NATIONAL LIBRARY OF MEDICINE

Recommendations on NLM Digital Repository Software

Prepared by the
NLM Digital Repository Evaluation and Selection Working Group

Submitted December 2, 2008

Contents

1. Executive Summary

The Digital Repository Evaluation and Selection Working Group recommends that NLM select Fedora as the core system for the NLM digital repository. Work should begin now on a pilot using four identified collections from NLM and the NIH Library. Most of these collections already have metadata and the NLM collections have associated files for loading into a repository.

The Working Group evaluated many options for repository software, both open source and commercial systems, based on the functional requirements that had been delineated by the earlier Digital Repository Working Group. The initial list of 10 potential systems/software was eventually whittled down to 3 top possibilities: two open source systems, DSpace and Fedora, and DigiTool, an Ex Libris product. The Working Group then installed each of these systems on a test server for extensive hands on testing. Each system was assigned a numeric rating based on how well it met the previously defined NLM functional requirements.

While none of the systems met all of NLM's requirements, Fedora (with the addition of a front end tool, Fez) scored the highest and has a strong technology roadmap that is aggressively advancing scalability, integration, interoperability, and semantic capabilities. The consensus opinion is that Fedora has an excellent underlying data model that gives NLM the flexibility to handle its near and long-term goals for acquisition and management of digital material.

Fedora is a low-risk choice because it is open-source software, so there are no software license fees, and it will provide NLM a good opportunity to gain experience in working with open source software. It is already being used by leading institutions that have digital project goals similar to NLM's, and these institutions are an active development community who can provide NLM with valuable advice and assistance. Digital assets ingested into Fedora can be easily exported, if NLM were to decide to take a different direction in the future.

Implementing an NLM digital repository will require a significant staffing investment for the Office of Computer and Communications Systems (OCCS) and Library Operations (LO). This effort should be considered a new NLM service, and staffing levels will need to be increased in some areas to support it. Fedora will require considerable customization. The pilot project will entail workflow development and selection of administrative and front end software tools which would be utilized with Fedora.

The environment regarding repositories and long term digital preservation is still very volatile. All three systems investigated by NLM have new versions being released in the next 12 months. In particular, Ex Libris is developing a new commercial tool that holds some promise, but will not be fully available until late 2009. The Working Group believes NLM must go forward now in implementing a repository; the practical experience gained from the recent testing and a pilot implementation would continue to serve NLM with any later efforts. After the pilot is completed, NLM can re-evaluate both Fedora and the repository software landscape.

2. Introduction and Working Guidelines

2.1. Introduction

In order to fulfill the Library's mandate to collect, preserve and make accessible the scholarly and professional literature in the biomedical sciences, irrespective of format, the Library has deemed it essential to develop a robust infrastructure to manage a large amount of material in a variety of digital formats. A number of Library Operations program areas are in need of such a digital repository to support their existing digital collections and to expand the ability to manage a growing amount of digitized and born-digital resources.

In May 2007, the Associate Director for Library Operations approved the creation of the Digital Repository Evaluation and Selection Working Group (DRESWG) to evaluate commercial systems and open source software and select one (or combination of systems/software) for use as an NLM digital repository. The group commenced its work on June 12, 2007 and concluded its work December 2, 2008. Working Group members were: Diane Boehr (TSD/CAT), Brooke Dine (PSD/RWS), John Doyle (TSD/OC), Laurie Duquette (HMD/OC), Jenny Heiland (PSD/RWS), Felix Kong (PSD/PCM), Kathy Kwan (NCBI), Edward Luczak (OCCS), Jennifer Marill (TSD/OC), chair, Michael North (HMD/RBEM), Deborah Ozga (NIH Library) and John Rees (HMD/IA). Doron Shalvi (OCCS) joined the group in October 2007 to assist in the set up and testing of software.

The group's work followed that of the Digital Repository Working Group, which created functional requirements and identified key policy issues for an NLM digital repository to aid in building NLM's collection in the digital environment.

The methodology and results of the software testing are detailed in Sections 3-4 of this report. Section 5 provides the Working Group's recommendations for software selection and first steps needed to begin building the NLM digital repository.

2.2. Working Guidelines

2.2.1. Goals and Scope of the NLM Digital Repository

Institutional Resource
The NLM digital repository will be a resource that will enable NLM's Library Operations to preserve and provide long-term access to digital objects in the Library's collections.

Contents
The NLM digital repository will contain a wide variety of digital objects, including manuscripts, pamphlets, monographs, images, movies, audio, and other items. The repository will include digitized representations of physical items, as well as born digital objects. NLM's PubMed Central will continue to manage and preserve the biomedical and life sciences journal literature. NIH's CIT will continue to manage and preserve HHS/NIH videocasts.

Future Growth

The NLM digital repository should provide a platform and flexible development environment that will enable NLM to explore and implement innovative digital projects and user services utilizing the Library's digital objects and collections. For example, NLM could consider utilizing the repository as a publishing platform, a scientific e-learning/e-research tool, or to selectively showcase NLM collections in a very rich online presentation.

2.2.2. Resources

OCCS

Staff will provide system architecture and software development resources to assist in the implementation and maintenance of the NLM digital repository.

Library Operations

Staff will define the repository requirements and capabilities, and manage the lifecycle of NLM digital content.

3. Project Methodology and Initial Software Evaluation Results

3.1 Project Timeline

The Working Group held its kick-off meeting June 12, 2007 and completed all work by December 2, 2008.

- Phase 1: Completed September 25, 2007. A qualitative evaluation was conducted of 10 systems, and three were selected for in-depth testing.
- Phase 2: Completed October 22, 2007. A test plan was developed and a wide range of content types was selected to be used for testing.
- Phase 3: Completed October 13, 2008. Three systems were installed at NLM and hands-on testing and scoring of each was performed. On average, each system required 85 testing days or just over four months from start of installation to completion of scoring.
- Phase 4: Completed December 2, 2008. The final report was completed and submitted.

3.2. Project Start: Preliminary Repository List

Based on the work of the previous NLM Digital Repository Working Group, the team conducted initial investigations to construct a list of ten potential systems/software for qualitative evaluation. The group also identified various content and format types to be used during the in-depth testing phase.

3.3. Qualitative Evaluation of 10 Systems/Software

The Working Group conducted a qualitative evaluation of the 10 systems, by rating each system using a set of Master Evaluation Criteria established by the Working Group (see Appendix A). Members reviewed Web sites and documentation, and talked to vendors and users to qualitatively rate each system. Each system was given a rating of 0 to 3 for each criterion, with 3 being the highest rating. Advantages and risks were also identified for each system.

The Working Group was divided into four subgroups, and each subgroup evaluated two or three of the 10 systems. Each subgroup presented their research findings and initial ratings to the full Working Group. The basis for each rating was discussed, and an effort was made to ensure that the criteria were evaluated consistently across all 10 tools. The subgroups finalized their ratings to reflect input received from discussions with the full Working Group.

All 10 systems were ranked, and three top contenders were identified (see Appendix B). DigiTool, DSpace, and Fedora were selected for further consideration and in-depth testing. Below are highlights of the evaluation of the 10 systems.

ArchivalWare

- Developed by: PTFS (commercial).

- Advantages:
  - Strong search capabilities.
- Risks:
  - Small user population.
  - Reliability and development path of vendor unknown.

CONTENTdm

- Developed by: University of Washington and acquired by OCLC in 2006 (commercial).
- Advantages:
  - Good scalability.
- Risks:
  - No interaction with third party systems.
  - Data stored in proprietary text-based database and does not accommodate Oracle.
  - Development path of vendor unknown.

DAITSS

- Developed by: Florida Center for Library Automation (FCLA) (open source) and released under the GNU GPL license as a digital repository system for 11 public universities.
- Advantages:
  - Richest preservation functionality.
- Risks:
  - Back-end/archive system.
  - Must use DAITSS in conjunction with other repository or access system.
  - Planned re-architecture over next 2 years.
  - Limited use and support; further development dependent on FCLA (and FL state legislature).

DigiTool

- Developed by: Ex Libris (commercial) as an enterprise solution for the management, preservation, and presentation of digital assets in libraries and academic environments.
- Advantages:
  - "Out-of-the-box" solution with known vendor support.
  - Provides good overall functionality.
  - Has ability to integrate and interact with other NLM systems.
  - Scalability and flexibility may be issues.
- Risks:
  - NLM may be too dependent on one commercial vendor for its library systems.

DSpace

- Developed by: MIT Libraries and HP Labs (open source) as one of the first open source platforms created for the storage, management, and distribution of collections in digital format.
- Advantages:
  - "Out-of-the-box" open source solution.
  - Provides some functionality across all functional requirements.
  - Community is mature and supportive.
- Risks:
  - Planned re-architecture over next year.
  - Current version's native use of Dublin Core metadata is somewhat limiting.

EPrints

- The Subgroup decided to discontinue the evaluation due to EPrints (open source) lack of preservation capabilities and its ability to only provide a small-scale solution for access to pre-prints.

Fedora

- Developed by: University of Virginia and Cornell University libraries (open source).
- Advantages:
  - Great flexibility to handle complex objects and relationships.
  - Fedora Commons received multi-million dollar award to support further development.
  - Community is mature and supportive.
- Risks:
  - Complicated system to configure according to NLM research and many users.
  - Need additional software for fully functional repository.

Greenstone

- Developed by: Cooperatively by the New Zealand Digital Library Project at the University of Waikato, UNESCO, and the Human Info NGO (open source).
- Advantages:
  - Long history, with many users in the last 10 years.
  - Strong documentation with commitment by original creators to develop and expand.
  - Considered "easy" to implement a simple repository out of the box.
  - DL Consulting available for more complex requirements.
  - Compatible with most NLM requirements.
- Risks:
  - Program is being entirely rewritten (C++ to Java) to create Greenstone 3. Delivery date unknown.

    - Development community beyond the originators is not as rich as other open source systems.
    - DL Consulting recently awarded grant "to further improve Greenstone's performance when scaled up to very large collections" -- implies it may not do so currently.
    - Core developers and consultants in New Zealand.

Keystone DLS

- Developed by: Index Data (open source).
- Advantages:
    - Some strong functionality.
- Risks:
    - Relatively small user population.
    - Evaluators felt it should be strongly considered only if top 3 above are found inadequate.
    - No longer actively being developed as of August 2008.

VITAL

- Developed by: VTLS, Inc. (commercial) as a commercial digital repository product that combines Fedora with additional open source and proprietary software and provides a quicker start-up than using Fedora alone.
- Advantages:
    - Vendor support for Fedora add-ons.
- Risks:
    - Vendor-added functionality may be in conflict with open-source nature of Fedora.

3.4. In-depth Testing of 3 Systems/Software

DSpace, DigiTool, and Fedora were selected as the top three systems to be tested and evaluated. Four subgroups of the Working Group (Access, Metadata and Standards, Preservation and Workflows, Technical Infrastructure) were formed to evaluate specific aspects of each system.

System testing preparation included:

- Creating a staggered testing schedule to accommodate all three systems.
- Selecting simple and complex objects from the NLM collection lists.
- Identifying additional tools that would be helpful in testing DSpace and Fedora (e.g. Manakin and Fez).
- Developing test scenarios and plans for all four subgroups based on the functional requirements.

A Consolidated Digital Repository Test Plan was created based on the requirements enumerated in the NLM Digital Repository Policies and Functional Requirements Specification. The Test

Plan contains 129 specific tests, and is represented in a spreadsheet. Each test was allocated to one of the four subgroups, who were tasked to conduct that test on all three systems.

DSpace 1.4.2, DigiTool 3.0, and Fedora 2.2/Fez 2 Release Candidate 1 were installed on NLM servers for extensive hands-on testing. OCCS conducted demonstrations and tutorials for DSpace and Fedora, and Ex Libris provided training on DigiTool, so that members could familiarize themselves with the functionalities of each system. The Consolidated Digital Repository Test Plan guided the testing and scoring of the three systems. Details of the testing are available in the next section.

4. Final Software Evaluation Results

The Technical Infrastructure, Access, Metadata and Standards, and Preservation and Workflows subgroups conducted the test plan elements allocated to their subgroup in the Consolidated Digital Repository Test Plan. Selecting from a capability/functionality scale of 0 to 3 (0=None, 1=Low, 2=Moderate, 3=High), the subgroups assigned scores to each element, indicating the extent to which the element was successfully demonstrated or documented. Scores were added up for each subgroup's set of test elements. A cumulative score for each system was calculated by totaling the four subgroup scores.

The Fedora platform and Fez interface were evaluated as a joint system.

4.1 Summary of Hands-on Evaluation

Subgroup	DSpace	DigiTool	Fedora (w/Fez)
Technical Infrastructure	36	51	49.75
Access	40	66	52.5
Metadata and Standards	16	27.5	40.75
Preservation and Workflows	42	45	56.5
Total Score	**134**	**189.5**	**199.5**

4.1.1. DSpace 1.4.2 Evaluation

See Appendix C for complete testing results.

4.1.1.1. Technical Infrastructure, score=36

- Data model well suited for academic faculty deposit of papers but does not easily accommodate other materials.
- All bitstreams uniquely identified via handles and stored with checksums.
- Very limited relationships between bitstreams (html document can designate the primary bitstream, hiding the secondary files that make up a web page).
- Workflow limited to three steps.
- Dublin Core metadata required for ingest. Other metadata can be accepted as a bitstream but would not be searchable.
- Versioning of objects/bitstreams not supported.
- Some usage and inventory reporting built-in.
- DSpace uses the database to store content organization and metadata, as well as administrative data (user accounts, authorization, workflow status, etc).

4.1.1.2. Access, score=40

- User access controls are moderate, with authorizations logic restricting functions to admin users or authenticated users.
- Although objects can have text files associated as licenses, there is not application logic to make use of license data, and no built-in way to facilitate content embargoes/selective user access.
- Entire collections can be hidden to anonymous users, but metadata remains viewable.
- Audit history written to a cumulative log which must be parsed by scripts into human-readable formats, and metadata actions are only sparsely logged.
- External automated access to Dublin Core metadata via OAI-PMH.
- Content is searchable by Dublin Core metadata and full text.
- Files are listed in the order they were ingested and cannot be sorted.

4.1.1.3. Metadata and Standards, score=16

- Dublin Core metadata required for ingest.
- Other metadata can be accepted as a bitstream but would not be searchable.
- Metadata validation not possible.
- Exporting of objects as METS files, but METS not currently supported as an ingest format.

4.1.1.4. Preservation and Workflows, score=42

- Exported data can be re-ingested with a replace function.
- Checksum checker can periodically monitor the bitstreams for integrity.
- No normalization capability.
- No referential integrity checks.
- No tools for file migration.
- Provenance for record updates is lacking.

4.1.1.5. System support issues

- **Platform support**: DSpace runs on Solaris, Linux, other UNIX, or Windows servers. It is a Java application, and uses Apache Tomcat, Apache Ant, and other open source Java tools. DSpace uses a relational database that can be Oracle, PostgreSQL, or MySQL.
- **Deployment and maintenance**: OCCS personnel installed several copies of DSpace on Windows computers for initial testing and demonstration. OCCS then installed DSpace on an NLM Solaris server using an Oracle database for full testing and evaluation. DSpace is relatively simple to install and build, and has limited but adequate documentation. DSpace includes user interfaces for public access and repository administration; however, these interfaces are very plain, and difficult to customize.

Installation and usage problems can often be solved by asking for assistance from members of the DSpace community, by posting a request on the DSpace email list server.

- **Development and user organizations**: DSpace has a very active user community and open source development community, with over 400 institutional users worldwide including NLM LHC for the SPER research project. DSpace was initially developed with support from MIT and HP. In 2007, the DSpace Foundation was formed to continue development of the open source software and support its community.
- **Future roadmap**: Future plans for DSpace are not crystal clear, but there is good promise for continued development and community support:
  - A DSpace 2.0 architecture has been defined that will introduce major improvements to the tool, and development of these enhancements has already begun.
  - Plans are being made for significant collaboration with the Fedora Commons community, to address needs and functions that are common to these two tools. Grant funding for planning joint activities has recently been obtained from the Andrew W. Mellon Foundation.

4.1.1.6. User Visits/Calls

- University of Michigan (May 14, 2008)

4.1.2. DigiTool 3.0 Evaluation

See Appendix D for complete testing results.

4.1.2.1. Technical Infrastructure, score=51

- Overall, the group was impressed with the broad range of tools and continued to discover new functionality, although the discovery was difficult at times.
- The ingest process is one example of the difficulty the group experienced: understanding the use of the legacy Meditor and the web ingest tool and the difference between deposit and ingest. Ingest workflows seemed overly complex.
- Certain challenges were a result of the NLM environment: the security lockdown, the Meditor installation, and ActiveX.
- Quite a few tests were conducted. The group was particularly happy with the range of file types (DigiTool really shines in this area) and areas of metadata handling, especially in terms of METS.
- Other positive aspects are the automatic format configurations and the support of relationships between digital entities (parent-child, for example).
- Weak areas include lack of specific support for quality assurance and audit functionality and the overall system configuration management.
- Standards support is good.

4.1.2.2. Access, score=66

- The group's evaluation considered staff users as well as end user needs and functionality.

- Access features in both areas were pretty strong, in terms of granularity of permissions, access protocols (Z39.50, OAI-PMH, etc.), and the search results display.
- The group would like to see more flexibility in search options, such as relevance ranking, proximity, and "more like this." Poor browsing features and no leveraging of authority control. The group recognizes many of these features are available via Primo and through some customization of Oracle.
- Good faith effort towards Section 508 compliance is well-documented by the vendor.
- Generally, the feeling is that DigiTool very strong in the access area.

4.1.2.3. Metadata and Standards, score=27.5

- Ingest of multiple format types is a feature the group likes.
- The limitation to Dublin Core mapping is a hindrance.
- The group would like to see more information on validation (for example, validation that a MeSH heading is MeSH).
- Updating and adding metadata fields are easy.
- The group did not see metadata checking for batch files, only individual files.

4.1.2.4. Preservation and Workflows, score=45

- DigiTool has many rich features, especially the use of METS extraction, JPEG 2000 thumbnail creation, and tagging master files in two ways.
- The rollback feature is good.
- Weak areas include the lack of confirmation for ingest and individual rather than batch ingest.
- The group recognizes that most preservation functionality will be offered with the Ex Libris Digital Preservation System (DPS), currently in development. Many customers will continue using DigiTool and have no need for the enhanced preservation functionality that will be offered by the DPS.

4.1.2.5. System support issues

- **Platform support**: DigiTool runs on either a Solaris or Linux server, with an embedded Oracle database. The Meditor administrative client software runs on a desktop PC.
- **Deployment and maintenance**: Installation was performed by Ex Libris on an NLM Solaris server; the vendor will not allow the software to be installed by the user organization. The installation requirements presented no particular difficulties, with the exception of the Meditor client software which required administrator privilege to install on user PCs. Parts of the code base are very old, having been migrated from a legacy COBOL product. Ex Libris provided detailed training on the use of the software, and was responsive in answering questions.
- **Development and user organizations**: The DigiTool product development team is located in Israel, and is accessible via web conference and teleconference. A separate team at Ex Libris is also developing a new repository product, the Digital Preservation System. Contacted users reported mixed experiences with DigiTool - a few are happy (e.g., Boston College), but others were disappointed and abandoned the product (e.g.,

University of Maryland, University of Tennessee, and Brandeis University). A small but active user group exists.

- **Future road map**: Ex Libris recently indicated to NLM that DigiTool will cease to be an independent product, and will be reformulated as a module that can be optionally used with the new Ex Libris Digital Preservation System. These plans have not yet been publicly announced.
- **Security**: OCCS conducted a web application security scan of DigiTool using IBM's AppScan scanning tool, and found 126 high-severity issues and 22 medium-severity issues. The high-security issues included Cross-Site Scripting vulnerabilities and Blind SQL Injection vulnerabilities. An additional 229 low-severity issues and information issues were detected by the scan. Details are provided in the DRESWG Security Scan Results.

4.1.2.6. User Visits/Calls

- Boston College (May 2, 2008)
- Oak Ridge National Library (May 7, 2008)
- University of Tennessee, Knoxville (email exchange on DigiTool 3 beta testing in 2005; May 28, 2008)
- Center for Jewish History and The Jewish Theological Seminary (May 30, 2008)

4.1.3. Fedora 2.2/Fez 2 Release Candidate 1 Evaluation

See Appendix E for complete testing results.

4.1.3.1. Technical Infrastructure, score=Fedora: 40.5; Fez: 35.5; Combined Fedora/Fez maximum: 49.75

- Fedora is very strong in the range of files that can be ingested, metadata requirements, versioning, relationships, and audit trails.
- Fedora's web services-based interface to repository content makes it easy to integrate with external tools and custom front-ends.
- Fedora is weak in workflow capabilities. Fez ranges from minimum to adequate in workflow capabilities.
- Fedora provides good support for standards compliance: SOAP, OAI, Unicode, METS, PREMIS, etc.
- One question is whether Fedora can catch transmission errors when a file is ingested from a directory, a function available in SPER. Fedora can compute a checksum and add it to the SIP, and it will verify checksums, but there appears to be a bug: the checksums always match. This problem should be fixed in version 3.0.

4.1.3.2. Access, score=Combined Fedora/Fez: 52.5

- Fedora provides great flexibility and granularity re: access controls at the user, collection, object, datastream and disseminator levels. The downside to this flexibility is that it requires custom policies to be written using a specialized markup - learning curve for the admin/developer staff.

- Fez also has granular security options, including Active Directory integration. The Group was not able to successfully test some of the access control logic. A big downside to the administration of the controls is the need to multi-select values using the Ctrl key, making it very easy to accidently deselect values which may not even be visible to the user.
- Fedora includes an OAI-PMH service which can provide the Dublin Core metadata associated with an object. This service could run (on Fedora) with a Fez implementation as well.
- Fedora has a very basic default end-user interface but is extremely flexible in its ability to integrate with third-party front-ends. Fez offers a rich end-user UI including UTF8 character support, controlled keyword searching, and output into RSS. Both systems do not adequately highlight a preferred version of an object over other versions also made visible to the end user.
- Full text searching is available with both systems via a third-party indexing plug-in.
- Fedora's disseminator approach offers much flexibility to content delivery, and Fez's inability to leverage the dissemination is a significant downside to the Fez product.

4.1.3.3. Metadata and Standards, score=Fedora 40.75; Fez 33.75; Combined Fedora/Fez: 40.75

- Most of the ratings assigned were 3s.
- The most difficult aspect of Fedora is determining workflows.
- Fedora conducts all the metadata checks that are needed.
- Fedora is difficult to use, as is DigiTool; Fez is easier.
- Fez uses only schemas, not DTDs.
- Dublin Core, MODS, and so on can be used as long as they are built into the workflow.
- MARC is ingested as a datastream.
- Disseminator architecture and other Fedora data model features should enable NLM to implement metadata linkage or exchange between Fedora and Voyager.

4.1.3.4. Preservation and Workflows, score=Fedora: 55; Fez: 41.5; Combined Fedora/Fez maximum: 56.5

- Fedora provides a solid core set of preservation capabilities that can be extended with companion tools (e.g. JHOVE for technical metadata extraction).
- Fedora/Fez does not create a physical AIP package but generates a FOXML/METS file that contains metadata and links to all datastreams during ingest.
- Fedora assigns a PID and generates a checksum for each ingested datastream.
- Fez can generate three different .jpg derivatives for each ingested image datastream. The subgroup was unable to test Fedora's disseminator.
- GSearch (the Fedora Generic Search Service) may be implemented with Fedora to index all metadata captured in FOXML/METS but style sheets must be written to enable GSearch functionality.

- Fedora allows data to be exported in three different ways: archive, migrate and public access but Fez has a very limited data export function.
- Fedora/Fez provides ingest confirmation on screen but no summary statistics. The subgroup was unable to test mail notification functionality because the mail server was not set up.
- The purge function in Fez does not delete an object from the repository. In Fedora, purging deletes an object.
- Still have a need for workflows, if not for the software itself than for external business functions.

4.1.3.5. System support issues

- **User interface**: Fedora does not include a public web access user interface, so an external interface must be added. Options include open source tools designed for use with Fedora such as Fez and Muradora, or custom web pages developed in-house. The Fez product restricts Fedora's flexibility in some key areas (access controls and content modeling) and appears to be more tightly integrated into Fedora than other front ends (which could be swapped out without touching the content or core services). New versions of the Fez and Muradora tools are expected to be released in the next few months, and the Fedora Commons organization is now focusing attention on the Fedora community's need for a flexible user interface approach.
- **Search**: Fedora includes an optional search component called GSearch that can search any metadata or text data in the repository. Because of time limitations, only the more limited default Fedora search component was tested. The full GSearch component should be implemented with Fedora. Resource Index database for storing relationships among objects as semantic concepts for querying by discovery tools.
- **Platform support**: Fedora runs on Solaris, Linux, other Unix, or Windows servers. It is a Java application, and uses Apache Tomcat, Apache Ant, and other open source Java tools. Fedora uses a relational database that can be Oracle, MySQL, PostgreSQL, McKoi, or others.
- **Deployment and maintenance**: OCCS personnel installed several copies of Fedora on Windows computers for initial testing and demonstration. OCCS then installed Fedora on an NLM Solaris server using an Oracle database for full testing and evaluation. Fedora is easy to install and is accompanied by clear and comprehensive documentation. An installation script is provided that guides the installation and configuration process. Fedora 2.2.2 was the production release version of the software when the NLM evaluation began, and was the version installed for testing. During testing, Fedora 3.0 was released, a significant upgrade with new features and simplified code base. NLM spoke with several Fedora users, and all plan to upgrade to version 3.0. Fedora 3.0 should be used instead of earlier versions.
- **Development and user organizations**: Fedora has an active user community, with more than 100 user institutions listed in the Fedora Commons Community Registry. The first prototype of Fedora was begun in 1997, and the project was led for several years by University of Virginia and Cornell University with grant money obtained from the Andrew W. Mellon Foundation. In 2007, Fedora Commons was incorporated as a non-profit organization, and received nearly $5 million in grant money from the Gordon and

Betty Moore Foundation to continue development of the Fedora software, and to provide the resources needed to build a strong open source community. Fedora Commons supports the user and developer community with an active project web site, a wiki, and several email lists. All source code is managed on SourceForge. The Moore grant funds a leadership team, chief architect, lead developer, and several software developers. Several dozen additional developers are actively involved in the community at user institutions. Fedora is being used by leading institutions that have digital projects goals similar to NLM's. The users NLM has contacted are enthusiastic and confident in their choice of Fedora. They are building effective digital collections, and they can provide valuable advice and lessons-learned to NLM. Fedora is built using technologies that OCCS is prepared to support, including Java, Tomcat, XML, and web services.

- **Future roadmap**: The Fedora Commons Technology Roadmap is published on the Fedora Commons web site, and defines the Fedora vision, goals, priorities, and five major projects, with detailed development plans and schedules. Some projects are primarily directed by Fedora Commons, and others are collaborations with other open source projects.
- **Security**: OCCS conducted a web application security scan of Fedora using IBM's AppScan scanning tool, and found 1 high-severity and 1 low-severity issues. The high-security issue was a Cross-site scripting vulnerability. The remediation for this vulnerability is to filter out hazardous characters from user input. This issue should be addressed in consultation with the Fedora Commons community leadership. The AppScan tool provides detailed information about the vulnerability and the coding approach needed to correct it. Additional details of the security scan are provided in the DRESWG Security Scan Results.

4.1.3.6. User Visits/Calls

- University of Maryland (August 7, 2007 Site Visit)
- University of Virginia (Sept 11, 2008)
- Indiana University (Sept 16, 2008)
- Tufts University (Sept 17, 2008)
- Rutgers University (Sept 18, 2008)
- Presentation from Thornton Staples of Fedora Commons (Sept 29, 2008)
- Yale University (Oct 3, 2008)

5. Recommendations

5.1. Recommendation to use Fedora and Conduct a Phase 1 Pilot

The Digital Repository Evaluation and Selection Working Group recommends Fedora as the core system for the NLM digital repository and to start now on a phase 1 pilot to involve real collections. Fedora's architecture should enable NLM to ingest, manage, and deliver exotic content as well as the typical digital scans of print originals. It has the potential to encourage creative approaches to digital library research and development, e-publishing, e-scholarship, and e-science.

Fedora has been implemented by a number of institutions involved in innovative digital services, including Indiana University, Rutgers University, Tufts University, the University of Virginia, the Max Planck Society (eSciDoc), the National Science Foundation (The National Science Digital Library), the Public Library of Science, and the Inter-University Consortium for Political and Social Research.

Drawbacks include the extensive customization, training, and support required to implement and manage the complex architecture. Considerable time also will be invested in developing detailed workflows for Fedora. These risks, while significant, do not outweigh the system's benefits.

5.1.1 Key reasons for Fedora

- Provides the flexibility that will be needed to handle NLM's near-term and foreseeable future needs.
- Has a strong technology roadmap that is aggressively advancing scalability, integration, interoperability, and semantic capabilities.
- Is being used by leading institutions that have digital projects goals similar to NLM's.
- Has an active open source development community that is well-funded with grant money. Fedora is cutting edge yet bounded by a strong commitment to standards.
- Strongest and most flexible metadata support of all candidates - it is not bound to any single scheme.
- Hands-on functional testing has demonstrated that Fedora by itself scored well against NLM functional requirements, and, with the Fez add-on front-end tool, scored higher than DSpace and DigiTool.
- Fedora is a low-risk choice for NLM at this time:
    - Fedora is open source software, so there are no software license fees.
    - Other institutions like NLM are building effective digital collections using Fedora, and they can provide valuable advice and lessons-learned.
    - Digital assets ingested into Fedora can be easily exported, if NLM were to decide to take a different direction in the future.
    - Fedora is a good opportunity for NLM to gain experience with open source software.
    - Fedora is developed and maintained using technologies that OCCS can support.

5.1.2. Future Actions Needed

After the completion of a pilot, NLM should evaluate its work. Evaluation is a prudent plan to mitigate any risks associated with using Fedora. The pilot group should also re-evaluate the repository software landscape as new versions of all the tools examined are coming out over the next 12 months, including:

- Fedora just released version 3.1 which makes significant improvements in defining the content model.
- DSpace architecture will undergo major improvements with a new version, DSpace 2.0.

Plans are also being made for significant collaboration between the DSpace and Fedora communities and NLM should keep abreast of how these plans could support NLM's use of Fedora.

The pilot group may also want to determine if NLM should conduct a formal test of the Ex Libris Digital Preservation System (DPS). DPS is an emerging new commercial tool that offers future promise for digital repository applications:

- DPS is being developed to meet the requirements of the National Library of New Zealand (NLNZ), which rejected DigiTool.
- Release 1.0 is expected to be generally available by end of 2008/early 2009.
- NLNZ has gone live with DPS and is happy with the results so far.

5.2. Phase 1 Pilot Recommendations

NLM should start with Fedora 3.1, the latest production release version. NLM hasn't exhaustively tested 3.x but is starting to examine the code and new key features. Other institutions which the group has spoken with are planning to migrate from 2.x to 3.x.

5.2.1. Companion Tools

- Use of Fedora open source software gives NLM the opportunity to select and incorporate "best-of-breed" companion tools.
- NLM can replace or add new tools as better alternatives become available.
- Tool awareness, evaluation, and selection will be a part of NLM's repository evolution process.
- Companion tool investigation needed during phase 1 pilot:
  - **Administrative interface tools**: The pilot group should not commit immediately to Fez but should investigate alternative administrative interface tools such as Muradora or the Rutgers Workflow Management System.
  - **Preservation tools**: Determine use of JHOVE and related tools such as DROID for file identification, verification and characterization.
  - **Public user interface tools**: Research and implement either open source or commercial page turning or other front end access capabilities and software.

5.2.2. Workflows

- The pilot group should make workflow recommendations over time and workflows may be tied to the collection or type of material.
- Workflows to be initially examined probably include metadata needed for SIPs (Submission Information Package) and format characterization.

5.2.3. Suggested Phase 1 Pilot Scope and Time Frame

6-8 months:

- Develop a first pilot collection that already has metadata and associated files. Produce a "quick" success to show progress.
- Manage the content in one secure place.
- Focus on defining the core functions in the areas of: data models, metadata, preservation and SIP creation.
- Investigate interfaces with Voyager to maximize use of existing metadata.
- Provide an initial public presentation using a simple Web interface.
- Investigate and begin to implement key preservation aspects to ensure master files are preserved.

8-18 months:

- Implement an additional one or two pilot collections (of the 4 proposed in section 5.4).
- Begin making recommendations on institutional workflows.
- Implement an administrative interface or collaborate with other users to evolve some open source alternative, or integrate/develop our own.
- Implement one or two unique public access capabilities (e.g., a page turning application).

5.2.4. NLM's Role in the Fedora Open Source Community

- NLM should investigate potential participation in the Fedora Commons community, e.g., the Fedora Preservation and Archiving Solution Community group. Participation could enable NLM to influence future software features. NLM should also investigate potential partnerships with leading Fedora users, e.g., University of Maryland, University of Virginia, or others. (These are strategic/management decisions.)
- NLM should consider contributing source code to the Fedora community only after the pilot phase, if NLM decides to continue its use of Fedora. NLM should become a participant rather than a "lurker."
- Before NLM shares any code it may want to consult with NIH legal counsel.

5.3. Phase 1 Pilot Resources Needed

The following summarized resources are estimated for the phase 1 pilot. Additional resource needs may be identified during the pilot and may be dependent on the collection(s) to be implemented.

5.3.1. LO

- .8 FTE Project Manager and Analyst. Develops phase 1 pilot plan including scope, schedule and deliverables. Tracks changes to requirements and monitors project progress. Provides technical input and oversight of all major functional areas.
- .5 FTE Metadata Specialist
- 2.1 FTE Analyst
- All the above to perform the following:
    - Analyze and develop workflows for various ingest and process models. (Refers to both single-file and batch mode).
    - Determine metadata schema(s) and element requirements for technical and descriptive metadata.
    - Define user community and access permissions. Develop specifications, specify requirements for interfaces with other internal systems and assist in developing integration plans for identified tools.
    - Develop specifications for management, preservation, and statistical reports including access methods, file formats, and delivery options.
    - Define data requirements including file formats, directory structure and information package for ingest.
    - Develop QA checklists for automatic and manual processes including data integrity checks and file format identification, validation and characterization.
    - Specify automatically generated error/confirmation/summary reports. (Refers to master, derivative and metadata files). Define derivative requirements.
    - Develop preservation plan including master file management, integrity checks, backup plan, file migration, etc.

- .5 FTE User Interface Analyst. Takes lead in designing staff and public web interfaces, including search options and viewing capabilities. Insures that usability testing, performance analysis, and 508 compliance are conducted according to NLM guidelines and standards. Additional guidelines may need to be developed depending on user needs for repository collections and formats.

5.3.2. OCCS

- 1 FTE Systems Architect/Analyst/Engineering Project Manager. Responsible for working with LO on implementation specifications, advising on technical options, tracking development progress, providing status updates, coordinating implementation efforts among different OCCS groups, building development team, etc. Performs analysis of open source and commercial software tools, including discussions with users, community members, and vendors.
- 1 FTE Software Engineer/Programmer. Responsible for installing, developing and testing programs and scripts. Provides overview and demonstrates new tools. Implements and tests integration of new and existing tools.
- .3 FTE Web Developer/User Interface Specialist. Primary responsibility for public interface design and programming. Works with User Interface Analyst on designing usable administrative/staff interfaces.

- Systems Engineer responsible for server preparation, network setup, system software configuration, etc.
- Database Administrator responsible for database configuration and administration.

5.4. Pilot Collections

The Working Group recommends the following digital collections as pilots for the repository in order to gain early implementation experience with many of the key capabilities of the selected NLM digital repository software. The files and metadata needed for the proposed collections are already available or can be compiled without significant effort. The Working Group recommends a variety of collection and file types be selected.

5.4.1. Cholera Monographs

HMD/RBEM and PSD/PCM have already scanned over 400 English language monographs in the collection relating to cholera dating from 1830 to 1890. HMD has already loaded many of the files online on a web site called Cholera Online, but the site is not searchable, except as part of the general NLM web search. Many of the PDFs are too large to download easily without a high speed connection. LO has high resolution tiff files with high quality technical metadata and METS/ALTO packages, of which the NLM digital repository should be able to use. Descriptive metadata for the materials already exists in Voyager. The Working Group would like to see a page turner installed for easy viewing of the materials in an online book-like format.

5.4.2. Digitized Motion Pictures

HMD has digitized a number of its historical audiovisuals for preservation and access purposes, and those created by the government are in the public domain. Metadata for these historical films already exists in Voyager. The Working Group proposes that as a pilot project, LO attempt to load about ten of these historical audiovisuals into the NLM digital repository. NLM may need to gain a waiver to post material in the NLM digital repository that are not 508 compliant; in the case of digitized motion pictures, this would require expensive closed captioning of any films put into the NLM digital repository.

5.4.3. Image Files from Historical Anatomies on the Web

HMD has selected and digitized over 500 images from important historical anatomical atlases in the collection and put them onto the web site, Historical Anatomies on the Web. The images are not searchable, however, by subject, artist, or author. Metadata does not exist for these individual images, so the Working Group proposes to add about 50 of the images from two of the most famous atlases (Vesalius' _De Fabrica_ and Albinus' _Tabulae sceletai_) in order to allow the pilot team to learn how to handle image files and enter metadata into the system.

5.4.4. NIH Institute Annual Reports (jointly with NIH Library)

Each year NIH Institutes and Centers issue annual reports, documents that provide historical perspective on research activities. Annual reports consist of a list of investigators for each research project and a project summary. More detail may be provided through individual project reports, which describe research objectives, methods, major findings, and resultant publications. In the mid-1990s, digital copies of many of the reports began to appear on Institute and Center web sites. Since 1998, intramural reports also have been submitted to the NIH Intramural Database for searching and viewing by NIH staff and the public (see NIDB Resources at http://intramural.nih.gov/mainpage.html). The NIH Library maintains a collection of older print NIH annual reports, totaling more than 700 volumes. To fill gaps in digital access, the Library plans to digitize the annual report collection, beginning with reports issued by the Clinical Center. The Clinical Center annual reports span thirty-five years, from 1958 to 1993. A pilot collection of eleven volumes has been selected for digitization and deposit in the NLM digital repository, covering fiscal years 1981 through 1993.

Appendix A - Master Evaluation Criteria Used for Qualitative Evaluation of Initial 10 Systems

NLM Digital Repository Master Evaluation Criteria

Updated August 13, 2007

Purpose

- Provide a decision method to select 3-4 systems for installation and testing at NLM from the initial list of 10 digital repository candidate systems.

Context

- The Digital Repository Evaluation and Selection Working Group (DRESWG) has begun evaluating the initial list of 10 candidate systems against a list of approximately 175 functional requirements specified in the *NLM Digital Repository Policies and Functional Requirements Specification,* March 16, 2007.
  - A weighted numerical scoring method is being used to compute a total score for each candidate system.
- The Functional Requirements score is one of the master evaluation criteria.
- Additional master evaluation criteria address other programmatic factors and risks that should be considered in the down-selection decision.

Master Evaluation Criteria

- **Functionality** - Degree of satisfaction of the requirements enumerated in the *NLM Digital Repository Functional Requirements Specification OR*
  - Evaluation: Numeric score as assessed by the Working Group
- **Scalability** – Ability for the repository to scale to manage large collections of digital objects.
  - Evaluation: 0-3 assessment scale (see below)
- **Extensibility** – Ability to integrate external tools with the repository to extend the functionality of the repository, via provided software interfaces (APIs), or by modifying the code-base (open source software).
  - Evaluation: 0-3 assessment scale (see below)
- **Interoperability –** Ability for the repository to interoperate with other repositories (both within NLM and outside NLM) and with the NLM ILS.
  - Evaluation: 0-3 assessment scale (see below)
- **Ease of deployment –** Simplicity of hardware and software platform requirements; simplicity of installation; ease of integration with other needed software.
  - Evaluation: 0-3 assessment scale (see below)
- **System security** – How well does the system meet HHS/NIH/NLM security requirements?
  - Evaluation: 0-3 assessment scale (see below)
- **System performance** – How well the system performs overall; response time (accomplished via load testing). System availability (24x7 both internally and externally?).
  - Evaluation: 0-3 assessment scale (see below)
- **Physical environment** – Ability for multiple instances for offsite recovery; ability to function with the NIH off-site backup facility (NCCS); ability for components to reside at different physical locations; ability for development, testing and production environments; capability for disaster recovery.
  - Evaluation: 0-3 assessment scale (see below)
- **Platform support** – Operating system and database requirements. Are these already supported by OCCS? Is there staff expertise to deal with required infrastructure?
  - **Preferable:** O/S: Solaris 10 (container); Storage: On NetApp via NFS; DB: Oracle; Web: java-tomcat or other application tier technology (OCCS will evaluate)

    - **Acceptable:** O/S: Windows 2003, Linux Red Hat ES; DB: MySQL; Web: (no constraints for now – OCCS will evaluate)
    - Evaluation: 0-3 assessment scale (see below)
- **Demonstrated successful deployments** – Relative number of satisfied users (organizations).
    - Evaluation: 0-3 assessment scale (see below)
- **System support –** Quality of documentation, and responsiveness of support staff or developer/user community (open source) to assist with problems.
    - Evaluation: 0-3 assessment scale (see below)
- **Strength of development community** – Reliability and support track record of the company providing the software; or size, productivity, and cohesion of the open source developer community.
    - Evaluation: 0-3 assessment scale (see below)
- **Stability of development organization –** Viability of the company providing the software; or stability of the funding sources and organizations developing open source software.
    - Evaluation: 0-3 assessment scale (see below)
- **Strength of technology roadmap for the future** – Technology roadmap that defines a system evolution path incorporating innovations and "next practices" that are likely to deliver value.
    - Evaluation: 0-3 assessment scale (see below)

To be considered only after the functional and technical criteria above are addressed:

- **Cost** – Expected total cost of software deployment, including initial cost of software, plus cost of software integration, modifications, and enhancements.
    - Evaluation: 0-highest cost 3-lowest cost

Assessment Scale

- 0 – None
- 1 – Low
- 2 – Moderate
- 3 – High

Appendix B - Results of Qualitative Evaluation of Initial 10 Systems

Final Systems Evaluation Matrix **Last updated: September 25, 2007**

		Type (open source, vendor)	Advantages	Risks	For further investigation	Notes
Top contenders	Fedora	Open source	Great flexibility to handle complex objects and relationships. Fedora Commons received multi-million dollar award to support further development. Community is mature and supportive.	Complicated system to configure according to our research and many users. Need additional software for fully functional repository.		
	DigiTool (Ex Libris)	Vendor	"Out-of-the-box" solution with known vendor support. Provides good overall functionality. Has ability to integrate and interact with other NLM systems.	Scalability and flexibility may be issues. NLM may be too dependent on one vendor for its library systems.	Ingest issues	
	DSpace	Open source	"Out-of-the-box" open source solution. Provides some functionality across all functional requirements (7.1-7.6) Community is mature and supportive.	Planned re-architecture over next year. Current version's native use of Dublin Core metadata somewhat limiting.		
Further evaluation and discussion needed	DAITSS	Open source	Richest preservation functionality	Back-end/archive system. Must use DAITSS in conjunction with other repository or access system. Planned re-architecture over next 2 years. Limited use and support; further development dependent on FCLA (and FL state legislature).	If selected for testing, code base needs examination for robustness.	
	Greenstone	Open source	Long history, with many users in the last 10 years. Strong documentation with commitment by original creators to develop and expand. Considered "easy" to implement (library school students have used it to create projects) a simple repository out of the box; DL Consulting available for more complex requirements. Compatible with most NLM requirements.	Program is being entirely rewritten (C++ to Java) to create Greenstone 3. Delivery date unknown. Development community beyond the originators is not as rich as other open-source systems. DL Consulting recently awarded grant "to further improve Greenstone's performance when scaled up to very large collections"—implies it may not do so currently. Core developers and consultants in New Zealand.		If selected for testing, not entirely clear whether Greenstone 3 (in beta) or Greenstone 2 (robust but going away) would be best to test with. Developers claim any system implemented in Greenstone 2 will be compatible with Greenstone 3. Should probably contact Greenstone developers and/or DL Consulting with this question if we select it.

		Type (open source, vendor)	Advantages	Risks	For further investigation	Notes
	Keystone DLS	Open source	Some strong functionality.	Relatively small user population. Evaluators felt it should be strongly considered only if top 3 above are found inadequate.		
No further consideration needed at this time	ArchivalWare (PTFS)	Vendor	Strong search capabilities.	Small user population. Reliability and development path of vendor unknown.		Very low rating across all master criteria.
	CONTENTdm (OCLC)	Vendor	Good scalability.	No interaction with third party systems. Data stored in proprietary text-based database and does not accommodate Oracle. Development path of vendor unknown.		Lower ratings across majority of master criteria.
	EPrints	Open source				Lower ratings across majority of master criteria.
	VITAL (VTLS)	Vendor	Vendor support for Fedora add-ons	Vendor-added functionality may be in conflict with open-source nature of Fedora.		If full evaluation of Fedora is successful, VITAL may be considered as an add-on.

Appendix C – DSpace Testing Results

Consolidated Digital Repository Test Plan Last updated: March 4, 2008		Source Requirements	Sub-group See Note 1	DSpace 1.4.2 Tests		
Test ID	**Test Plan Element**			**Test Procedure and Results**	**Score (0-3) Note 2**	**Notes**
7.1.1 Ingest - Receive Submission			**T**			
7.1.1.7	**File types** - Demonstrate that the system can ingest content in all the file formats listed as "supported" in Appendix B of the NLM DR Functional Requirements document (plus MP3 and JPEG2000), specifically: MARC, PDF, Postscript, AIFF, MPEG audio, WAV, MP3, GIF, JPEG, JPEG2000, PNG, TIFF, HTML, text, RTF, XML, MPEG. Demonstrate that the system can ingest the following types of content: articles, journals, images, monographs, audio files, video files, websites, numeric data, text files, and databases. Conduct this test element by ingesting the set of files listed in the Test File spreadsheet. (The files listed in this spreadsheet contain examples of all the file formats, and all the content types identified above.)	7.1.1.7 7.1.1.9	**T**	All files can be ingested. It is an implementation decision as to how the files/content are structured. Testing of "primary bit stream" for HTML files (KK): Shows primary bit stream file but hides all other files regardless of how related to HTML doc. Does not change original links in HTML doc.	3	
7.1.1.1	**Manual review** - Demonstrate that the system has the capability to require that submitted content be manually reviewed before it is accepted into the repository. Demonstrate that the system maintains submitted content in a staging area before it is accepted. Demonstrate that the system notifies a reviewer when new content is ready for review. (Also see tests for 7.1.4.1, 7.1.4.2, and 8.1.2.)	7.1.1.1	**T**	Workflow limited to 3 steps, although this will be generalized in next release, 1.5.	3	
7.1.1.2	**Review and acceptance workflow** - Demonstrate that the system supports a workflow for the review and acceptance of submitted content. Demonstrate that the workflow includes the following functions: - Receive and track content from producers; YES - Validate content based on submitter, expected format, file quality, duplication, and completeness; NO - Normalize content by converting content into a supported format for final ingestion into the repository; NO - Human review of content; YES - Acceptance or rejection of content or file format. YES	7.1.1.2, 7.1.1.10	**T**	JHOVE or similar needed for file validation. Tools/scripts available to parse log files.	2	

7.1.1.3	**Reason for rejection** - Demonstrate that the system records a set of identifying information or metadata that describes the reason for the rejection of submitted content. Demonstrate two cases: (1) automatic rejection, and (2) rejection by a human reviewer.	7.1.1.3	**T**	DSpace doesn't record the reason for rejection anywhere. The text of the reason that is manually entered by a reviewer is sent in an email back to the submitter, but the reason is not recorded in the DSpace database or the log file. The rejected item is kept as an "Unfinished Submission" in the submitter's My DSpace area, but the reason for rejection is not included with the item.	0	
7.1.1.4	**Rejection filter** - Demonstrate that the system allows the creation of a filter that can be used to automatically reject submitted content. (This capability will eliminate the need for manual review of some submissions and resubmissions.)	7.1.1.4	**T**			
7.1.1.5	**Rejection notification** - Demonstrate that the system can notify the producer or donor when submitted content is rejected. Demonstrate two cases: (1) notification after immediate rejection by an automated process, and (2) notification after rejection by manual review.	7.1.1.5, 7.1.1.11	**T**	1 - No 0 2 - Yes by email	1	
(7.1.1.8)	**Metadata types** - Demonstrate that the system can ingest content with associated metadata in the following formats: all NLM DTDs, Dublin Core, MARC21, MARCXML, ONIX, MODS, EAD, TEI, PREMIS, METS. (NOTE: This test is covered by tests 8.1.1, 8.1.8, and 8.1.9)	7.1.1.8, 8.1.1, 8.1.8, 8.1.9	**M/T**	Dublin Core only	1 (M & T)	
7.1.1.10	**Format conversion** - Demonstrate that the system has the capability to convert the format of a file being ingested to a desired supported format. As a test case, demonstrate that a WAV file can be converted to MP3 format when it is ingested. (An external tool may be needed to perform the conversion. If this is the case, demonstrate that the system can invoke the required external tool.)	7.1.1.10, 7.1.1.2	**T**	Definitely not a showstopper. External tool could possibly be used.	0	
7.1.1.12	**Resubmission** - Demonstrate that the system can ingest a SIP that is resubmitted after an error in the SIP was detected and corrected. Demonstrate two cases: the resubmission can occur after an error was detected in (1) the content of the SIP, and (2) the metadata of the SIP.	7.1.1.12	**T**	If an item is rejected by a reviewer, an email containing the reason for rejection is sent to the submitter. The rejected item is kept in the submitter's My DSpace area as an "Unfinished Submission." The submitter can edit the item, correct any errors, and resubmit it. When format errors are detected during batch submission, the error is reported in the command window where the batch submission command is run. The administrator can manually correct the format errors, and resubmit the item in another batch submission. There is no duplication checking.	2	
7.1.1.14	**Versions** - Demonstrate that the system can store, track, and link multiple versions of a file.	7.1.1.14	**T**	Planned for version 1.6 or 2.0	0	

7.1.1.15a	**Unique identifiers** - Demonstrate that the system assigns a unique identifier to each object ingested. Demonstrate two cases: (1) a unique identifier assigned to a digital object, which may be comprised of a set of component files, and (2) a unique identifier assigned to each of the component files of a digital object.	7.1.1.15a, 7.1.1.15b	**T**	A handle is associated with each item. Each bitstream is uniquely identified. The original Handle ID is retained during re-ingest and a new Handle ID is added when exported data are re-ingested. However, if the “replace” option is used, the re-ingest will only replace the files without adding a new Handle ID.	3	
7.1.1.15b	**Relationships** - Demonstrate that the system can represent a parent-child relationship between content items. Demonstrate two cases: (1) an object having multiple components (e.g., a document having multiple pages, each in a separate file), and (2) an object having multiple manifestations (e.g., an image having both TIFF and JPEG files).	7.1.1.15b	**T**	Item=parent; bitstreams=children Bitstreams can be "bundled," though this is not apparent to users. HTML page can be designated as "primary"	1.5	
7.1.1.16	**Audit trail** - Demonstrate that the system maintains an audit trail of all actions regarding receiving submissions (SIPs).	7.1.1.16	**T**	Info contained in log file but not easily usable.	1	
7.1.2 Ingest - Quality Assurance			**T**			
7.1.2.1	**Virus checking** - By design analysis, confirm that the system performs automatic virus checking on submitted content files.	7.1.2.1	**T**	Could be handled by external tool as part of pre-ingest process	0	
7.1.2.2	**Transmission errors** - Demonstrate that the system uses MD5, CRC, checksums, or some other bit error detection technique to validate that each data file submitted is received into the repository staging area without transmission errors.	7.1.2.2	**T**	MD5 computed and stored with each bitstream. SPER project added code to compute own MD5, which is part of SIP.	1	
7.1.2.3	**Submission validation** - Demonstrate that the system verifies the validity of submitted content based on the following criteria: submitter; expected file format; file quality (e.g., actual format of file matches the filename extension, and content of file is well-formed); duplication (e.g., existence of object in the repository); completeness of metadata; completeness of file set (e.g., all expected files are included in the submission).	7.1.2.3	**T**		0	
7.1.2.4	**QA UI** - Demonstrate that the system allows NLM staff to perform manual/visual quality assurance on staged SIPs via a user-friendly interface.	7.1.2.4	**T**			
7.1.2.5	**Reaction to QA errors** - Demonstrate that the system can react to specified QA errors in two ways: (1) request that the producer correct and resubmit the content, or (2) automatically modify the submission (e.g., converting to a supported format).	7.1.2.5	**T**	1 - Rejection email sent back to submitter. 2 - No automated way 2	1	
7.1.2.6	**File/batch accept/reject** - Demonstrate that the system enables NLM staff to accept or reject submitted content (SIPs) at the file or batch level.	7.1.2.6	**T**	File review is manual (one x one). Batch review is not automated.	1.5	
7.1.2.7b	**Error reports** - Demonstrate that the system generates error reports for ingest quality assurance problems.	7.1.2.7b	**T**	The DSpace statistics reports show a count of the number of item rejections and rejection notifications. The reports do not classify reasons for rejection, and do not include the text reason entered by the rejecting reviewer. Successful and unsuccessful batch ingests are not included in the statistics reports.	1	
7.1.2.8	**Adjustable level of manual QC** - By design analysis, confirm that the system has the ability to adjust the level of manual ingest quality control needed, based on the origin of the file.	7.1.2.8	**T**			

0

7.1.2.9	**Audit trail** - Demonstrate that the system maintains an audit trail of all actions regarding ingest quality assurance.	7.1.2.9	**T**	The DSpace log records when items are submitted, approved, and rejected. Reasons for rejection are not recorded. Successful and unsuccessful batch ingests are logged.	1	
7.1.4 Ingest - Generate Descriptive Information / Metadata			**M**			
7.1.4.1	**Additional metadata** - Demonstrate the entry of additional metadata (e.g. subject headings, names, dates, “curatorial” descriptive metadata - evaluative information that explains why an object is important, whether it was part of a larger collection (e.g., an exhibit), etc.).	7.1.4.1	**M**	Rather clunky	2	
7.1.4.2	**Validate metadata** - Demonstrate ability to validate specified metadata elements.	7.1.4.2	**M**			
7.1.4.4	**Metadata storage** - Demonstrate that metadata is stored in the database in a manner that conforms to repository reformatting and linked to their corresponding objects via an identifier. o Demonstrates that basic descriptive metadata is also stored with the objects (e.g., unique identifier, title and date stored in the TIFF header) so that the objects can still be identified in the event that information in the database is corrupted. o See Appendix D for examples of TIFF header metadata requirements. (Use of external tool probable)	7.1.4.4	**M**	First bullet - 2; second bullet - 2 0	2	
7.1.4.5	**Required descriptive elements** - Demonstrate the ability to recognize required descriptive elements.	7.1.4.5	**M**	would need an external tool; could write a program to do this	1	
7.1.4.7	**Audit trail** - Demonstrate the creation of an audit trail of all actions.	7.1.4.7	**M**			
7.1.3 Ingest - Generate AIP		Note 3	**P**			
7.1.5 Ingest - Coordinate Updates		Note 3	**P**			
7.2.1 Archival Storage - Receive Data		Note 3	**P**			
7.2.2 Archival Storage - Manage Storage Hierarchy		Note 3	**P**			
7.2.3 Archival Storage - Replace Media		Note 3	**P**			
7.2.4 Archival Storage - Error Checking and Disaster Recovery		Note 3	**P**			
7.2.5 Archival Storage - Provide Data		Note 3	**P**			
7.3.1 Data Management - Administer Database		Note 3	**P**			
7.3.2 Data Management - Administer Perform Queries		Note 3	**P**			
7.3.3 Data Management - Generate Report		Note 3	**P**			
7.3.4 Data Management - Receive Database Updates		Note 3	**P**			
7.4 Administration		Note 3	**P**			
P1 - Generate AIP			**P**			
P1-1	**Generate AIP** - Demonstrate the generation of AIPs from ingested SIPs that do not need normalization.	7.1.3.1, 7.1.3.2, 7.1.3.3, 7.4.1	**P**	Yes	2	

P1-2	**Generate AIP with normalization** - Demonstrate the generation of AIPs from ingested SIPs that need normalization - Transform an unsupported format to an accepted format (See Appendix B).	7.1.3.1, 7.1.3.2, 7.1.3.3, 7.4.1	**P**	NO: No normalization and submission auditing (check the title field only).	0	
P1-3	**Derivative files** - Demonstrate the generation of AIPs that consist of master files and derivatives.	7.1.3.6	**P**	Yes	1	
P1-4	**Master files** - Demonstrate the generation of AIPs that consist of master files only.	7.1.3.6	**P**	Yes	1	
P1-5	**Store AIP in archival storage** - Demonstrate the ability to transfer AIPs to Archive Storage.	7.1.5.1, 7.2.1.1, 7.2.1.2	**P**	Yes	2	
P1-6	**Store metadata in DB** - Demonstrate the ability to generate and transfer Descriptive Information (metadata) to Data Management Database.	7.1.5.2, 7.3.4.1	**P**	Yes	2	
P1-7	**Link metadata and objects** - Demonstrate the ability to store identification information in the Data Management database and link digital objects in the Archive Storage.	7.1.5.4	**P**	Yes	2	
P1-8	**Send confirmation** - Demonstrate the ability to automatically send confirmation to ingest and/or receiver when AIP and metadata transfers are completed.	7.1.5.1, 7.1.5.3, 7.2.1.3, 7.3.3.3	**P**	Yes for manual ingest. Batch ingest has only on-screen confirmation and can optionally invoke workflow process. Batch ingest provides on-screen confirmation of item ingest; data shown includes all metadata values and bitstream file names. (Ed verified)	2	
P1-9	**Send statistical reports** - Demonstrate the ability to automatically send statistical reports to ingest and/or receivers when AIP and metadata transfers are completed.	7.1.5.1, 7.1.5.3, 7.2.1.3, 7.3.3.3	**P**	No for manual ingest. The existing DSpace statistics reports do not include counts for items ingested via batch ingest. (Ed verified.)	0	
P1-10	**Send error reports** - Demonstrate the ability to automatically send error reports to ingest and/or receivers when AIP and/or metadata transfers fail.	7.1.5.1, 7.1.5.3, 7.2.1.3, 7.3.3.3	**P**	No for manual ingest. Error reports are not sent for batch ingest. (Ed verified.)	0	
P2 - Administer Archival Storage & Database			**P**			
P2-1	**Monitor transfer integrity** - Demonstrate the built-in function to automatically monitor and report if any AIPs and metadata are altered or corrupted during data transfer and media change (refresh or replace).	7.2.2.1, 7.2.3.2, 7.2.4.1	**P**	Yes. The DSpace Checksum Checker verifies that checksum of every bitstream file in the repository has not changed. The Checker can be configured to run regularly using the Unix cron. The Checker creates a log file that contains the results of the checksum checker run.	2	
P2-2	**Check data/referential integrity** - Demonstrate the built-in function to perform routine and special referential and data integrity checks (CRC or checksums) on files in the Archive Storage and Data Management Database.	7.2.4.2, 7.3.1.1, 7.3.1.2, 7.4.4	**P**	Data integrity checks (checksum) for data transfer but not for version upgrades and format migration (7.4.4.). Also no referential integrity checks (7.3.1.2).	1	
P2-3	**Routine configuration for data/referential integrity** - Demonstrate the ability to allow for routine configuration.	7.2.4.2, 7.3.1.1, 7.3.1.2, 7.4.4	**P**	See comments in P2-2.	1	

P2-4	**Disaster recovery** - Demonstrate the ability to allow for disaster recovery including data backup, off-site data storage, and data recovery.	7.2.4.3	**P**	Yes. Can be recovered from backup or exported data)	2	
P2-5	**User views** - Demonstrate the ability to allow for customized user views of the contents of the storage (create, maintain, and access).	7.3.1.4	**P**	Yes with external tools.	2	
P2-6	**System CM** - Demonstrate the ability to allow for configuration management of the system hardware and software.	7.4.2	**P**	Yes	2	
P2-7	**Database CM** - Demonstrate the ability to allow for configuration management of the Data Management Database such as table, schema definitions, etc.	7.3.1.3	**P**	Yes	2	
P2-8	**Delete AIPs** - Demonstrate the ability to allow the authorized staff to delete AIPs from the repository including: removing the digital object's files and retaining associated metadata, or removing both the files and metadata.	7.4.3.4	**P**	Yes	2	
P2-9	**Coordinate AIP removal** - Demonstrate the ability to generate an alert and coordinate the removal of an AIP with maintenance of metadata held in other systems.	7.4.3.5	**P**	No	0	
P2-10	**File migrations** - Demonstrate the ability to allow the authorized staff to schedule and perform file migrations or migration on request for batched and individual files by authorized staff.	7.4.3.6	**P**	No	0	
P2-11	**Request DIPs for update** - Demonstrate the ability to allow the authorized staff to request DIPs for file migrations and data updates.	7.3.4.1, 7.3.4.2, 7.3.4.3, 7.4.3.1, 7.4.3.2, 7.4.6.2	**P**	Yes. DIPs can be requested and exported. However, it provides no tools for file migration and data updates.	1	
P2-12	**Re-ingest updated DIPs** - Demonstrate the ability to allow the authorized staff to reingest updated DIPs as SIPs.	7.4.3.3	**P**	Yes. Exported data can be re-ingested with a replacing option. Ed will verify whether the re-ingest will also remove a deleted file from an item.	2	
P2-13	**Support query requests** - Demonstrate the ability to receive, retrieve, display, and deliver data for query requests from other functions such as Ingest, Access, and Administration.	7.3.2.1, 7.3.2.3, 7.4.6.1	**P**	Yes	2	
P2-14	**Query requests from different storage locations** - Demonstrate the ability to handle query requests with required data to be sourced from different storage locations.	7.3.2.2	**P**	Yes. Files can be optionally stored on a network file system.	2	
P2-15	**Queries against all metadata** - Demonstrate the ability to run data queries against all metadata used to manage the repository.	7.3.2.4	**P**	Yes	2	
P2-16	**Audit trial** - Demonstrate the creation of an audit trail of all actions including who, when, how, what and where for Archive Storage and Data Management Database.	7.1.3.4, 7.1.5.6, 7.2.1.4, 7.2.2.3, 7.2.5.2, 7.3.2.5, 7.3.3.7, 7.3.4.6, 7.4.3.7, 7.4.6.4	**P**	No provenance for record update and no email/screen confirmation for delete/withdrawal.	1	

P2-17	**Generate reports** - Demonstrate the ability to receive, generate, display, and deliver management information reports and statistics such as summaries of repository holdings by category, summaries of updates by category, user codes, etc., usage statistics for access to repository holdings, and descriptive information for a specific AIP.	7.3.3.1, 7.3.3.2, 7.3.3.5, 7.3.4.4	**P**	Perl was installed to enable DSpace's statistics tools to be exercised, and the reports were viewed by the entire working group. // Monthly and total repository lifetime reports can be generated that show the total number of archived items, but the items are not broken down into categories (e.g., collections, submitters). The total number of archived items shown in the report includes all items ingested via the web interface and by batch; the "items ingested" counts only those items ingested via the web interface (does not include items ingest by batch). Counts are shown for creation, update, and deletion of items, bitstreams, bundles, collection, and communities; but these counts are only for "all items," "all collections," etc. Updates to the metadata of items are not counted or otherwise reported. Counts are reported for total user logins, total item views, bitstream views, searches performed. User logins are also reported by userid. Items are identified that were viewed more than a certain number of times. Note: Some actions shown in the "All Actions Taken Report" were unclear.	1	
P2-18	**Schedule reports** - Demonstrate the ability to generate reports in an ad-hoc manner, automatically or to be triggered by a calendar or by a specific system event.	7.3.3.4	**P**	The Perl-based DSpace statistics report generator can be run manually by an administrator, or scheduled to run at any desired frequency (e.g., daily, weekly, monthly) using the Unix cron task scheduler. However, only two type of reports can be generated: total repository activity up through the current date/time, and monthly activity reports. DSpace can be configured so that reports can be viewed by all users, or only by administrators.	1	
P2-19	**Time period for reports** - Demonstrate the ability to allow the user to specify a time period or set of time periods for reports and statistics.	7.3.3.6	**P**	DSpace's Perl-based statistics and report generation tools only enable the creation of monthly reports.	0	
P3 - Generate DIP			**P**			
P3-1	**Generate DIP for access requests** - Demonstrate the generation of DIPs by putting AIPs and Descriptive Information back together for access requests.	7.1.5.5, 7.2.5.1, 7.4.6.2	**P**	Yes	2	

P3-2	**Generate DIP for object maintenance** - Demonstrate the generation of DIPs by putting AIPs and Descriptive Information back together for content/metadata update, versions upgrades and format migration by authorized staff.	7.4.6.2, 7.4.3	**P**	Yes	2	
7.4.1 Administration - Negotiate Submission Agreement			**T**			
7.4.1.1	**Manage submission agreements** - Demonstrate that the system manages information regarding submission agreements: that it tracks negotiation status and written submission agreements, and that it maintains schedules.	7.4.1.1	**T**			
7.4.1.2	**Edit submission agreements** - Demonstrate that the system allows submission agreements to be edited, based on the access level of the user.	7.4.1.2	**T**	0		
7.4.1.5	**Terms of submission agreements** - Demonstrate that the system stores the terms of submission agreements, and uses the terms to monitor, review, and process submissions.	7.4.1.5	**T**	0		
7.4.1.6	**Audit trail** - Demonstrate that the system maintains an audit trail of all actions related to submission agreements.	7.4.1.6	**T**	0		
7.4.2 Administration - Manage System Configuration			**T**			
7.4.2.1	**Monitor repository functionality** - Demonstrate that the system monitors the functionality of the entire repository.	7.4.2.1	**T**	Functions are monitored but not entire repository. 0	0	
7.4.2.2	**System configuration** - By design analysis, confirm that the system maintains the integrity of the system configuration.	7.4.2.2	**T**			
7.4.2.3	**Audits operations** - Demonstrate that the system audits system operations, performance, and usage.	7.4.2.3	**T**	In log file. 0	2	
7.4.2.4	**Data management information** - Demonstrate that the system collects and can display system information concerning Data Management.	7.4.2.4	**T**	Statistics reports show the number of items, communities, collections, bitstreams, and bundles that are created, updated, and deleted in each month, and totaled for all months of operation. Additional detailed information is collected in the log file, but no report is generated containing this detail. No information collected or reported on metadata updates. Reports don't breakdown activity by specific user, community, or collection.	1	
7.4.2.5	**Operational statistics** - Demonstrate that the system collects and can display operational statistics concerning Archival Storage.	7.4.2.5	**T**	Statistics reports show total number of items in archive, and number of items archived each month. Batch imported items are included in "total in archive", but not included in "items archived". No breakdowns by collection, community, user. No totals for bitstreams. Additional detailed information is recorded in the log file and history files, but no tool is currently available to report on this information.	1	
7.4.3 Administration - Archival Information Update						

7.4.5 Administration - Audit Submission			**T**			
7.4.5.1	**Audits** - Demonstrate that the system can support an audit procedure to verify that submissions (SIP or AIP) meet specified requirements of the repository. The audit method may be based on sampling, periodic review, or peer review. [See NLM DRD Functional Requirements document, section 7.4.5 for description of audit requirements.] (Also partially covered by 7.2.4.2)	7.4.5.1	**T**	0		
7.4.5.2	**Metadata audit** - Demonstrate that the system can audit metadata as part of the audit procedure.	7.4.5.2	**T**			
7.4.5.3	**Audit rejection -** Demonstrate that the system can reject components of audited information packages, based on specified audit requirements.	7.4.5.3	**T**	0		
7.4.5.4	**Audit report** - Demonstrate that the system can generate an audit report, based on the results of periodic audits of SIPs and AIPs.	7.4.5.4	**T**	0		
7.4.5.5	**Audit trail** - Demonstrate that the system maintains an audit trail of all actions regarding the auditing of SIPs and AIPs.	7.4.5.5	**T**	0		
7.4.6 Administration - Activate Requests			**P**			
7.6.1 Access - Coordinate Access Activities - User Access			**A**			
7.6.1.1	**Manage user permissions** - Demonstrate the access controls for multiple permission levels and user privileges.	7.6.1.1	**A**	Discrete admin accounts; authenticated users can be limited to specific functions or collections. Collections can be hidden from public (anonymous) view.	2	
7.6.1.2	**Manage user restrictions** - Demonstrate multiple levels of access restrictions for NIH employees and general public based on licensing terms, embargo periods, IP range restrictions, workstation access, and other possible legal restrictions.	7.6.1.2, 7.6.1.3	**A**	No built-in logic to tie access controls to licensing data. No built-in access controls based on IP ranges. Collections could be hidden from anonymous users via the Authorizations policies.	1	
7.6.1.4	**Manage user settings** - Demonstrate access settings allow staff to add or edit descriptive metadata	7.6.1.4	**A**			
7.6.1.7	**Audit users** - Demonstrate access mechanisms can identify individual users and maintain audit log of user actions.	7.6.1.7	**A**	Metadata edit actions are not logged in a useable way. 3	0	
7.6.1.5	**Perform maintenance tasks** - Demonstrate maintenance access including adding new files, manipulating images, editing metadata, performing format conversions/migrations, and troubleshooting system problems.	7.6.1.5	**A**	Maintenance actions demonstrated by Ed. Some can be through the UI, some are command-line only.	1	
7.6.1.6	**Manage system rights** - Demonstrate ultimate system rights access for NLM system administrators and programmers.	7.6.1.6	**A**	OCCS staff testing has demonstrated system-level access for both the system files and the Oracle schema.	1	
7.6.1 Access - Coordinate Access Activities - Rights/Data Control of Objects			**A**			
7.6.1.8	**Manage access rights** - Demonstrate access rights and conditions to materials and storage directories provide for a combinational of create/write; edit; read; delete privileges.	7.6.1.8	**A**	via Authorizations section	3	

7.6.1.9	**Manage metadata rights** - Demonstrate access rights may be associated with the metadata relating to an individual object	7.6.1.9	**A**	Authorizations do not apply to metadata, only to Communities, Collections, Items, Bundles and Bitstreams. Item metadata is always viewable.	0	
7.6.1.13	**Manage relationships** - Demonstrate access rights and conditions can be inherited from a parent object to any child object.	7.6.1.13	**A**	via Authorizations section	3	
7.6.1.14	**Manage relationships** - Demonstrate access rights and conditions can be assigned to an object on an individual or group basis at same time.	7.6.1.14	**A**	via Authorizations section	3	
7.6.1.16	**Automated retrieval** - Demonstrate objects in the repository are accessible for data mining or automated retrieval.	7.6.1.16	**A**	DSpace does not internally facilitate automated retrieval of its objects, only the metatdata via OAI-PMH. Full-text indexing by external source may be facilitated through a Java API in the future (JSR-170).	0	
7.6.1.17	**Metadata access** - Demonstrate access to deleted and retracted metadata is retained.	7.6.1.17	**A**	Minimal audit history in a cumulative log file accessible via scripts, but history of metadata actions is sparse.	0	
7.6.1.18	**Metadata harvesting** - Demonstrate metadata harvesting following the OAI-PMH guidelines.	7.6.1.18	**A**	DSpace will allow external hosts to harvest its metadata via OAI-PMH. It does not do harvesting (bring in metatdata).	2	
7.6.1.10	**Access rights** - Demonstrate access rights and conditions of use are applied to each digital object and its related metadata and are machine readable and actionable.	7.6.1.10, 7.6.1.11	**A**	License can be associated with a Collection or a specific item. There is no logic triggered by the license, which is just an unstructured text file.	1	
7.6.1.12	**Access conditions** - Demonstrate access conditions are specific to a digital object.	7.6.1.12	**A**	Only collection-level restriction - denial of read access to anonymous.	0	
7.6.1.15	**Free/Restricted access** - Demonstrate free (items available via internal/external delivery mechanisms) and restricted access (access permission must be satisfy various criteria) status for objects, files, metadata, etc.	7.6.1.15	**A**	Verified collection-level restriction via Authorizations (read access denied to anonymous users). No file-level restriction is available.	1	
7.6.1 Access - Coordinate Access Activities - Search and Retrieval			**A**			
7.6.1.19	**508 compliance** - Demonstrate the search interface is web-accessible and Section 508 compliant.	7.6.1.19	**A**	User interface is entirely web-based. Tested with Fangs and Accessibility Add-on in Firefox. Tables are used for layout, but alt tags may be input for images. Content scaled logically when CSS is disabled. Unable to validate HTML code.	2	

7.6.1.20	**Search features** - Demonstrate search includes: metadata, full-text, standard boolean, proximity, "more like" this"	7.6.1.20, 7.6.1.21, 7.6.1.22, 7.6.1.23, 7.6.1.24	**A**	Among the features listed, only metadata and, probably, full-text, are supported. Searching across DSpace needs additional investigation and it isn't clear how much configuration of Lucene (the underlying search engine) can be done.	1	
7.6.1.25	**Search results display** - Demonstrate search results display includes date sort; relevancy ranking; alpha by author or source.	7.6.1.25	**A**	Help mentions category search - how?	0	
7.6.1.26	**Relevancy ranking** - Demonstrate whether relevancy ranking can be manipulated via system as well as user defined settings.	7.6.1.26	**A**			
7.6.1.29	**Federated search** - Demonstrate federated searching of different repository sites.	7.6.1.29	**A**	0		
7.6.1.30	**Advanced search** - Demonstrate advanced search includes search history; saved searches; saved citation lists/bibliographies; alerts; various functions and formats; dynamic selection of delivery media without recreating search query.	7.6.1.30	**A**	0		
7.6.1.31	**Display formats** - Demonstrate a variety of standard display formats are provided and whether they are customizable by user	7.6.1.31	**A**	0		
7.6.1.32	**Alternate search interfaces** - Demonstrate availability of alternate search interfaces for mechanisms such as handhelds and PDAs.	7.6.1.32	**A**	0		
7.6.1.33	**Object access** - Demonstrate access to the appropriate copy of the identified item (text, image, video, etc.)	7.6.1.33	**A**	The record does describe the bitstream formats, but doesn't suggest the "appropriate" one 0	1	
7.6.1.34	**Library holdings** - Demonstrate integration of search results with library holdings.	7.6.1.34	**A**			
7.6.1.35	**Response time** - Demonstrate acceptable response time.	7.6.1.35	**A**	Good so far, but with very limited content. SPER testing suggests response time may suffer with large data sets. 0	1	
7.6.1.36	**External search engines** - Demonstrate searching by outside search engines such as usa.gov, Google, and Yahoo.	7.6.1.36	**A**	Several DSpace installations have been indexed by search engines.	2	
7.6.1.37	**External system access** - Demonstrate external access to other repositories or systems performing web harvesting functions.	7.6.1.37	**A**			
7.6.1.38	**Language support** - Demonstrate how multiple languages and non-Roman scripts are supported in search, retrieval and display.	7.6.1.38	**A**	demonstrated display of Chinese characters. Verified search & retrieval using Feng Chia University DSpace instance. 2	2	
7.6.1.39	**Versioning** - Demonstrate access to all versions of digital objects in the repository is provided.	7.6.1.39	**A**	No versioning functionality present. Any and all parts of an item will be accessible, but no relationships between parts can be conveyed.	0	
7.6.1.40	**Search settings** - Demonstrate system settings and user-defined settings in the search functions are provided.	7.6.1.40	**A**	Only default system-provided search settings are offered, through regular and advanced search interface.	0	

7.6.2 Access - Generate DIP			**A**			
7.6.2.1	**Integrate holdings** - Demonstrate integration of search results with library holdings.	7.6.2.1	**A**	Open-URLs can be utilized on item pages for possibly OPAC querying against the DC metadata associated with the item.	1	
7.6.2.2	**Retrieval and notification** - Demonstrate the generation function accepts a dissemination request, retrieves AIP from archival storage and moves a copy of the data to a staging area for further processing, and creates and sends a report request to data management to obtain appropriate metadata.	7.6.2.2, 7.6.2.3, 7.6.2.4	**A**	3		
7.6.2.7	**Audit trail** - Demonstrate an audit trail of all actions is created and stored.	7.6.2.7	**A**	Info contained in log file but not easily usable.	0	
7.6.2.5	**Response and delivery** - Demonstrate that the prepared DIP response is placed in the staging area and a message is generated and sent to Coordinate Access Activities that the DIP is ready for delivery.	7.6.2.5	**A**	This aspect of OAIS is not currently modeled by DSpace. DSpace does not appear to use a staging area but serves requested content directly from the Asset Store.	0	
7.6.2.6	**Storage retrieval** - Demonstrate that Generate function accesses data objects in staging storage and applies the requested processes if special processing is required.	7.6.2.6	**A**	See above.	0	
7.6.3 Access - Deliver Response			**A**			
7.6.3.1	**Web-accessibility** - Demonstrate the display interface is web-accessible.	7.6.3.1	**A**	Checked using Fangs and the Accessibility Checker Add-on. It uses tables for layout purposes, but was unable to validate the HTML output.	1	
7.6.3.2	**Downloading** - Demonstrate export function that provides XML output for batch downloads	7.6.3.2	**A**	Must be done through externally scripting	0	
7.6.3.3	**Saving content** - Demonstrate users are allowed to save digital content to a hard-drive, e-mail, and/or save search results.	7.6.3.3	**A**	Documents may be downloaded. There does not appear to be a function for emailing or saving search results.	1	
7.6.3.5	**System notification** - Demonstrate a confirmation message is returned to the Coordinate Access Activities section after response has been sent.	7.6.3.5	**A**	This aspect of OAIS is not currently modeled by DSpace.	0	
7.6.3.6	**Audit trail** - Demonstrate an audit trail of all actions is created and stored.	7.6.3.6	**A**	Info contained in log file but not easily usable.	0	
7.6.3.4	**Response request** - Demonstrate a response request is received from Coordinate Access Activities	7.6.3.4	**A**	Demonstrated retrieval of objects via the UI without issue.	2	
8.1 Metadata Requirements			**M**			
8.1.1	**Metadata formats** - Demonstrate that the system can accept metadata associated with objects in at least the following formats: All NLM DTDs, Dublin Core, MARC21, MARCXML, ONIX, MODS, EAD, TEI.	8.1.1	**M/T**	Does accept it but minimally	1 (M & T)	
8.1.2	**Metadata checks** - Demonstrate the built-in checks on the incoming metadata. Records not containing the minimally defined set of fields should be flagged as problems, either to be returned to the submitter, or sent locally for metadata enhancement.	8.1.2	**M**	Batch - 0; manual - 1	0-1	
8.1.5	**Metadata updates** - Demonstrate the ability to allow for metadata updates.	8.1.5	**M**	Rather clunky	2	

8.1.6a	**Metadata search and display** - Demonstrate the ability to search and display metadata (use of external tool possible).	8.1.6a	**M**			
8.1.8	**PREMIS** - Demonstrate standards compliance for PREMIS (use of external tool possible).	8.1.8	**M/T**	2	0 (M & T)	
8.1.9	**METS** - Demonstrate standards compliance for METS (use of external tool possible).	8.1.9	**M/T**	Only exports collections/files in METS (via command line, not the web interface) and is working on METS import capability.	1 (M & T)	
App A	**Descriptive metadata** - Demonstrate that the minimum descriptive metadata requirements described in Appendix A are accepted.	App A	**M**			
9.1 Additional Technical Infrastructure Requirements			**T**			
9.1.1	**OAI-PMH** - Demonstrate that the system can respond to OAI-PMH requests as a data provider.	9.1.1	**T**	DSpace can be a provider but does not itself harvest and incorporate other data. Key functions: identify (handshake), listsets (list of collections), listidentifiers (list of IDs), listmetadataformat (list of metadata formats), listrecords (metadata only, no bit stream), etc. 2	2	
9.1.2	**Z39.50** - By design analysis, confirm that the system can respond to data requests using the Z39.50 standard.	9.1.2	**T**			
9.1.3	**SRU/SRW** - By design analysis, confirm that the system can respond to data requests using the SRU and SRW data access standards.	9.1.3	**T**	0		
9.1.4	**SOAP** - Demonstrate that the system can respond to web service requests using SOAP.	9.1.4	**T**	0		
9.1.5	**UNICODE** - Demonstrate that the system supports UNICODE.	9.1.5	**T**	0		
9.1.6	**OpenURL** - By design analysis, confirm that the system is compliant with OpenURL.	9.1.6	**T**	3		
9.1.7	**Z39.87** - By design analysis, confirm that the system supports the Z39.87 image metadata standard.	9.1.7	**T**	0		
				0		
Notes:	1. **Subgroups:** A=Access, M=Metadata, P=Preservation, T=Technical Infrastructure					
	2. **Score** indicates the extent to which the test element could be demonstrated: 0=None, 1=Low, 2=Moderate, 3=High					
3.	**Preservation tests** - These sections of the functional requirements are covered by Test Plan sections P1, P2, and P3, which were defined by the Preservation subgroup to facilitate testing.					
	4. Test elements having **blue background** are the subject of outstanding questions from the Access subgroup.					
10. Additional Observations						
10.1	Back button on browser cannot function as a navigation tool all the time, and there is no other navigation tool. (KK)					
10.2	Files are just listed in the order they were ingested and cannot be sorted by item (KK).					

10.3	Sub-community hierarchies get lost in the collection selection window during submission (FK).				
10.4	Batch ingest does allow customized licenses to be included for different items (EL).				
10.5	Internal errors were generated in the following instances: 1. When more than one user to edit a record simultaneously. The record does not appear to be locked and DSpace generates internal system errors (JM, DB). 2. While trying to edit policies for a collection (FK)				
10.6	DSpace creates two XML files for item(s) containing both DC and NLM/DC metadata (such as permanence) during the export process. The dublincore.xml for all DC elements and metadata_nlm.xml for all NLM/DC elements. (EL)				
10.7	DSpace tracks every action as an entry in a text-based log file but the log file doesn't reveal the specific action taken. The History file, on the other hand, records more specific details than the log file but some of the entries don't seem related to the actual action. For example, adding a subject and modifying an item type are recorded as updated bit streams and updated bundles in the history file. No tools are provided to access the history file. We should check with the DSpace community to see if there are any available tools already. (EL)				

Appendix D – DigiTool Testing Results

Consolidated Digital Repository Test Plan Last updated: June 13, 2008		Source Requirements	Sub-group See Note 1	DigiTool 3.0 Tests		
Test ID	**Test Plan Element**			**Test Procedure and Results**	**Score (0-3) Note 2**	**Notes**
7.1.1 Ingest - Receive Submission			**T**			
7.1.1.7	**File types** - Demonstrate that the system can ingest content in all the file formats listed as "supported" in Appendix B of the NLM DR Functional Requirements document (plus MP3 and JPEG2000), specifically: MARC, PDF, Postscript, AIFF, MPEG audio, WAV, MP3, GIF, JPEG, JPEG2000, PNG, TIFF, HTML, text, RTF, XML, MPEG. Demonstrate that the system can ingest the following types of content: articles, journals, images, monographs, audio files, video files, websites, numeric data, text files, and databases. Conduct this test element by ingesting the set of files listed in the Test File spreadsheet. (The files listed in this spreadsheet contain examples of all the file formats, and all the content types identified above.)	7.1.1.7 7.1.1.9	**T**	Can automatically create thumbnails when ingesting JPG2000, PDF. (Some initial PDF ingest tests failed to produce thumbnails due to unusual PDF format.)	3	Answer was received to Question QT4.
7.1.1.1	**Manual review** - Demonstrate that the system has the capability to require that submitted content be manually reviewed before it is accepted into the repository. Demonstrate that the system maintains submitted content in a staging area before it is accepted. Demonstrate that the system notifies a reviewer when new content is ready for review. (Also see tests for 7.1.4.1, 7.1.4.2, and 8.1.2.)	7.1.1.1	**T**	3		
7.1.1.2	**Review and acceptance workflow** - Demonstrate that the system supports a workflow for the review and acceptance of submitted content. Demonstrate that the workflow includes the following functions: a - Receive and track content from producers; b - Validate content based on submitter, expected format, file quality, duplication, and completeness; c - Normalize content by converting content into a supported format for final ingestion into the repository; d - Human review of content; e - Acceptance or rejection of content or file format.	7.1.1.2, 7.1.1.10	**T**	a - yes b - no c - no d - yes e - yes	2	
7.1.1.3	**Reason for rejection** - Demonstrate that the system records a set of identifying information or metadata that describes the reason for the rejection of submitted content. Demonstrate two cases: (1) automatic rejection, and (2) rejection by a human reviewer.	7.1.1.3	**T**	1 - no 2 - maybe - needs further testing	1	
7.1.1.4	**Rejection filter** - Demonstrate that the system allows the creation of a filter that can be used to automatically reject submitted content. (This capability will eliminate the need for manual review of some submissions and resubmissions.)	7.1.1.4	**T**	This is not filter based but rather template based	1	

7.1.1.5	**Rejection notification** - Demonstrate that the system can notify the producer or donor when submitted content is rejected. Demonstrate two cases: (1) notification after immediate rejection by an automated process, and (2) notification after rejection by manual review.	7.1.1.5, 7.1.1.11	**T**	Failure indication (instead of "success") upon immediate rejection.	1	
(7.1.1.8)	**Metadata types** - Demonstrate that the system can ingest content with associated metadata in the following formats: all NLM DTDs, Dublin Core, MARC21, MARCXML, ONIX, MODS, EAD, TEI, PREMIS, METS. (NOTE: This test is covered by tests 8.1.1, 8.1.8, and 8.1.9)	7.1.1.8, 8.1.1, 8.1.8, 8.1.9	**M/T**		T=2.5 M=2.5	
7.1.1.10	**Format conversion** - Demonstrate that the system has the capability to convert the format of a file being ingested to a desired supported format. As a test case, demonstrate that a WAV file can be converted to MP3 format when it is ingested. (An external tool may be needed to perform the conversion. If this is the case, demonstrate that the system can invoke the required external tool.)	7.1.1.10, 7.1.1.2	**T**	Can automatically create JPG and JP2 when ingesting TIFF. Can automatically create JPG when ingesting JP2. Can automatically create JPG thumbnail when ingesting JP2 and PDF. Can add other external file converters.	2	Answer was received to Question QT1
7.1.1.12	**Resubmission** - Demonstrate that the system can ingest a SIP that is resubmitted after an error in the SIP was detected and corrected. Demonstrate two cases: the resubmission can occur after an error was detected in (1) the content of the SIP, and (2) the metadata of the SIP.	7.1.1.12	**T**	Failed ingests can be rolled back, edited, and reingested.	2	
7.1.1.14	**Versions** - Demonstrate that the system can store, track, and link multiple versions of a file.	7.1.1.14	**T**	Alternate manifestations can be created but there are no "Versions"	0	
7.1.1.15 a	**Unique identifiers** - Demonstrate that the system assigns a unique identifier to each object ingested. Demonstrate two cases: (1) a unique identifier assigned to a digital object, which may be comprised of a set of component files, and (2) a unique identifier assigned to each of the component files of a digital object.	7.1.1.15a, 7.1.1.15b	**T**	3		
7.1.1.15 b	**Relationships** - Demonstrate that the system can represent a parent-child relationship between content items. Demonstrate two cases: (1) an object having multiple components (e.g., a document having multiple pages, each in a separate file), and (2) an object having multiple manifestations (e.g., an image having both TIFF and JPEG files).	7.1.1.15b	**T**	3		
7.1.1.16	**Audit trail** - Demonstrate that the system maintains an audit trail of all actions regarding receiving submissions (SIPs).	7.1.1.16	**T**			
7.1.2 Ingest - Quality Assurance			**T**			
7.1.2.1	**Virus checking** - By design analysis, confirm that the system performs automatic virus checking on submitted content files.	7.1.2.1	**T**			
7.1.2.2	**Transmission errors** - Demonstrate that the system uses MD5, CRC, checksums, or some other bit error detection technique to validate that each data file submitted is received into the repository staging area without transmission errors.	7.1.2.2	**T**	MD5 is created during ingest and is saved with the file. However, an MD5 generated pre-ingest cannot be compared with the DigiTool-created MD5 to verify that transmission errors have not occurred. 0	1	Answer was received to Question QT2.

7.1.2.3	**Submission validation** - Demonstrate that the system verifies the validity of submitted content based on the following criteria: submitter; expected file format; file quality (e.g., actual format of file matches the filename extension, and content of file is well-formed); duplication (e.g., existence of object in the repository); completeness of metadata; completeness of file set (e.g., all expected files are included in the submission).	7.1.2.3	**T**	No submission validation other than JHOVE checksum. Checksum done at ingest and no capability to compare externally provided checksum with that done during ingest.	1	
7.1.2.4	**QA UI** - Demonstrate that the system allows NLM staff to perform manual/visual quality assurance on staged SIPs via a user-friendly interface.	7.1.2.4	**T**			
7.1.2.5	**Reaction to QA errors** - Demonstrate that the system can react to specified QA errors in two ways: (1) request that the producer correct and resubmit the content, or (2) automatically modify the submission (e.g., converting to a supported format).	7.1.2.5	**T**	After failed ingest user can rollback, edit, and resubmit. No automatic modifications performed. 1	1	
7.1.2.6	**File/batch accept/reject** - Demonstrate that the system enables NLM staff to accept or reject submitted content (SIPs) at the file or batch level.	7.1.2.6	**T**			
7.1.2.7b	**Error reports** - Demonstrate that the system generates error reports for ingest quality assurance problems.	7.1.2.7b	**T**	1.5		
7.1.2.8	**Adjustable level of manual QC** - By design analysis, confirm that the system has the ability to adjust the level of manual ingest quality control needed, based on the origin of the file.	7.1.2.8	**T**	0 0		
7.1.2.9	**Audit trail** - Demonstrate that the system maintains an audit trail of all actions regarding ingest quality assurance.	7.1.2.9	**T**			
7.1.4 Ingest - Generate Descriptive Information / Metadata			**M**			
7.1.4.1	**Additional metadata** - Demonstrate the entry of additional metadata (e.g. subject headings, names, dates, “curatorial” descriptive metadata - evaluative information that explains why an object is important, whether it was part of a larger collection (e.g., an exhibit), etc.).	7.1.4.1	**M**	3		
7.1.4.2	**Validate metadata** - Demonstrate ability to validate specified metadata elements.	7.1.4.2	**M**	1.5		
7.1.4.4	**Metadata storage** - Demonstrate that metadata is stored in the database in a manner that conforms to repository reformatting and linked to their corresponding objects via an identifier. o Demonstrates that basic descriptive metadata is also stored with the objects (e.g., unique identifier, title and date stored in the TIFF header) so that the objects can still be identified in the event that information in the database is corrupted. o See Appendix D for examples of TIFF header metadata requirements. (Use of external tool probable)	7.1.4.4	**M**	3		
7.1.4.5	**Required descriptive elements** - Demonstrate the ability to recognize required descriptive elements.	7.1.4.5	**M**			
7.1.4.7	**Audit trail** - Demonstrate the creation of an audit trail of all actions.	7.1.4.7	**M**	3		
7.1.3 Ingest - Generate AIP		Note 3	**P**			
7.1.5 Ingest - Coordinate Updates		Note 3	**P**			
7.2.1 Archival Storage - Receive Data		Note 3	**P**			
7.2.2 Archival Storage - Manage Storage Hierarchy		Note 3	**P**			
7.2.3 Archival Storage - Replace Media		Note 3	**P**			

7.2.4 Archival Storage - Error Checking and Disaster Recovery		Note 3	**P**			
7.2.5 Archival Storage - Provide Data		Note 3	**P**			
7.3.1 Data Management - Administer Database		Note 3	**P**			
7.3.2 Data Management - Administer Perform Queries		Note 3	**P**			
7.3.3 Data Management - Generate Report		Note 3	**P**			
7.3.4 Data Management - Receive Database Updates		Note 3	**P**			
7.4 Administration		Note 3	**P**			
P1 - Generate AIP			**P**			
P1-1	**Generate AIP** - Demonstrate the generation of AIPs from ingested SIPs that do not need normalization.	7.1.3.1, 7.1.3.2, 7.1.3.3, 7.4.1	**P**	Can generate an XML-based digital entity which contains points to link all objects and metadata but not physical AIP package.	2	
P1-2	**Generate AIP with normalization** - Demonstrate the generation of AIPs from ingested SIPs that need normalization - Transform an unsupported format to an accepted format (See Appendix B).	7.1.3.1, 7.1.3.2, 7.1.3.3, 7.4.1	**P**	No normalization.	0	
P1-3	**Derivative files** - Demonstrate the generation of AIPs that consist of master files and derivatives.	7.1.3.6	**P**	Can convert a file from TIFF to JP2 or from TIFF/JP2 to JPEG. Can also generate JP2 thumbnail.	1.5	
P1-4	**Master files** - Demonstrate the generation of AIPs that consist of master files only.	7.1.3.6	**P**	**QP1**: How does DigiTool manage manifestation relationships and identify which is the master that may or may not be in TIFF? **Opher**: Masters are handled in two contexts: (1) a preservation context, where the preservation_level field of the digital entity can be set to a value designated for masters (typically "Preservation Master" or "High") and used to differentiate between storage rules; (2) an application context, where the usage_type field of the digital entity (e. g. "main", "archive") can determine if the object would be delivered.	2	
P1-5	**Store AIP in archival storage** - Demonstrate the ability to transfer AIPs to Archive Storage.	7.1.5.1, 7.2.1.1, 7.2.1.2	**P**	See P1-1	2	
P1-6	**Store metadata in DB** - Demonstrate the ability to generate and transfer Descriptive Information (metadata) to Data Management Database.	7.1.5.2, 7.3.4.1	**P**	Generate (extract) and transfer.	3	
P1-7	**Link metadata and objects** - Demonstrate the ability to store identification information in the Data Management database and link digital objects in the Archive Storage.	7.1.5.4	**P**	Yes	3	

P1-8	**Send confirmation** - Demonstrate the ability to automatically send confirmation to ingest and/or receiver when AIP and metadata transfers are completed.	7.1.5.1, 7.1.5.3, 7.2.1.3, 7.3.3.3	**P**	View in Success/Failed log on screen only. **QP2**: Can DigiTool send confirmation in an email to receivers? **Opher**: view logs and folder status of ingests at all phases through the web module, but emails are not sent.	1	
P1-9	**Send statistical reports** - Demonstrate the ability to automatically send statistical reports to ingest and/or receivers when AIP and metadata transfers are completed.	7.1.5.1, 7.1.5.3, 7.2.1.3, 7.3.3.3	**P**	Has to click on each ingest ID to view details in the Success/Failed log. **QP3**: Can DigiTool send statistical reports (or along with the confirmation) in an email to receivers? **Opher**: view logs and folder status of ingests at all phases through the web module, but emails are not sent.	1	
P1-10	**Send error reports** - Demonstrate the ability to automatically send error reports to ingest and/or receivers when AIP and/or metadata transfers fail.	7.1.5.1, 7.1.5.3, 7.2.1.3, 7.3.3.3	**P**	Only indicated in the Failed log. **QP4**: Why is there a rollback icon for a failed ingest in the Failed log? Does a failed one get ingested anyway? **Opher**: Yes – in certain cases ingests partially fail (e. g. not all tasks could be 100% completed) but digital entities are created, and the rollback allows for the staff to analyze the problem, correct the cause for the partial failures, and re-ingest.	1	
P2 - Administer Archival Storage & Database			**P**			
P2-1	**Monitor transfer integrity** - Demonstrate the built-in function to automatically monitor and report if any AIPs and metadata are altered or corrupted during data transfer and media change (refresh or replace).	7.2.2.1, 7.2.3.2, 7.2.4.1	**P**	**QP5 (**Same as QT2 from Ed): Does DigiTool compare the checksum that is generated before the ingest with the one that is generated after the ingest? **Opher**: DigiTool creates a file-level checksum during ingest and can check the validity of this checksum of repository items as an ongoing post-ingest maintenance procedure. This procedure only supports checksums generated during the ingest, not prior to it.	2	
P2-2	**Check data/referential integrity** - Demonstrate the built-in function to perform routine and special referential and data integrity checks (CRC or checksums) on files in the Archive Storage and Data Management Database.	7.2.4.2, 7.3.1.1, 7.3.1.2, 7.4.4	**P**	No referential integrity check.	1	
P2-3	**Routine configuration for data/referential integrity** - Demonstrate the ability to allow for routine configuration.	7.2.4.2, 7.3.1.1, 7.3.1.2, 7.4.4	**P**	No referential integrity check.	1	

P2-4	**Disaster recovery** - Demonstrate the ability to allow for disaster recovery including data backup, off-site data storage, and data recovery.	7.2.4.3	**P**	**QP6**: Can we reingest exported files for data recovery to recreate the repository? **Opher**: Yes - the entire repository (or parts of it) can be exported as digital entities (with respective file streams) to be re-ingested.	2	
P2-5	**User views** - Demonstrate the ability to allow for customized user views of the contents of the storage (create, maintain, and access).	7.3.1.4	**P**	Not with external tool either.	0	
P2-6	**System CM** - Demonstrate the ability to allow for configuration management of the system hardware and software.	7.4.2	**P**	Limited/restricted local control.	1.5	
P2-7	**Database CM** - Demonstrate the ability to allow for configuration management of the Data Management Database such as table, schema definitions, etc.	7.3.1.3	**P**	No database table or schema changes.	0	
P2-8	**Delete AIPs** - Demonstrate the ability to allow the authorized staff to delete AIPs from the repository including: removing the digital object's files and retaining associated metadata, or removing both the files and metadata.	7.4.3.4	**P**	Yes with rollback function.	3	
P2-9	**Coordinate AIP removal** - Demonstrate the ability to generate an alert and coordinate the removal of an AIP with maintenance of metadata held in other systems.	7.4.3.5	**P**	No alert.	0	
P2-10	**File migrations** - Demonstrate the ability to allow the authorized staff to schedule and perform file migrations or migration on request for batched and individual files by authorized staff.	7.4.3.6	**P**	**QP7**: How to schedule and perform file migration and migration on request for batched or individual files? **Opher**: Filestreams for individual objects can be exported, imported, deleted or replaced using the Meditor. Batch migrations cannot be performed.	0	
P2-11	**Request DIPs for update** - Demonstrate the ability to allow the authorized staff to request DIPs for file migrations and data updates.	7.3.4.1, 7.3.4.2, 7.3.4.3, 7.4.3.1, 7.4.3.2, 7.4.6.2	**P**	**QP8**: How to request DIPs for file migrations and data updates? **Opher**: This can be done on an individual basis only. Objects can be "pulled" from the repository into the Meditor in various ways for filestream maintenance.	1	
P2-12	**Re-ingest updated DIPs** - Demonstrate the ability to allow the authorized staff to reingest updated DIPs as SIPs.	7.4.3.3	**P**	No duplication check or overwrite option.	1	
P2-13	**Support query requests** - Demonstrate the ability to receive, retrieve, display, and deliver data for query requests from other functions such as Ingest, Access, and Administration.	7.3.2.1, 7.3.2.3, 7.4.6.1	**P**	Yes.	2	
P2-14	**Query requests from different storage locations** - Demonstrate the ability to handle query requests with required data to be sourced from different storage locations.	7.3.2.2	**P**	NFS or URL retrieval.	2.5	
P2-15	**Queries against all metadata** - Demonstrate the ability to run data queries against all metadata used to manage the repository.	7.3.2.4	**P**	Yes	2.5	

P2-16	**Audit trial -** Demonstrate the creation of an audit trail of all actions including who, when, how, what and where for Archive Storage and Data Management Database.	7.1.3.4, 7.1.5.6, 7.2.1.4, 7.2.2.3, 7.2.5.2, 7.3.2.5, 7.3.3.7, 7.3.4.6, 7.4.3.7, 7.4.6.4	**P**	**QP9**: Does DigiTool keep an audit trail for all actions including who, when, how, how and where for the archive storage and database? **Opher**: Such preservation-oriented needs are better addressed by our Preservation system. In DigiTool, the History metadata provides a partial audit trail.	1	
P2-17	**Generate reports** - Demonstrate the ability to receive, generate, display, and deliver management information reports and statistics such as summaries of repository holdings by category, summaries of updates by category, user codes, etc., usage statistics for access to repository holdings, and descriptive information for a specific AIP.	7.3.3.1, 7.3.3.2, 7.3.3.5, 7.3.4.4	**P**	**QP10**: Which specific reports and statistics can be generated? **Opher**: Currently: Repository DE and stream count, Depositor Statistics Reports, Digital Entities Viewing Reports. We are also planning on implementing the BIRT reporting system in DigiTool later in 2008 – this will allow for more extensive reporting.	1	
P2-18	**Schedule reports** - Demonstrate the ability to generate reports in an ad-hoc manner, automatically or to be triggered by a calendar or by a specific system event.	7.3.3.4	**P**	**QP11**: Need clarification on how available reports/statistics can be generated in an ad-hoc manner, automatically or to be triggered by a calendar or by a specific system event. **Opher**: These can be run ad-hoc (immediately or postponed), and can be scheduled, by calendar, for ongoing running using cron.	2	
P2-19	**Time period for reports** - Demonstrate the ability to allow the user to specify a time period or set of time periods for reports and statistics.	7.3.3.6	**P**	**QP12**: Need clarification on how a user can specify a time period or set of time periods for reports and statistics. **Opher**: DE viewing reports can be filtered by date. With the implementations of BIRT this will be expanded.	1	
P3 - Generate DIP			**P**			
P3-1	**Generate DIP for access requests** - Demonstrate the generation of DIPs by putting AIPs and Descriptive Information back together for access requests.	7.1.5.5, 7.2.5.1, 7.4.6.2	**P**	Yes	2	
P3-2	**Generate DIP for object maintenance** - Demonstrate the generation of DIPs by putting AIPs and Descriptive Information back together for content/metadata update, versions upgrades and format migration by authorized staff.	7.4.6.2, 7.4.3	**P**	Pending for answers to **QP6, QP7** and **QP8**.	2	
7.4.1 Administration - Negotiate Submission Agreement			**T**			
7.4.1.1	**Manage submission agreements** - Demonstrate that the system manages information regarding submission agreements: that it tracks negotiation status and written submission agreements, and that it maintains schedules.	7.4.1.1	**T**		0	

7.4.1.2	**Edit submission agreements** - Demonstrate that the system allows submission agreements to be edited, based on the access level of the user.	7.4.1.2	**T**			
7.4.1.5	**Terms of submission agreements** - Demonstrate that the system stores the terms of submission agreements, and uses the terms to monitor, review, and process submissions.	7.4.1.5	**T**	0 0		
7.4.1.6	**Audit trail** - Demonstrate that the system maintains an audit trail of all actions related to submission agreements.	7.4.1.6	**T**			
7.4.2 Administration - Manage System Configuration			**T**			
7.4.2.1	**Monitor repository functionality** - Demonstrate that the system monitors the functionality of the entire repository.	7.4.2.1	**T**			
7.4.2.2	**System configuration** - By design analysis, confirm that the system maintains the integrity of the system configuration.	7.4.2.2	**T**	0		
7.4.2.3	**Audits operations** - Demonstrate that the system audits system operations, performance, and usage.	7.4.2.3	**T**	0 0		BIRT reporting system coming late 2008
7.4.2.4	**Data management information** - Demonstrate that the system collects and can display system information concerning Data Management.	7.4.2.4	**T**			
7.4.2.5	**Operational statistics** - Demonstrate that the system collects and can display operational statistics concerning Archival Storage.	7.4.2.5	**T**	Must use SQL reporting 0	0	
7.4.3 Administration - Archival Information Update						
7.4.5 Administration - Audit Submission			**T**			
7.4.5.1	**Audits** - Demonstrate that the system can support an audit procedure to verify that submissions (SIP or AIP) meet specified requirements of the repository. The audit method may be based on sampling, periodic review, or peer review. [See NLM DRD Functional Requirements document, section 7.4.5 for description of audit requirements.] (Also partially covered by 7.2.4.2)	7.4.5.1	**T**	Ex Libris sees audit procedures as a preservation concern, and will be provided in the new Preservation tool (DPS).	0	Answer was received to Question QT3.
7.4.5.2	**Metadata audit** - Demonstrate that the system can audit metadata as part of the audit procedure.	7.4.5.2	**T**			
7.4.5.3	**Audit rejection -** Demonstrate that the system can reject components of audited information packages, based on specified audit requirements.	7.4.5.3	**T**	0		
7.4.5.4	**Audit report** - Demonstrate that the system can generate an audit report, based on the results of periodic audits of SIPs and AIPs.	7.4.5.4	**T**	0		
7.4.5.5	**Audit trail** - Demonstrate that the system maintains an audit trail of all actions regarding the auditing of SIPs and AIPs.	7.4.5.5	**T**	0		
7.4.6 Administration - Activate Requests			**P**			
7.6.1 Access - Coordinate Access Activities - User Access			**A**			
7.6.1.1	**Manage user permissions** - Demonstrate the access controls for multiple permission levels and user privileges.	7.6.1.1	**A**	Good, but not robust. Staff Privileges configuration governs a fairly granular list of rights for Staff Users, Admin Users. Automated patron registration and authentication possible using LDAP.	2	
7.6.1.2	**Manage user restrictions** - Demonstrate multiple levels of access restrictions for NIH employees and general public based on licensing terms, embargo periods, IP range restrictions, workstation access, and other possible legal restrictions.	7.6.1.2, 7.6.1.3	**A**	Embargo period - Kaplan explained they hoped to do more with this feature.	2	

7.6.1.4	**Manage user settings** - Demonstrate access settings allow staff to add or edit descriptive metadata	7.6.1.4	**A**	Meditor is used to add or edit metadata.	2	
7.6.1.7	**Audit users** - Demonstrate access mechanisms can identify individual users and maintain audit log of user actions.	7.6.1.7	**A**	Audit trails were detailed and human-readable	3	
7.6.1.5	**Perform maintenance tasks** - Demonstrate maintenance access including adding new files, manipulating images, editing metadata, performing format conversions/migrations, and troubleshooting system problems.	7.6.1.5	**A**	Some of these functions are also addressed below. File and metadata access is provided via web interface and Meditor client.	2	
7.6.1.6	**Manage system rights** - Demonstrate ultimate system rights access for NLM system administrators and programmers.	7.6.1.6	**A**	**Question for technical infrastructure?** Too ambiguous for Access group to answer.	?	
7.6.1 Access - Coordinate Access Activities - Rights/Data Control of Objects			**A**			
7.6.1.8	**Manage access rights** - Demonstrate access rights and conditions to materials and storage directories provide for a combinational of create/write; edit; read; delete privileges.	7.6.1.8	**A**	How are storage directories integrated into DigiTool? If deleted in DigiTool does it delete from servers?	3	
7.6.1.9	**Manage metadata rights** - Demonstrate access rights may be associated with the metadata relating to an individual object	7.6.1.9	**A**	Embargo information is part of metadata - how is this done?	2	
7.6.1.13	**Manage relationships** - Demonstrate access rights and conditions can be inherited from a parent object to any child object.	7.6.1.13	**A**	Manifestations and complex objects can share descriptive metadata and/or usage rights.	2	
7.6.1.14	**Manage relationships** - Demonstrate access rights and conditions can be assigned to an object on an individual or group basis at same time.	7.6.1.14	**A**	Batch metadata changes can be performed via management jobs (but _not_ via Object Manager).	2	
7.6.1.16	**Automated retrieval** - Demonstrate objects in the repository are accessible for data mining or automated retrieval.	7.6.1.16	**A**	Open URL compliant, OAI-PMH for metadata retrieval. Resource Discovery has full-text searching, but no built-in data mining services.	1	
7.6.1.17	**Metadata access** - Demonstrate access to deleted and retracted metadata is retained.	7.6.1.17	**A**	Once it's gone, it's gone.	0	
7.6.1.18	**Metadata harvesting** - Demonstrate metadata harvesting following the OAI-PMH guidelines.	7.6.1.18	**A**	DigiTool will allow external hosts to harvest its metadata via OAI-PMH. It does not do harvesting (bring in metatdata) using this protocol but can do so via Z39.50.	2	
7.6.1.10	**Access rights** - Demonstrate access rights and conditions of use are applied to each digital object and its related metadata and are machine readable and actionable.	7.6.1.10, 7.6.1.11	**A**	Rights and use conditions can be recorded into controlled metadata fields such as Usage Type, which can restrict harvesting into Silos.	2	
7.6.1.12	**Access conditions** - Demonstrate access conditions are specific to a digital object.	7.6.1.12	**A**	Viewable in Object Manager	3	
7.6.1.15	**Free/Restricted access** - Demonstrate free (items available via internal/external delivery mechanisms) and restricted access (access permission must be satisfy various criteria) status for objects, files, metadata, etc.	7.6.1.15	**A**	Viewable in Object Manager, but embargo is not sophisticated	2	
7.6.1 Access - Coordinate Access Activities - Search and Retrieval			**A**			

7.6.1.19	**508 compliance** - Demonstrate the search interface is web-accessible and Section 508 compliant.	7.6.1.19	**A**	Good faith effort well-documented by ExLibris. Meditor is likely to be weak in this area, but it's on the way out.	2	
7.6.1.20	**Search features** - Demonstrate search includes: metadata, full-text, standard boolean, proximity, "more like" this"	7.6.1.20, 7.6.1.21, 7.6.1.22, 7.6.1.23, 7.6.1.24	**A**	No proximity and no "More like this", but Object Manager has a controlled metadata search.	2	
7.6.1.25	**Search results display** - Demonstrate search results display includes date sort; relevancy ranking; alpha by author or source.	7.6.1.25	**A**	No date ranking, but it is promised in an upcoming service pack.	2	
7.6.1.26	**Relevancy ranking** - Demonstrate whether relevancy ranking can be manipulated via system as well as user defined settings.	7.6.1.26	**A**	Unclear how relevancy is determined; cannot be modified.	1	
7.6.1.29	**Federated search** - Demonstrate federated searching of different repository sites.	7.6.1.29	**A**	Resource Discovery searches across admin units by harvesting content into silos.	2	
7.6.1.30	**Advanced search** - Demonstrate advanced search includes search history; saved searches; saved citation lists/bibliographies; alerts; various functions and formats; dynamic selection of delivery media without recreating search query.	7.6.1.30	**A**	Search can be refined by adding additional values (using AND, OR or WITHOUT). Search history available during session. Can save results to "my space" (one object at a time).	1	
7.6.1.31	**Display formats** - Demonstrate a variety of standard display formats are provided and whether they are customizable by user	7.6.1.31	**A**	A limited number of Resource Discovery default Preferences can be set by user.	1	
7.6.1.32	**Alternate search interfaces** - Demonstrate availability of alternate search interfaces for mechanisms such as handhelds and PDAs.	7.6.1.32	**A**			
7.6.1.33	**Object access** - Demonstrate access to the appropriate copy of the identified item (text, image, video, etc.)	7.6.1.33	**A**	Metadata can indicate the "use" 0 manifestation of an object.	1	
7.6.1.34	**Library holdings** - Demonstrate integration of search results with library holdings.	7.6.1.34	**A**	Can be done using Primo, but not internal to DigiTool.	0	
7.6.1.35	**Response time** - Demonstrate acceptable response time.	7.6.1.35	**A**	Response time in NLM development environment can be very slow, although ExLibris demos generally showed acceptable response	1	
7.6.1.36	**External search engines** - Demonstrate searching by outside search engines such as usa.gov, Google, and Yahoo.	7.6.1.36	**A**	ExLibris is in the process of making search-engine friendly site maps - will be introduced in a service pack. (Commercial search engines do not want to spider via OAI-PMH.)	1	
7.6.1.37	**External system access** - Demonstrate external access to other repositories or systems performing web harvesting functions.	7.6.1.37	**A**	via OAI-PMH and Z39.50	2	
7.6.1.38	**Language support** - Demonstrate how multiple languages and non-Roman scripts are supported in search, retrieval and display.	7.6.1.38	**A**	Resource Discovery has multiple languages, Unicode support	2	

7.6.1.39	**Versioning** - Demonstrate access to all versions of digital objects in the repository is provided.	7.6.1.39	**A**	All objects / manifestations can be provided, with metadata distinguishing between manifestations.	2	
7.6.1.40	**Search settings** - Demonstrate system settings and user-defined settings in the search functions are provided.	7.6.1.40	**A**	Only default system-provided search settings are offered, through regular and advanced search interface.	0	
7.6.2 Access - Generate DIP			**A**			
7.6.2.1	**Integrate holdings** - Demonstrate integration of search results with library holdings.	7.6.2.1	**A**	Could be done via Primo, but not internal to DigiTool	0	
7.6.2.2	**Retrieval and notification** - Demonstrate the generation function accepts a dissemination request, retrieves AIP from archival storage and moves a copy of the data to a staging area for further processing, and creates and sends a report request to data management to obtain appropriate metadata.	7.6.2.2, 7.6.2.3, 7.6.2.4	**A**	AIP and DIP are conceptual in DigiTool, but system provides the bitstream through a pre-determined viewer, and metadata is also provided.	2	
7.6.2.7	**Audit trail** - Demonstrate an audit trail of all actions is created and stored.	7.6.2.7	**A**	Usage reporting is available to administrators.	1	
7.6.2.5	**Response and delivery** - Demonstrate that the prepared DIP response is placed in the staging area and a message is generated and sent to Coordinate Access Activities that the DIP is ready for delivery.	7.6.2.5	**A**	Items are already 'harvested' into the silo for public access - this is probably comparable to staging/delivery.	2	
7.6.2.6	**Storage retrieval** - Demonstrate that Generate function accesses data objects in staging storage and applies the requested processes if special processing is required.	7.6.2.6	**A**	As above.	2	
7.6.3 Access - Deliver Response			**A**			
7.6.3.1	**Web-accessibility** - Demonstrate the display interface is web-accessible.	7.6.3.1	**A**	Resource Discovery is completely web-based. Some object viewers are served through the browser, some objects rely on local PC software.	3	
7.6.3.2	**Downloading** - Demonstrate export function that provides XML output for batch downloads	7.6.3.2	**A**	no batch downloading of objects available to end user, but comprehensive exporting available from management side.	1	
7.6.3.3	**Saving content** - Demonstrate users are allowed to save digital content to a hard-drive, e-mail, and/or save search results.	7.6.3.3	**A**	Users can save / email, add to e-shelf, SFX / Primo integration possible.	3	
7.6.3.5	**System notification** - Demonstrate a confirmation message is returned to the Coordinate Access Activities section after response has been sent.	7.6.3.5	**A**			
7.6.3.6	**Audit trail** - Demonstrate an audit trail of all actions is created and stored.	7.6.3.6	**A**	usage reporting is available to 0 administrators.	1	
7.6.3.4	**Response request** - Demonstrate a response request is received from Coordinate Access Activities	7.6.3.4	**A**	if patron is authenticated, DigiTool can evaluate patron's rights before delivering content. Otherwise, Resource Discovery delivers according to anonymous rights.	1	
8.1 Metadata Requirements			**M**			

8.1.1	**Metadata formats** - Demonstrate that the system can accept metadata associated with objects in at least the following formats: All NLM DTDs, Dublin Core, MARC21, MARCXML, ONIX, MODS, EAD, TEI.	8.1.1	**M/T**	Mapping to DC only	T=2 M=2	TEI/EAD not that great
8.1.2	**Metadata checks** - Demonstrate the built-in checks on the incoming metadata. Records not containing the minimally defined set of fields should be flagged as problems, either to be returned to the submitter, or sent locally for metadata enhancement.	8.1.2	**M**	Batch=1; Manual = 2	1-2	
8.1.5	**Metadata updates** - Demonstrate the ability to allow for metadata updates.	8.1.5	**M**			
8.1.6a	**Metadata search and display** - Demonstrate the ability to search and display metadata (use of external tool possible).	8.1.6a	**M**	3		
8.1.8	**PREMIS** - Demonstrate standards compliance for PREMIS (use of external tool possible).	8.1.8	**M/T**		1.5	T=0 M=1
8.1.9	**METS** - Demonstrate standards compliance for METS (use of external tool possible).	8.1.9	**M/T**			T=3 M=2
App A	**Descriptive metadata** - Demonstrate that the minimum descriptive metadata requirements described in Appendix A are accepted.	App A	**M**			
9.1 Additional Technical Infrastructure Requirements			**T**			
9.1.1	**OAI-PMH** - Demonstrate that the system can respond to OAI-PMH requests as a data provider.	9.1.1	**T**	OAI data provider only.	2	
9.1.2	**Z39.50** - By design analysis, confirm that the system can respond to data requests using the Z39.50 standard.	9.1.2	**T**			
9.1.3	**SRU/SRW** - By design analysis, confirm that the system can respond to data requests using the SRU and SRW data access standards.	9.1.3	**T**	2		
9.1.4	**SOAP** - Demonstrate that the system can respond to web service requests using SOAP.	9.1.4	**T**	0		
9.1.5	**UNICODE** - Demonstrate that the system supports UNICODE.	9.1.5	**T**	3		
9.1.6	**OpenURL** - By design analysis, confirm that the system is compliant with OpenURL.	9.1.6	**T**	3		
9.1.7	**Z39.87** - By design analysis, confirm that the system supports the Z39.87 image metadata standard.	9.1.7	**T**	3		
					1.5	
Notes:	1. **Subgroups:** A=Access, M=Metadata, P=Preservation, T=Technical Infrastructure					
	2. **Score** indicates the extent to which the test element could be demonstrated: 0=None, 1=Low, 2=Moderate, 3=High					
3.	**Preservation tests** - These sections of the functional requirements are covered by Test Plan sections P1, P2, and P3, which were defined by the Preservation subgroup to facilitate testing.					
	4. Test elements having **blue background** are the subject of outstanding questions from the Access subgroup.					

Appendix E – Fedora Testing Results

Consolidated Digital Repository Test Plan Last updated: October 16, 2008		Source Requirements	Sub-group See Note 1	Fedora 2.2/Fez 2 Release Candidate 1 (Score reflects Fedora/Fez total)				
				Test Procedure and Results			Score (0-3) Note 2	Notes
Test ID	Test Plan Element			Fedora	Fez	Both or Not Sure		
7.1.1 Ingest - Receive Submission			**T**					
7.1.1.7	**File types** - Demonstrate that the system can ingest content in all the file formats listed as "supported" in Appendix B of the NLM DR Functional Requirements document (plus MP3 and JPEG2000), specifically: MARC, PDF, Postscript, AIFF, MPEG audio, WAV, MP3, GIF, JPEG, JPEG2000, PNG, TIFF, HTML, text, RTF, XML, MPEG. Demonstrate that the system can ingest the following types of content: articles, journals, images, monographs, audio files, video files, websites, numeric data, text files, and databases. Conduct this test element by ingesting the set of files listed in the Test File spreadsheet. (The files listed in this spreadsheet contain examples of all the file formats, and all the content types identified above.)	7.1.1.7 7.1.1.9	**T**	3	3		3	Tests and demo examples show that all file types are supported.
7.1.1.1	**Manual review** - Demonstrate that the system has the capability to require that submitted content be manually reviewed before it is accepted into the repository. Demonstrate that the system maintains submitted content in a staging area before it is accepted. Demonstrate that the system notifies a reviewer when new content is ready for review. (Also see tests for 7.1.4.1, 7.1.4.2, and 8.1.2.)	7.1.1.1	**T**	0 - Fedora provides no manual review.	1 - Fez can be configured with a workflow that includes manual review, but no notification is sent.		1	
7.1.1.2	**Review and acceptance workflow** - Demonstrate that the system supports a workflow for the review and acceptance of submitted content. Demonstrate that the workflow includes the following functions: a - Receive and track content from producers; b - Validate content based on submitter, expected format, file quality, duplication, and completeness; c - Normalize content by converting content into a supported format for final ingestion into the repository; d - Human review of content; e - Acceptance or rejection of content or file format.	7.1.1.2, 7.1.1.10	**T**	0 - Fedora provides no review and acceptance workflow.	1.5 - Fez provides (a), (d), and part of (e).		1.5	Fez provides some limited workflow capabilities: manual accept and reject; staging area for submissions; JHOVE invoked to get file format information. Missing from Fez: no notifications; no normalization; no reject bin; no comments on rejection; no submitter-based content validation; no rejection based on JHOVE results; no duplicate and completeness checks.
7.1.1.3	**Reason for rejection** - Demonstrate that the system records a set of identifying information or metadata that describes the reason for the rejection of submitted content. Demonstrate two cases: (1) automatic rejection, and (2) rejection by a human reviewer.	7.1.1.3	**T**	0 - Fedora provides no review and rejection workflow.	1 - Fez provides manual rejection, but no reason for rejection, and no automatic rejection.		1	

7.1.1.4	**Rejection filter** - Demonstrate that the system allows the creation of a filter that can be used to automatically reject submitted content. (This capability will eliminate the need for manual review of some submissions and resubmissions.)	7.1.1.4	**T**	0 - Fedora has no review / rejection workflow or rejection filter.	0 - Fez provides no rejection filter.	0	
7.1.1.5	**Rejection notification** - Demonstrate that the system can notify the producer or donor when submitted content is rejected. Demonstrate two cases: (1) notification after immediate rejection by an automated process, and (2) notification after rejection by manual review.	7.1.1.5, 7.1.1.11	**T**	0 - no review / rejection workflow or rejection notification.	0 - no rejection notification	0	
(7.1.1.8)	**Metadata types** - Demonstrate that the system can ingest content with associated metadata in the following formats: all NLM DTDs, Dublin Core, MARC21, MARCXML, ONIX, MODS, EAD, TEI, PREMIS, METS. (NOTE: This test is covered by tests 8.1.1, 8.1.8, and 8.1.9)	7.1.1.8, 8.1.1, 8.1.8, 8.1.9	**M/T**	M=2.75, T=3	M=2.75,T =2	, T=3	Takes schemas very well but not necessarily DTDs. Needs disseminators to be configured.
7.1.1.10	**Format conversion** - Demonstrate that the system has the capability to convert the format of a file being ingested to a desired supported format. As a test case, demonstrate that a WAV file can be converted to MP3 format when it is ingested. (An external tool may be needed to perform the conversion. If this is the case, demonstrate that the system can invoke the required external tool.)	7.1.1.10, 7.1.1.2	**T**	2 - Fedora disseminators can provide converted files at the time of access.	2 - Fez uses ImageMagik to convert image formats, create thumbnails. M=2.75	2	Fedora's ImageManip service can convert images between gif, jpg, tiff, png, and bmp; can also resize, crop, watermark, adjust brightness, convert to grayscale.
7.1.1.12	**Resubmission** - Demonstrate that the system can ingest a SIP that is resubmitted after an error in the SIP was detected and corrected. Demonstrate two cases: the resubmission can occur after an error was detected in (1) the content of the SIP, and (2) the metadata of the SIP.	7.1.1.12	**T**	1	1	1	SIPs can be resubmitted, but neither Fedora nor Fez has any specific support (e.g. no rollback).
7.1.1.14	**Versions** - Demonstrate that the system can store, track, and link multiple versions of a file.	7.1.1.14	**T**	3 - Every datastream in an object (external file or embedded XML) can have multiple versions, which are stored and linked from the object, and can be individually retrieved.	2 - Fez add-on packages ("Version Viewing" and "Versioning of Content" expose the underlying Fedora versioning capability to the Fez UI.	3	
7.1.1.15a	**Unique identifiers** - Demonstrate that the system assigns a unique identifier to each object ingested. Demonstrate two cases: (1) a unique identifier assigned to a digital object, which may be comprised of a set of component files, and (2) a unique identifier assigned to each of the component files of a digital object.	7.1.1.15a, 7.1.1.15b	**T**	3 - Unique identifier for every object, and every datastream (file or metadata) within the object.	3 - Fez exposes underlying Fedora unique identifiers.	3	

7.1.1.15b	**Relationships** - Demonstrate that the system can represent a parent-child relationship between content items. Demonstrate two cases: (1) an object having multiple components (e.g., a document having multiple pages, each in a separate file), and (2) an object having multiple manifestations (e.g., an image having both TIFF and JPEG files).	7.1.1.15b	**T**	3 - Relationships stored as RDF in RELS-EXT datastream of every object. RDF Triplestore used for quick search and retrieval of relationships. Extendable ontology of object relationships provided.	1 - Limited exposure of Fedora's underlying relationships.		3	
7.1.1.16	**Audit trail** - Demonstrate that the system maintains an audit trail of all actions regarding receiving submissions (SIPs).	7.1.1.16	**T**	2.5 - AUDIT datastream (XML) included in every object's FOXML, records submission events.	2 - Fez workflows store submission events in PREMIS datastream.		2.5	
7.1.2 Ingest - Quality Assurance			**T**					
7.1.2.1	**Virus checking** - By design analysis, confirm that the system performs automatic virus checking on submitted content files.	7.1.2.1	**T**	0	0		0	Virus checking must be performed pre-ingest with external tools.
7.1.2.2	**Transmission errors** - Demonstrate that the system uses MD5, CRC, checksums, or some other bit error detection technique to validate that each data file submitted is received into the repository staging area without transmission errors.	7.1.2.2	**T**	1 - See note.	0 - no transmission error checks		1	Fedora design allows MD5 or other checksum to be provided in SIP, and Fedora will validate that no transmission error occurs during ingest (score=3). However, this feature inoperative in Fedora 2.2.3 due to code bug (score=1).
7.1.2.3	**Submission validation** - Demonstrate that the system verifies the validity of submitted content based on the following criteria: submitter; expected file format; file quality (e.g., actual format of file matches the filename extension, and content of file is well-formed); duplication (e.g., existence of object in the repository); completeness of metadata; completeness of file set (e.g., all expected files are included in the submission).	7.1.2.3	**T**	0	1 - Fez invokes JHOVE to check file format, but does not reject if incorrect or bad format found.		1	
7.1.2.4	**QA UI** - Demonstrate that the system allows NLM staff to perform manual/visual quality assurance on staged SIPs via a user-friendly interface.	7.1.2.4	**T**	0	2		2	
7.1.2.5	**Reaction to QA errors** - Demonstrate that the system can react to specified QA errors in two ways: (1) request that the producer correct and resubmit the content, or (2) automatically modify the submission (e.g., converting to a supported format).	7.1.2.5	**T**	0	0		0	
7.1.2.6	**File/batch accept/reject** - Demonstrate that the system enables NLM staff to accept or reject submitted content (SIPs) at the file or batch level.	7.1.2.6	**T**	0	1 - Batch or single file, no email		1	

7.1.2.7b	**Error reports** - Demonstrate that the system generates error reports for ingest quality assurance problems.	7.1.2.7b	**T**	0	0		0	No statistics or error reports
7.1.2.8	**Adjustable level of manual QC** - By design analysis, confirm that the system has the ability to adjust the level of manual ingest quality control needed, based on the origin of the file.	7.1.2.8	**T**	0	0		0	
7.1.2.9	**Audit trail** - Demonstrate that the system maintains an audit trail of all actions regarding ingest quality assurance.	7.1.2.9	**T**	1	1		1	
7.1.4 Ingest - Generate Descriptive Information / Metadata			**M**					
7.1.4.1	**Additional metadata** - Demonstrate the entry of additional metadata (e.g. subject headings, names, dates, “curatorial” descriptive metadata - evaluative information that explains why an object is important, whether it was part of a larger collection (e.g., an exhibit), etc.).	7.1.4.1	**M**	3	3		3	Creating disseminators for flat metadata is not too hard it's digging out chunk of the objects that are hard and can cause the system to crash when dealing with terabytes.
7.1.4.2	**Validate metadata** - Demonstrate ability to validate specified metadata elements.	7.1.4.2	**M**	2	2		2	Can setup required elements
7.1.4.4	**Metadata storage** - Demonstrate that metadata is stored in the database in a manner that conforms to repository reformatting and linked to their corresponding objects via an identifier. o Demonstrates that basic descriptive metadata is also stored with the objects (e.g., unique identifier, title and date stored in the TIFF header) so that the objects can still be identified in the event that information in the database is corrupted. o See Appendix D for examples of TIFF header metadata requirements. (Use of external tool probable)	7.1.4.4	**M**	3	3		3	Both bullet 1 and 2 are 3
7.1.4.5	**Required descriptive elements** - Demonstrate the ability to recognize required descriptive elements.	7.1.4.5	**M**	3	3		3	
7.1.4.7	**Audit trail** - Demonstrate the creation of an audit trail of all actions.	7.1.4.7	**M**	3	3		3	
7.1.3 Ingest - Generate AIP		Note 3	**P**					
7.1.5 Ingest - Coordinate Updates		Note 3	**P**					
7.2.1 Archival Storage - Receive Data		Note 3	**P**					
7.2.2 Archival Storage - Manage Storage Hierarchy		Note 3	**P**					
7.2.3 Archival Storage - Replace Media		Note 3	**P**					
7.2.4 Archival Storage - Error Checking and Disaster Recovery		Note 3	**P**					
7.2.5 Archival Storage - Provide Data		Note 3	**P**					
7.3.1 Data Management - Administer Database		Note 3	**P**					
7.3.2 Data Management - Administer Perform Queries		Note 3	**P**					
7.3.3 Data Management - Generate Report		Note 3	**P**					
7.3.4 Data Management - Receive Database Updates		Note 3	**P**					
7.4 Administration		Note 3	**P**					
P1 - Generate AIP			**P**					
P1-1	**Generate AIP** - Demonstrate the generation of AIPs from ingested SIPs that do not need normalization.	7.1.3.1, 7.1.3.2, 7.1.3.3, 7.4.1	**P**	2	2	2		Fedora/Fez can generate a FOXML or METS file that contains metadata and links to all datastreams but does not create a physical AIP package.

P1-2	**Generate AIP with normalization** - Demonstrate the generation of AIPs from ingested SIPs that need normalization - Transform an unsupported format to an accepted format (See Appendix B).	7.1.3.1, 7.1.3.2, 7.1.3.3, 7.4.1	**P**	0	0	0		No normalization.
P1-3	**Derivative files** - Demonstrate the generation of AIPs that consist of master files and derivatives.	7.1.3.6	**P**	3	1.5	3		Fedora's disseminator can be configured to accept images in the gif, jpg, tiff, png and bmp formats and can also convert images between these formats. Fez can create three jpg derivatives for each ingested image datastream: thumbnail, web-preview and access copy.
P1-4	**Master files** - Demonstrate the generation of AIPs that consist of master files only.	7.1.3.6	**P**	2	2		2	Fedora assigns a PID to each datastream. Fez uses a prefix (archival, preview or thumbnail) to label each datastream.
P1-5	**Store AIP in archival storage** - Demonstrate the ability to transfer AIPs to Archive Storage.	7.1.5.1, 7.2.1.1, 7.2.1.2	**P**	2	2		2	
P1-6	**Store metadata in DB** - Demonstrate the ability to generate and transfer Descriptive Information (metadata) to Data Management Database.	7.1.5.2, 7.3.4.1	**P**	3	3		3	With Fedora's GSearch, all metadata in the FOXML/METS can be indexed.
P1-7	**Link metadata and objects** - Demonstrate the ability to store identification information in the Data Management database and link digital objects in the Archive Storage.	7.1.5.4	**P**	3	3		3	
P1-8	**Send confirmation** - Demonstrate the ability to automatically send confirmation to ingest and/or receiver when AIP and metadata transfers are completed.	7.1.5.1, 7.1.5.3, 7.2.1.3, 7.3.3.3	**P**	1	1		1	Both can provide a confirmation on the screen. The email capability is not available for testing.
P1-9	**Send statistical reports** - Demonstrate the ability to automatically send statistical reports to ingest and/or receivers when AIP and metadata transfers are completed.	7.1.5.1, 7.1.5.3, 7.2.1.3, 7.3.3.3	**P**	0	1		1	Only Fez has a limited on-screen view of "My Created Items". The email capability is not available for testing.
P1-10	**Send error reports** - Demonstrate the ability to automatically send error reports to ingest and/or receivers when AIP and/or metadata transfers fail.	7.1.5.1, 7.1.5.3, 7.2.1.3, 7.3.3.3	**P**	1	1	1		Both will display an error message on the screen if the ingest fails but Fedora also records it in the log file. The email capability is not available for testing.
P2 - Administer Archival Storage & Database			**P**					
P2-1	**Monitor transfer integrity** - Demonstrate the built-in function to automatically monitor and report if any AIPs and metadata are altered or corrupted during data transfer and media change (refresh or replace).	7.2.2.1, 7.2.3.2, 7.2.4.1	**P**	3	0	3		Fedora generates a checksum for every ingested datastream. A pre-generated checksum could be supplied in the ingest process but the validation process seems having bugs (fail any supplied checksum, good or bad).

P2-2	**Check data/referential integrity** - Demonstrate the built-in function to perform routine and special referential and data integrity checks (CRC or checksums) on files in the Archive Storage and Data Management Database.	7.2.4.2, 7.3.1.1, 7.3.1.2, 7.4.4	**P**	2	0	2		Fedora maintains a checksum for each datastream in the repository but provides no referential integrity check.
P2-3	**Routine configuration for data/referential integrity** - Demonstrate the ability to allow for routine configuration.	7.2.4.2, 7.3.1.1, 7.3.1.2, 7.4.4	**P**	1	1	1		No referential integrity check.
P2-4	**Disaster recovery** - Demonstrate the ability to allow for disaster recovery including data backup, off-site data storage, and data recovery.	7.2.4.3	**P**	2	1		2	In Fedora there are three ways to export data: Archive (the exported XML file includes all metadata and Base64-encoded datastreams); Migrate (the exported XML file contains metadata and links to datastreams - for migration of objects from one repository to another); Public Access (similar to Migrate but for use outside the context of a Fedora repository). As long as all the datastreams are backed up, Fedora claims that the FOXML file can be used to rebuild the entire repository. Fez has a very limited export function that can only output the metadata and links to datastreams in spreadsheet/CSV format wrapped in XML.
P2-5	**User views** - Demonstrate the ability to allow for customized user views of the contents of the storage (create, maintain, and access).	7.3.1.4	**P**	2	2		2	
P2-6	**System CM** - Demonstrate the ability to allow for configuration management of the system hardware and software.	7.4.2	**P**	2	2.5		2.5	Fedora has a limited admin client but Fez provides a GUI interface for system configuration management.
P2-7	**Database CM** - Demonstrate the ability to allow for configuration management of the Data Management Database such as table, schema definitions, etc.	7.3.1.3	**P**	2	2		2	
P2-8	**Delete AIPs** - Demonstrate the ability to allow the authorized staff to delete AIPs from the repository including: removing the digital object's files and retaining associated metadata, or removing both the files and metadata.	7.4.3.4	**P**	3	3	3		Fedora provides a purge function that can physically remove an object from the repository. Fez has a delete function but it only marks an object for delete instead of removing it from the repository. Using Fedora to purge an object that was marked for delete by Fez may not completely remove all associated files/data.
P2-9	**Coordinate AIP removal** - Demonstrate the ability to generate an alert and coordinate the removal of an AIP with maintenance of metadata held in other systems.	7.4.3.5	**P**	0	0		0	

P2-10	**File migrations** - Demonstrate the ability to allow the authorized staff to schedule and perform file migrations or migration on request for batched and individual files by authorized staff.	7.4.3.6	**P**	0	0		0	
P2-11	**Request DIPs for update** - Demonstrate the ability to allow the authorized staff to request DIPs for file migrations and data updates.	7.3.4.1, 7.3.4.2, 7.3.4.3, 7.4.3.1, 7.4.3.2, 7.4.6.2	**P**	3	0	3		Fedora can export metadata and/or datastream (in Base64 encoding). Fez can only export metadata and links to datastreams in spreadsheet/CSV format wrapped in XML but not datastreams.
P2-12	**Re-ingest updated DIPs** - Demonstrate the ability to allow the authorized staff to reingest updated DIPs as SIPs.	7.4.3.3	**P**	2	0		2	Fedora allows the user to specify changes in the FOXML/METS file for re-ingest but it does not allow the same UID to be reingested. Fez is not capable of re-ingesting its own exported content.
P2-13	**Support query requests** - Demonstrate the ability to receive, retrieve, display, and deliver data for query requests from other functions such as Ingest, Access, and Administration.	7.3.2.1, 7.3.2.3, 7.4.6.1	**P**	2	2		2	
P2-14	**Query requests from different storage locations** - Demonstrate the ability to handle query requests with required data to be sourced from different storage locations.	7.3.2.2	**P**	3	2	3		Fedora supports data sourced from local, external (remote in FOXMAL) or redirect (not disseminated). Fez supports data sourced from local or redirect.
P2-15	**Queries against all metadata** - Demonstrate the ability to run data queries against all metadata used to manage the repository.	7.3.2.4	**P**	2.5	2		2.5	With Fedora GSearch, all metadata captured in the FOXMAL/XML file can be indexed for search. Fez has a built-in function that can be used to manage all searchable keys.
P2-16	**Audit trial** - Demonstrate the creation of an audit trail of all actions including who, when, how, what and where for Archive Storage and Data Management Database.	7.1.3.4, 7.1.5.6, 7.2.1.4, 7.2.2.3, 7.2.5.2, 7.3.2.5, 7.3.3.7, 7.3.4.6, 7.4.3.7, 7.4.6.4	**P**	2.5	1.5		2.5	Fedora/Fez can record all actions in FOXML.
P2-17	**Generate reports** - Demonstrate the ability to receive, generate, display, and deliver management information reports and statistics such as summaries of repository holdings by category, summaries of updates by category, user codes, etc., usage statistics for access to repository holdings, and descriptive information for a specific AIP.	7.3.3.1, 7.3.3.2, 7.3.3.5, 7.3.4.4	**P**	1	1	1		Fedora has a limited “Repository Reports” capability that can be invoked from the REST interface (.../fedora/report). The report lists all objects in the repository of a specified type that have been modified or created in a specified timeframe. Fez also has a limited reporting capability that allows the “admin” user to view a list of

								"My Created Items" on the screen.
P2-18	**Schedule reports** - Demonstrate the ability to generate reports in an ad-hoc manner, automatically or to be triggered by a calendar or by a specific system event.	7.3.3.4	**P**	0	0			
P2-19	**Time period for reports** - Demonstrate the ability to allow the user to specify a time period or set of time periods for reports and statistics.	7.3.3.6	**P**	1	1	1		Fedora's limited "Repository Reports" capability allows the user to specify a time period for the report, e.g., all objects created or modified in the past 24 hours, 7 days, etc. Fez allows the "admin" user to specify a "before" or "after" date to find items only in "My Created Items".
P3 - Generate DIP			**P**					
P3-1	**Generate DIP for access requests** - Demonstrate the generation of DIPs by putting AIPs and Descriptive Information back together for access requests.	7.1.5.5, 7.2.5.1, 7.4.6.2	**P**	2	2	2		Fedora has both REST and SOAP interfaces available in its access API (API-A). A coordinated set of web service calls can be made to retrieve all the metadata and datastreams of an object, which can be combined and displayed to a user. Fez provides similar functions for access requests.
P3-2	**Generate DIP for object maintenance** - Demonstrate the generation of DIPs by putting AIPs and Descriptive Information back together for content/metadata update, versions upgrades and format migration by authorized staff.	7.4.6.2, 7.4.3	**P**	2	2		2	The Fedora Admin Client enables authorized administrators to edit metadata, import new versions of datastreams, and export entire objects for migration. Command line utilities provide key functions of the management API (API-M) that can be invoked directly or from customized scripts. DIP objects can be exported in FOXML/METS format, and can include all metadata and all datastreams (base64-encoded) in a single XML file. Fez has a workflow-based export function that allows the "admin" user to export selected community, collection or record in CSV or spreadsheet format

								wrapped in XML. The exported XML file contains only metadata and file names of datastreams.
7.4.1 Administration - Negotiate Submission Agreement			**T**					
7.4.1.1	**Manage submission agreements** - Demonstrate that the system manages information regarding submission agreements: that it tracks negotiation status and written submission agreements, and that it maintains schedules.	7.4.1.1	**T**	0	0		0	
7.4.1.2	**Edit submission agreements** - Demonstrate that the system allows submission agreements to be edited, based on the access level of the user.	7.4.1.2	**T**	0	0		0	
7.4.1.5	**Terms of submission agreements** - Demonstrate that the system stores the terms of submission agreements, and uses the terms to monitor, review, and process submissions.	7.4.1.5	**T**	0	0		0	
7.4.1.6	**Audit trail** - Demonstrate that the system maintains an audit trail of all actions related to submission agreements.	7.4.1.6	**T**	0	0		0	
7.4.2 Administration - Manage System Configuration			**T**					
7.4.2.1	**Monitor repository functionality** - Demonstrate that the system monitors the functionality of the entire repository.	7.4.2.1	**T**	0	0		0	
7.4.2.2	**System configuration** - By design analysis, confirm that the system maintains the integrity of the system configuration.	7.4.2.2	**T**	0 - Info stored in FOXML objects; some info saved to relational DB	2 - Fez utility to manually check site installation configuration		2	Easy-to-use command-line function that rebuilds Resource Index and relational DB if corruption occurs.
7.4.2.3	**Audits operations** - Demonstrate that the system audits system operations, performance, and usage.	7.4.2.3	**T**	1 - Log file contains errors for sysadmin and programmer. Log files at file level. Audit trail for each object in FOXML.	1 - Some limited Fez logs		1	
7.4.2.4	**Data management information** - Demonstrate that the system collects and can display system information concerning Data Management.	7.4.2.4	**T**	0	0		0	
7.4.2.5	**Operational statistics** - Demonstrate that the system collects and can display operational statistics concerning Archival Storage.	7.4.2.5	**T**	0	0		0	
7.4.3 Administration - Archival Information Update								
7.4.5 Administration - Audit Submission			**T**					

7.4.5.1	**Audits** - Demonstrate that the system can support an audit procedure to verify that submissions (SIP or AIP) meet specified requirements of the repository. The audit method may be based on sampling, periodic review, or peer review. [See NLM DRD Functional Requirements document, section 7.4.5 for description of audit requirements.] (Also partially covered by 7.2.4.2)	7.4.5.1	**T**	0	0		0	
7.4.5.2	**Metadata audit** - Demonstrate that the system can audit metadata as part of the audit procedure.	7.4.5.2	**T**	0	0		0	
7.4.5.3	**Audit rejection -** Demonstrate that the system can reject components of audited information packages, based on specified audit requirements.	7.4.5.3	**T**	0	0		0	
7.4.5.4	**Audit report** - Demonstrate that the system can generate an audit report, based on the results of periodic audits of SIPs and AIPs.	7.4.5.4	**T**	0	0		0	
7.4.5.5	**Audit trail** - Demonstrate that the system maintains an audit trail of all actions regarding the auditing of SIPs and AIPs.	7.4.5.5	**T**	0	0		0	
7.4.6 Administration - Activate Requests			**P**					
7.6.1 Access - Coordinate Access Activities - User Access			**A**					
7.6.1.1	**Manage user permissions** - Demonstrate the access controls for multiple permission levels and user privileges.	7.6.1.1	**A**	User permissions are controlled via XACML. Custom policies can be created, and policies can be nested logically.	The need to hold down ctrl while adding members to groups is a little risky - too easy to deselect members.		2	
7.6.1.2	**Manage user restrictions** - Demonstrate multiple levels of access restrictions for NIH employees and general public based on licensing terms, embargo periods, IP range restrictions, workstation access, and other possible legal restrictions.	7.6.1.2, 7.6.1.3	**A**	XACML policies can be written to allow or deny access at every level of object aggregation, using IP range, inactive/deleted status of datastreams, etc. Fedora supports LDAP simple user/password out of the box, but other sources can be configured.	Access restrictions to communities are granular but not as visible as we would like. AD integration is very attractive.		2	

7.6.1.4	**Manage user settings** - Demonstrate access settings allow staff to add or edit descriptive metadata	7.6.1.4	**A**	XACML policies can be written to allow or deny access at the datastream level. Metadata editing requires the Fedora client.	Granular, role-based access to add or edit descriptive metadata. This takes some up-front configuration, but works OK.	1	
7.6.1.7	**Audit users** - Demonstrate access mechanisms can identify individual users and maintain audit log of user actions.	7.6.1.7	**A**	Every change to a datastream can be versioned with audit trail record.	Premis event synopsis is viewable in the public view, more detailed log is available.	2	
7.6.1.5	**Perform maintenance tasks** - Demonstrate maintenance access including adding new files, manipulating images, editing metadata, performing format conversions/migrations, and troubleshooting system problems.	7.6.1.5	**A**	Fedora allows adding files, and files can be manipulated via disseminators. Some troubleshooting will require the client or command line actions.	Fez allows adding new files and editing metadata. Image manipulation and format conversion is not directly supported, but Fez can manage content after it has been externally manipulated or converted. System troubleshooting is excellent, with a very thorough sanity checker to detect common installation problems. Run-time errors are saved to the log and can be optionally sent to the browser, with configurable levels of error detail (time, object, method, parameters).	1	

7.6.1.6	**Manage system rights** - Demonstrate ultimate system rights access for NLM system administrators and programmers.	7.6.1.6	**A**	Some admin access is controlled by database and OS accounts, but Fedora user privileges are controlled via the XACML policies.	The ability to add users or change user privileges can be isolated to users with specific application administrative privileges. There is also a Community Administrator role. Rights are stored in the Fez DB.	2	
7.6.1 Access - Coordinate Access Activities - Rights/Data Control of Objects			**A**				
7.6.1.8	**Manage access rights** - Demonstrate access rights and conditions to materials and storage directories provide for a combinational of create/write; edit; read; delete privileges.	7.6.1.8	**A**	Granular access control to objects\datastreams\disseminators or aggregates\repository-wide policies via XACML. Custom policies can be created, and policies can be nested logically.	Editing security options at the community, collection and item level appears intuitive and powerful, but we have been unable to successfully test most of this area.	2	
7.6.1.9	**Manage metadata rights** - Demonstrate access rights may be associated with the metadata relating to an individual object	7.6.1.9	**A**	Granular access control to objects\datastreams\disseminators, including metadata datastreams, via XACML policies.	Access to the object's record should be controllable. Unable to test this successfully with granular permissions.	2	
7.6.1.13	**Manage relationships** - Demonstrate access rights and conditions can be inherited from a parent object to any child object.	7.6.1.13	**A**	XACML policies can utilize the RELS-EXT values to allow or deny access.	Security settings allow for parent-child propagation of security values.	2	
7.6.1.14	**Manage relationships** - Demonstrate access rights and conditions can be assigned to an object on an individual or group basis at same time.	7.6.1.14	**A**	XACML policies can be assigned to a content model or by PID.	Child objects can inherit parent access controls or have their own independent controls. As with 7.6.1.8, this has not been successfully tested.	1	

7.6.1.16	**Automated retrieval** - Demonstrate objects in the repository are accessible for data mining or automated retrieval.	7.6.1.16	**A**	Automated retrieval is not facilitated, but comprehensive indexing of metadata and fulltext is available with indexing plug-in.	Automated retrieval is not facilitated, but comprehensive indexing of metadata and fulltext is available with indexing plug-in.	1	
7.6.1.17	**Metadata access** - Demonstrate access to deleted and retracted metadata is retained.	7.6.1.17	**A**	Fedora supports write-once, where any changes to datastreams are versioned.	Versioning of underlying datastreams is delegated to Fedora. Metadata, attached files and hyperlinks can be versioned through Fez.	1	
7.6.1.18	**Metadata harvesting** - Demonstrate metadata harvesting following the OAI-PMH guidelines.	7.6.1.18	**A**	Fedora includes an OAI provider to expose content for harvesting. Recently rewritten for Fedora 3.0.	Fez can utilize the Fedora OAI provider.	1	
7.6.1.10	**Access rights** - Demonstrate access rights and conditions of use are applied to each digital object and its related metadata and are machine readable and actionable.	7.6.1.10, 7.6.1.11	**A**	XACML policies are machine-readable by design.	Rights can be applied per datastream, object and higher-level aggregations.	2	
7.6.1.12	**Access conditions** - Demonstrate access conditions are specific to a digital object.	7.6.1.12	**A**	Policies can be applied at the datastream level and all higher aggregations of content.	Rights can be applied per datastream, object and higher-level aggregations.	2	

7.6.1.15	**Free/Restricted access** - Demonstrate free (items available via internal/external delivery mechanisms) and restricted access (access permission must be satisfy various criteria) status for objects, files, metadata, etc.	7.6.1.15	**A**	XACML policies can be written to allow or deny access at every level of object aggregation, using IP range, inactive/deleted status of datastreams, etc. Policies should be able to accommodate embargo logic ("moving wall").	Access controls are granular (in theory, unable to test successfully). No "embargo" logic is present.	2	
7.6.1 Access - Coordinate Access Activities - Search and Retrieval			**A**				
7.6.1.19	**508 compliance** - Demonstrate the search interface is web-accessible and Section 508 compliant.	7.6.1.19	**A**	Fedora's thick client does not appear to be Section 508 compliant. However, NLM staff could use alternative methods for ingesting and managing content such as running UNIX commands or via a Web UI. Section 508 compliance is a design goal in any upcoming UI development.	Fez is an Australian product, so it is not bound by the Section 508 requirements. Since the product is open-source, NLM could easily tweak the HTML templates, etc. to create accessible UIs, etc. were feasible.	1.5	
7.6.1.20	**Search features** - Demonstrate search includes: metadata, full-text, standard boolean, proximity, "more like" this"	7.6.1.20, 7.6.1.21, 7.6.1.22, 7.6.1.23, 7.6.1.24	**A**	Metadata searching with some operators, less GUI than Fez. GSearch supports full text indexing and	No explicit "or" searching in our environment, but UQ has it. Lots of metadata searching, with wildcards. No proximity or "more like".	1	

				searching, proximity.			
7.6.1.25	**Search results display** - Demonstrate search results display includes date sort; relevancy ranking; alpha by author or source.	7.6.1.25	**A**	no custom ordering of results. Default order is by PID.	No Author or source, but date, relevance, title, description.	2	
7.6.1.26	**Relevancy ranking** - Demonstrate whether relevancy ranking can be manipulated via system as well as user defined settings.	7.6.1.26	**A**	n/a	Not accessible through admin interface.	0	
7.6.1.29	**Federated search** - Demonstrate federated searching of different repository sites.	7.6.1.29	**A**		Can search across all or select communities/collections via advanced search.	2	
7.6.1.30	**Advanced search** - Demonstrate advanced search includes search history; saved searches; saved citation lists/bibliographies; alerts; various functions and formats; dynamic selection of delivery media without recreating search query.	7.6.1.30	**A**	none of these functions are present	Can save searches as RSS feeds,	1	
7.6.1.31	**Display formats** - Demonstrate a variety of standard display formats are provided and whether they are customizable by user	7.6.1.31	**A**	Fedora lets you select the fields to display.	Can be saved as XML, RSS, citation-only. Not customizable by user.	1	
7.6.1.32	**Alternate search interfaces** - Demonstrate availability of alternate search interfaces for mechanisms such as handhelds and PDAs.	7.6.1.32	**A**			0	
7.6.1.33	**Object access** - Demonstrate access to the appropriate copy of the identified item (text, image, video, etc.)	7.6.1.33	**A**	Datastreams have no preference.	Unclear how to identify the appropriate datastream, although one is highlighted. Can identify differences in datastream descriptions.	0	
7.6.1.34	**Library holdings** - Demonstrate integration of search results with library holdings.	7.6.1.34	**A**			0	
7.6.1.35	**Response time** - Demonstrate acceptable response time.	7.6.1.35	**A**	Response time is acceptable, within our test environment	Response time is acceptable, within our test environment and limited collection.	1	

				and limited collection.			
7.6.1.36	**External search engines** - Demonstrate searching by outside search engines such as usa.gov, Google, and Yahoo.	7.6.1.36	**A**	so far, evidence suggests only library web pages with a "browse view" external to Fedora are spidered.	uq.edu's espace browse pages appear to be indexed by Google	1	
7.6.1.37	**External system access** - Demonstrate external access to other repositories or systems performing web harvesting functions.	7.6.1.37	**A**	Fedora has a built-in OAI-PMH Provider Interface, and all objects have a compliant DC record. Only the DC metadata may be disseminated, however.	Fez could delegate the OAI-PMH service to Fedora.	1	
7.6.1.38	**Language support** - Demonstrate how multiple languages and non-Roman scripts are supported in search, retrieval and display.	7.6.1.38	**A**	Chinese characters do not display in test record (fedorans:137).	Chinese characters displayed in search results (fedorans:137), but not searchable.	1	
7.6.1.39	**Versioning** - Demonstrate access to all versions of digital objects in the repository is provided.	7.6.1.39	**A**	Fedora objects can be versioned at every level, including disseminators.	All versions are accessible, but no versioning functionality for uploaded content. This is in the works for a future release.	2	
7.6.1.40	**Search settings** - Demonstrate system settings and user-defined settings in the search functions are provided.	7.6.1.40	**A**	Only default system-provided search settings are offered.	Only default system-provided search settings are offered.	1	
7.6.2 Access - Generate DIP			**A**				
7.6.2.1	**Integrate holdings** - Demonstrate integration of search results with library holdings.	7.6.2.1	**A**	No functionality built-in to	No functionality built-in to Fez for this.	0	

				Fedora for this.			
7.6.2.2	**Retrieval and notification** - Demonstrate the generation function accepts a dissemination request, retrieves AIP from archival storage and moves a copy of the data to a staging area for further processing, and creates and sends a report request to data management to obtain appropriate metadata.	7.6.2.2, 7.6.2.3, 7.6.2.4	**A**	AIP/DIP is conceptual, but the search API can result in a list of any and all datastreams, including all metadata associated with the object.	AIP/DIP is conceptual. Search interface can provide links to multiple derivatives of an object, the archival master and associated metadata.	2	
7.6.2.7	**Audit trail** - Demonstrate an audit trail of all actions is created and stored.	7.6.2.7	**A**	Tomcat logs can provide dissemination requests (according to Indiana Univ. DLP).	Fez can track downloads per file, but this is not working in testing.	1	
7.6.2.5	**Response and delivery** - Demonstrate that the prepared DIP response is placed in the staging area and a message is generated and sent to Coordinate Access Activities that the DIP is ready for delivery.	7.6.2.5	**A**	This aspect of OAIS is not currently modeled by Fedora. Fedora does not appear to use a staging area but serves requested content directly from the repository.	This aspect of OAIS is not currently modeled by Fez. Fez does not appear to use a staging area but serves requested content directly from the repository.	0	
7.6.2.6	**Storage retrieval** - Demonstrate that Generate function accesses data objects in staging storage and applies the requested processes if special processing is required.	7.6.2.6	**A**	No staging storage area per se, but disseminators can process the master file(s), separating it from the DIP.	No staging storage area per se, and Fez architecture inhibits the disseminator functionality of Fedora.	1	Disseminator layer of Fedora is quite powerful and flexible, but cumbersome to configure in 2.2.3. Fez's inability to leverage the Fedora disseminators is a big downside.
7.6.3 Access - Deliver Response			**A**				
7.6.3.1	**Web-accessibility** - Demonstrate the display interface is web-accessible.	7.6.3.1	**A**	Fedora has a fairly limited web interface for retrieval.	Fez is entirely web-accessible.	2	

7.6.3.2	**Downloading** - Demonstrate export function that provides XML output for batch downloads	7.6.3.2	**A**	Objects can be exported as METS packages, and some individual datastreams are downloadable as XML.	Fez can export some (but not all) metadata into XML. It cannot then re-ingest from the export output. Export is intended for spreadsheet manipulation of metadata.		1	
7.6.3.3	**Saving content** - Demonstrate users are allowed to save digital content to a hard-drive, e-mail, and/or save search results.	7.6.3.3	**A**	Files may be downloaded. There does not appear to be a function for emailing or saving search results.	Files may be downloaded. There does not appear to be a function for emailing or saving search results.		1	
7.6.3.5	**System notification** - Demonstrate a confirmation message is returned to the Coordinate Access Activities section after response has been sent.	7.6.3.5	**A**	This aspect of OAIS is not currently modeled by Fedora.	This aspect of OAIS is not currently modeled by Fez.		0	
7.6.3.6	**Audit trail** - Demonstrate an audit trail of all actions is created and stored.	7.6.3.6	**A**	Tomcat logs can provide dissemination requests (according to Indiana Univ. DLP).	Fez can track downloads per file, but this is not working in testing.		1	
7.6.3.4	**Response request** - Demonstrate a response request is received from Coordinate Access Activities	7.6.3.4	**A**	Demonstrated retrieval of objects via the UI without issue.	Demonstrated retrieval of objects via the UI without issue.		2	
8.1 Metadata Requirements			**M**					
8.1.1	**Metadata formats** - Demonstrate that the system can accept metadata associated with objects in at least the following formats: All NLM DTDs, Dublin Core, MARC21, MARCXML, ONIX, MODS, EAD, TEI.	8.1.1	**M/T**	T=3 M=3	T=3 Any metadata could be added as a datastream; M=3.		T=3 M=3	Fedora is completely agnostic about what kinds of metadata and number of metadata objects that can be assigned to any object
8.1.2	**Metadata checks** - Demonstrate the built-in checks on the incoming metadata. Records not containing the minimally defined set of fields should be flagged as problems, either to be returned to the submitter, or sent locally for metadata enhancement.	8.1.2	**M**	3	3		3	Fedora would need an additional tool to perform checks.
8.1.5	**Metadata updates** - Demonstrate the ability to allow for metadata updates.	8.1.5	**M**	3	2.5		3	Fedora client only has one template field for descriptive title; actual object metadata box can take anything just need disseminator to do something with it.
8.1.6a	**Metadata search and display** - Demonstrate the ability to search and display metadata (use of external tool possible).	8.1.6a	**M**	3	2.5		3	

8.1.8	**PREMIS** - Demonstrate standards compliance for PREMIS (use of external tool possible).	8.1.8	**M/T**	T=3 M=3	T=2 Fez limited. Fez won't display Premis metadata if it was added in Fedora. M=2		T=3 M=3	Fez creates PREMIS metadata for each object, stored as Fedora datastream in the object.
8.1.9	**METS** - Demonstrate standards compliance for METS (use of external tool possible).	8.1.9	**M/T**	T=3 Fedora can ingest METS SIPs, and export objects in METS format. M=3	T=1 Fez limited. METS could be stored as a datastream. M=2		T=3 M=3	Fedora can store METS metadata as a datastream in an object, e.g. to drive a METS-based page-turner.
App A	**Descriptive metadata** - Demonstrate that the minimum descriptive metadata requirements described in Appendix A are accepted.	App A	**M**	3	2		3	
9.1 Additional Technical Infrastructure Requirements			**T**					
9.1.1	**OAI-PMH** - Demonstrate that the system can respond to OAI-PMH requests as a data provider.	9.1.1	**T**	2	0		2	Fedora has a basic OAI-PMH capability, and an extended capability using the optional OAI-Provider tool (in the Fedora Service Framework).
9.1.2	**Z39.50** - By design analysis, confirm that the system can respond to data requests using the Z39.50 standard.	9.1.2	**T**	0	0		0	
9.1.3	**SRU/SRW** - By design analysis, confirm that the system can respond to data requests using the SRU and SRW data access standards.	9.1.3	**T**	0	0		0	VTLS provides SRU/SRW for Arrow project.
9.1.4	**SOAP** - Demonstrate that the system can respond to web service requests using SOAP.	9.1.4	**T**	3	0		3	
9.1.5	**UNICODE** - Demonstrate that the system supports UNICODE.	9.1.5	**T**	3	2		3	UNICODE filename not displayed properly in Fez. UNICODE file content handled ok.
9.1.6	**OpenURL** - By design analysis, confirm that the system is compliant with OpenURL.	9.1.6	**T**	0	0		0	
9.1.7	**Z39.87** - By design analysis, confirm that the system supports the Z39.87 image metadata standard.	9.1.7	**T**	0	0		0	
Notes:	1. **Subgroups:** A=Access, M=Metadata, P=Preservation, T=Technical Infrastructure							
	2. **Score** indicates the extent to which the test element could be demonstrated: 0=None, 1=Low, 2=Moderate, 3=High							
3.	**Preservation tests** - These sections of the functional requirements are covered by Test Plan sections P1, P2, and P3, which were defined by the Preservation subgroup to facilitate testing.							
	4. Test elements having **blue background** are the subject of outstanding questions from the Access subgroup.							