Towards Interoperable Preservation Repositories (TIPR)

Final Report to the Institute for Museum and Library Services

LG-07-08-0097-08

Florida Center for Library Automation
Cornell University Library
New York University Libraries
Overview

Towards Interoperable Preservation Repositories (TIPR), a joint project of the Florida Center for Library Automation, the Cornell University Library, and the New York University Libraries, was designed to develop and test a standard packaging format for the transfer of rich archival information packages (AIPs) from one preservation repository to another. The project developed the Repository eXchange Package (RXP) and tested its use in exchanging AIPs among three technically heterogeneous and geographically distributed preservation repositories. Project outputs include a specification for the RXP, tools to help validate and use the RXP, a framework for inter-repository service agreements, and a finer understanding of use cases for the transfer of archived content from one preservation repository to another.

Background

The field of digital preservation has advanced to the point where cultural heritage institutions have a number of options for long-term preservation solutions. Many university libraries and library consortia have built their own custom repository systems or have joined or formed Private LOCKSS Networks like the MetaArchive. Vendors provide commercial product options such as Ex Libris’ Rosetta and Tessella’s Safety Deposit Box. Open source institutional repository systems such as DSpace and EPrints are adding some preservation functionality, and open source repository applications designed specifically for preservation, such as DAITSS and Archivematica, are becoming available. Fee-based third-party repository services are lagging but will be increasingly available as vendors move their services to the cloud. A distributed national and international network of heterogeneous preservation repositories is becoming a reality.

In this environment, there is a need for preservation repositories in geographically separated places, run by unrelated organizations, and using different repository software applications, to transfer stored archival content among them. The most obvious use case is in when an existing repository must for some reason cease operations. Trustworthy Repositories: Audit and Certification: Criteria and Checklist (TRAC) requires a repository to have a formal succession plan in place (“A1.2 Repository has an appropriate, formal succession plan, contingency plans, and/or escrow arrangements in place in case the repository ceases to operate or the governing or funding institution substantially changes its scope.”). In many cases the preferred plan would be for content to be transferred from the original repository to another, more viable, preservation archive. Other use cases were developed as part of the TIPR project and will be discussed in Analysis, below.

In the Reference Model for an Open Archival Information System (OAIS), repositories ingest (take in) content packaged as Submission Information Packages (SIPs), process and transform these into Archival Information Packages (AIPs), and store the AIPs safely in Archival Storage. Once content is stored in an AIP in one repository, the original SIP may no longer exist. The only way to get a copy of the content into a second repository is to export it from the first repository and send it to the second repository for ingest. In OAIS terms, the sending repository must use the stored AIP to create a Dissemination Information Package (DIP) and transfer the DIP to the receiving repository, which must treat the DIP as a SIP for ingest.
The problem with this seemingly simple scenario is that there is no standard format for either a SIP or an AIP – each repository system (and in some cases, each local implementation of a given repository system) will have its own requirements. Moreover, in the process of creating an AIP from a SIP, a repository will enhance the AIP with additional metadata and in some cases, with additional content. A trustworthy repository will maintain a history of the chain of custody of each package it archives, and a record of all actions it performed that modified the package or were in any way significant to its preservation (for example, a validation check). When content is transferred from one repository to another, all of that information must transfer with it, or the digital provenance of the content will be broken. Information about the transfer itself must also be maintained for the same reason.

In the absence of uniform standards for SIPs, DIPs and AIPs (something which would be difficult or impossible to achieve, given the diversity of content and contexts) there are only three options for interoperability:

1) The sending repository produces a DIP in its own format. The DIP is then translated, by either the sender or the receiving repository, into the SIP format required by the receiving repository. This could be workable for a very small number of bi-lateral transfers, but gets complicated in a many-to-many environment, where potentially every repository system would have to know the preferred format of every other system.

2) The sending repository produces a DIP in its own format. The DIP is then translated by a central translation resource that knows all formats into the SIP format required by the receiving repository. This is the model explored by the University of Illinois at Urbana Champaign in their Hub and Spoke Framework. The problem here is finding a stable central organization willing to maintain current translations for existing and new repository systems and implementations.

3) The sending repository produces a DIP in its own format, and transforms the DIP to a common format used for repository-to-repository exchange. The receiving repository transforms the common format into its own required SIP format. This obviates the need for all repositories to know the formats required by all other repositories, since each repository only needs to know its own format and the common exchange format. It also gets around the problem of maintaining a central conversion facility. However, it requires that the community develop and adopt a common repository-to-repository transfer format.

The purpose of the TIPR project was to address this need by developing a common, XML-based transfer format; testing the format in various exchanges between heterogeneous repository systems; and promoting and publicizing the format to encourage its adoption in the digital preservation community. Specific goals as stated in the project proposal were to:

- demonstrate the feasibility of repository-to-repository transfer of rich archival information packages;
- advance the state of the art by identifying and resolving issues that impede such transfers;
- develop a stable, standards-based transfer format, building on prior work;
- disseminate results to the international preservation community and the relevant standards activities.
Activities

The project was carried out in four overlapping phases: Analysis, Coding, Testing and Dissemination.

Analysis

In Phase 1, Analysis, the goals were to develop specifications for the common transfer format, develop use cases for transfer, and plan transfer tests. The project partners FCLA, Cornell, and NYU became familiar with the storage formats and requirements of each other’s repository applications. FCLA used DAITSS, a preservation repository application developed locally with help from a grant from the IMLS. Cornell used aDORe, an archive and federated archive framework developed at the Los Angeles National Laboratory (LANL). NYU used a preservation repository based on the open source DSpace institutional repository system.

The transfer format specification was named the Repository eXchange Package (RXP) and is described in the section Deliverables below.

The meaning and intent of the use cases changed during this phase of the project. Originally, use cases were conceived in terms of content. An attempt was made to identify the various types of content archived in the partners’ repositories (digitized books, audiovisual material, oral histories, etc.) and to determine which types of content should be included in testing the RXP. During the course of discussions it became apparent that the nature and structure of the content itself were not the important factors affecting a successful and interpretable transfer. Rather, we found an institution’s motivation for doing a repository-to-repository transfer could affect that institution’s idea of success, and that use cases for wanting to do such exchanges would be useful for working out a test plan. As a result, some effort was given to identifying these use cases, which are described here:

Succession. Trustworthy repositories should have a formal succession plan in place in the event that the repository ceases to operate or changes its scope. The preferred plan might be to transfer content from the terminating repository to another more viable repository service. Source repositories can package AIPs as RXPs and transfer the RXPs to successor repositories. The successor repositories can convert the RXPs into SIPs, ingest the SIPs, ingest and update the associated digital provenance, provide notification of receipt, and formally take custody of the content.

Disaster Recovery. Two geographically distributed preservation repositories may establish a reciprocal storage agreement in which each one agrees to allocate a certain amount of storage for the partner’s AIPs. The repositories package their AIPs as RXPs and send them to the partner repository for ingest. Each repository then processes the incoming RXPs as in the Succession case. Should one repository experience a disaster, the other can export its AIPs as RXPs and send them back to the originating repository. As part of the RXP-generation process, the sending repository exports all of the digital provenance events that occurred while the AIPs were in its custody so the receiving repository can restore its AIPs while maintaining an unbroken chain of digital provenance.

Temporal Interoperability. In a conventional filesystem backup environment, the backup system replicates the files as they appear on disk. If a preservation repository uses an opaque file renaming strategy that distributes AIP files across different directories, then disaster recovery from backup tape is dependent upon having an operating instance of the repository that was in existence when the backup
tapes were created. However, if the repository first packages its AIPs as RXPs and writes the RXPs to backup tape instead, then the AIPs can be recovered by any preservation repository implementation able to read the RXP format. This use case may be beneficial to preservation repositories that write their AIPs directly to tape.

**Software Migration.** A repository that upgrades its repository software application might need to migrate all of the AIPs from its current system to the new system. The AIPs in the current system can be exported as RXPs which can be ingested into the new system. This scenario is an example of technological and short-term temporal interoperability

**Diversification.** Those in the OAIS Producer role may request that particularly valuable content be stored in technologically heterogeneous preservation repositories. Storing AIPs in technologically heterogeneous repositories reduces the risk that a software bug in one repository software application will result in the loss or corruption of all copies of an AIP.

**Specialized Content Processing.** A general-purpose repository may have some content in a format for which it has no particular expertise. However, there is another repository which specializes in content in that format. The first repository can enter into a repository exchange agreement in which it exports selected AIPs and sends these the second repository as RXPs; the specialized repository converts the RXPs to its own SIP format and ingests them, performs format-specific processing such as description, migration and/or normalization, updates the digital provenance as appropriate, and exports the new AIPs as RXPs back to the first repository for ingest.

These use cases raise different requirements for the success of a transfer. Succession, Temporal Interoperability, Software Migration, and Diversification require that an AIP can be transferred successfully from the originating repository into a receiving repository. In all cases except Succession, the two repository services will run different repository systems by definition. In the cases of Temporal Interoperability and Software Migration, the institution responsible for the content does not change; while in the cases of Succession and Diversification, responsibility for the transferred package falls to the institution running the target repository. Here success requires that the target repository can successfully ingest the package; identify and retain the original file and package identifiers; understand and retain the digital provenance of the package; and, importantly, understand the structure and content of the package, which it must continue to manage in the future.

In the cases of Disaster Recovery and Specialized Content Processing, the package must make a round trip, moving from the originating repository to the target repository and back into the originating repository again. In these cases it could be argued that the target repository has no real obligation to understand the structure and content of the original AIP, because in neither case does it have long-term responsibility for maintaining the AIP. In these cases the most important thing is that the originating repository loses no information in the process, and retention of original identifiers is critical.

A distinction was drawn between “shallow” and “deep” ingest on the part of the target repository. In a shallow ingest, such as required for disaster recovery, the receiving repository has no obligation to truly “understand” the content it is archiving, it must only be able to give that content back to the originating repository. A deep ingest, such as required for succession, means the receiving repository fully understands and takes custody of the content.
Towards Interoperable Preservation Repositories

To a large extent the RXP and the transfer tests of the RXP were designed around these requirements. The issues described below were also important in the design of the RXP.

Standards

A transfer format leveraging established standards would have more traction, a better chance of being adopted, and a better chance of being maintained in the future after the end of the funded TIPR project. METS and PREMIS were chosen as vehicles for carrying information that had to be understandable to the target repository, primarily because they were already in widespread use within the digital preservation community. METS is a container format used widely as a manifest for SIPs and DIPs. METS requires a section of structural metadata called the StructMap and allows optional sections of other types of metadata to be included. PREMIS is a data dictionary for preservation metadata describing Objects, Agents, Rights and Events. The RXP uses METS for structural metadata describing the package and uses PREMIS Events for representing digital provenance information for the package itself. The TIPR partners felt that it would not be a large burden to expect providers of preservation services for the cultural heritage community to understand METS syntax and PREMIS semantics.

Representations

In PREMIS terminology, a representation is a set of files required to assemble a renderable version of a digital object. For example, if the object in question is a (simple) Web page, the representation might consist of an HTML file, a style sheet, and a couple of image files, all of which have to be present to render the page in a browser. Should one or more of the files become unusable in their original form (for example, imagine that browsers no longer support HTML and require XHTML universally) a migrated version of that file may be created. In our example, we would now have two representations each containing four files: the original which includes an HTML file, and the new one which substitutes an XHTML file instead.

There is great variation in how preservation systems handle representations. Some discard the original and keep only the current representation, while others keep all representations or some meaningful subset. Some keep all representations in the same AIP, while still others will create a new AIP for each new representation. The TIPR partners could not make any assumptions about how representations would be treated, but had the requirement that the target repository must be able to discern how many distinct representations are contained with a package, and which of these was considered by the sending repository to be the current or active representation.

Provenance

The most important tool for maintaining the authenticity of archived content is a contemporaneous record of the chain of custody and the change history of the object. This is called digital provenance, and in the PREMIS data dictionary it is recorded as Events that are linked to associated Objects and Agents. When a SIP is ingested by a repository and turned into an AIP, the repository should record digital provenance information along with other metadata. When an AIP is exported, transformed, and sent to another repository for ingest, not only must the digital provenance recorded by the originating repository be maintained, but the change of custody and any changes made to the packages during the transfer must also be recorded and preserved. The transfer format had to make it possible to maintain an unbroken record of digital provenance at both the package and the file levels.
Coding

The Coding Phase required that each of the three TIPR partners make their repositories able to export and ingest a package in RXP format. In all cases the partners decided to write utilities external to their repository application that would a) take an RXP and convert it to the native SIP format required by the repository, and b) take a native DIP exported from the repository and convert it to RXP format. This was found to suffice for shallow ingest. NYU and Cornell found that coding specific to the source repository would be needed for deep ingest of a repository’s content.

FCLA and NYU were able to complete the coding required during the first year of the project. Cornell was delayed by that institution’s decision to migrate from aDORe to a Fedora-based repository. Cornell finished code to export an RXP from the aDORe repository, but never did manage to create one from the Fedora repository. The RXP specification changed as issues arose and were resolved, so code changes were ongoing throughout the project.

Schematron schema were written to validate RXP documents. Schematron is a rule-based validation language that uses pattern matching to make assertions about XML trees. It provides a more flexible way of expressing constraints than other validation schema. Schematron “assert” elements allow you to confirm that an XML document conforms to the schema. “Report” elements can be used to notify you of some feature of the document contents. The Schematron files also had to be modified to keep up with the changing RXP specification.

There was some discussion whether the RXP specification should reference canonical Schematron files or whether the specification should detail everything. This issue should be familiar to anyone experienced with metadata schemes expressed as XML schema. The specification was ultimately preferred as it provided a place to explain the reasoning behind various choices.

During the course of the project it became clear that the RXP specification alone would not be sufficient to ensure interoperability, and that in fact exchange partners would have to agree among themselves whether and how to use certain elements, as well as other details related to a specific transfer, such as the transfer protocol. An RXP element defined as optional in the specification could be mandatory for a particular bilateral transfer. We decided to provide a set of “base” Schematron files such that repositories could create their own versions that inherit from the base. Requirements in the RXP specification are treated as assertions in the base Schematron, and optional items in the specification are treated as report elements.

Testing

Partners agreed to use BagIt as the packaging format for file transfer because it is very simple and allows an arbitrary number of packages to be transmitted at once. FCLA wrote a simple tool in Ruby to bag up RXPs.

Transfer tests were iterative and repeated as needed. A normal pattern was for partners to conduct a transfer test that raised errors, issues or questions that would be discussed at the next bi-weekly conference call. Many issues were determined to lie within the provenance of the inter-repository
agreement and required only that partners agree on a particular practice for the purpose of the exchange. Others led to changes in the specification, which then had to be propagated to the Schematron files and the repositories' import and export programs. The test would then be repeated until all partners were satisfied.

The initial tests were simple bilateral transfers from one repository to another, considered to be successful if the receiving repository could ingest the package without loss of critical information. A second set of tests concerned bilateral round-trip transfers, from repository A to B and back to A. In this case repository A would compare its original AIP with the new AIP. The third set of tests, called ring transfers, were supposed to test what would happen when a package was sent from A to B to C. These were more difficult to achieve as Cornell’s Fedora repository could not import or export RXPs. However, an RXP created from Cornell’s original aDORe repository was used to initiate a ring transfer passing from Cornell to FCLA to NYU.

A final set of tests used packages deliberately altered to contain certain errors, the purpose being to see whether the receiving repository could discover and identify the errors and report them back to the originating repository.

**Dissemination**

The accomplishments of the TIPR project will only be useful if the RXP specification is known in the community and either adopted as is or used as the basis of further work that will be adopted for repository-to-repository transfers. Ideally, developers of repository applications will create software to transform their native DIPs to RXPs, and to ingest externally created RXPs. TIPR partners felt it would be premature to try develop the RXP as a formal standard until there is consensus in the community that the need for a repository-to-repository transfer standard exists, and that the RXP is a reasonable way to address that need. Therefore a lot of effort was devoted to publicizing the project and specifically to explicating use cases for transfer.

Dissemination efforts began the first year of the project with the establishment of the project website. In May 2009, Joseph Pawletko gave an overview of the project at the Digital Library Federation’s Spring Forum. In March 2010 the project was invited to participate in a NIST workshop on Digital Preservation Interoperability Framework. The goal of the workshop was to establish a US national roadmap on long-term digital preservation standardization by identifying requirements, technologies, and best practices. The US roadmap would feed into the Study Group on Digital Content Management and Protection (SGDCMP) of the Joint Technical Committee (JTC 1) of the International Organization for Standardization and the International Electrotechnical Commission (ISO/IEC). The SGDCMP is charged with standardizing a digital preservation interoperability framework for effectively and reliably access the preserved digital contents between interoperable digital preservation repositories. That August Joseph Pawletko was invited to present about TIPR to an SGDCMP meeting in Shenzhen, China.

Dissemination efforts escalated in 2010 and 2011 with more conference presentations and publications in journals and conference proceedings. Speaking at conferences was particularly useful as it also offered a channel for feedback from likely future users of the RXP. One major criticism we heard was that descriptive metadata, which was not included in early drafts of the RXP, must be allowed in the root level METS document. This led to our changing the specification.
A list of all publications and presentations is given in the Outputs section of this report and is also on the project website.

**Outputs**

The primary deliverables of the project are the RXP specification, a framework for inter-repository service agreements, the project website, this report, and papers and presentations prepared to disseminate information about the project. Secondary deliverables include registered METS profiles for the RXP, sample RXP files, Schematron validation files and scripts, and a Ruby tool for bagging RXPs for transfer.

**The RXP Specification**

The Repository eXchange Package specification describes a set of files which together constitute the RXP. The minimal RXP structure is shown below in Figure 1.

![RXP Minimal Structure](image)

Figure 1. RXP minimal structure.

The OBJID attribute in the root METS element contains a URI which uniquely identifies the RXP itself, as opposed to the AIP content it contains. The METS header points to an agent element which identifies the sending repository and the RXP version in use.

All of the files in the source AIP are in the /files/ directory. At least 5 constructed files, not part of the AIP, must also be included in the RXP package. The files rxp.xml, rxp-digiprov.xml, and rxp-rights.xml describe the RXP package itself. The files rxp-rep-n.xml and rxp-rep-n-digiprov.xml describe each representation in the /files/ directory, where “n” is a unique integer (in character form) equal to or
greater than “1”. Since all packages must contain at least one representation, the files rxp-rep-1.xml and rxp-rep-1-digiprov.xml are required.

The file rxp.xml is a METS document which describes the package structure and references all of the constructed files in the package, but none of the files in the /files/ directory. Each representation is referenced by a div within the structMap. Embedded divs may have an optional attribute ORDER to indicate the order of representations when there are multiple representations. Although use of ORDER is recommended, its meaning is up to the partners in a transfer to define. It may indicate successive versioning, for example, or fixed characteristics (e.g. ORDER=“2" designates a normalized version). Exactly one representation must be flagged as “active” with the TYPE attribute. The precise meaning of active is also left for bilateral agreement.

The file rxp-digiprov.xml is a PREMIS document which describes the current and all previous dissemination events pertaining to this AIP.

Each representation description file (rxp-rep-n.xml) is paired with representation provenance file (rxp-rep-n-digiprov.xml). Flocat elements in the fileSec of the representation description have an OWNERID attribute that should match the URI identifier of the file in the representation provenance. OWNERID serves as a link between the location of a file and its provenance. If digital object identifiers differ in the source and target repositories, the representation provenance file must contain “alias" events providing the tie between the two identifiers.

The RXP is described completely in the RXP specification and in three registered METS profiles. An annotated set of sample RXP files is provided as a concrete example of how the specification could be implemented. Schematron files are also provided for validating RXP METS and PREMIS documents.

**The Inter-Repository Agreement**

From its conception, TIPR was a practically-oriented project, looking for solutions that could work in the real world and not just in a research or sponsored environment. One of the issues that plagues all standards developers is the tradeoff between flexibility and interoperability. On the one hand, the more invariable a standard is, the more it facilitates interoperability. On the other hand, to gain wide adoption of a standard, prospective implementers must feel that their individual needs can be met, which often boils down to allowing options in how the standard is implemented. Some standards like METS and Dublin Core allow so much variability that communities of users have had to develop profiles, additional sets of rules constraining use of the standard, to facilitate interoperability within the community.

The TIPR partners tried to err on the side of interoperability in most cases, making the specification more rather than less proscriptive. However, some variability had to be allowed to accommodate the known diversity of preservation repositories. The project assumed that for repository-to-repository transfer to be successful, transfer partners would have to agree to a shared set of rules which at a minimum would document which options were used where options were allowed.

As transfer testing proceeded, it became clear that repositories would have to agree on far more than rules for the RXP. For example, the entire mechanism or protocol for transferring the RXP from one site
to another lay entirely outside of the RXP itself. Over time the TIPR partners developed guidelines for what we believe must be documented in an inter-repository service agreement. These include:

- details of RXP composition by the source repository in this particular transfer, documenting the options chosen where the RXP specification allows options;
- how the RXP will be transferred from the source to the target repository;
- actions to be performed by the target repository on receipt of the RXP;
- archiving and preservation treatment of the ingested RXP by the target repository;
- rights and permissions agreed upon by the source and target repositories;
- financial arrangements between source and target repositories;
- legal aspects of the arrangement.

Suggested guidelines for drafting an Inter-Repository Service Agreement is provided on the project website.

The Project Website

The project website at wiki.fcla.edu:8000/TIPR was the active wiki for communication among the partners as the project progressed. These “partner pages” are not well organized and are not intended for public consumption but are left in place for whatever historical value they provide.

Outward facing pages include a list of links to all deliverables, including the RXP specification; a list of links to publications and presentations about the TIPR project; and grant-related information including the original project proposal, and interim and final reports.

FCLA intends to keep the project website available indefinitely, although the platform and/or the URL address may change. A stable PURL to the website, http://purl.fcla.edu/purl/TIPR, should always resolve to the correct location.

Papers and Presentations


Priscilla Caplan, Joseph Pawletko, "Towards Interoperable Preservation Repositories," U.S. Workshop on


**Evaluation**

The proposal to the IMLS for the TIPR project recognized that true outcomes of a successful project would be long term. It stated:

If these results have good take-up within the preservation community, future preservation repositories will be demonstrably more interoperable and better positioned to be certified as trustworthy. At a data roughly three years from the conclusion of this project, one would expect to see:

- the transfer format described by this project serving as a standard (formal or *de facto*) for the repository-to-repository exchange of information packages;
- any recommended changes to PREMIS and/or METS incorporated into the published versions of those standards;
- a majority of the most commonly-used repository systems developing or planning to develop transfer functionality based on this model;
- a small but growing number of operational repositories with succession plans involving transfer of content to other repositories; and
- at least one future project funded to build on the work of this project, as this has built on earlier work.

At this time only the second point is verifiably in process. Some small changes to PREMIS have already been included in the standard, and a major change requested by the project, the ability to use PREMIS to describe an Intellectual Entity, will be included in version 3.0 of the PREMIS Data Dictionary, expected to be released in the spring of 2012. We suspect that the third point, the incorporation of RXP capability into currently used repository applications, may require direct negotiations with the developers of these systems, something the TIPR partners may not be in a position to do. However, one very popular system, Archivematica, has publicly indicated interest in the RXP.

The project proposal indicated that short term evaluation would focus on whether the promised deliverables were successfully achieved. These deliverables and their status are noted here.
1. Testing of all use cases defined for the project. Use cases were given particular attention, as noted in the Analysis section above. Transfer testing was successfully completed, as noted under Testing.

2. Completion of all dissemination steps [listed in the grant proposal]. The project website went online very early in the project, and a press release was sent to preservation-oriented discussion lists. Project briefings were presented at the Digital Library Federation Forum, Open Repositories, the 2009 and 2010 International Conference on Digital Preservation (iPRES), and IS&T Archiving 2011. Papers on the project were accepted and published in *D-Lib Magazine* and the *International Journal of Digital Curation*. In addition, papers and presentations were done for NIST and the ISO/IEC JTC-1 Study Group on Digital Content Management and Protection.

3. Schema definition and documentation for the transfer format, possibly as a registered METS profile. Since the RXP uses existing schema (METS and PREMIS) no new XML schema were required, but Schematron validation schema were written and made available on GitHub, a major dissemination site for open source software. Three METS profiles were registered and are available on the Library of Congress’ METS website, [http://www.loc.gov/standards/mets/mets-profiles.html](http://www.loc.gov/standards/mets/mets-profiles.html).

4. Formal recommendations to the PREMIS Editorial Committee and the METS Editorial Board. Recommendations to the METS editorial board were not required. In May 2009 we took a request to the PREMIS Editorial Committee (PEC) to consider extending the PREMIS entity model to include intellectual entities in order to enable application of PREMIS Rights and Events information to intellectual entities. The PEC found that other implementers, including the British Library, shared this need. Version 3.0 of the PREMIS Data Dictionary will make Intellectual Entities a first level Object type, along with representations, files, and bitstreams.

5. A complete, format project report, describing goals, methodology and results.

6. Documentation of issues impeding interoperability, and recommendations (where possible) on how these can be addressed. To the large part this deliverable morphed into the outline for the Inter-Repository Service Agreement, which not only shows not only where agreement on options for RXP implementation is required but also what conditions outside the scope of the RXP need to be decided upon.

**Future Work**

The first, most important step in future work is for institutions running and/or developing repository systems to try to produce and process RXP-format packages using the RXP specification and other tools available from the TIPR project. This would demonstrate whether the documentation produced by the TIPR project was sufficient for use, and would extend testing beyond repository applications and institutional needs of the three TIPR partners.

A second activity would be to directly engage developers of repository systems (vendors of commercial systems, developers of open source systems, institutions with locally-developed preservation systems). These parties should understand the use cases for repository-to-repository transfer, and could be convinced to start assessing the demand from their users for implementing this capability.
A third useful step would be to see if current standards bodies (e.g. NISO) or activities (e.g. the MOIMS (Mission Operations Information Management Services) Repository Audit and Certification Working Group) would be interested in initiating an activity around repository-to-repository transfer involving the RXP and (potentially) other proposed approaches.