Self-contained Information Retention Format (SIRF) Use Cases and Functional Requirements

Working Draft - Version 0.5a
September 27th, 2010

Publication of this Working Draft for review and comment has been approved by the Long Term Retention (LTR) Technical Work Group. This draft represents a "best effort" attempt by the LTR Technical Work Group to reach preliminary consensus, and it may be updated, replaced, or made obsolete at any time. This document should not be used as reference material or cited as other than a "work in progress." Suggestions for revision should be directed to http://www.snia.org/feedback/.

Revision History

Version	Date	Originator	Comments
0.5	August 20th, 2010	Simona Rabinovici-Cohen	Version for ballot within LTR TWG.
0.5a	September 27th, 2010	Simona Rabinovici-Cohen	Added required text from SNIA template and some editorial changes. This version is candidate to TC approval.

TABLE OF CONTENTS

FIGURES

TABLES

Foreword

Abstract

This document describes the use cases and functional requirements of Self-contained Information Retention Format (SIRF).

SNIA Web Site

Current SNIA practice is to make updates and other information available through their web site at http://www.snia.org

SNIA Address

Requests for interpretation, suggestions for improvement and addenda, or defect reports are welcome. They should be sent to the Storage Networking Industry Association, 425 Market Street, Suite #1020, San Francisco, CA 94105, U.S.A.

Acknowledgements

The SNIA Long Term Retention Technical Working Group, who developed this document, would like to recognize the significant contributions made by the following members:

Company	Contributor
HP	Mary Baker
HP	Samuel Fineberg
IBM	Simona Rabinovici-Cohen
Symantec	Roger Cummings

We would also want to acknowledge Gary Zasman (NetApp) and Michael Peterson (Individual) who contributed to the initiation of SIRF.

1. INTRODUCTION

Many organizations now have a requirement to preserve and maintain access to large volumes of digital content indefinitely into the future. Regulatory compliance and legal issues require preservation of email archives, medical records and information about intellectual property. Web services and applications compete to provide storage, organization and sharing of consumers' photos, movies, and other creations. And many other fixed-content repositories are charged with collecting and providing access to scientific data, intelligence, libraries, movies and music.

Unfortunately, preserving and maintaining access to large amounts of digital information is still difficult, error-prone, and expensive. Long-term digital content suffers from many threats, including corruption of the digital content, attack, organizational changes, and obsolescence of hardware and software. For affordability and efficiency, any processing to address these threats must be performed at scale.

For the same reason, archivists and records managers of physical items avoid processing individual items (e.g. documents, objects, records, etc.). Instead, they gather together a group of items that are related in some manner - by usage, by association with a specific event, by timing, etc - and then perform all of their processing on that group as a unit. The group itself may be known as a series, a collection, or even in some cases as a record or a record group. Once assembled, an archivist will place the series in a physical container (e.g. a file folder or a filing box of standard dimensions), and that container will be marked with a name and a reference number and placed in a known location. Information about the series will be included in a "finding aid" such as an online catalog that conforms to a defined schema which gives the name and location of the series, its size and an overview of its contents.

We propose an approach to digital content preservation that leverages the knowledge of the archival profession and helps archivists remain comfortable with the digital domain. One of the major needs to make this strategy possible is a digital equivalent to the physical container - the archival box or file folder - that defines a series, and that can be labeled with standard information in a defined format to allow retrieval when needed. The Self-contained Information Retention Format (SIRF) is defined to be that equivalent - a logical container for a set of (digital) preservation objects that also contains catalogs and metadata related to the entire contents of the container as well as to the individual objects. This logical container makes it easier and more efficient to provide many of the processes needed to address threats to digital content.

This document describes SIRF and the motivation for it along with its use cases and requirements. The use case model is used to derive the desired functional requirements of the SIRF format and the system that implements and uses it. The model uses graphical symbols and text to specify how users or applications in specific roles use SIRF. The document describes the use cases from a usage point of

view: it doesn't describe how systems implement SIRF internally, nor does it describe SIRF internal structures or mechanisms.

1.1 Purpose and Methodology

The main purpose of the use case model is to specify the functionalities and attributes needed in SIRF, so they can be agreed and form the basis for development of the SIRF specification. The use case model also:

- Provides a basis for communication between end-users and system developers.
- Provides a basis for identifying objects, object functionality, interaction, and interfaces.
- Serves as the basis for validation during the SIRF specification development, ensuring that the specification actually meets the defined requirements.
- Provides a basis for producing user support materials and documentation.

The methodology used in this document is as follows:

- Describe SIRF and its objectives.
- Define actors involved in use cases related to SIRF.
- Define use cases and flows among the actors.
- For each use case, extract the derived functional requirements.
- Aggregate all functional requirements and map use cases to them.
- Categorize the functional requirements to several categories.
- Prioritize the functional requirements. Note that some of the requirements may conflict with each other.

1.2 LTR TWG Overview

This document is developed within the Long Term Retention (LTR) Technical Working Group (TWG) in the Storage Networking Industry Association (SNIA). The mission of the LTR TWG is:

"The TWG will lead storage industry collaboration with groups concerned with, and develop technologies, models, educational materials and practices related to, data & information retention & preservation."

The LTR group's charter is defined as follows:

"The TWG will ensure that SNIA plays a full part in addressing the "grand technical challenges" of long term digital information retention & preservation, namely both physical ("bit") and logical preservation. The TWG will generate reference architectures, create new technical definitions for formats, interfaces and services, and author educational materials. The group will work to ensure that digital information can be efficiently and effectively preserved for many decades, even when devices

are constantly replaced, new technologies, applications and formats are introduced, consumers (designated communities) often change, and so on."

The LTR TWG Program of Work comprises four technical areas, namely:

- Reference model and related materials - generating a number of materials to aid in communication archival and preservation concepts to storage professions. These include reference architecture, a terminology "bridge" document, an online knowledgebase, and a survey of relevant industry standards.
- Logical preservation - addressing logical preservation needs by developing requirements and definitions for a Self-contained Information Retention Format (SIRF) that can act as a container for multiple preservation objects. The requirements gathering part of this work is based on a unique approach of defining preservation in terms of both generic and workload-based use cases, and deriving requirements directly from those cases. Use cases appropriate to cloud applications are also included.
- Education - presenting tutorials such as "Retaining Information for 100 years" at the Storage Networking World conference. Members of the group have also given keynote speeches at the SNIA Developer Conference, the Creative Storage Conference, Storage Visions Conference and the IEEE Massive Storage Systems and Technologies Conference. The group has conducted bidirectional briefing sessions with key representatives of the Academy of Motion Picture Arts and Sciences Technical Committee and the LDS Family Search organization, and with a leader of the OAIS & CASPAR projects. The group has participated in the creation of a "terminology bridge" document that attempts to define a consistent terminology for digital retention and preservation by adopting and extending existing terminology in use in archiving and records management.
- Bit Preservation - address bit (or physical) preservation needs by creating an information base on why, how and when physical storage loses integrity. This work will build on a ground breaking study "A Fresh Look at the Reliability of Long-term Digital Storage" lead by a TWG member that was presented at Eurosys.

1.3 Document Organization

The SIRF use cases and functional requirements document is divided into five main related sections. Section 1 provides a brief introduction to the document. In section 2, we describe SIRF definition, motivation, and objectives. In section 3, we describe other related specifications and their relation to SIRF. Section 4 describes the actors and use cases including the interactions, and data elements used in those interactions. Finally, Section 5 includes the functional requirements derived from the use cases and their categorization.

2. SIRF Description

Long term digital retention and preservation is the ability to sustain the understandability and usability of digital objects in the distant future regardless of changes in technologies and in the "designated communities" that use these digital objects (that is, the data consumers). Specialized preservation systems and processes are needed to enable and support long term retention. A key component in those preservation systems is the storage subsystem where the preservation objects are located for most of their lifecycle.

We cannot predict or insist upon what features future storage subsystems will provide, so the most practical way to solve our problem is to make sure the content itself provides the means to be migrated without losing either its metadata or our ability to identify its format. To make it easier to move content between systems and technologies, while ensuring it remains complete and interpretable, we need a standard way to store that information that is self-contained, self-described, and extensible.

The key properties of a long term storage container format are:

- *Self-contained*: Long-term retention requires the preservation of both data and its surrounding metadata, which can become disaggregated. To prevent this from happening, the unit of storage for an object should include to the possible extent both the data and its metadata, so that they are treated and moved together as a single storable unit that will be kept intact for the life of the object. Similarly, the unit of storage for the objects' container should include to the possible extent both the objects and the metadata about the objects and their interrelationship. The metaphor we use here is a closed bottle that includes all the information needed to understand the bottle's content in another point in time (see SIRF visual identifier).
- *Self-described:* It should be possible to look at a data package and determine what it is, so that we can interpret it correctly. For example, it should be possible to determine the objects within the container and their associated metadata. One problem is that the self-description of the container must also be interpretable. If it is complex, then it too must be self-describing. Because of this recursive problem, a completely self-describing format is impossible to achieve. However, self-describing formats remain useful if at the root of the recursion they use only very widely used formats, such as ASCII, and the self-description itself can be updated over time. While it is possible to create self-describing proprietary formats, widely used industry standard formats are more likely to have a long life.
- *Extensible:* It is impossible to predict all of the changes likely to be needed for information retained for decades. As these changes occur, we want to preserve information about what changes we made and when. For example, we need to record information about format migrations and may also want to keep the original container tied to its rendition in a new format. As another example, we may want to add information about changes in custody of the container or be able to add new types of contact information to existing information about customers. A good long-term storage container format must allow for additions and extensions while preserving the integrity of the original data.

2.1 What is SIRF?

SIRF is a logical container format for the storage subsystem appropriate for the long-term storage of digital information. It is a logical data format of a mountable unit e.g. a filesystem, a block device, a stream device, an object store, a tape, etc. It assumes the mountable unit includes an object interface layer that constructs objects out of the sectors and blocks. Some advanced storage subsystems provide a built-in object interface as in the case of Object storage, Cloud storage and XAM storage. Other, more lower level storage subsystems, have specialized media dependent standards to expose object interfaces as in the case of UDF (Universal Disk Format-ISO/IEC 13346) for DVDs, CDFS (Compact Disc File System-ISO 9660) for CDs, FAT (File Allocation Table) for HDDs, and LTFS (Long Term File System) for tapes.

The following figure schematically depicts a SIRF container that includes:

- A magic object that identifies whether this is a SIRF container and its version. The magic object is independent of the media and has an agreed defined name and a fixed size. It includes means to access the SIRF catalog.
- Numerous preservation objects that are immutable. The container may include multiple versions of a preservation object and multiple copies of each version. See next section for a detailed definition and description of preservation objects.
- A catalog that is updateable and contains metadata needed to make the container and its preservation objects portable into the future without relying on functions external to the storage subsystem.

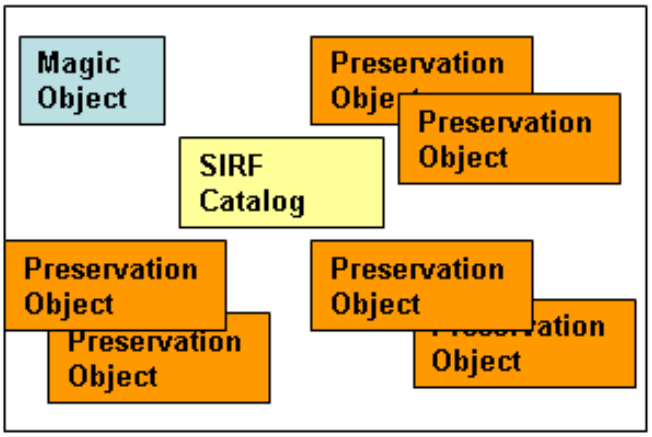

Figure 1: SIRF Components

SIRF is defined using a layered approach with two levels. The SIRF level 1 catalog contains unique metadata that is not included within the preservation objects, but is mandatory to make those preservation objects portable into the future. Examples of such metadata include retention hold,

reference counts, preservation object fixity algorithms, fixity values and fixity calculation dates, etc. The SIRF level 2 catalog includes information that may also be included within the preservation objects but is needed for fast access to the preservation objects. Examples of such metadata are links to representation information needed to assure referential integrity, metadata about the relationship among the preservation objects, packaging format, etc.

2.2 Preservation Object

A preservation object is a digital information object that includes the raw data to be preserved plus additional embedded or linked metadata needed to enable the sustainability of the information encoded in the raw data for decades to come. The preservation object is the basic unit in a preservation system, and may be subject to physical and logical migrations making it an updateable object over time. An updated preservation object is a new version of the original and its audit log records the changes that have occurred so authenticity may be verified. The OAIS Archival Information Package (AIP) standard [1] is an example of a preservation object. OAIS provides a reference model and describes the elements that should be within an AIP without specifying their format or how they are packaged together. Some standards are emerging for specific designated communities that provide specification for the actual format and packaging of a preservation object. Examples of such standards are the XML Formatted Data Unit (XFDU) [2] for space data, the VERS Encapsulated Object (VEO) [3] for electronic records, the Metadata Encoding and Transmission Standard (METS) [4] for digital libraries, PREservation Metadata: Implementation Strategies (PREMIS) [5], and Long Term Archiving and Retrieval (LOTAR) for aerospace data.

This document (and all other SIRF documents) does not specify the preservation object format. Preservation objects are generally created by applications and services defined outside of the storage subsystem and their formats tend to be domain-specific. Moreover, the storage subsystem may include multiple formats of preservation objects and this will be supported by SIRF. Specifically, SIRF is scoped to define the metadata and format in its catalog, which includes information about the preservation objects, the relationship among these objects and information to support implementation of preservation processes.

One of the processes performed upon a preservation object is migration, which is essential for long term digital retention and preservation. The migration process includes the act of moving data from one system to another because of a change. The nature of the change may include (but not limited to) one of the following:

- Possible decay of storage media
- Obsolete hardware or software (encompasses obsolete file formats)
- Change in availability of software or documentation (copyright issues)
- Change in external environment e.g., organization, staff

Migration is a major characteristic in preservation environments. The OAIS reference model identifies four primary digital migration types:

- **Refreshment** - bit-to-bit copy of the entire media's contents onto newer media of the same type, without changing the bit sequence of either the packaging or content information, or the placement of the data objects. As a result, the existing archival storage mapping infrastructure, without alteration, is able to continue to locate and access the preservation object.
- **Replication** - copying data onto newer media that is not necessarily of the same type, but without changing the bit sequence of either the packaging or content information. Note that refreshment is also replication, but replication may require changes to the archival storage mapping infrastructure.
- **Repackaging** - copying data while changing the placement of the components within the preservation object. This changes the bits of the packaging information but not the content information object itself.
- **Transformation** - copying data while performing format change on the data. This may change the bit sequence of both the packaging and content information object. Data that is transformed runs the risk of losing some of the original functionality since newer formats may be incapable of capturing all the functionality of the original format, or the converter itself may be unable to interpret all the nuances of the original format. The latter is often a concern with proprietary data formats.

Once created, the preservation objects are generally immutable, but new versions may be created over time. SIRF needs to support these immutable objects and migration processes.

2.3 SIRF Benefits

SIRF is self-describing namely it can be interpreted by different systems and in different points in time. SIRF is also self-contained namely all data needed for the preservation objects interpretation is contained within the container. This facilitates containment of any information losses - loss of a single mountable unit does not impact other mountable units.

SIRF facilitates transparent logical and physical migration and movement in order to support long term retention and preservation where:

- Media, subsystem or bitstream movement may include removing the mountable unit from one system and attaching it to a new system.
- Transparent migration and movement means that the original system is not involved. All the information needed for the new system to understand the mountable unit is self-described and self-contained within the mountable unit.
- Long term may include several years and above [6].
- Preservation includes sustaining the understandability and usability of the data and not just the bits.

SIRF makes it possible to reduce the cost of preservation, as the preservation processes can be done in a lower level of the system stack and can be performed close to the data in more robust, efficient and automatic methods. Additionally, with the advent of new storage media with longer life expectancy such as holographic versatile disk (HVD), SIRF enables reducing the number of migrations by moving

the media to future preservation systems without depending on today's systems to extract, interpret and export the preservation objects.

The following table summarizes the behavior and benefits when using SIRF for long term retention and preservation:

Without SIRF	With SIRF
Sets of linked preservation objects are moved individually between systems; thus referential integrity and context may be lost	Sets of linked preservation objects are moved between systems while maintaining referential integrity and full context
Only the original application that created the preservation objects can read and interpret them	Any SIRF compliant application can read and interpret the preservation objects
Export and import processes are needed to migrate objects	Objects can be migrated without export and import processes
Hard to sustain Preservation Objects for long-term	Preservation Objects can survive longer

Table 1: SIRF Benefits

3. SIRF and Related Specifications

There are several specifications that are related to SIRF and we discuss some of them in this section along with their relation to SIRF. Yet, SIRF is unique because it:

- Preserves collections of objects and their relationships
- Includes generic metadata that can be extended with domain specific information for fast access
- Can be mapped to and physically migrated between a wide variety of underlying storage systems

3.1 SIRF and OAIS

The current reference model for long term digital preservation is the Open Archival Information System (OAIS) [1] ISO standard. OAIS includes a functional model that describes the entities as well as their functions and process flows in a preservation system.

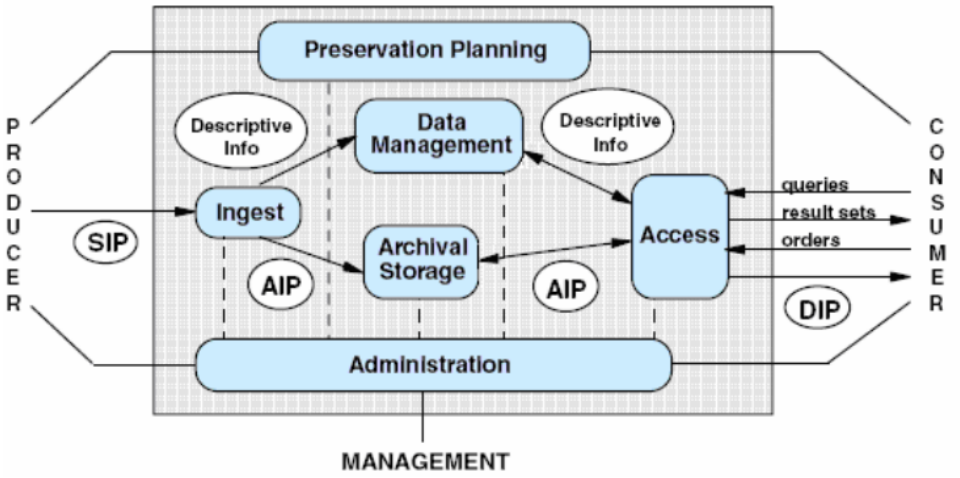

Figure 2: OAIS Functional Model

The *Ingest* entity is responsible for accepting information submitted by producers and preparing it for inclusion in the archival storage, while the *Access* entity manages the services and processes by which consumers locate, request and receive delivery of items from the archive.

The *Archival Storage* entity is responsible for ensuring that archived content resides in appropriate forms of storage and remains complete and renderable over the long-term. This is done by periodic

media refreshment or format migration, as well as implementation of safeguard mechanisms such as error-checking procedures and disaster recovery policies. The *data management* component maintains databases of descriptive metadata identifying and describing the archived information. It supports search and retrieval of the OAIS' archived content.

Preservation Planning is responsible for monitoring the environment and developing recommendations for updating the OAIS policies and procedures to accommodate these changes.

Administration is in charge of managing the day-to-day operations of the OAIS and coordinating the activities with the other five high-level OAIS services.

OAIS includes also an information model. One of the main concepts in the information model is the Archival Information Package (AIP), which is the basic object stored in a preservation system. AIP serves as an example of a preservation object. As depicted in the figure below, an AIP contains zero or one Content Information compartments and one or more Preservation Description Information (PDI) compartments. More specifically, Content Information contains the Content Data Object (raw data) that is the focus of the preservation, plus the Representation Information (RepInfo) which is needed to render the object intelligible to its designated community. This may include information regarding the hardware and software environment needed to view the content data object. The PDI compartments include additional metadata focused on describing the past and present states of the Content Information, ensuring it is uniquely identifiable, and ensuring it has not been altered in an undocumented manner. The PDI contains the following five sections:

- **Reference** – contains identifiers for the content information. At least one of these identifiers should be globally unique and persistent.
- **Provenance** – documents the history and the origin of the content information and any changes that may have taken place since it was originated. Provenance information also documents who has had custody of the content information since it was originated.
- **Context** – documents the reasons for the creation of the content information and relationships to its environment.
- **Fixity** – an integrity check that demonstrates that the particular content information has not been altered in an undocumented manner.
- **Access Rights** - the information that identifies the access restrictions pertaining to the content information, including the legal framework, licensing terms, and access control.

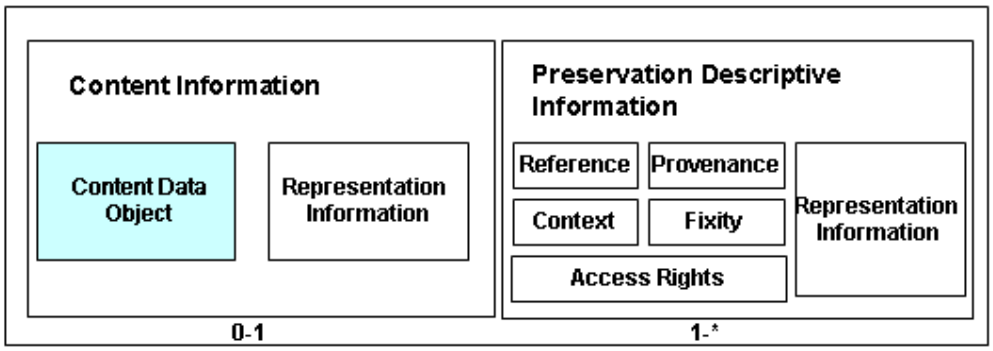

Figure 3: OAIS AIP Logical Structure

Preservation Objects within SIRF may utilize OAIS AIP. In such cases, a SIRF implementation that is OAIS-aware can enable the access to the various AIP parts including CDO, RepInfo, reference, provenance, context, fixity, and access rights.

3.2 SIRF and XAM

Some of the implementations of SIRF may utilize Extensible Access Method (XAM) [7]. XAM is a SNIA initiative to define a standard interface between consumers (application and management software) and providers (storage systems). A XAM storage system includes one or more XSystems with each XSystem being a logical container of XSet records. An XSet, which is the basic artifact in XAM, is a data structure that is a package of multiple pieces of data and metadata, bundled together for access under a common globally unique external name, called an XUID. An XSet is a collection of XSet Fields. There are two types of XSet Fields: *Properties* and XStreams. A property holds contents of a simple datatype (Boolean, int64, uint64, float64, string, datetime, or xuid), checked and enforced by the storage system. *XStreams*, on the contrary, include unbounded byte streams. These can be of any valid MIME-type, but the datatype is not checked or enforced by the storage system.

As mentioned earlier, XAM can be used to provide an object interface for SIRF, and the XAM interface can be used to access the SIRF container and the contained preservation objects. Moreover, in some implementations of SIRF, a preservation object may be implemented as an XSet object with properties for short typed metadata and XStreams for the actual content to be preserved.

3.3 SIRF and JHOVE

The open source JHOVE characterization tool has proven to be an important component of many digital repositories and preservation workflows. The Library of Congress, under its National Digital Information Infrastructure Preservation Program (NDIIPP) initiative, is now funding the development

of next-generation JHOVE2 architecture [8] for format-aware characterization. JHOVE2 is based on DROID (and PRONOM) which perform automatic format identification of a file.

JHOVE is orthogonal to SIRF and the combination of the two can be very powerful. Given a SIRF-compliant storage subsystem, the application can read the preservation objects (e.g. OAIS AIPs) included in that stream. Then, for each preservation object, the application can read its Content Data Object (CDO - actual data to be preserved), and its external-to-CDO metadata e.g. representation information, provenance and fixity. However, SIRF is agnostic to the content inside the CDO.

JHOVE2 is a tool to be used upon the CDO to identify the characterization of this CDO including its format. This characterization can be used as additional representation information to enrich the preservation object of that CDO stored in a SIRF-compliant storage subsystem. Or, it can be used to identify the format of the CDO after it was read from SIRF-compliant storage.

3.4 SIRF and BagIt

BagIt is a hierarchical file packaging format developed by the Library of Congress and published as an internet draft of the Internet Engineering Task Force (IETF) [9]. A bag consists of a payload that is the custodial focus of the bag and is treated as semantically opaque. The bag also includes tags that are metadata files intended to facilitate and document the storage and transfer of the bag. The tags include information such as the listing of payload files and corresponding checksums, the organization transferring the content, the date that the content was prepared for delivery.

SIRF is a logical container format of a mountable unit. While BagIt is more intended for a single preservation object, SIRF is focused on a container of multiple preservation objects. It includes metadata in its catalog and numerous preservation objects. The catalog metadata includes much broader information than that provided in BagIt to help interpret the preservation objects as well as the interrelationship among those preservation objects in the container. Yet, once this SIRF catalog metadata is defined, we may choose to format it in a way similar to the BagIt format.

4. USE CASE MODEL

This section first contains descriptions of the actors that are involved in the preservation system in relation to SIRF. Following are then descriptions of the actual use cases and the requirements derived from each use case. The use cases are divided to generic use cases and workload-based use cases. The former are not specific to a type of data or application, while the latter are specialized for concrete workloads.

4.1 Actors

The human actors in a preservation system are:

- Archive Employee
- Consumer
- Preservation Manager
- Producer
- System Administrator
- Auditors

The non-human actors in a preservation system that relate to SIRF are:

- Storage - Storage subsystem that persists numerous preservation objects
- TP-Service - Today's preservation service, e.g., OAIS ingest service, transformation service
- FP-Service - Future preservation service which may be unknown today
- T-App - Today's application that generates digital data, e.g., a word processor, eMail application
- F-App - Future application which may be unknown today
- Reg – Registry that stores representation information of the used storage formats, e.g., the specification documents of the used formats.

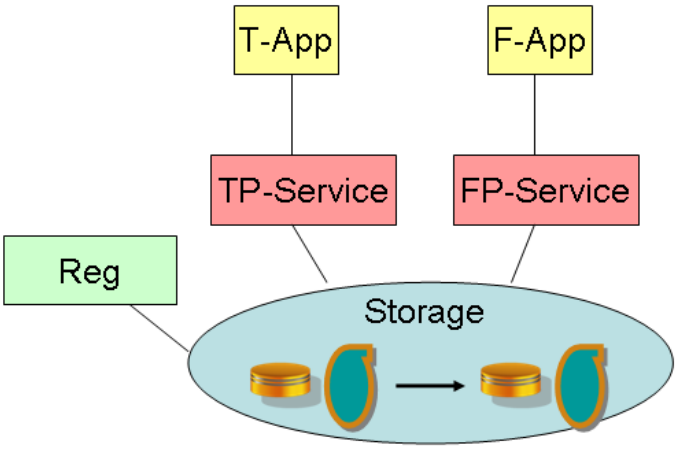

Figure 4: Preservation System Actors

The following table maps the defined actors to the entities in the OAIS Functional Model:

SIRF Actors	OAIS Functional Model
Storage	Archival Storage
TP-Service, FP-Service	Data management, Ingest, Access, Administration, Preservation Planning
T-App, F-App	Producer, Consumer
Reg	-

Table 2: Actors and OAIS Entities

4.2 Generic Use Cases

This section describes the generic use cases that appear with any application or type of data because of changes in technology (and thus the environment) over time. The first use case is where there is no change in the environment, and subsequent cases add more changes in the system due to the passage of time. For each use case, a flow is given and a set of requirements derived.

4.2.1 UC1: Ingest and Access with Same Application

The use case flow is:

1. T-App ingests a Preservation Object at 10:00 using a standard interface. The operation is agnostic to media, platform and vendor.
2. An hour passed and there is no change in environment.

3. T-App access the Preservation Object at 11:00 using a standard interface.

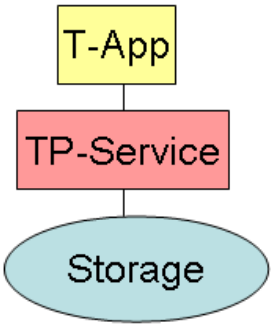

Figure 5: Ingest and Access with Same Application

The main requirements derived from this use case are:

- Support for standard interfaces, e.g., NFS, CIFS, XAM
- Agnostic to media, platform, vendor

4.2.2 UC2: Ingest and Access with Different Applications

The use case flow is:

1. T-App ingests a Preservation Object today, e.g., an object with meteorological data from a specific satellite.
2. Time passes and a newer application called F-App is developed for the same type of data, e.g., for meteorological data from satellite. Note that although it's the same type of data, it may be now in a different format.
3. F-App access the Preservation Object in the future using one of TP-Service's supported interfaces.

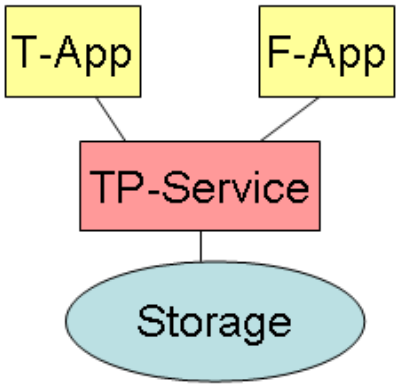

Figure 6: Ingest and Access with Different Applications

The main requirements derived from this use case are:

- Support multiple versions of preservation objects
- Support multiple data models and multiple formats for the raw data

4.2.3 UC3: Ingest and Access with Different Preservation Services

The use case flow is:

1. T-App ingests a Preservation Object today via TP-Service.
2. Time passes and the preservation services changed. New preservation services called FP-Service were developed.
3. F-App access the Preservation Object in the future via FP-Service.

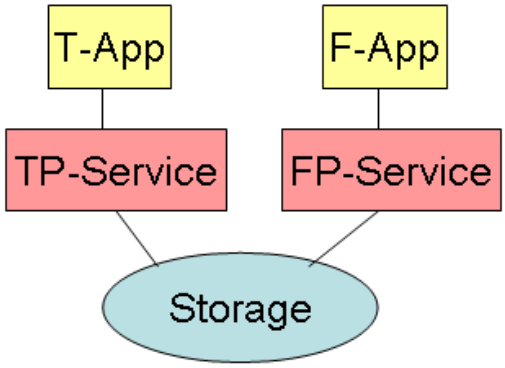

Figure 7: Ingest and Access with Different Preservation Services

The main requirements derived from this use case are:

- Persistent globally unique identifiers for the preservation objects so the object identifiers and references continue to work
- Self-contained data so nothing is lost when moving from TP-Service to FP-Service

4.2.4 UC4: Storage Format Change

The use case flow is:

1. T-App ingests a Preservation Object today via TP-Service.
2. Time passes and the storage subsystem migrates to a new one with a new container format standard that replaced SIRF.
3. F-App access the Preservation Object in the future via FP-Service.

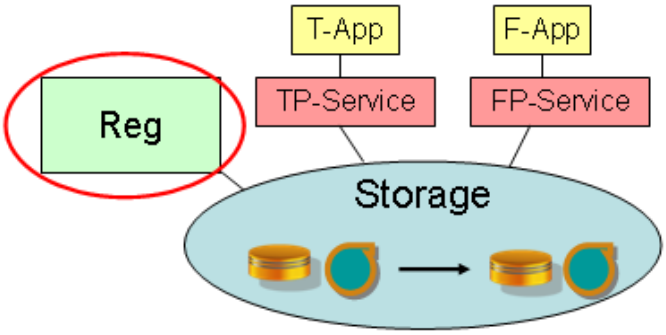

Figure 8: Storage Format Change

The main requirements derived from this use case are:

- Self-describing via a simple formalized meta-language that itself should be changeable to support SIRF format migration.
- SIRF Representation Information should be preserved in an external registry. This registry should be recursively preservable as well. The recursion ends when the Representation Information is described in a simple format that can be preserved by the community, e.g., a simple text file.

4.3 Workload-based Use Cases

This section describes the workload-based use cases that include data which needs to be accessed and used in the future in spite of technology changes. For each use case, a flow is given and a set of requirements derived.

4.3.1 UC5: eDiscovery

Discovery is the formal legal process of finding information relevant to a legal matter and delivering it to opposing council. More loosely defined, discovery can include formal legal requests as well as internal investigations that may never reach a court. eDiscovery is discovery as applied to electronically stored information. Preservation objects, like any other electronic information, can be subject to eDiscovery.

The following eDiscovery Terminology is defined:

- Case – a legal matter, i.e., lawsuit or investigation
- Responsive – information that is related to a specific case is “responsive” to it
- Legal hold – a means for ensuring that responsive information is not deleted or modified while a case is pending. A specific preservation object may be subject to any number of holds and must be maintained until all of them have been released.
- Identification – determining what data is potentially relevant to a legal inquiry
- Collection – the process of gathering all identified information

- Preservation – ensuring that potentially relevant information cannot be destroyed or altered
- Processing, Review, and Analysis – the process of sifting through collected information either electronically or manually to identify which objects are responsive and which are not
- Retention Policy – A policy governing when and for how long an object must be retained by a storage system
- Disposition Policy – A policy that defines what actions to perform at the end of an object's lifecycle.

The use case flow is:

1. T-App ingests a Preservation Object today via TP-Service.
2. Time passes and the data becomes subject to eDiscovery.
3. Potentially responsive preservation objects are identified using provenance, context and content information stored with preservation objects.
4. Identified objects are put on "legal hold," preventing deletion or modification.
5. Identified objects are copied from the preservation system and collected to a case repository for processing, review, and analysis.
6. At some future date the "legal hold" is removed. The object may become subject to other legal holds or retention /disposition policies at any time.

The main requirements derived from this use case are:

- Support for retention holds on preservation objects that prevent their deletion or modification
- Support for verification of document provenance and authenticity, regardless of migrations whether logical or physical
- Support methodology for verification of completeness and correctness
- Support for storing audits. The audits can include records about modification, possibly records about access, etc.
- It needs to be possible to identify, collect, and preserve Preservation Objects that are relevant to a legal matter.

4.3.2 UC6: eMail Archive

eMail data may include interrelated objects and a lot of repetitions. An email thread includes one or more messages where each message is an email by itself and can contain zero or more attachments. The following flow is one method of preserving emails used to derive SIRF requirements, but other methods may exist.

The use case flow is:

1. T-App ingests an e-mail thread today via TP-Service. This includes ingesting a collection of several interrelated Preservation Objects (POs) as follows:
   - Ingest a new PO for the thread. The PO metadata should include all mail header information, auditable date information, keywords, etc., including allowance for organizational-unique metadata.

- For each message within the thread, check if a PO already exists for that message. If it does, create a link from the thread PO to the message existing PO. If not, ingest a new PO for the message and a link from the thread PO to the newly created message PO.
- For each file attachment within the message, ingest a PO for that attachment and a link from the message PO to the attachment PO.
- Ingest one or more POs for information upon which the thread depends, such as a PO for the address book, POs for organizational processes, POs for data leakage policies, etc.

2. Time passes and the organization changes scope, name, undergoes a merger, etc. As a result, FP-Service creates a set of new version POs. These include a new version PO for the address book, new version POs for the new organizational processes, new version POs for data leakage policies. Note that the thread, message and attachment POs created in step 1 are not affected.
3. More time passes and F-App searches the metadata of threads, messages and attachments in parallel to find relevant POs. F-App creates POs for the search results to raise performance of future searches and ingests them to the preservation system via FP-service. Those new POs may contain links to the thread, messages and attachments created in step 1.

The main requirements derived from this use case are:

- Support for links between objects that are immutable as the objects. The links can be either "hard links" that require the existence of the linked objects within SIRF or "soft links" that can reference objects external to SIRF.
- Support for auditable time stamps that are immutable and created by known authority
- Support for "special" POs such as a PO that includes address book, a PO that includes search results.
- Generic support for organizational unique metadata (perhaps extended TLV like OrgID/Type/PrimitiveType/Length/Value)

4.3.3 UC7: Consumer Archive on the Cloud

An individual wants to preserve his family photos and documents on a cloud that provides preservation services, so that forthcoming generations will be able to access that data and study their roots.

The use case flow is:

1. A user creates a genealogy container for his genealogy-related documents on a cloud that provides SLAs for preservation.
2. The user uses T-App to ingest a genealogy-related document via TP-service on the cloud.
3. TP-service on the cloud ingests the PO with the original document as well as transforms the document to a standardized format believed to be more sustainable such as pdf/a and ingests the resulting PO version to the same genealogy container.
4. Time passes and the grandchildren would like to get that document.
5. FP-service will validate the grandchildren identity and will provide appropriate credentials to access the genealogy container and the document.
6. F-App which is a future application executing on technology used at that future time, access via FP-Service the latest version of the document and renders the pdf/a document.

The main requirements derived from this use case are:

- Support for transformations of preservation objects e.g., support various versions of the PO and the tree structure they create
- Support for managing identifiers over time
- Support secured access to the data that is updatable over time e.g., when a security mechanism becomes weak
- Support cloud containers to be SIRF-compliant, so containers can be migrated to other clouds with all the required preservation information
- Verification of document provenance and authenticity, regardless of migrations whether logical or physical

4.3.4 UC8: BioMedical Bank

A large hospital which also has an adjacent academic medical research center stores the patients' biomedical data in a biomedical bank in which data is preserved for decades. The data is used at the point of care as well as for biomedical research by the adjacent research center.

The use case flow is:

1. T-App ingests via TP-service a PO that includes a standardized Digital Imaging and Communications in Medicine (DICOM) image of the leg of a patient that is a minor.
2. Time passes and the patient, who is now an adult, scheduled an appointment to check a new problem he has encountered in his leg.
3. FP-service will identify the data needed for the scheduled appointment using reference, context and provenance information.
4. The identified Preservation Objects will be a-priory brought from an offline media to an online media to be timely accessible for the appointment.
5. F-App at point of care accesses the identified POs for the patient via FP-Service.
6. More time passes and a researcher from the adjacent academic medical research center wants to access that image for research purposes. According to HIPAA regulations, the researcher can get just a de-identified image.
7. F-App accesses the de-identified PO via FP-Service.

The main requirements derived from this use case are:

- Support hierarchical storage management, e.g., support unique IDs for the POs regardless of the storage tier, support on-line and off-line storage
- Support masking of sensitive data, e.g., support storing POs for de-identification modules within the SIRF container
- Verification of document's provenance and authenticity, regardless of migrations whether logical or physical

4.3.5 UC9: Merged Cloud Repositories

The use case flow is:

1. T-App ingests via TP-service a PO in a cloud that is provided by company “FirstCloud”.
2. T-App also ingests via TP-service a second PO in a second cloud provided by company “SecondCloud”.
3. Time passes and the two companies “FirstCloud” and “SecondCloud” are merged and their two cloud repositories are combined. This is possible as the POs identifiers are globally unique.
4. F-App access via FP-Service the two POs in the combined cloud provided by the merged company.

The main requirements derived from this use case are:

- Support cloud containers to be SIRF-compliant, so containers can be interpreted by other clouds
- Persistent **globally** unique identifiers for the preservation objects

5. REQUIRMENTS

The following is a list of the derived requirements divided to categories.

General Requirements:

- Media agnostic
  - Tape, disk, future media
  - Direct random access and serial access
  - Support mixture of storage technologies
  - Required by: all use cases
- Vendor and Platform agnostic
  - Required by: all use cases
- Support different standard storage technologies and interfaces e.g. NFS, CIFS, XAM
  - Required by: use case 1
- Extensible
  - Support additional information which may be added in the future
  - Required by: use cases 2, 3, 5-8

Format Requirements:

- Self-describing
  - The amount of "a priori" information is small and can be acquired in stages
  - Interpretable by both humans and machines
  - Ability to do offline inspection
  - Required by: use cases 2-9
- Support self-contained data
  - Include means to represent internal links and cross references
  - Required by: use cases 3-9
- Support different SIRF formats and versions preserved in a way independent of SIRF itself e.g. preserve the SIRF formats in an external registry
  - Required by: use case 4
- Interoperability
  - Ability to migrate data between different systems without loss of information – data should be interpretable after migrations
  - Can be interpreted in the future
  - Required by: use cases 3-9
- Support methodology for verification of completeness and correctness
  - Required by: use cases 3-9

Preservation Object Data Model Requirements:

- Allow different data models for preservation objects
  - Allow different object data models at one time
  - Allow complex data structures like collections of objects

    - Allow migrating objects from one data model to an alternative data model
    - Required by: use cases 3-8
- Can handle any proper data format for the raw data
    - No restrictions on file formats
    - Required by: use case 2
- Enable keeping various versions of the same preservation object with their relations
    - References from new to existing preservation objects of the same version series
    - Required by: use cases 2-8
- Support a persistent identifier for each preservation object
    - Include additional external identifiers
    - Required by: use case 3, 9
- Support for retention holds
    - Required by: use case 5
- Verification of document provenance and authenticity, regardless of migrations whether logical or physical
    - Required by: use case 4, 5, 7, 8
- Support for storing audits. The audits can include records about modification, possibly records about access, etc.
    - Required by: use case 5
- Support for “special” (secondary catalog) preservation objects
    - Required by: use case 6
- Support for auditable time stamps that are immutable and created by known authority
    - Required by: use case 6
- Support for managing identifiers over time
    - Required by: use case 7
- Support secured access to the data
    - Required by: use case 7, 8

Performance Requirements:

- Performance
    - Need to have good performance even for data that includes text and binaries
    - Support large objects, e.g., web archiving objects, database archiving objects, movies
    - Do not require complete scanning for access
    - Required by: all use cases
- Enable parallel data migration
    - Enable parallel reads and writes
    - Required by: all use cases

REFERENCES

List of referenced documents:

1. ISO 14721:2003, Blue Book. Issue 1. *CCSDS 650.0-B-1: Reference Model for an Open Archival Information System (OAIS)*, 2002. See http://public.ccsds.org/publications/archive/650x0b1.pdf
2. XML Formatted Data Unit (XFDU). See http://public.ccsds.org/sites/cwe/rids/Lists/CCSDS%206610R1/Attachments/661x0r1.pdf
3. Management of Electronic Records PROS 99/007 (Version 2). The Victorian Electronic Records Strategy (VERS). See http://www.prov.vic.gov.au/vers/standard/version2.asp
4. Metadata Encoding and Transmission Standard (METS). See http://www.loc.gov/standards/mets/
5. PREservation Metadata: Implementation Strategies (PREMIS). See http://www.loc.gov/standards/premis/
6. The 100 Year Archive Requirements Survey, SNIA-DMF, January 2007
7. Extensible Access Method, SNIA-DMF. See http://www.snia-dmf.org/xam/
8. JHOVE - http://confluence.ucop.edu/display/JHOVE2Info/Home
9. BagIt - http://tools.ietf.org/html/draft-kunze-bagit-04