The Storage Networking Industry Association (SNIA)
Data Preservation and Metadata Projects

Bob Rogers, Application Matrix
Overview

- The Self Contained Information Retention Format
  - Rationale & Objectives
  - Requirements & Use Cases
- SNIA’s eXtensible Access Method (XAM)
  - The metadata issue
- The SNIA Resource Domain Model
  - Applying policies to information management
What is the Self-Contained Information Retention Format (SIRF)?

- A logical container format for the storage subsystem appropriate for the long-term storage of digital information
  - A logical data format of a mountable unit e.g. a filesystem, a block device, a stream device, an object store, a tape, etc.
  - Includes a cluster of “interpretable” preservation objects that can be understood in the future
  - Self-describing — can be interpreted by different systems
  - Self-contained — all data needed for the preservation objects interpretation is contained within the preservation objects cluster
    - If a mountable unit is damaged or lost, the effect is contained — the information in the other mountable units is still valid!
    - Need to define how and when external references are supported
- A work effort by SNIA’s Long Term Retention Technical Work Group
Longest Retention Requirement

- **Long-term needs are real:**
  - 68% over 100 Years
  - 83% over 50 Years

- **Requirements vary by organization type, information type, and compliance rules/risk**
  - Leading Organizations: Education, gov., IT services, manufacturing, finance, insurance, pharma, & health-care
  - Leading Info-Types: Source-files, government, history, customer, & database records

http://www.snia.org/forums/dmf/registration

**Bar Graph:**

- **Percent of Responses**
  - >10 Yrs beyond Project: 4%
  - Life of Company: 2%
  - Life of Product: 1%
  - >5 Yr: 4%
  - >10 Yr: 4%
  - >20 Yr: 2%
  - >25 Yr: 3%
  - >50 Yr: 13%
  - >100 Yr: 15%
  - Permanent: 53%

Source: 100 Year Archive Requirements Survey, SNIA 2007

N=104
SIRF Objectives

- **Facilitate transparent logical and physical migration and movement in order to support long term preservation**
  - Media, subsystem or bitstream movement — remove the mountable unit from system A and put it at system B.
  - Transparent — system A is not involved. All the information needed for system B to understand the mountable unit is self-described and self-contained within the mountable unit.
  - Long term — 15 years and above (according to 100 years archive requirements survey).
  - Preservation — sustain the understandability and usability of the data and not just the bits.

- **Considering multiple implementations of SIRF to utilize:**
  - the Open Archival Information System (OAIS) ISO standard
  - SNIA’s eXtensible Access Method (XAM)
Problem SIRF is Addressing

Without SIRF

- Cannot move cluster of preservation objects between systems by itself
- Only the original application who wrote the preservation objects can read and interpret them
- Utilize export and import processes
- Preservation Objects cannot be sustained over the long-term

With SIRF

- Can move cluster of preservation objects between systems by itself
- Any SIRF compliant application can read and interpret the preservation objects
- No need for export and import processes
- Preservation Objects can survive longer
### Application Information Layer
- Get the interpretation of the data
- Example: render a Word 3.0 document
- Diverse applications and data types

### Preservation Information Layer
- Get the basic preservation objects
- Example: interprets OAIS AIPs
- Can it be standardized?

### Bit Layer
- Get the bits from the media
- Example: Linear Tape-Open (LTO) for tapes
- May depend on media type
What is a Preservation Object?

- The raw data to be preserved plus additional metadata needed to enable the preservation of the raw data for decades to come
  - The preservation object may be subject to physical and logical migrations
  - The preservation object may be dynamic and change over time
  - The updated preservation object is a new version of the original preservation object.

- An example of a preservation object is OAIS Archival Information Package (AIP)
  - An AIP includes recursive representation information that enables future interpretation of the raw data
SIRF Use Cases and Functional Requirements
SIRF Initial Requirements —
General

- **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: subset of use cases

- **Extensible**
  - Support additional information which may be added in the future
  - Required by: subset of use cases
SIRF Initial Requirements — Format

- **Self-describing**
  - The amount of required information is small and can be acquired in stages
  - Interpretable by both humans and machines
  - Ability to do offline inspection

- **Support self-contained data**
  - Include means to represent internal links and cross references

- **Support different SIRF formats preserved in an external registry**

- **Interoperability**
  - Ability to migrate data between different systems without loss of information — data should be interpretable after migrations
  - Can be interpreted in the future
  - Support methodology for verification of completeness and correctness
Support different data models for preservation objects
  - Support different object data models at one time
  - Support complex data structures like collections of objects
  - Support migrating objects from one data model to an alternative data model

Can handle any proper data format for the raw data
  - No restrictions on file formats

Enable retention of multiple versions of the original preservation object with their relations
  - References from new to existing preservation objects of the same version series

There must be a persistent identifier for each preservation object
  - Include additional external identifiers
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
- 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
SNIA’s eXtensible Access Method (XAM)
A New Fixed Content API

**XAM Provides:**
- Interoperability: Applications can work with any XAM conformant storage system; information can be migrated and shared
- Compliance: Integrated record retention and disposition metadata,
- ILM Practices: Framework for classification, policy, and implementation
- Migration: Ability to automate migration process to maintain long-term readability
- Discovery: Application-independent structured discovery avoids application obsolescence.

**XAM is a SNIA Architecture**
- The XAM Architecture spec defines the normative semantics of the API for use by applications and implementation by storage systems

**XAM is an Application Programming Interface (API)**
- The XAM Java and C API specs define bindings of the XAM Architecture to the Java and C Languages

**XAM is SNIA Software**
- The XAM SDK provides a common library and reference implementation to promote widespread adoption of the standard
XAM Interface

- First interface to standardize system metadata for retention of data
- Implements basic capability to Read and Write Data (through Xstreams)
- Locates any XSet with a query or by supplying the XUID
- Allows Metadata to be added to the data and keeps both in an XSet object
- Uses and produces system metadata for each Xset
- For example Access and Commit times (Storage System Metadata)
- But it also uniquely specifies Data System Metadata for Retention Data Services

XAM User metadata is un-interpretable by the system, but stored with the other data and is available for use in queries

Given this we can see that XAM is a data storage interface that is used by both Storage and Data Services (functions)
The Resource Domain Model

This model shows the logical layering of the different domains and the role of policies for each domain. The services in each domain play a different role, but leverage common, standard interfaces.
Resource Domains are a way of classifying services into specific areas that each deal with a different aspect of the problem.

An information domain application creates data and associates MetaData with it.

Certain Data Storage Interfaces can accommodate both Data and MetaData (XAM, Filesystems with extended attributes).

MetaData aware Data Services interpret Data System MetaData as the requirements for it’s lifecycle and implement policies for retention, placement, lifecycle, etc.

Other Data Storage interfaces (based on blocks or objects) provide virtualized Containers for the Data bits and the management of those containers.

Storage services are employed to meet those requirements at this point in the data’s lifecycle, however the storage services are unaware of the data’s requirements.
Summary

- Long Term Retention Technical Work Group was launched this year
- Preliminary work focuses on:
  - Terminology
  - SIRF Requirements
  - SIRF Use Cases
- XAM SDK released this year
- Storage Resource Domain Model in review
- Contributors are welcome
The DMF would like to thank the following individuals for their contributions to the development of this presentation:

- Michael Peterson
- Gary Zasman
- Simona Cohen
- Mark Carlson
For More Information

- **SNIA Long Term Retention Technical Work Group**

- **XAM Home**
  - [http://www.snia.org/xam](http://www.snia.org/xam)

- **XAM Technical Work Group**

- **XAM Software Development Kit (SDK) TWG**

- **SNIA Community Forum**