Bruce Barkstrom, NASA Langley Research Center

Status of This Memo: Informational

This memo provides information to the Grid community specifying the requirements for systems that support the long-term archiving of digital material. In particular, data grids provide the capabilities that address each of the archiving requirements.

Long-Term Digital Archive Requirements

Abstract

The core requirements for long-term digital archives can be expressed as management policies for both the digital records and the infrastructure that supports the digital records. Sixteen core requirements have been identified. This document explains the significance of each core requirement, and proposes multiple levels of support for each core requirement.

Contents

b.r.barkstrom@larc.nasa.gov.

1. Preservation in Archives

Archives exist to maintain a long-term record of their contents. In practical terms, this means preserving the records in the archive for at least 200 years. While much of the archival community's attention has been focused on dealing with the technological issues involved in this preservation requirement, it seems likely that the more difficult issues are those that arise from sociological and resource issues. Indeed, it may be that the most difficult issue is preserving a community that understands the contents and uses of the archived material.

2. Fundamental Requirements for Long-Term Digital Archives

A long-term digital archive provides support for authenticity (the assurance that the material in the digital archive is correctly linked to descriptions of its origin), integrity (the assurance that the material in the archive is uncorrupted, that the chain of custody can be tracked, and that the information content remains unchanged), and infrastructure independence (the assurance that the digital archive has not imposed any proprietary standards that prevent migration of the contents of the digital archive to another choice of technology).

Use of data in an archive can be rendered impossible by at least five risk factors:

1. Malicious or inadvertent destruction
2. Technological Obsolescence (hardware or software)
3. Loss of context (undocumented features; loss of understanding; aging, retirement, or death of key community members)
4. Competition for resources with new opportunities
5. Institutional instability

These factors impact two features of long-term archives in very important ways. They create two fundamental requirements:

1. A requirement to reduce operation costs as much as possible
2. A requirement to minimize errors in all of the archive operations.

3. Level 1 Derived Requirements

3.1 Core Requirements for Long-Term Digital Archives

There are sixteen Core Requirements that derive directly from the Fundamental Requirements for a long-term digital archive:

1. Unless the probability of loss per year from intrusion is less than 0.00005, a long-term archive will require off-line (and off-site) storage.
2. The data provider and the archive must agree on a valuation and risk assessment for the data and metadata to be stored in the archive.
3. The archive needs to provide redundant systems to avoid single point of failure modes that would lead to loss of data, and mechanisms to validate the consistency of the data across the redundant systems.
4. An archive system needs to be independent of the language in which it is implemented and of the operating system on which it runs. This derived requirement is necessary in order to ensure that the archive has independent modes of failure arising from system design errors and errors in system implementations. Thus, if possible, a long-term

archive should be independently implemented in at least two languages, and preferably three or more.
5. A long-term archive cannot rely on hardware provided by a single vendor.
6. A long-term archive cannot rely on software provided by a single vendor with proprietary software.
7. The archive needs to document its contents, the software components of its systems, as well as the procedures it uses to ingest, store, and distribute data.
8. An archive needs a method of registering permanent names for the unique content in its collections, for the archivists running the system, and for the processes used to manage the content.
9. Because metadata may "reach inside" files, an archive needs a method of registering permanent names for data elements inside files, independent of possible permutative rearrangement of either the files or their format.
10. The archive needs to make systematic plans for preserving and evolving the intellectual capital represented by its data, metadata, and documentation through the changes required by the technological and sociological evolution of the archive and its environment.
11. The archive needs to actively create a dispersed community of practice and discourse that is familiar with the contents and procedures of the archive. This requirement suggests that wherever possible, the archive should consider the way in which it can foster federations with other archives (to reduce the probability of loss by dispersing the archive's contents) and a vibrant Open Source community.
12. A long-term archive needs to actively work to automate as much of its operations as possible. General experience suggests that over long time periods, costs of human activity are the largest element in the cost of operations.
13. Specific experience in the NASA ESE data centers also suggests that automation must be designed into the archive's systems, rather than added to them later.
14. An archive needs a rigorous cost model for its operational costs.
15. The archive cost model needs to use statistical information from the actual history of the archive to project future costs where this history is available. In other words, the archive systems must be designed to collect both a record of archive activities and archive costs.
16. An archive needs to develop an Open Source archival community that can accept stewardship for preservation of the intellectual capital contained in archives.

4. Factors That Create Difficulties for Long-Term Digital Archives

4.1 Malicious or Inadvertent Destruction

Under normal archival standards, data needs to have a high probability of surviving more than 200 years. If the archive is connected to the Internet, it appears that a reasonable probability for an intrusion with potential consequences for the data, the metadata, or the chain of custody for both is about 10% per year. This rather high risk means that without strenuous effort on the part of the archive, the probability of having data survive 200 years is (1-0.1)**200 or about 7x10**(-10). These are clearly rather stiff odds. At present the number of intrusion incidents is rising. At the same time, an archive can take a number of steps to decrease the probability of loss – most notably by sending data to a site away from the archive. Such off-site storage, in which the data and metadata are not connected to the network, appears to be a very important component in reducing the risk of data loss.

It is also important to note that the quantification of the risk we have just identified must be considered in the light of the cost of replacing the data and metadata. Technically, this cost is part of the valuation of the data and metadata – and, in this case, is based on replacement costs. If the loss of the data is of no particular consequence, then we might not consider it worth-while to undergo the expense and hassle of off-site storage. To be careful, we need to ensure that the archive and the data provider have carefully evaluated the probability of loss – and the value of

the replacement. In other words, the archive and the data provider must quantify the probability of risk and the cost of preservation.
This pessimistic assessment leads to two derived requirements:

1. Unless the probability of loss per year from intrusion is less than 0.00005, a long-term archive will require off-line (and off-site) storage.
2. The data provider and the archive must agree on a valuation and risk assessment for the data and metadata to be stored in the archive.

4.2 Technological Obsolescence (Hardware or Software)

Unfortunately, data are not safe – even when we put it on tapes and store it deep under a mountain. Every five years or so (at least for the foreseeable future), the vendors of the hardware used to read the storage media (tapes or CDs or holographic media …) produce new models of the devices that read or write the data onto the storage media. Likewise, the vendors of software produce new versions of their wares on a time scale of about eighteen months. These facts of life mean that we must expect to move data in storage (or in the archive itself) from one medium to another about once every five years. This means that the transfer process will only occur about forty times in two hundred years – not two hundred times. While such transfers appear to be reasonably safe, experience in ASDC (and in other archives) suggests that the probability of successful transfers from one medium to another has a definitely finite probability of loss, which we estimate at perhaps 2% per transfer. This numerical value is derived by considering our experience with both large-scale data transfers and routine operations within ASDC. For example, in one recent incident, a router failure corrupted about 10% of the data in a rather large dataset delivered from another data center. This problem has happened relatively rarely, but such incidents appear to happen about once every five years or so.

Quantitatively, with a 2% probability of loss per transfer, the probability of having a particular file survive 200 years is (1 – 0.02)**40 or about 0.45. This probability is much higher than that for loss through malicious or inadvertent destruction. At the same time it is unacceptably low for real archival work, where the probability for survival in each data migration needs to be well above 0.99. Translated into an allowable probability of loss per transfer (and assuming a transfer will need to occur once every five years), this means that the allowable probability of loss per transfer needs to be kept below about 0.0002. This is rather stringent.

In terms of the data contamination incident we just described, it also means that an archive will need to play a very active role in reducing the probability of errors – which may arrive in quite unexpected forms (such as unexpected vendor hardware failures). Given the stringency of this requirement, the archive needs a carefully planned strategy to reduce the role of happenstance. A key element of such a strategy is to reduce the probability of single point failures – or to increase the redundancy of the system, or to increase the number of end-to-end validation steps used to assure integrity. In other words, if the archive can reasonably assume that data loss incidents will occur independently on separately instantiated systems, then the probability of unsustainable data loss over 200 years may be reduced to more acceptable levels. In other words, we have another set of derived requirements:

3. The archive needs to provide redundant systems to avoid single point of failure modes that would lead to loss of data, and mechanisms to validate the consistency of the data across the redundant systems.
4. An archive system needs to be independent of the language in which it is implemented and of the operating system on which it runs. This derived requirement is necessary in order to ensure that the archive has independent modes of failure arising from system design errors and errors in system implementations. Thus, if possible, a long-term archive should be independently implemented in at least two languages, and preferably three or more.
5. A long-term archive cannot rely on hardware provided by a single vendor.

6. A long-term archive cannot rely on software provided by a single vendor with proprietary software.

4.3 Loss of Context

While we are familiar with the corrosive impact of Moore's law on the stability of hardware and software, a long-term archive also needs to consider the longer term impact of changes in a number of "sociological" factors in its environment. These include the loss of knowledge regarding undocumented features of system software or hardware and the loss of understanding associated with aging, retirement, or death of key community members. For example, in the ASDC context, the criteria used by instrument teams to select data for calibration coefficients are probably lost once an instrument team disbands. Once this information is lost, it cannot be recovered.

While loss of context is probably a phenomenon with a longer time constant than either loss by malicious activities or technological obsolescence, an archive must still actively work to prevent it from causing additional data loss. This leads to at least five additional derived requirements:

7. The archive needs to document its contents, the software components of its systems, as well as the procedures it uses to ingest, store, and distribute data.
8. An archive needs a method of registering permanent names for the unique content in its collections, for the archivists running the system, and for the processes used to manage the content.
9. Because metadata may "reach inside" files, an archive needs a method of registering permanent names for data elements inside files, independent of possible permutative rearrangement of either the files or their format.
10. The archive needs to make systematic plans for preserving and evolving the intellectual capital represented by its data, metadata, and documentation through the changes required by the technological and sociological evolution of the archive and its environment.
11. The archive needs to actively create a dispersed community of practice and discourse that is familiar with the contents and procedures of the archive. This requirement suggests that wherever possible, the archive should consider the way in which it can foster federations with other archives (to reduce the probability of loss by dispersing the archive's contents) and a vibrant Open Source community.

4.4 Competition of Resources

While the concern over loss drives archive requirements toward increasing the investment in redundancy (both of hardware and of software), the environment in which archives operate can severely constrain the available resources. In general, archive budgets compete for resources required by organizations that see new opportunities. For example, NASA's Earth Science Enterprise (when that existed during the last decade of the Twentieth Century) faced severe competition between the requirements for operating a data and information system that had not been designed for low total cost of operation and its need to develop new missions. As another example, libraries in universities compete with other campus organizations for capital and operating budgets.

This competition leads to a new set of derived requirements:

12. A long-term archive needs to actively work to automate as much of its operations as possible. General experience suggests that over long time periods, costs of human activity are the largest element in the cost of operations.
13. Specific experience in the NASA ESE data centers also suggests that automation must be designed into the archive's systems, rather than added to them later.
14. An archive needs a rigorous cost model for its operational costs.

15. The archive cost model needs to use statistical information from the actual history of the archive to project future costs where this history is available. In other words, the archive systems must be designed to collect both a record of archive activities and archive costs.

4.5 Institutional Instability

Finally, as identified by the Library of Congress National Digital Information Infrastructure Preservation Program workshops and planning report, archives must be prepared for institutional instability. A specific example of this "instability" arises in the question of how to preserve the unique record of Earth observations created by a massive investment by NASA, although there are expectations that the long-term archival responsibility lies with NOAA. There are a number of potential loss mechanisms that may afflict the transfer of an operational archive from NASA to NOAA:

- Loss or data by name changes
- Loss of knowledge owing to perceptions that no knowledge or activity will be required to recreate higher level data products, even though science teams have spent hundreds or thousands of person hours in validating these products, which could not be reconstructed in a reasonable length of time after such a transfer
- Reduction in resources available – leading to loss of data by design, even though there are active communities still working with the data

It is not easy to deal with the issues raised by these large-scale movements of institutional resources. Perhaps the best we can do is to explore an implicit requirement:

16. Develop an Open Source archival community that can accept stewardship for preservation of the intellectual capital contained in archives.

5. Categories of Level 1 Requirements

It will be helpful to reorganize the list of Level 1 requirements we have derived from the two basic starting principles that the archive has to be prepared to avoid errors and that it needs to be as cost effective as possible. The headings that follow are organized in four categories. In some cases, we can combine Level 1 requirements. For example both 12 and 13 in the original list require system automation.

In addition, we can identify some additional requirements that we need to add to make the list more complete.

5.1 Low Total Cost of Ownership

- **Automation.** A long-term archive needs to actively work to automate as much of its operations as possible. General experience suggests that over long time periods, costs of human activity are the largest element in the cost of operations. Specific experience in the NASA ESE data centers also suggests that automation must be designed into the archive's systems, rather than added to them later. (Requirements 12 and 13)
- **Cost Model.** An archive needs a rigorous cost model for its operational costs. The archive cost model needs to use statistical information from the actual history of the archive to project future costs where this history is available. In other words, the archive systems must be designed to collect both a record of archive activities and archive costs. At the same time, the cost model must be able to incorporate new technology that may provide a substantial cost savings. (Requirements 14 and 15)
- **Commodity Computers and Data Storage.** A recent National Research Council (NRC) Report strongly recommended using commodity computers and data storage at government data centers. We concur with this recommendation. This kind of equipment lessens an archive's dependence on proprietary solutions to its problems. It also allows the equipment manufacturers to amortize their investment and increases the pool of available suppliers. Commodity data storage may increase the amount of validation that

is required for integrity checking as the systems may not be as reliable. A tradeoff analysis between labor spent on validation versus capital cost of the storage systems should be maintained for the archive.

- **Use of Open Source Software to Reduce Licensing Costs.** As with hardware, using software created and maintained by the Open Source community can substantially reduce licensing costs. The maintenance model for this software is different than that for proprietary software. Open Source software usually relies on e-mail and bug lists to maintain the software configuration. However, to the extent that we do not expect a mass market for archives, there is a requirement that the software be understood by the archival engineers. The archive will need to maintain their own testing and validation facility for the Open Source software, and may need to port their own local modifications to the Open Source software. If an archival consortium collaborates on the management of the software, the costs per institution can be minimized.

5.2 High Reliability

- **Balanced Approach to Redundancy and Dispersed Storage.** Unless the probability of loss per year from intrusion is less than 0.00005, a long-term archive will require off-line (and off-site) storage. The archive needs to provide redundant systems to avoid single point of failure modes that would lead to loss of data. (Requirements 1 and 3 with consideration of Low Total Cost of Ownership)
- **Valuation and Risk Assessment.** The data provider and the archive must agree on a valuation and risk assessment for the data and metadata to be stored in the archive. (Requirement 2)
- **Avoidance of Dependence on Proprietary Sources.** A long-term archive cannot rely on hardware provided by a single vendor. A long-term archive cannot relay on software provided by a single vendor with proprietary software. (Requirements 5 and 6)
- **Robust and Graceful Exception handling.** Neither computer hardware nor the software we develop is perfect. In addition, “things break”. In a well designed system, performance degrades gracefully, with a useful record of where the systems encountered faults and suggested fixes. In a poorly designed system, when something “breaks”, the system responds “brittlely”, with no record of where the system noted the first breakage. There are interactions between this requirement and low total cost of ownership –it’s much easier and cheaper to repair a system that identifies a fault, suggests confirmatory tests, and is able to bypass the problem until it can be tended by a help team.
- **Designed-In Security.** Security intrusions disrupt the archive’s operations and reduce its reliability. Most experts recommend that security be designed into the system “from the ground up”.

5.3 Evolvability

- **Language and Implementation Independence.** An archive system needs to be independent of the language in which it is implemented and of the operating system on which it runs. This derived requirement is necessary in order to ensure that the archive has independent modes of failure arising from system design errors and errors in system implementations. Thus, if possible, a long-term archive should be independently implemented in at least two languages, and preferably three or more. (Requirement 4)
- **Modularity of Architecture with Well-Designed Message Protocols.** In a general sense, modularity of architecture requires the architect to make parts of the system that do not need to know about each other entirely separate. With good, message-passing object-oriented design, we enforce this general dictum by starting with use cases that ensure that the components of the system that do not need to “talk” with each other are ignorant and unaffected by the parts of the system that can be treated as independent. By further taking care to formalize the message protocols, we improve the modularity of the system.

- **Formalization of Operational and Evolutionary Procedures.** By formalizing the operational procedures, subjecting them to the rigor of procedural simplification (or "business process engineering"), we create a system that we can reason about. In addition, with formalization, it is more straightforward to modify the procedures when the archive needs to evolve its systems. Such modifications need to be undertaken carefully and systematically. Modularity of the architecture and formalization of procedures aid in simplifying the process.
- **Complete Documentation.** The archive needs to document its contents, the software components of its system, as well as the procedures it uses to ingest, store, and distribute data. (Requirement 7)
- **Systematic Planning for Evolution**. The archive needs to make systematic plans for preserving and evolving the intellectual capital represented by its data, metadata, and documentation through the changes required by the technological and sociological evolution of the archive and its environment. (Requirement 10)
- **Outside Participation in Design, Development, and Evolution.** In order to maximize the probability of long-term survival of knowledge, it is helpful to spread understanding of the system over a wide range of communities and to solicit their input into the system design, development, and evolution.
- **Dispersed Archive Community.** The archive needs to actively create a dispersed community of practice and discourse that is familiar with the contents and procedures of the archive. This requirement suggests that whenever possible, the archive should consider the way in which it can foster federations with other archives (to reduce the probability of loss by dispersing the archive's contents) and a vibrant Open Source community. The archive needs to develop an Open Source archival community that can accept stewardship for preservation of the intellectual capital contained in archives. (Requirement 11 and 16)

5.4 Maintenance of Data Provenance and Integrity

- **Permanent Naming.** An archive needs a method of registering permanent names for the unique content in its collections. Because metadata may "reach inside" files, an archive needs a method of registering permanent names for data elements inside files, independent of possible permutative rearrangement of either the files or their format. (Requirements 8 and 9)
- **Provenance Tracking.** Data and metadata provenance are critical elements of an archive. Thus, the design of an archive's systems must include a process that updates the provenance of items within the archive after any archival operation and a process that verifies the provenance when needed. Given the volume of data and the high rate at which digital content can flow through an archive's systems, it is important to fully engage the computer portions of the archive in helping with this maintenance.
- **Transactional Basis for System Operations.** When the archive's systems transfer files or metadata form location to location, the archive needs to automate the process of ensuring that the transfer was authorized and completed satisfactorily. Transactions that can be rolled back if they are not successfully completed provide the assurance that the metadata and data will be consistent with one another.
- **Transaction Auditing and Reconciliation.** Just as an accountant reconciles the journals with the ledgers in a business setting, so the archive's systems must allow the record of transactions to be reconciled with the actual state of the archive's content inventory. Likewise, it is a requirement that the transaction accounts should be auditable – and that there be procedures for performing that work.

6. Summary

The requirements that must be supported by a digital archive can be cast as constraints on both the choice of system architecture and the management policies required to maintain the digital

archive. An examination of the experiences of the NASA Langley Research Center result in a set of sixteen recommendations for digital archive requirements.

7. Author Information

Bruce Barkstrom
Atmospheric Science Data Center
NASA Langley Research Center
b.r.barkstrom@larc.nasa.gov.

8. Acknowledgements

The ideas expressed here were developed under projects with the National Aeronautics and Space Administration.

9. Glossary

The terms used to describe digital archives are listed in this section.

Archival engineer – the system administrator of a digital archive.
Chain of custody – the organizations that maintain the archive, the storage systems used to hold the data, and the processes that have been applied to the data while in the archive.
Context – the ancillary information needed to understand the relevance of data, including the ability to interpret, use, and apply the data in research.
Digital archive – the software and hardware systems used to manage data for periods of time exceeding the lifetime of any single software or hardware component.
Exception handling – the application of automated processes to identify problems and manage the response to the problems, while notifying the archival engineer.
Intellectual capital – the standard digital reference data sets that are used to support research within a scientific community, along with the metadata that
Open Source – software systems for which the source code is distributed, and for which user-specified modifications can be incorporated independently of the distributor.
Permanent names – persistent identifiers for data, archivists, metadata, access constraints, and storage resources that remain invariant when new technology or media are incorporated in the archive.
Provenance – the set of metadata describing the origin of the data, and the calibration files and calibration programs used to generate a derived data product or the simulation code used to generate simulation output.
Technological obsolescence – the replacement of software or hardware components by new technology that is more cost effective.

10. Intellectual Property Statement

The GGF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this recommendation. Please address the information to the GGF Executive Director.

11. Full Copyright Notice

12. References

1. InterPares Preservation Task Force, “How to Preserve Authentic Records”, Oct. 2001, http://www.interpares.org/book/interpares_book_o_app06.pdf
2. Moore, R., A. Rajasekar, “Common Consistency Requirements for Data Grids, Digital Libraries, and Persistent Archives”, Grid Procotol Architecture Research Group draft, Global Grid Forum, April 2003
3. Moore, R., “The San Diego Project: Persistent Objects”, Proceedings of the Workshop on XML as a Preservation Language, Urbino, Italy, October 2002.
4. Moore, R., A. Merzky, “Persistent Archive Concepts,” Global Grid Forum, December 2003.
5. Moore, R. (2000a), “Knowledge-based Persistent Archives,” Proceedings of La Conservazione Dei Documenti Informatici Aspetti Organizzativi E Tecnici, in Rome, Italy, October, 2000.
6. OAIS - Reference Model for an Open Archival Information System (OAIS). submitted as ISO draft, http://www.ccsds.org/documents/pdf/CCSDS-650.0-R-1.pdf, 1999.
7. Thibodeau, K., “Building the Archives of the Future: Advances in Preserving Electronic Records at the National Archives and Records Administration”, U.S. National Archives and Records Administration, http://www.dlib.org/dlib/february01/thibodeau/02thibodeau.html
8. Underwood, W. E., “The InterPARES Preservation Model: A Framework for the Long-Term Preservation of Authentic Electronic Records”. Choices and Strategies for Preservation of the Collective Memory, Toblach/Dobbiaco Italy 25-29 June 2002. To be published in Archivi per la Storia.

Areas Contributing to Design Requirements

Core Requirements

Table 1. Automated Generation of Search Interface

Level	Complexity	Minimum Level to Meet Core Requirement
1	Hand Development of Web Pages for Search Interface	Below
2	Single Page Template with Static Web Page Generation	Below
3	2 + Multi-Type Page templates with Automated Static Web Page	Below
4	3 + Active/Interactive Web Pages (XUL, CSS, Jscript)	Below
5	4 + Alternate Technologies (DHTL, CSS, Jscript) with On-the-Fly Page Generation	Above
6	5 + User Controllable Attribute Visibility	Above

Table 5. Commodity Computers and Data Storage

Level	Complexity	Minimum Level to Meet Core Requirement
1	Custom-Designed Components with Proprietary Software	Below
2	Commodity Computers and Data Storage Elements with Open Source Software	Above

Note: The search interface may use a reserved vocabulary that is derived from provenance metadata or a thesaurus.
The use of Open Source software may require that the archival engineers maintain , build, and validate the software used within the archive.

Table 2. Cost-Effective Operations

Level	Complexity	Minimum Level to Meet Core Requirement
1	Ad Hoc, Manual Record Keeping Sufficient for Time and Attendance; Informal Planning	Below
2	Periodic, Hierarchical Planning; Automated Record Keeping	Below
3	Periodic, Hierarchical Planning Process, using Deterministic Cost Model for Forecasting	Above
4	Periodic, Hierarchical Planning Process, using Stochastic Schedule and Resource Forecasting	Above

Table 3. Automated Trouble Ticket Process

Level	Complexity	Minimum Level to Meet Core Requirement
1	Manual, Ad Hoc Process for Fault Detection, Diagnosis, and Correction	Below
2	Written, Manual Procedures for Fault Detection, Diagnosis, and Correction	Below
3	Automated Fault Detection with Written Manual Procedures for Fault Diagnosis and Correction	Below
4	Automated Fault Detection and Diagnosis with Written Procedures for Fault Correction	Below
5	Automated Fault Detection and Diagnosis with Automated Fault Correction or By-Pass; On-Call Help	Above

Note: The planning process should also evaluate the cost-effectiveness of new technology.

Table 4. Automated System Installation

Level	Complexity	Minimum Level to Meet Core Requirement
1	Manual Installation of All Components and Content	Below
2	Manual Installation of Base Software (e.g., Compilers, Scripting Languages, Databases), Archive Infrastructure, and Scripted Installation of Archive Contents	Below
3	Manual Installation of Archive Infrastructure; Automated Installation of Base Software and of Archive Contents	Below
4	Automated Installation of All Components and Content (Necessary for Automated Archive Replication)	Above

Table 6. Open Source Software to Belowuce Licensing Costs

Level	Complexity	Minimum Level to Meet Core Requirement
1	System Built from Proprietary Software	Below
2	Less than Half of System Components (by cost) from Proprietary Software	Below
3	More than 90 percent of System Components (by cost) from Open Source Software	Above

Note: The management costs for maintaining an appropriate version of the Open Source software for the local archive should be compared with commercially supported software, or amortized through an archival consortitum.

Table 7. Automated Hardware and Software Inventory and Configuration Management

Level	Complexity	Minimum Level to Meet Core Requirement
1	Manual, Ad-Hoc Processes	Below
2	Hardware and Software Inventory in Database; Manual Configuration Management	Below
3	Hardware and Software Inventory and System Configuration in Database with Manual Updates	Below
4	Hardware and Software Inventory in Database with Automated Checks of Inventory; System Configuration Automatically Maintained	Above

Table 8. Migration

Level	Complexity	Minimum Level to Meet Core Requirement
1	System Backed Up to Temporary Storage; Ad Hoc Procedures	Below
2	On-Site Backup Only; Data and System Transfer Done with Ad Hoc Procedures	Below
3	Off-Site Backup, Automated Data and System Transfer with Verification	Below
4	3 + Deterministic Technology Migration Model(e.g. "Moore's Law" and related "rules of thumb") to Plan Future Migration Steps	Above
5	3 + Stochastic Technology Migration Model and Options-Based Pricing of Investments for Future Migration Steps	Above

Note: The automation of the network configuration is equally important, especially when managing data that has been replicated to another site.

Table 9. System Disaster Recovery

Level	Complexity	Minimum Level to Meet Core Requirement
1	Archive Reconstruction	Below
2	1 + Valuation and Risk Analysis	Below
3	2 + Off-Site Backup with Weekly Deliveries of New Archive Contents	Below
4	3 + Periodic Disaster Rehearsals	Below
5	4 + Inventory and System Reconstruction Checks on a Systematic Basis	Above
6	5 + Multi-Site Replication with Voting and Verified Formal Model of Disaster Probabilities	Above

Table 10. Automatic Diagnostics

Level	Complexity	Minimum Level to Meet Core Requirement
1	System Error Messages	Below
2	Unified Error and Exception Handling Tied to Use Cases	Below
3	2 + Automated Fault Detection Log with Robust Exception Handling	Above

Note: The ability to rebuild the name spaces used to identify data, archivists, resources, provenance metadata, and access controls across the multiple sites is essential for minimizing risk of data loss.

Table 11. Designed-in Security

Level	Complexity	Minimum Level to Meet Core Requirement
1	System Provides Passive Protection Behind Firewall	Below
2	1 + System Actively Monitors Most Interactions with External Entities	Below
3	2 + System Actively Monitors Internal Data and Components	Below
4	3 + System Actively Monitors and REconciles all Internal Transactions	Above

Table 12. Multi-Language Implementation

Level	Complexity	Minimum Level to Meet Core Requirement
1	Core Archive Components Implemented in One Language	Below
2	Core Archive Components Implemented in Two Languages	Above
3	Core Archive Components Implemented in Three Languages	Above
4	Core Archive Components Implemented in Two or More Languages and Maintained by Open Source Community	Above
5	Open Source and Proprietary Systems for Highly Robust Archive	Above

Note: The management of security risk due to the compromise of system administrator accounts implies the need for a deep archive that will not be accessible by users.

Table 13. Use Cases Become System Manuals

Level	Complexity	Minimum Level to Meet Core Requirement
1	No Use Cases	Below
2	Use Cases Developed to Identify System Objects	Below
3	2 + Design and User Manuals Based on Use Cases	Below
4	3 + Test Procedures Based on Use Cases	Above

Table 14. Systematic Procedure for Updating and Validating Use Case Evolution

Level	Complexity	Minimum Level to Meet Core Requirement
1	No Use Cases	Below
2	Use Cases Developed, But Not Updated	Below
3	Periodic Review of Use Cases, with Updates for Documentation and Test Cases	Above

Table 15. Object-Oriented Design Traceable to Use Cases

Level	Complexity	Minimum Level to Meet Core Requirement
1	Design Decoupled from Use Cases **Unacceptable Design Practice**	Below
2	Original Design Derived from Use Cases	Below
3	2 + Substantial Portion of Test Procedures Derived from Use Cases	Below
4	3 + Design Coevolves with Use Cases	Above

Table 16. Formalization of Message-Passing Protocols

Level	Complexity	Minimum Level to Meet Core Requirement
1	No Systematic Organization of Message-Passing Protocols **Unacceptable Design Practice**	Below
2	Internal and External Protocols Documented from Objects Derived from Design Based on Use Cases	Below
3	2 + Protocols Coevolve with Use Cases and Object Design	Above

Table 17. Systematic Procedure for Documenting Protocol Evolution

Level	Complexity	Minimum Level to Meet Core Requirement
1	No Documented Protocols	Below
2	Protocol Documentation Not Updated After Initial Design	Below
3	Protocols Periodically Reviewed and Systematically Updated	Above

Note: Protocols that ensure backwards compatibility minimize risk of data loss when migrating an archive to software systems using a new protocol.

Table 18. Formalization of Operational Procedures (CMMI)

Level	Complexity	Minimum Level to Meet Core Requirement
1	Undocumented, Ad Hoc Operational Procedures	Below
2	Documented Procedures, with Some Ad Hoc Deviations	Below
3	Documented Procedures Periodically Reviewed and Updated with Statistical Data	Above
4	Documented Procedures Based on Formal Model of Effective Organizational Communication Patterns	Above

Table 19. Training

Level	Complexity	Minimum Level to Meet Core Requirement
1	Ad Hoc	Below
2	Training Necessary for Understanding and Using Base Components of System (e.g., OS, Compilers, Scripting)	Below
3	2 + Training Necessary for Understanding and Using Infrastructure Components of System (e.g., Objects Internal to Archive)	Below
4	3 + Training Necessary for Understanding and Modifying Archive Contents	Below
5	4 + Training Necessary for Evolving the Archive	Above

Table 20. Procedures for Operational Procedure Evolution

Level	Complexity	Minimum Level to Meet Core Requirement
1	Undocumented Procedures **(Bad Operational Practice)**	Below
2	Operational Procedures Documented	Below
3	2 + Systematic and Periodic Review	Above

Table 21. Peer Review of Design and Documentation

Level	Complexity	Minimum Level to Meet Core Requirement
1	No Documentation	Below
2	Peer Review within Local Developer Community	Above
3	2 + On-Site, Authorized User Peer Review	Above
4	3 + Full User Participation in All Development Activities	Above

Notes: Peer Review is generally regarded as highly beneficial to producing system components of high quality. Peers are members of the development team or individuals respected by the development team for their knowledge of the system and their skills in solving problems. Peer Reviewers must take the time to understand and review the content of the material being reviewed. They may also suggest solutions to problems that are discovered in the course of the review.

An important intent of peer review is to make the design, code, and procedures part of the development community's knowledge, rather than being the property of a single individual.

Peer Review is not equivalent to a Process that engages in large, Formal Reviews, in which the development team invites large numbers of people to watch specially prepared review packages. The Airlie Software Council has identified Formal Reviews as one of the nine worst software development practices. In the experience of the authors, Formal Reviews are political in nature and waste extraordinary amounts of time an energy. They also exhaust the developers and waste large amounts of paper and other scarce resources.

Table 22. Open Source Publication of Design and Code

Level	Complexity	Minimum Level to Meet Core Requirement
1	No Publication	Below
2	Publication of Design and Code	Above

Table 23. Ability to Interact with Open Source Community and Incorporate Open Source Contributions

Level	Complexity	Minimum Level to Meet Core Requirement
1	No Open Source Community Involvement	Below
2	Publication of Design and Source Code in Open Source Form	Above
3	2 + Active Solicitation of Contributions from Open Source Community	Above
4	3 + Formal Organization of Open Source Project	Above

Table 24. Ability to Cooperate in Archive Federations that Maintain Local Autonomy

Level	Complexity	Minimum Level to Meet Core Requirement
1	No Participation in Archive Federations	None given
2	Voluntary Participation in Federations with Goal of Interoperability	None given
3	2 + Resource Sharing Between Federation Members	None given

Note: There have been a number of studies and simulations of communities that engage in resource sharing. These suggest that community members may participate in resource sharing without the need for centralized oversight. We recognize the variability in the needs of various kinds of archives, which may allow some archives to operate in a solitary fashion, while others benefit greatly from the participation in federations. Accordingly, we do not set a minimum level of cooperation in order to meet Archive Core Requirements.

Table 25. Intellectual Property Rights Considerations Included In Design

Level	Complexity	Minimum Level to Meet Core Requirement
1	No Recognition of Intellectual Property Rights in Use Cases or Design	Below
2	Intellectual Property Rights Controlled by Access Control Lists	Below
3	2 + Allowance fo Internal System Partitioning of IPR	Above
4	3 + Procedures Developed to Allow Narrowing or Broadening of IPRs	Above

Table 26. Permanent File and File Content Naming and Registration

Level	Complexity	Minimum Level to Meet Core Requirement
1	File Names and File Content Names Assigned without Consideration of Permanence	Below
2	Files Have Permanently Registerable Identifiers as well as Names	Above
3	2 + File Contents have Structural Indexes that Allow Permanent Reference to and Retrieval of Individual Data Elements– even in the event of Data Migrations and Transformations	Above

Note: This set of requirements arises from the need to be able to uniquely identify files and their contents. We envision that the contents of the files must also be permanently identifiable, even if the data must migrate from one storage medium to another–or reformatted as a result of software or hardware obsolescence.

A simple example of the kind of capability needed for file content naming is the identification of the pixels in a satellite image that have been identified by a data mining algorithm as belonging to a particular hurricane instance. It is quite possible that the file would be reorganized during data migration, so that what may have been included in the original data have been reorganized by data reformatting into a very different format–or even broken into separate files.

Implementation of this capability will almost certainly require creation of Information Packages (in the sense defined by the OAIS Reference Model) that can refer to format transformations endured by the data after its original archival.

Table 27. Verifiable System for Maintaining Chain-of-Custody

Level	Complexity	Minimum Level to Meet Core Requirement
1	No Verifiable System Incorporated into Design for Maintaining Chain-of-Custody of Archive Contents	Below
2	System Design incorporates systematic procedure for recording data transfers based on the Negotiated Submission Agreement Between a Data Provider and an Archive– although the Archive maintains no automated procedure for verifying the transfers from the Data Provider or within the Archive	Below
3	2 + Automatic Recording of Data Ingest and Transfer	Below
4	3 + Transaction Based Data Ingest and Transfer Operations	Below
5	4 + Auditing, Reconciliation, and Periodic Inventory of Archive Contents and User Accesses	Above

Table 28. Access Control and Authentication

Level	Complexity	Minimum Level to Meet Core Requirement
1	No Access Control or Authentication Procedures	Below
2	Access Control Lists Maintained with Manual Procedures	Below
3	Access Control Lists Maintained with Automated Procedures and Transactional Auditing and Reconciliation	Above
4	3 + Authentication of Users (both Internal and External)	Above
5	4 + Physical Isolation of Archive	Above

Table 29. Transactional Basis for System Operation

Level	Complexity	Minimum Level to Meet Core Requirement
1	No Record of System Messages Between Objects	Below
2	Messages Generate Logs Used with Manual Monitoring	Below
3	All System Activities Operate as Transactions that can be rolled back	Below
4	3 + Automated Fall-back and Recovery Mechanisms	Above

Table 30. Transaction Auditing and Reconciliation

Level	Complexity	Minimum Level to Meet Core Requirement
1	No Transactions	Below
2	Messages Recorded in Logs with Manual Monitoring	Below
3	All Activities Operate on a Transactional Basis with Automated roll-back, backup, and restore capability	Above
4	3 + Transaction and REconciliation used to develop statistical data used for monitoring the system reliability in data handling operations	Above

Engineering Requirements

Table 31. Logical Name Space Complexity

Level	Complexity	Minimum Level to Meet Core Requirement
1	Flat–One Layer	x
2	Multi-Layer, Homogeneous	x
3	Multi-Layer, Inhomogeneous Nodes	x

Table 32. Metadata Complexity

Level	Complexity	Minimum Level to Meet Core Requirement
1	Only Permanent ID and Full File Path Name (Logical Name) in Flat (One-Level) Hierarchy	x
2	Multi-Level Logical Name Space with Permanent ID and Full File Path Name	x
3	Multi-Level Logical Name Space with Single Table (no one-to-many or many-to-many relations) of Attribute at each LNS Node	x
4	3 + Junction Tables (one-to-many and many-to-many relations) of Attributes at each LNS Node	x
5	4 + Multiple Path Search Nodes	x
6	5 + Non-Homogeneous Logical Name Space Nodes	x

Probable User Success Requirements

Table 33. Search Interface Complexity

Level	Complexity	Minimum Level to Meet Core Requirement
1	One File per Search–Permanent ID and Full File Path Name Searches Only Level) Hierarchy	x
2	Multiple Files per Search–Permanent ID and Full FIle Path Name Seaches Only	x
3	1 + Simple Logical Name Space Traversal (Multi-Click Navigation	x
4	3 + Visibility of Node Metadata as a Table	x
5	4 + Multiple Files Selectable per Search	x
6	5 + SQL Query using Visible Buttons in Web Pages	x
7	6 + Multi-Entry Concept Maps	x
8	7 + Multi-Persona Searches	x

Table 34. User Help Complexity

Level	Complexity	Minimum Level to Meet Core Requirement
1	User Manuals Only	x
2	1 + User Help Desk with Phone and Email Responses	x
3	2 + Interface Input Error Messages	x
4	3 + On-Line Tutorials in Context	x

Table 35. Data Distribution Complexity

Level	Complexity	Minimum Level to Meet Core Requirement
1	FTP Pull	x
2	1 + FTP Push	x
3	2 + Single Kind of Media Distribution	x
4	3 + Multiple Kinds of Media Distribution	x
5	2 + Specialized Subsetting and Reformatting Web Services	x
6	5 + Specialized Transformational Web Services (e.g., Visualization)	x