.

Successful end condition:

The digital custodian has received the requested data and an export report.

Failed end condition:

The digital custodian has not received all of the requested data, but has received an export report.

Primary actors:

Digital custodian

Trigger:

Digital custodian submits an export request to the bit preservation system.

References:

Interface mechanisms, page 28
Authentication keys, page 27
Data export, page 33

Notes:

The steps in this flow are asynchronous.

Basic flow:

1. *Request*: Digital custodian submits a key and an export request manifest to the bit preservation system. The export request contains the grouping, the part/whole of the grouping being requested, and the location for the export.
2. *Validate*: The bit preservation system validates the key and the export request manifest.
3. The bit preservation system acknowledges that it has received the export request manifest.
4. *Export*: The bit preservation system exports a copy of the data to the specified location.

5. *Verify*: The bit preservation system verifies the data that has been exported.
6. The bit preservation system provides an export confirmation report to the digital custodian indicating that the data has been exported.

Validate failed alternative flow:

If an error is encountered during *Validate* then *Validate* is followed by:

1. The bit preservation system notifies the digital custodian that the export request failed and the reasons for the failure.

Export failed alternative flow:

If an error is encountered during *Export* then *Export* is followed by:

1. The bit preservation system provides an export report to the digital custodian.

Possible errors include the export location running out of storage space, the bit preservation system not have appropriate permissions, or the export location's file system not supporting all of the necessary features (e.g., long filenames). Also, note that the bit preservation system is not required to remove any previously written data.

Verification failed alternative flow:

If a verification error or is encountered during *Verification* then *Verification* is followed by:

1. The bit preservation system determines if the problem is with the file/directory as exported or as stored by the bit preservation system.
2. The bit preservation system provides an export report to the digital custodian .

Note that the bit preservation system is not required to remove any previously written data.

Export elapsed alternative flow:

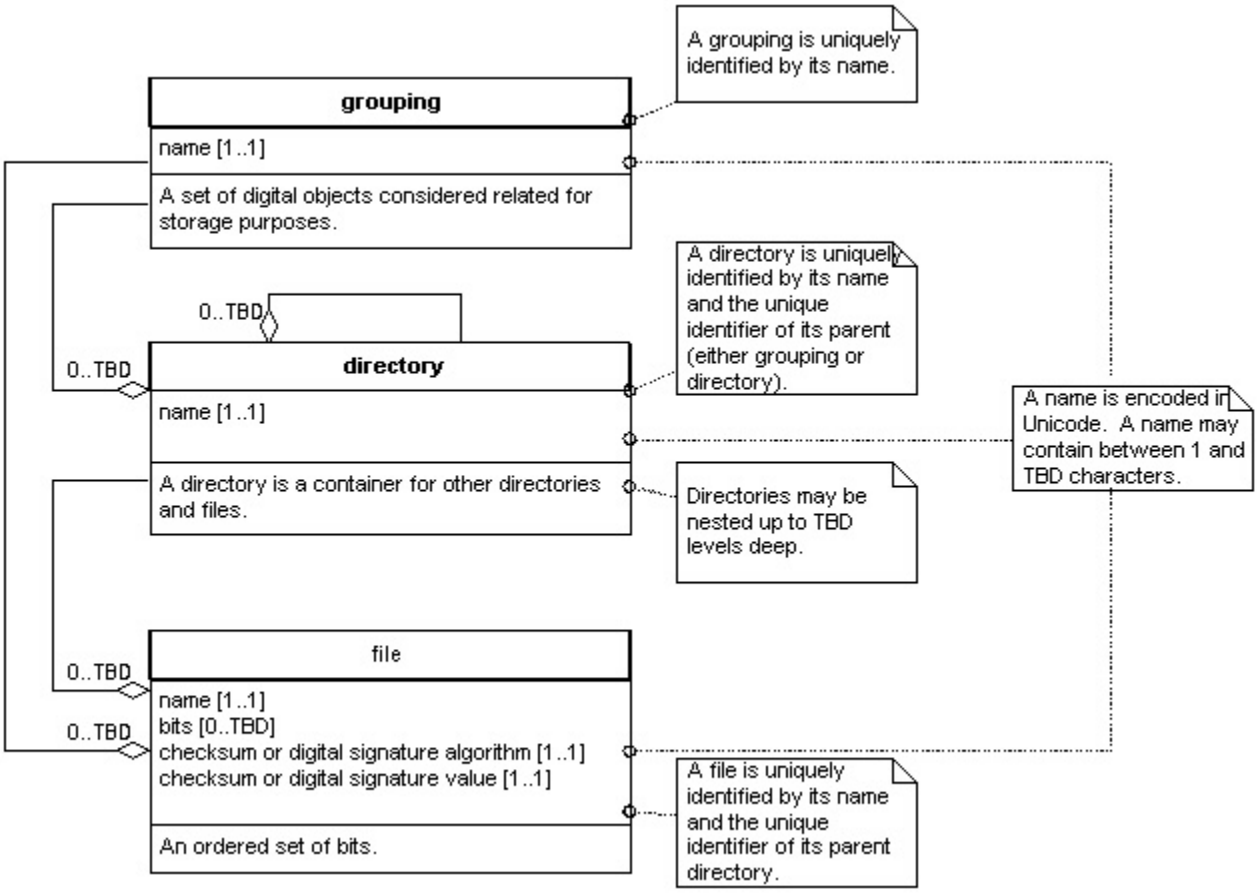
If during any step, the bit preservation system is unable to complete the export within the required time, the bit preservation system interrupts the current step and:

1. The bit preservation system provides an export report to the digital custodian providing all of the errors encountered.

Functional and non-functional requirements

1 Data

1.1 Logical data model



1.1.1 Grouping

1.1.1.1 A grouping is a set of digital objects considered related for storage purposes. It is up to the digital custodian to determine what constitutes an appropriate grouping.

1.1.1.2 A grouping has one name.

1.1.1.3 A grouping is uniquely identified by its name.

1.1.1.4 A grouping is composed of 0 to TBD directories and 0 to TBD files.

1.1.2 Directory

1.1.2.1 A directory is a container for other directories and files.

1.1.2.2 A directory has one name.

1.1.2.3 A directory is uniquely identified by its name and the unique identifier of its parent. (In other words, the directory name and path.)

1.1.2.4 A directory is composed of 0 to TBD directories and 0 to TBD files.

1.1.2.5 Directories may be nested up to TBD levels deep.

1.1.3 Files

1.1.3.1 A file is an ordered set of bits.

1.1.3.2 A file has one name.

1.1.3.3 A file has 0 to TBD bits. (Thus, the maximum file size is TBD.)

1.1.3.4 A file has 1 checksum or digital signature algorithm.

1.1.3.5 A file has 1 checksum or digital signature value.

1.1.3.6 A directory is uniquely identified by its name and the unique identifier of its parent. (In other words, the file name and path.)

1.1.4 Names
1.1.4.1 Names are encoded using Unicode.
1.1.4.2 Names contain 0 and TBD characters.
1.1.5 Fixity algorithms
1.1.5.1 Permitted checksum algorithms are TBD.
1.1.5.2 Permitted digital signature algorithms are TBD. (Note that all digital signature algorithms must support the embedding of the public key in the digital signature value.)
1.2 Data requirements for storage
1.2.1 Data that a digital custodian wishes to store in the bit preservation system must conform to the logical data model.
1.2.1.1 This may require that the digital custodian map her data to this data model.
1.2.2 Similarly, the bit preservation system must accept data that conforms to the logical data model.
1.2.2.1 The bit preservation system may need to map from the logical data model to the actual storage systems.

2 Storage

2.1 Storage mechanisms
2.1.1 A storage mechanism is a system for the storage of data. Each storage system may store one or more copies of a particular piece of data. (For example, a tape storage system may have multiple copies of a particular piece of data.)
2.1.2 All data and all related records necessary for performing and validating an export must be stored by at least TBD storage mechanisms. Thus, there is at least TBD copies of all data (where TBD is at least 2). (This is referred to as the "multiple copies requirement".)
2.1.3 At least one copy of the data and all related records necessary for performing and validating an export for one of the storage mechanism must be stored remotely. (This is referred to as the "remote storage requirement".)
2.1.3.1 By remotely is meant at least TBD from the bit preservation system.
2.1.4 The bit preservation system may migrate data from one storage mechanism to another (e.g., for a move to a newer technology) or refresh media as long as the multiple copies and remote storage requirements are satisfied.
2.1.5 Technology
2.1.5.1 A storage mechanism will be based on commercially reasonable technology.
2.1.5.2 Each component of a storage mechanism will be technologically distinct from the comparable component of the other storage mechanism. So, for example, each storage mechanism should use a distinct:
2.1.5.2.1 Operating systems (e.g., Linux vs Unix vs Windows)
2.1.5.2.2 File systems (e.g., NTFS vs EXT3)
2.1.5.2.3 Backup software
2.1.5.2.4 Disk drive types (e.g., serial ATA vs iSCSI)
2.1.5.2.5 Tape drive types (e.g., LTO vs DLT)
2.1.5.2.6 Optical storage types (e.g., CD vs DVD)
2.1.5.3 Note, however, that both Storage Mechanisms can use the same storage medium. So, for example, both Storage Mechanisms can use hard drives.

2.1.6 For each storage mechanism for which copies are used to satisfy the remote storage requirement, at least one instance of the storage mechanism (i.e., all hardware and software) must also be stored remotely.
2.1.6.1 This instance must be capable of supporting the retrieval of data, though with relaxed performance requirements.
2.1.6.2 This instance must be tested every TBD.

2.2 Data status
2.2.1 A file or directory which satisfies the multiple copies and remote storage requirements is protected.
2.2.2 A file or directory which does not satisfy the multiple copies and remote storage requirements is at risk if there is at least one copy of the file or directory.
2.2.3 A file or directory for which there are no copies is lost.
2.2.4 Note that a data does not have a status until it is initially protected. In other words, data is not considered at risk or lost after it has been submitted by a digital custodian but before the multiple copies and remote storage requirements have been satisfied.
2.2.5 Discovery of a problem with a copy of a file or directory (e.g., if a file was found to be corrupt) is referred to as a "data error." The discovery of a data error may (but not necessarily) result in a change in data status.

3 Events

3.1 Various reports will be described in terms of the events they are reports on. For example, an export report will contain information on export events.

3.2 The following lists events and the information that must be reported on for each event:
3.2.1.1 Add file (reported after file is protected)
3.2.1.1.1 Unique identifier for file (i.e., file name and path)
3.2.1.1.2 Checksum or digital signature algorithm
3.2.1.1.3 Checksum or digital signature value
3.2.1.1.4 Datetime of addition
3.2.1.1.5 Errors encountered (including data errors)
3.2.1.2 Replace file (reported after file is protected)
3.2.1.2.1 Unique identifier for file (i.e., file name and path)
3.2.1.2.2 Checksum or digital signature algorithm of existing file
3.2.1.2.3 Checksum or digital signature value of existing file
3.2.1.2.4 Checksum or digital signature algorithm of new file
3.2.1.2.5 Checksum or digital signature value of new file
3.2.1.2.6 Datetime of replacement
3.2.1.2.7 Errors encountered (including data errors)
3.2.1.3 Add directory (reported after directory is protected)
3.2.1.3.1 Unique identifier for directory (i.e., directory name and path)
3.2.1.3.2 Datetime of addition
3.2.1.3.3 Errors encountered (including data errors)
3.2.1.4 Delete file or directory
3.2.1.4.1 Unique identifier for file or directory
3.2.1.4.2 Datetime of deletion
3.2.1.4.3 Errors encountered (including data errors)
3.2.1.5 Rename file or directory (reported after file or directory is protected)

3.2.1.5.1 Unique identifier for file or directory
3.2.1.5.2 Existing name
3.2.1.5.3 New name
3.2.1.5.4 Datetime of rename
3.2.1.5.5 Errors encountered (including data errors)
3.2.1.6 Update file fixity information (reported after file or directory is protected)
3.2.1.6.1 Unique identifier for file
3.2.1.6.2 Existing checksum or digital signature algorithm
3.2.1.6.3 Existing checksum or digital signature value
3.2.1.6.4 New checksum or digital signature algorithm
3.2.1.6.5 New checksum or digital signature value
3.2.1.6.6 Datetime of update
3.2.1.6.7 Errors encountered (including data errors)
3.2.1.7 Audit directory
3.2.1.7.1 Unique identifier for file or directory
3.2.1.7.2 Datetime of audit
3.2.1.7.3 Storage mechanism name
3.2.1.7.4 Errors encountered (including data errors)
3.2.1.8 Audit file
3.2.1.8.1 Unique identifier for file or directory
3.2.1.8.2 Datetime of audit
3.2.1.8.3 Storage mechanism name
3.2.1.8.4 Errors encountered (including data errors)
3.2.1.8.5 Checksum or digital signature algorithm of audited file
3.2.1.8.6 Checksum or digital signature value of audited file
3.2.1.9 Add or modify grouping
3.2.1.9.1 Grouping name
3.2.1.9.2 Security information (if added or modified)
3.2.1.9.3 Notification information (if added or modified)
3.2.1.9.4 Auditing information (if added or modified)
3.2.1.9.5 Datetime of addition or modification
3.2.1.9.6 Errors encountered
3.2.1.10 Export directory
3.2.1.10.1 Unique identifier for file or directory
3.2.1.10.2 Datetime of export
3.2.1.10.3 Location of export
3.2.1.10.4 Errors encountered (including data errors)
3.2.1.11 Export file
3.2.1.11.1 Unique identifier for file or directory
3.2.1.11.2 Datetime of export
3.2.1.11.3 Location of export
3.2.1.11.4 Checksum or digital signature algorithm of exported file
3.2.1.11.5 Checksum or digital signature value of exported file
3.2.1.11.6 Errors encountered (including data errors)
3.2.1.12 File or directory at risk discovery (reported when bit preservation system discovers that file or directory is at risk)
3.2.1.12.1 Unique identifier for file or directory

3.2.1.12.2 Datetime of discovery
3.2.1.12.3 Description of discovery
3.2.1.13 File or directory at risk recovery (reported when the file or directory is restored to protected)
3.2.1.13.1 Unique identifier for file or directory
3.2.1.13.2 Datetime of recovery
3.2.1.13.3 Description of how recovery performed
3.2.1.14 File or data lost discovery (reported when bit preservation system discovers that file or directory is lost)
3.2.1.14.1 Unique identifier for file or directory
3.2.1.14.2 Datetime of discovery
3.2.1.14.3 Description of discovery

4 Security

4.1 The bit preservation system must be able to manage (e.g., issue, verify, and remove) authentication keys and authenticity digital signatures.
4.2 Authentication keys
4.2.1 Authentication keys will be used to provide access control to operations on a grouping by a digital custodian.
4.2.1.1 Note that this is to control external access to the bit preservation system. It is not required that these keys be used within the bit preservation system to control access to individual components of the bit preservation system.
4.2.2 The implementation of the keys is TBD. (However, the implementation must be commercially reasonable and such that the bit preservation system can check the key without storing a copy of the key. Possible implementations include public/private key digital signatures and strong passwords with the bit preservation system storing a checksum for the password.)
4.2.3 If the bit preservation system creates a key for a grouping for a digital custodian, it must not retain a copy.
4.2.4 The bit preservation system must store the necessary information to check the key for each grouping. So, for example, if public/private keys are used, the bit preservation system must store a copy of each public key for each grouping and not rely on the digital custodian to provide the public key with each request.
4.3 Authenticity digital signatures
4.3.1 Digital signatures will be used to prove the authenticity of various requests, e.g., requests to add/modify data. As described in the use cases, digital signatures will be applied by the digital custodian and the bit preservation system. This is intended primarily for nonrepudiation purposes, e.g., proving what was actually requested by the digital custodian, rather than authenticating current requests (which is handled by the authentication key).
4.3.2 The implementation of digital signatures is TBD.
4.4 Note that if digital signatures are used for both authentication and authenticity, the same key may be used for both.
4.5 Commercially reasonable means will be used to insure the physical and network security of the bit preservation system, including the storage of all copies of the data.

5 Interface mechanisms

5.1 Request mechanisms are TBD.
5.2 Acknowledgment mechanisms are TBD.
5.3 Notification mechanisms are TBD.
5.4 Report delivery mechanisms are TBD.

6 Grouping management

6.1 Requesting a new grouping
- 6.1.1 A request for a new grouping must include the following information:
  - 6.1.1.1 Grouping name
  - 6.1.1.2 Security information
    - 6.1.1.2.1 Information necessary to verify authentication key (e.g., public key to verify a private key) or request that the bit preservation system create authentication key
    - 6.1.1.2.2 Authenticity digital signature public key or request that the bit preservation system create an authenticity digital signature public/private key.
  - 6.1.1.3 Notification information
    - 6.1.1.3.1 Information to provide notifications to digital custodian
  - 6.1.1.4 Auditing information
    - 6.1.1.4.1 Periodicity of audits
    - 6.1.1.4.2 Coverage of audits

6.2 Deletion of grouping
- 6.2.1 Deletion of a grouping entails deletion of all data.
- 6.2.2 Deletion of a grouping entails that all related records, e.g., stored Submission Manifests or activity logs, are no longer available to the digital custodian for the grouping.
- 6.2.3 It is up to the bit preservation system to determine how, when, or if to dispose of any records related to the grouping.
- 6.2.4 Deletion request
  - 6.2.4.1 The deletion request must include the name of the grouping.
  - 6.2.4.2 The deletion request must be signed by the digital custodian.
- 6.2.5 Grouping deletion report
  - 6.2.5.1 The confirmation report indicates if the deletion was successful or unsuccessful.
  - 6.2.5.2 The submission report contains information on the following events that are related to the grouping deletion:
    - 6.2.5.2.1 Delete file or directory
  - 6.2.5.3 If the grouping deletion was successful, the grouping deletion report includes the deletion request, which is also signed by the bit preservation system.
  - 6.2.5.4 If the grouping deletion was unsuccessful, data was deleted, and recovery cannot be performed, the grouping deletion report lists each of the files and directories that cannot be recovered. The grouping deletion report must be signed.
  - 6.2.5.5 The format for the confirmation report is TBD.
- 6.2.6 The acceptable delay for a deletion to be performed is TBD.

7 Data submission

7.1 Submission mechanisms
- 7.1.1 The bit preservation system must support various mechanisms for making data accessible for ingest. These include TBD. (Possible mechanisms include direct connection of external hard drives, SCP to scratch storage, network connected scratch storage, and network connected storage.)

7.2 Submission manifests
- 7.2.1 A submission manifest must include the following information:
  - 7.2.1.1 Grouping name
  - 7.2.1.2 Root location of data
  - 7.2.1.3 For each file to be added or replaced:
    - 7.2.1.3.1 Unique identifier for file (i.e., file name and path)
    - 7.2.1.3.2 Checksum or digital signature algorithm
    - 7.2.1.3.3 Checksum or digital signature value
  - 7.2.1.4 For each directory to be replaced:
    - 7.2.1.4.1 Unique identifier for directory (i.e., directory name and path)
  - 7.2.1.5 For each file or directory to be deleted:
    - 7.2.1.5.1 Unique identifier for file or directory
  - 7.2.1.6 For each file or directory to be renamed:
    - 7.2.1.6.1 Unique identifier for file or directory
    - 7.2.1.6.2 New name
  - 7.2.1.7 For each file for which fixity information is to be updated:
    - 7.2.1.7.1 Unique identifier for file
    - 7.2.1.7.2 New checksum or digital signature algorithm
    - 7.2.1.7.3 New checksum or digital signature value
- 7.2.2 The format for the submission manifest is TBD.
- 7.2.3 The bit preservation system must store copies of signed submission manifests.

7.3 Deletion of data
- 7.3.1 Deletion of data entails TBD. (At the least, it means preventing access to the data by the digital custodian. It may also entail destruction of all copies of the stored bits.)

7.4 Validation
- 7.4.1 Validation for all data includes:
  - 7.4.1.1 Validating that the grouping exists.
  - 7.4.1.2 Validating that the data conforms with the Data requirements for storage.
- 7.4.2 Validation for added files includes:
  - 7.4.2.1 Validating that all files listed in the submission manifest as being added are accessible.
  - 7.4.2.2 Validating that the fixity information provided in the submission manifest matches the actual files.
- 7.4.3 Validation for added directories includes:
  - 7.4.3.1 Validating that all directories listed in the submission manifest as being added are accessible.
- 7.4.4 Validation for replaced files includes:
  - 7.4.4.1 Validating that all files listed in the submission manifest as being replaced are accessible.
  - 7.4.4.2 Validating that all files listed in the submission manifest for replacement exist in

the bit preservation system.

7.4.4.3 Validating that the fixity information provided in the submission manifest match the actual files.

7.4.5 Validation for deleted files and directories includes:

7.4.5.1 Validating that all files and directories listed in the submission manifest for deletion exist in the bit preservation system

7.4.6 Validation for renamed files and directories includes:

7.4.6.1 Validating that all files and directories listed in the submission manifest for renaming exist in the bit preservation system.

7.4.7 Validation for changing the fixity information for files includes:

7.4.7.1 Validating that the checksum or digital algorithm is a permitted algorithm.

7.4.7.2 Validating that all files listed in the submission manifest for changing fixity information exist in the bit preservation system.

7.4.7.3 Validating that the fixity information is correct for the files stored in the bit preservation system.

7.5 Submission report

7.5.1 The submission report indicates if the submission could be completed (i.e., was successful) or could not be completed (i.e., was unsuccessful).

7.5.2 The submission report contains information on the following events that are related to the submission:

7.5.2.1 Add file

7.5.2.2 Replace file

7.5.2.3 Add directory

7.5.2.4 Delete file or directory

7.5.2.5 Rename file or directory

7.5.2.6 Update file fixity information

7.5.3 If the submission was successful, the confirmation report includes the submission manifest, which is also signed by the bit preservation system.

7.5.4 If the submission was unsuccessful, data was deleted, and recovery cannot be performed, the submission report lists each of the files and directories that cannot be recovered. The submission report must be signed.

7.5.5 The format for the confirmation report is TBD.

7.6 The acceptable delay for data validation and ingest is TBD.

8 Auditing

8.1 Performing an audit for a storage mechanism involves:

8.1.1 Checking that each file and directory for the grouping is stored by the storage mechanism.

8.1.2 Checking that no additional files or directories are stored by the storage mechanism for the grouping.

8.1.3 Calculating and checking the fixity value for each file.

8.2 An audit requires checking each storage mechanism that stores a grouping. This includes the copy of the data that is stored remotely (though the remote copy requirement must be maintained at all times).

8.3 Scheduled audits

8.3.1 Each grouping has an audit period and an audit coverage.

8.3.1.1 The audit period is the period within which an audit must be completed.
8.3.1.2 The audit coverage is the percentage of the collection that must be audited.
8.3.1.2.1 Percentage is to be determined based on the total number of files and directories.
8.3.1.2.2 100% audit coverage entails a complete audit of the grouping.
8.3.1.3 For example, if the audit period for a grouping is 2 years and the audit coverage is 100%, all data for each storage mechanism must be checked at least once within 2 years, i.e., a complete audit. If the audit period for a grouping is 1 month and the audit coverage is 5%, 5% of all data for each storage mechanism must be checked at least every month, i.e., a spot check.
8.3.2 The selection of data to audit should be random.
8.3.3 Audits of different storage mechanisms of a particular file or directory should be offset by at least TBD percent of the audit period.
8.4 Requested audits
8.4.1 For a requested audit, an audit coverage must be provided.
8.4.2 The acceptable delay between an audit being requested and notification of completion being provided is TBD.
8.5 Audit reports
8.5.1 An audit report must contain for each grouping:
8.5.1.1 Whether the audit was successful or unsuccessful. An audit was successful if it was completed.
8.5.1.2 Datetime of completion
8.5.2 The audit report contains information on the following events that are related to the audit:
8.5.2.1.1 Audit file
8.5.2.1.2 Audit directory
8.5.3 The format for an audit report is TBD.

9 Reporting

9.1 Component logs
9.1.1 The bit preservation system must enable logging for all internal components that support system logging. Where applicable, the logging should be set at the maximum verbosity short of a "debugging" verbosity.
9.1.2 The bit preservation system must create machine readable logs for all regularly occurring manual processes. For example, if tapes are regularly shipped to an offsite location, a log should be kept of the shipments.
9.1.3 All logs must be maintained for a period of TBD (even for components that are removed from the bit preservation system).
9.1.4 When providing a log, the bit preservation system may (but is not required to) filter messages that are exclusively related to the data of another grouping.
9.1.5 A log may be provided in its native format, as long as the format is text-based. Otherwise, it should be provided in a text-based format.
9.1.6 The acceptable delay for preparing a component log is TBD.
9.2 Activity reports
9.2.1 Activity report request
9.2.1.1 A request for an activity report must include:

9.2.1.1.1 A grouping
9.2.1.1.2 Unique identifier for a file or directory (optional)
9.2.1.1.3 A time period for the activity (optional)
9.2.1.1.4 A type of event (optional)

9.2.2 The acceptable delay between a request being submitted and the activity report being returned is TBD.

9.2.3 The submission report contains information on all events given in Events (unless the activity report is limited to a type of event).

9.2.4 The format of an activity report is TBD.

9.2.5 The acceptable delay for preparing an activity report is TBD.

9.3 Data status reports

9.3.1 Data status reports are provided every TBD.

9.3.2 Data status reports are activity reports that are limited to:

9.3.2.1 File or directory at risk discovery
9.3.2.2 File or directory at risk recovery
9.3.2.3 File or directory lost

9.3.3 Data status reports cover the period between the previous report and the current report.

9.4 Data reports

9.4.1 Data report request

9.4.1.1 A request for a data report must include:

9.4.1.1.1 A grouping

9.4.2 The acceptable delay for preparing a data report is TBD.

9.4.3 A data report must contain:

9.4.3.1 The grouping name
9.4.3.2 For each directory in the grouping:
9.4.3.2.1 Unique identifier for directory
9.4.3.2.2 Directory status
9.4.3.3 For each file in the grouping:
9.4.3.3.1 Unique identifier for the file
9.4.3.3.2 Checksum or digital signature algorithm
9.4.3.3.3 Checksum or digital signature value
9.4.3.3.4 File size
9.4.3.3.5 File status

9.4.4 The format for a data report is TBD.

9.5 System availability

9.5.1 Advance notification of system unavailability should be provided as far in advance as possible, but not to exceed TBD.

9.5.2 Even if advance notification of system unavailability is provided, notification should be provided at the time the bit preservation system becomes unavailable.

9.5.3 Notification should be provided when the system becomes available after a period of unavailability.

9.6 Historical lost data reports

9.6.1.1 The bit preservation system must maintain a permanent history of every instance of a lost file or directory.

9.6.1.2 The format for historical lost data reports is TBD.

9.6.1.3 The acceptable delay for preparing a lost data report is TBD.

9.6.2 Historical data storage reports
9.6.2.1 The bit preservation system must maintain a history of the number of file and directories stored on the first day of each month.
9.6.2.2 The format for historical data storage reports is TBD
9.6.2.3 The acceptable delay for preparing a data storage report is TBD.

10 Data export

10.1 Export request
10.1.1 An export request manifest must include the following information:
10.1.1.1 Grouping name
10.1.1.2 Export location
10.1.1.3 Data to be exported (part of grouping or entire grouping)
10.1.1.3.1 If part of grouping, then a list of files and/or directory identifiers must be provided.
10.1.1.3.2 If a directory export is requested, then all descendants of the directory will also be exported.
10.1.2 The format for an export request manifest is TBD.
10.2 Verification
10.2.1 Verification for each directory includes:
10.2.1.1 Validating that the directory exists with the appropriate name.
10.2.2 Verification for each file includes:
10.2.2.1 Validating that the file exists with the appropriate name.
10.2.2.2 Validating the fixity value of the file.
10.3 Export report
10.3.1 Whether the export was successful or unsuccessful.
10.3.2 The export report contains information on the following events that are related to the export:
10.3.2.1 Export file
10.3.2.2 Export directory
10.3.3 The format for an export confirmation is TBD.
10.4 Export performance
10.4.1 The acceptable delay between an export being requested and the export confirmation or failure report being returned is TBD.
10.5 Export mechanisms
10.5.1 The bit preservation system must support various mechanisms for placing exported data. These include TBD. (Possible mechanisms include direct connection of external hard drives, SCP to scratch storage, network connected scratch storage, and network connected storage.)
10.6 The bit preservation system must verify its ability to access and write to the export location.
10.7 If a file or directory with an identical name already exists at the export location, the bit preservation system should overwrite the existing file or directory.