NISO RP-23-2015

Protocol for Exchanging Serial Content (PESC)

Approved May 14, 2015

A Recommended Practice of the
National Information Standards Organization

About NISO Recommended Practices

A NISO Recommended Practice is a recommended "best practice" or "guideline" for methods, materials, or practices in order to give guidance to the user. Such documents usually represent a leading edge, exceptional model, or proven industry practice. All elements of Recommended Practices are discretionary and may be used as stated or modified by the user to meet specific needs.

This recommended practice may be revised or withdrawn at any time. For current information on the status of this publication contact the NISO office or visit the NISO website (www.niso.org).

Published by
National Information Standards Organization (NISO)
3600 Clipper Mill Road, Suite 302
Baltimore, MD 21211
www.niso.org

ISBN: 978-1-937522-66-7

Contents

Foreword

About this Recommended Practice

Many different organizations, such as libraries, archives, indexing services, content aggregators, publishers, and content creators exchange and work with the diverse digital files that comprise serial content. There are many reasons for copies of serial content to be transferred from organization to organization, and even within a single organization, many times during the lifecycle of the content. When exchanging content, the files that comprise a serial "publication" are packaged together in some manner and these packages can be highly variable.

There are a number of elements that may be present in the exchange of serial content. These include:

- The content itself, represented by digital files
- Descriptive metadata describing the intellectual content of the files
- Structural metadata describing how the files are related to one another
- A manifest describing what files should be present when more than one file is sent
- The package that holds all of the information described above

These elements can be combined and delivered in many different ways. For example, members of the NISO Protocol for Exchanging Serial Content Working Group have received or delivered such content in the following ways:

- an e-mail with multiple attachments (an XML file along with 3 JPEG files) that represent a single manuscript according to the text of the e-mail;
- a ZIP file that contains hundreds of separate files that may or may not be accompanied by metadata or a manifest file that explains their relationships;
- a tar file that represents an entire issue of content with directory structures that implicitly define relationships between files; and
- an EPUB, comprised of HTML, XML, and image files.

In addition to the variety of delivery methods and packaging, the packages themselves may contain a handful of files for a single issue or over a million files for a full journal title backfile.

If the amount of material being exchanged is small, or the exchange happens rarely, this variation might not be a problem, but for organizations that exchange large quantities of serial content on an ongoing basis, this variation leads to frustrations and inefficiencies.

To address these issues, NISO approved the formation of a Working Group in June 2013 with the charge to:

- Examine current practice and see what strategies are currently used in the community. Depending on the results of this exploration the Working Group might find that it is possible to either adopt a current method wholesale or to make changes to a current method to better address the specific needs of serial content, rather than creating a new method from scratch.
- Develop a Recommended Practice for the specifications of packaging that can be used for archiving and exchanging digital files related to periodic publications.

The guidelines in this document provide the recommendations developed by the PESC Working Group on the best way to manage all of the elements of serial content packaging in a manner that aids both the content provider and the content recipient in understanding what has been delivered and received.

NISO CCM Topic Committee Members

The Content and Collection Management (CCM) Topic Committee had the following members at the time it approved this Recommended Practice:

Marti Heyman, Co-chair
Cengage Learning

Betty Landesman, Co-chair
University of Baltimore

Éva Bolkovac
Yale University Library

Eric Childress
OCLC Research

Marjorie M.K. Hlava
Access Innovations / Data Harmony

Rebecca Kennison
K|N Consultants

Amy Kirchhoff
ITHAKA/JSTOR/Portico

Robert Klingenberger
Yale University Libraries

NISO PESC Working Group Members

The following individuals served on the NISO Protocol for Exchanging Serial Content (PESC) Working Group, which developed and approved this Recommended Practice:

Jean-Baptiste Bertrand
Chargé de système d'information documentaire - Cléo

Mark Donoghue
IEEE Publishing Technology

Beth Friedman
Data Conversion Laboratory, Inc.

Leslie Johnston, Co-chair
National Archives and Records Administration

Laurie Kaplan
SerialsSolutions/ProQuest

Amy Kirchhoff
ITHAKA/JSTOR/Portico

Andrea Kudzia (through January 2014)
Cengage

Jennifer Melinn (through December 2013)
Wolters Kluwer

Oliver Pesch
EBSCO

Laura Randall (from June 2014)
National Library of Medicine

Gabriele Schaefer (from February 2014)
Springer

Henning Schönenberger (through January 2014)
Springer

David Schott (through February 2014)
Copyright Clearance Center, Inc.

Linda Sussman
Cold Spring Harbor Laboratory Press

Lauren Syer (through February 2014)
Harvard University

Kimberly A. Tryka, Co-chair
National Institute of Standards and Technology

Stephen Want
U.S. Copyright Office

Elizabeth Windsor
Project MUSE

Amy Wood
Center for Research Libraries

Wei Zhao
OCUL/Scholars Portal

Acknowledgements

The PESC Working Group wishes to acknowledge those outside the formal working group membership who contributed to this effort:

Marti Heyman (Cengage Learning), Nettie Lagace (NISO), and Cynthia Hodgson (NISO)

Trademarks, Service Marks

Section 1: Introduction

1.1 Purpose and Scope

Serial publications represent a diverse content space ranging from popular magazines to scholarly journals, from content that is image-based to content that is text-based, from publications that have new content daily (even hourly) to those that might have new content only every few years. The manner in which the content creator packages the content for any particular serial publication may not match the needs of the recipient of that content and, given the proliferation of serial content, this creates a highly chaotic environment with different partners delivering significantly different looking content packages and making it difficult for all parties in the environment to work with content deliveries in a cost-efficient and effective way.

The recommendations in this document offer guidance to members of the scholarly communication community on preferred practices for the packaging and exchange of serial content that will enable the automation of processes to receive and manage serial content at scale. By following these practices, organizations can make it clear what content has been transmitted, how it is organized, and what processing is required when a new package is received.

1.2 Terms and Definitions

The following terms, as used in this recommended practice, have the meanings indicated.

Term	Definition
actors	Sending and receiving parties (see Appendix B.2).
BagIt	A hierarchical file packaging format designed to support disk-based or network-based storage and transfer of arbitrary digital content. *Source*: BagIt specification at: http://tools.ietf.org/html/draft-kunze-bagit-10
checksum	A short piece of text that is algorithmically derived from a file and allows the determination of whether the file has been corrupted.
conformance	Compliance with the guidelines of this recommended practice. *Note:* Conformance is defined at different levels.
descriptive metadata	Metadata that describes the intellectual content of a file, as opposed to the physical or structural characteristics of the file.
digital object identifier DOI®	A unique, persistent, and actionable digital identifier for a content object. *Note:* The DOI syntax is defined in ANSI/NISO Z39.84 and the entire DOI system is defined in ISO 26324.
full-text XML	Bibliographic metadata plus the full text of the article is included in an XML file.
ISSN	International Standard Serial Number, an eight digit number assigned to a continuing resource, according to the standard ISO 3297.

Term	Definition
manifest	Detailed listing of the items contained in a package.
metadata	A set of data that describes and gives information about other data; for example, data describing the full text of an article.
package	An aggregation of files being sent to another actor. *Note:* In the Open Archival Information System (OAIS), this package is often the Submission Information Package (SIP).
rendition	Electronic encoding of content. *Note:* Any individual article may have one or more renditions. Sometimes these are provided by the content provider, though sometimes they must be created by the recipient. For example, the following are all different renditions of the same content: • A PDF file of page images of the article • A set of TIFF images of each page of the article • A content owner-provided HTML file of the article • An HTML file that is created on the fly from full-text XML provided by the content owner
serial	A continuing resource issued in a succession of discrete issues or parts, usually bearing numbering, that has no predetermined conclusion. *Examples*: Journals, magazines, electronic journals, ongoing directories, annual reports, newspapers, monographic series, and also those journals, magazines, and newsletters of limited duration that otherwise bear all the characteristics of serials (e.g., newsletter of an event). *Source*: ISO 3297
serial item	The smallest self-contained portion of a serial. *Examples*: Articles, letters, reviews, case studies, front matter (in contrast to volumes or issues)
structural metadata	Metadata that is used to illustrate the relationships between a series of files and that allows the reconstruction of a whole from its parts.
tar	A file format that groups many files together while retaining their relationships within the file system.
use cases	Scenarios in which content is exchanged between actors.
ZIP	A file format that allows many files to be grouped together and compressed.

Section 2: Recommendations

2.1 Package

The package includes all of the content being sent along with other information needed to understand those files, as described in 2.1.1 through 2.1.4. Minimally, it must include content files and a manifest file. It may also include descriptive and structural metadata.

2.1.1 Package Content

The package may contain three types of information:

- Manifest – The manifest document describes the package and its content. It is required for all Conformance Levels (see 2.3).
- Content Files – Examples of content files are: full text of a serial item (PDF or XML), images, supplementary materials, metadata, and cited references.
- Metadata Files – Metadata files describe the content items being delivered. This is required only at Conformance Level 2 (see 2.3.3).

2.1.2 The Manifest Document

A manifest document must be contained in all packages and should be named "manifest" with the appropriate file extension for the format used. The manifest will include information related to the package as a whole and will list the files included (as well as their directory structure if there is one), with information about the files corresponding to the Conformance Level of the package (see 2.3).

The encoding of the manifest.txt file should be clearly stated. If using the BagIt packaging this would be stated in the bagit.txt file. If using XML the encoding is assumed to be UTF-8 unless explicitly stated otherwise in the XML declaration.

2.1.3 Package Structure

It is helpful for package recipients to have sufficient information in the package structure and file naming conventions to identify the set of serial items being delivered. To this end, the files within the package should be organized into a folder structure (see Figure 1) that represents the logical organization of the content as follows:

- Every package should have a manifest file that appears in the root directory of the package. It should be named "manifest" and have the proper extension for its format. (For example, `manifest.txt` or `manifest.xml`.)
- If multiple serial items are in the package, the contents of each item shall appear in a separate folder. To ensure unique folder names within a package, the folder names can be derived from the item's DOI® (or other identifier).
- If multiple sets of serial items (for example, journal issues) are included in a single package, each set shall be represented by a separate folder containing the serial item folders. In general, it is advisable for the hierarchy of the serial publication to be represented in the folder structure. (For example, an issue folder might be named with the journal ISSN, publication year, volume number, and issue number, and for an article in a journal without issues or volume, then the directory structure and file naming would contain the ISSN and

publication year). If multiple issues are included in the package, each issue shall be represented by a separate folder and should represent the volume and issue of the material being sent. When "ahead of print" materials are exchanged for serials that use volume/issue enumeration, it is recommended to group the "ahead of print" materials using volume/issue designation of `0`.

- If multiple journals are included in the package, each journal shall be represented by a separate folder with that journal's issues and articles appearing in sub-folders. The folder name for a journal should be the ISSN or other unique identifier of that journal as used in the package.

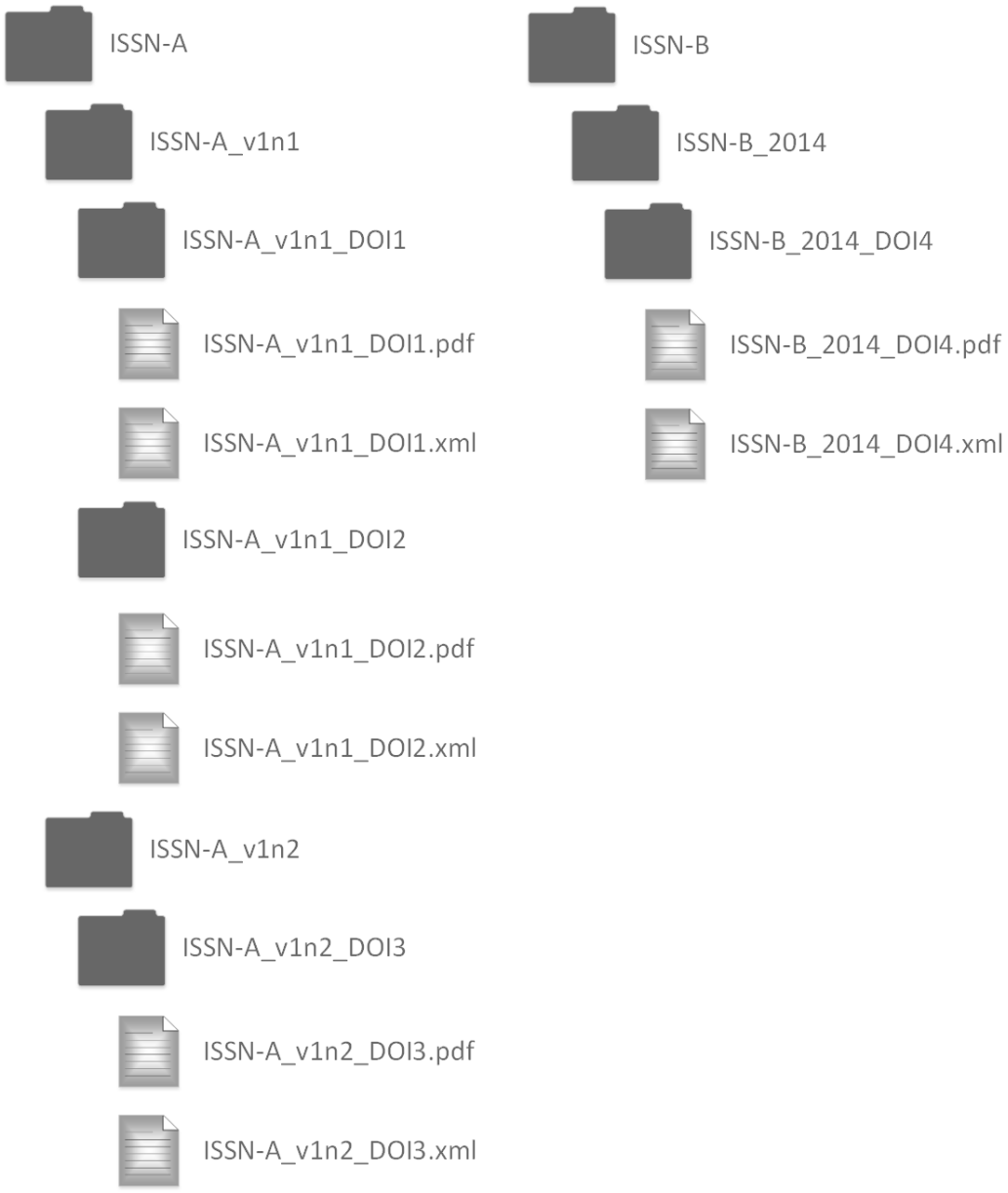

Figure 1: Example schematic of package structure

2.1.4 Package Format

The content being exchanged should be packaged using ZIP, tar, BagIt, or other commonly used formats. A suitable format will:

- Allow multiple files to be grouped within a single container (or file, for example a ZIP file) that will be used in the exchange
- Support a folder or directory structure so files can be logically grouped within the package
- Be mutually agreeable to both parties involved in the exchange

In the experiences of the PESC Working Group members, a package that contains no more than 100 articles is advisable.

2.2 Exchanging Packages

Organizations that receive large numbers of files from large numbers of partners over a prolonged period of time are likely to set up FTP (or SFTP) accounts for their partners to transmit the files. Other common transmission methods are rsync or OAI-ORE. These particular methods ensure that packages are sent to a pre-determined destination that can be automatically monitored for the arrival of new content or, in the case of OAI-ORE, packages can be fetched.

In the case of small packages sent on a one-off basis, it might be possible that both partners would consent to e-mail exchange rather than bother with the creation of FTP accounts. In general, this is not advisable as it is very easy for content to get lost in e-mail accounts and traceability is very difficult.

In rare instances where a partner might be sending very large amounts of content, such that the use of the regular FTP protocol would result in days of transfer, it might become practical to look into either high-speed file transfer software (such as Aspera) or shipping hard drives. This would need to be negotiated between the two organizations, including the selection of hard drives.

2.3 Conformance Levels

Level 1 should be considered the default level of conformance. Level 0 will allow adequate exchange between organizations and could be the best choice if one, or both, partners in the exchange don't have a workflow capable of easily producing or consuming the information in Level 1. Level 2 is only likely to be used in cases where both parties agree that a greater depth of information exchange will be of mutual benefit.

Examples of all Conformance Levels, and the package file structure they correspond to, are shown in Appendix A.

2.3.1 Conformance Level 0

For a package to conform to Level 0 it must contain the content files being sent as well as the following information in a distinct manifest document.

For the entire package:

- The Conformance Level of the package (0, 1, or 2)
- Creation date of the package

- The default purpose of the package (suggested values are new, replace, version, delete) – This can be superseded for individual items.
- Sender name and contact information
- What serial items are included

For each serial item:

- The files that belong to it

For each file:

- Its relative location in the package, i.e., the directory or folder structure
- The purpose of the item, if it differs from the default purpose of the package
- A unique identifier – A DOI or other globally unique identifier is preferred
  If a DOI is not available, use a proprietary ID that uniquely identifies work or item. The unique identifier facilitates effective processing of replace, version, and delete actions on items in subsequent packages. If using an identifier "type" tag (see examples in Appendix A), it is recommended to use the list of types from the Journal Article and Interchange Tag Library for the <pub-id-type> attribute (see http://jats.nlm.nih.gov/archiving/tag-library/1.1d1/?attr=pub-id-type).

2.3.2 Conformance Level 1

Conformance to Level 1 requires all the information required for Level 0 (content files and manifest file) with the additional information as noted below added to the manifest file.

For each individual file:

- Media type (in accordance with RFC 6838, *Media Type Specifications and Registration Procedures*)
- Checksum and checksum type (for example, an MD5 or SHA-1 cryptographic hash)
- Relationship to other files in the package. This is indicated by a <role> element in the manifest. It is expected that parties involved in an exchange of data will collaborate on defining appropriate roles. A list of suggested roles, as used in the sample files, is given below:
  - component: figure graphic
  - component: formula graphic
  - component: marked up header
  - component: media
  - component: other
  - component: other graphic
  - component: page images
  - component: supplemental file
  - component: table graphic
  - data: marked up

- rendition: multimedia
- rendition: page images
- rendition: web
- text: marked up full text
- text: marked up header
- text: plain

2.3.3 Conformance Level 2

Conformance to Level 2 requires all the information required for Level 1 with the addition for each serial item of:

- A metadata file contained in the package – This file can either contain the metadata or point to another location from which the metadata is publicly accessible.
- A file contained in the package that holds publication information – Again the file can either contain the information or point to another location from which the publication information is accessible.

2.4 Notes on Conformance Levels

It is possible to be compliant to one conformance level while still including some of the additional information from a higher level. Adding only some information from a higher conformance level, though, does not take the package to this higher level. For example, if a file contained all the information to make it compliant with Level 0 and then added checksum information (not required until Level 1), that file would still be considered conformant to level 0, unless *all* of the Level 1 information were included.

Section 3: Guidelines for Effective Implementation

This section offers additional guidance for maximizing the effectiveness of the Recommended Practices in Section 2:.

3.1 Material Inclusion/Exclusion

Packages conforming to this Recommended Practice should contain all of the material pertinent to the serial items and no extraneous material. Files referenced in the manifest must be present and all files called out by the files within the packages should also be present.

Material that is not pertinent to the serial items should not be captured in the package. This may include, but is not limited to, system-generated files like Windows-generated `thumbs.db` files, Mac-generated `__MACOSX` folders, or files supplemental to serial items not enumerated in the manifest.

3.2 Validation

If the packages contain XML data, whether for the serial item itself or supporting the serial item, that data should be valid XML. In cases where the XML data does not declare a schema, the XML should be well-formed.

3.3 File Naming

In an attempt to minimize difficulty with multiple systems and platforms handling the same content, care should be taken in file naming.

- **Characters** – Avoid elements such as spaces and punctuation in file names for all files inside the package and the package file itself. File names will ideally be limited to ASCII characters. Should non-ASCII characters be required, the encoding used must be communicated with and agreed to by the involved parties. Regardless of the encoding used, the file names must be accurately represented in the manifest file.
- **Uniqueness** – File names should be unique within and among packages; replace, version, and delete items should use IDs consistent with their previous iterations. Even in cases where packages contain multi-level hierarchies and uniqueness is not required, it should still be observed. Because the sender has no control over how the receiver handles the package and cannot guarantee that the hierarchy is preserved, it is important that file names be unique within a package.

3.4 Consistency

As with any project for which automated handling is a goal, consistency is fundamental. For the packaging of serial content, consistency applies to several different aspects.

3.4.1 Consistency in File Naming

File naming should be consistent both within and across deliveries. Once naming schemes are established and agreed upon by the sending and receiving parties, those schemes should not change. Should the need for a change to the scheme arise, all parties should communicate and agree to the change before the change is implemented.

3.4.2 Consistency of Delivered File Types

The type of package (ZIP, tar, etc.) should be consistent across deliveries.

3.4.3 Consistency with Serial Item Identifiers

The type of serial item identifiers used within and across the packages should be consistent. There will be cases where the identifier will vary according to the serial item type (a research article may have a DOI but the cover may not), however the type of serial item identifier used in each case should be consistent and understood by the involved parties.

Appendix A: Examples of Packages

A.1 One Full Issue

A publisher sends one full issue at a time. The issue is full text with PDF files for each article. In addition, each article may have figure graphic images and supplemental files.

A.1.1 Package Content

The content of the package conformant to Level 0 or Level 1 could be represented:

As a simple Zip package	**As a BagIt package**[1]
ZIP file: XYZv.18i4-08161021-1469331.zip ɬFile: manifest.xml ɬDir: ISSN-A_v1n1 ɬDir: ISSN-A_v1n1_DOI1 ɬ Dir: graphic ɬFile: DOI1.fig0001t.jpg ɬFile: DOI1.fig0001f.jpg ɬFile: DOI1.fig0002t.gif ɬFile: DOI1.fig0002f.jpg ɬ File: DOI1.pdf ɬ File: DOI1.xml ɬDir: ISSN-A_v1n1_DOI2 ɬ Dir: suppl ɬFile: DOI2.suppl00001.pdf ɬFile: DOI2.suppl00002.pdf ɬ File: DOI2.pdf ɬ File: DOI2.xml ɬDir: ISSN-A_v1n1_DOI3 ɬ Dir: graphic ɬFile: DOI3.fig0001f.jpg ɬ File: DOI3.pdf ɬ File: DOI3.xml ɬDir: ISSN-B_2014 ɬDir: ISSN-B_2014_DOI4 ɬ Dir: suppl ɬFile: DOI4.suppl00001.pdf ɬ File: DOI4.pdf ɬ File: DOI4.xml	Bag ZIP: XYZv18i4.20140916.zip ɬFile: bagit.txt ɬFile: manifest.txt ɬDir: data ɬDir: ISSN-A_v1n1 ɬDir: ISSN-A_v1n1_DOI1 ɬ Dir: graphic ɬFile: DOI1.fig0001t.jpg ɬFile: DOI1.fig0001f.jpg ɬFile: DOI1.fig0002t.gif ɬFile: DOI1.fig0002f.jpg ɬ File: DOI1.pdf ɬ File: DOI1.xml ɬDir: ISSN-A_v1n1_DOI2 ɬ Dir: suppl ɬFile: DOI2.suppl00001.pdf ɬFile: DOI2.suppl00002.pdf ɬ File: DOI2.pdf ɬ File: DOI2.xml ɬDir: ISSN-A_v1n1_DOI3 ɬ Dir: graphic ɬFile: DOI3.fig0001f.jpg ɬ File: DOI3.pdf ɬ File: DOI3.xml ɬDir: ISSN-B_2014 ɬDir: ISSN-B_2014_DOI4 ɬ Dir: suppl ɬFile: DOI4.suppl00001.pdf ɬ File: DOI4.pdf ɬ File: DOI4.xml

[1] Note that it is not technically possible to create a Level 0 conformant package in BagIt as BagIt requires checksums for all files.

A.1.2 Manifest – XML Version

The XML manifest for such a package would be as follows, depending on the conformance level:

A.1.2.1 Manifest Level 0

```
<manifest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="pescConform0-element.xsd">
    <package_info>
        <conformance>0</conformance>
        <created>2015-01-26</created>
        <default_update_state>new</default_update_state>
        <sender>
            <name>Jane Smith</name>
            <email>jsmith@pubx.com</email>
            <organization>Publisher X</organization>
        </sender>
        <recipient>
            <name>Fred Jones</name>
            <email>fredj@archive.org</email>
            <organization>Archive Y</organization>
        </recipient>
    </package_info>
    <container>
        <item>
            <file>1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/1741-
427X_v2014_10.1155%2f2014%2f982913.xml</file>
            <file>1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/1741-
427X_v2014_10.1155%2f2014%2f982913.pdf</file>
            <file>1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/graphic/1741-
427X_v2014_10.1155%2f2014%2f982913_001.tif</file>
            <file>1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/graphic/1741-
427X_v2014_10.1155%2f2014%2f982913_002.tif</file>
            <file>1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/graphic/1741-
427X_v2014_10.1155%2f2014%2f982913_003.tif</file>
            <file>1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/graphic/1741-
427X_v2014_10.1155%2f2014%2f982913_004.tif</file>
            <file>1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/graphic/1741-
427X_v2014_10.1155%2f2014%2f982913_005.tif</file>
        </item>
        <item>
            <update_state>replace</update_state>
            <file>1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982914/1741-
427X_v2014_10.1155%2f2014%2f982914.xml</file>
            <file>1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982914/1741-
427X_v2014_10.1155%2f2014%2f982914.pdf</file>
        </item>
    </container>
</manifest>
```

A.1.2.2 Manifest Level 1

```
<manifest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="pescConform1-element.xsd">
    <package_info>
        <conformance>0</conformance>
        <created>2015-01-26</created>
        <id>982913_20150126</id>
        <default_update_state>new</default_update_state>
        <sender>
            <name>Jane Smith</name>
            <email>jsmith@pubx.com</email>
            <organization>Publisher X</organization>
        </sender>
        <recipient>
            <name>Fred Jones</name>
            <email>fredj@archive.org</email>
            <organization>Archive Y</organization>
        </recipient>
    </package_info>
  <container>
   <item>
    <identifier>
    <type>doi</type>
    <value>10.1155/2014/982913</value>
    </identifier>
    <file>
        <loc>1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/1741-
427X_v2014_10.1155%2f2014%2f982913.xml</loc>
        <mime_type>text/xml</mime_type>
        <role>text: marked up full text</role>
        <checksum_type>sha512</checksum_type>
        <checksum_value>1e824a69eb88434d814ea19b0ea…</checksum_value>
    </file>
    <file>
        <loc>1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/1741-
427X_v2014_10.1155%2f2014%2f982913.pdf</loc>
        <mime_type>application/pdf</mime_type>
        <role>rendition: page images</role>
        <checksum_type>sha512</checksum_type>
        <checksum_value>6f84b71bac71890fe519d81e0a23f134e31a…</checksum_value>
    </file>
    <file>
        <loc>1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/graphic/1741-
427X_v2014_10.1155%2f2014%2f982913_001.tif</loc>
        <mime_type>image/tiff</mime_type>
```

<role>component: figure graphic hires**</role>**

<checksum_type>sha512**</checksum_type>**

<checksum_value>c0e3063080ef66ac4cc1b070ab93d4c6b0a…**</checksum_value>**

</file>

<file>

<loc>1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/graphic/1741-427X_v2014_10.1155%2f2014%2f982913_002.tif**</loc>**

<mime_type>image/tiff**</mime_type>**

<role>component: figure graphic hires**</role>**

<checksum_type>sha512**</checksum_type>**

<checksum_value>b25eb0eb98b283b898701ab6e81981d4981…**</checksum_value>**

</file>

<file>

<loc>1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/graphic/1741-427X_v2014_10.1155%2f2014%2f982913_003.tif**</loc>**

<mime_type>image/tiff**</mime_type>**

<role>component: figure graphic hires**</role>**

<checksum_type>sha512**</checksum_type>**

<checksum_value>9d73b89bca32859a81da9faeb590b93bc9729e…**</checksum_value>**

</file>

<file>

<loc>1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/graphic/1741-427X_v2014_10.1155%2f2014%2f982913_004.tif**</loc>**

<mime_type>image/tiff**</mime_type>**

<role>component: figure graphic hires**</role>**

<checksum_type>sha512**</checksum_type>**

<checksum_value>16c1e314446c05c0ca44807d51f2c335414471e…**</checksum_value>**

</file>

<file>

<loc>1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/graphic/1741-427X_v2014_10.1155%2f2014%2f982913_005.tif**</loc>**

<mime_type>image/tiff**</mime_type>**

<role>component: figure graphic hires**</role>**

<checksum_type>sha512**</checksum_type>**

<checksum_value>09827f4eeb51e4deaf39992fe06598bfe6cf16c0…**</checksum_value>**

</file>

</item>

<item>

<identifier>

<type>doi**</type>**

<value>10.1155/2014/982914**</value>**

</identifier>

<update_state>replace**</update_state>**

<file>

<loc>1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982914/1741-427X_v2014_10.1155%2f2014%2f982914.xml**</loc>**

<mime_type>text/xml**</mime_type>**

```
        <role>text: marked up full text</role>
        <checksum_type>sha512</checksum_type>
        <checksum_value>1e824a69eb88434d814ea19b0ea…</checksum_value>
    </file>
    <file>
        <loc>1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982914/suppl/1741-
427X_v2014_10.1155%2f2014%2f982914-suppl.pdf</loc>
        <mime_type>application/pdf</mime_type>
        <role>component: supplemental file</role>
        <checksum_type>sha512</checksum_type>
        <checksum_value>6f84b71bac71890fe519d81e0a23f134e31a…</checksum_value>
    </file>
    <file>
        <loc>1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982914/1741-
427X_v2014_10.1155%2f2014%2f982914-sup.pdf</loc>
        <mime_type>application/pdf</mime_type>
        <role>rendition: page images</role>
        <checksum_type>sha512</checksum_type>
        <checksum_value>6f84b71bac71890fe519d81e0a23f134e31a…</checksum_value>
    </file>
  </item>
 </container>
</manifest>
```

A.1.3 Manifest – Text Version (for use with BagIt)

A.1.3.1 Manifest files for Level 1

bagit.txt

```
Source-Organization: Publisher X
Contact-Name: Jane Smith
Contact-Email: jsmith@pubx.com
PESC-Conformance: 1
Bagging-Date: 2014-09-16
PESC-Update-State: new
External-Identifier: 982913_20150126
Recipient-Organization: Archive Y
Recipient-Name: Fred Jones
Recipient-Email: fredj@archive.org
```

manifest.txt

```
item: doi=10.1155%2f2014%2f982913
[CHECKSUM] 1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/graphic/1741-
427X_v2014_10.1155%2f2014%2f982913_001.tif mime_type= image/tiff role="component: figure
graphic hires"
[CHECKSUM] 1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/graphic/1741-
427X_v2014_10.1155%2f2014%2f982913_002.tif mime_type= image/tiff role="component: figure
graphic hires"
```

```
[CHECKSUM] 1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/graphic/1741-
427X_v2014_10.1155%2f2014%2f982913_003.tif mime_type= image/tiff role="component: figure
graphic hires"
[CHECKSUM] 1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/graphic/1741-
427X_v2014_10.1155%2f2014%2f982913_005.tif mime_type= image/tiff role="component: figure
graphic hires"
[CHECKSUM] 1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/graphic/1741-
427X_v2014_10.1155%2f2014%2f982913_001.tif mime_type= image/tiff role="component: figure
graphic hires"
[CHECKSUM] 1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/1741-
427X_v2014_10.1155%2f2014%2f982913.pdf mime_type=application/pdf role="rendition: page
images"
[CHECKSUM] 1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982913/1741-
427X_v2014_10.1155%2f2014%2f982913.xml mime_type=text/xml role="text: marked up full text"
item: doi=10.1155%2f2014%2f982914
[CHECKSUM] 1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982914/suppl/1741-
427X_v2014_10.1155%2f2014%2f982914-suppl.pdf mime_type=application/pdf role="component:
supplemental file"
[CHECKSUM] 1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982914/1741-
427X_v2014_10.1155%2f2014%2f982914.pdf  mime_type=application/pdf role="rendition: page
images"
[CHECKSUM] 1741-427X_v2014/1741-427X_v2014_10.1155%2f2014%2f982914/1741-
427X_v2014_10.1155%2f2014%2f982914.xml  mime_type=text/xml role="text: marked up full
text"
```

(Note that there is one file per line and the first value on each line is the checksum for the file.)

A.1.4 Conformance Level 2

For a package to be conformant to Level 2 there would be at least one of the following files included at the level of the manifest file: 1) a metadata file (named "`metadata`" with the appropriate extension) or 2) a file containing publication information (named "`pub-info`" with the appropriate extension). There is no specification for what these files include although it is recommended that they adhere to a recognized standard; for example the metadata file might follow the NISO JATS or Dublin Core standards.

Appendix B: Additional Detail on Use Cases

B.1 Introduction

The Use Cases subgroup was tasked with enumerating and describing scenarios for serial data exchange. The group identified the various institutional actors in the exchange process and characterized two broad categories of exchange: primarily full text or primarily metadata. The group then worked through the combinations of actors and broad categories, examining which actors were likely to participate in that exchange and why.

It should be noted that these are descriptions of potential exchanges and are not intended to represent a requirement that all, or indeed any, of the described transactions must occur between partners. And not all identified data types must be included in a given transaction.

Additionally, although a distinction is made between different actors and different exchange scenarios to refine the problem of exchange, in reality the problem of exchange generally collapses to the general case of one actor sending some type of content to another actor.

B.2 Actors

The following list describes the actors involved in the exchange of serial content. Some organizations may operate in multiple capacities (e.g., as a full-text aggregation service as well as a discovery service) and some organizations that belong to one category (library) might function in a particular scenario as described in another category (archive). Organizations reading this should look at the list from the point of view of the function they are carrying out with respect to serial content.

- **Publisher/Prepress Vendor –** The organization that is responsible for the original creation and tagging of the content. In some ways this is the starting point of the publishing flow. Examples of prepress vendors include Techset, Dartmouth Journal Services (DJS), and TnQ.
- **Publisher Host –** An organization that provides online access to the serial content. This could be the publisher itself or an organization such as Project MUSE®, HighWire, or Atypon®.
- **Discovery Service/Search Engine –** An organization providing a software-based service that receives metadata about, and often full text of, serial publications that is indexed to provide in-depth searching ability and relevance-ranked results in a publisher and aggregator agnostic way for users conducting research. Examples include EBSCO Discovery Service™, ProQuest® Summon® Service, Ex Libris® Primo®, and Google Scholar.
- **Full-Text Aggregator – A**n organization that receives content—often including full text, images, video, and/or music—and repackages that content to make it searchable and presentable to users conducting research. Examples include EBSCOhost®, ProQuest®, and Gale.
- **Archiving Service –** An organization that receives and stores serial content for archival purposes. Examples include Portico®, PubMed Central, institutional repositories, and national libraries.
- **Abstracting & Indexing (A&I) Service/Database Host –** An organization that creates A&I data or hosts the A&I data created by other organizations. Examples include Innodata®, GreenPoint, Special Libraries Cataloguing (A&I service), Gale (A&I database), MLA (A&I service and database).

- **Library –** Generally, the end customer who receives serial metadata or full content as a product to be used by its patrons. For example, a university library receiving e-journal content through a subscription or licensed service.
- **Conversion Vendor –** An organization that creates full-text XML, as well as image, metadata, and other files, from input sent from other organizations (such as a publisher or archiving service). Examples include Data Conversion Library (DCL™) and Apex CoVantage.

B.3 Scenarios

The scenarios listed below were those used to help understand the large universe in which the exchange of serial content occurs. Two broad use cases were identified: 1) the exchange of full-text content and 2) the exchange of metadata (and potentially full-text content) for discovery.

For each broad case, the general scenario and any variations that were considered are listed.

B.3.1 Full-Text Content Exchanged

1) Send newly released content for the full issue of a single journal.
    a. Send "early release" or "publish-ahead-of-print (PAP)" papers for an unassigned issue.
    b. Send "early release" or "publish-ahead-of-print (PAP)" papers for an assigned issue.
    c. Send newly released content for a single journal – individual articles or continuous publication (with or without volumes or issues).
    d. Send newly released content for a single journal – multiple issues.
    e. Send newly released content for multiple journals – multiple issues.
2) Send corrected or updated content (full text) for a journal – individual articles.
    a. Send updated content for "early release" or "publish-ahead-of-print (PAP)" for an unassigned issue.
    b. Send updated content for "early release" or "publish-ahead-of-print (PAP)" papers for an assigned issue.
    c. Send corrected or updated content (full text) for a journal – complete issue.
3) Send deletion notifications for serial items previously delivered.
    a. Send deletion notification for a complete issue.
    b. Send deletion notifications for articles previously delivered.
    c. Send deletion notification for a component of a previously delivered article.

B.3.2 Article Metadata Exchanged, With or Without Full Text, for Use in Discovery Only

1) Send new metadata about released content for a single journal (full issue) to search engines and A&I services – no full text.
2) Send new metadata about released content for a single journal (full issue) to search engines and A&I services – full text included for indexing.
    a. Send newly released content for a single journal – early release of articles.
    b. Send newly released content for a single journal – individual articles.
    c. Send newly released content for a single journal – multiple issues.
    d. Send newly released content for multiple journals and/or multiple issues.
    e. Send newly digitized content for older issues of a journal – backfile.
3) Send corrections/updates to previously delivered article metadata – no full text.
    a. Send replacement metadata for articles of record for "early version of article."
    b. Send corrected or updated article metadata – complete issue.
4) Send corrections/updates to previously sent article metadata – full text included for indexing.
    a. Send corrected or updated article metadata for a journal with full text included for indexing – complete issue.
5) Send deletion requests for previously delivered serial item metadata.
    a. Send deletion notification for a complete issue.
    b. Send deletion requests for previously delivered article metadata.
    c. Send deletion notification for component metadata of a previously delivered article.

Table 1 summarizes common exchanges of content between different actors. The table, like the scenarios above, is used for illustration, not to definitively enumerate any potential exchange that could happen between actors; actors are always able to negotiate what information is exchanged.

Table 1: Scenarios for sending of serials-related data between actors

to→ from ↓	Publisher / Prepress Vendor	Publisher Host	Discovery / Search Engine	Full-Text Aggregator	Archiving Service	A&I Service / Database Host	Library	Conversion Vendor
Publisher / Prepress Vendor		FT/MD	FT/MD	FT/MD	FT/MD	FT/MD	FT/MD	
Publisher Host			MD	MD	FT/MD	FT/MD	FT/MD	FT
Full-Text Aggregator			MD		FT/MD	MD		
Archiving Service		FT/MD	MD		FT/MD			
A&I Service / Database Host			MD			MD		
Conversion Vendor	FT/MD	FT/MD		FT/MD	FT/MD		FT/MD	

Appendix C: How to Apply for ISSN and DOI

The content of this section is taken from NISO RP-16-2013, *Recommended Practices for the Presentation and Identification of E-Journals (PIE-J)*.

C.1 ISSN

ISSN (International Standard Serial Number) is the mostly widely used international identifier for serials and similar publications. The scope of ISSN includes journals, magazines, newsletters, annuals, and resources such as ongoing databases and directories, whether in print or digital form. The ISSN uniquely identifies a publication and is a crucial identifier in systems and services such as link resolvers, electronic resource management systems, article databases, journal aggregations, and library catalogs. The ISSN is used by publishers, libraries, subscription agencies, postal services, rights management agencies, archives, digitizers, library union catalogs, and a host of others.

Some key points about the ISSN follow. More complete information can be found at the ISSN International Centre website (http://www.issn.org).

- The acronym 'ISSN' is both the singular and the plural form.
- ISSN are assigned by 88 national ISSN centers worldwide, coordinated by the ISSN International Centre located in Paris. Each national ISSN center assigns ISSN to continuing resources published in its own country. The ISSN International Centre assigns ISSN to international continuing resources and resources published in countries without an ISSN center.
- Applicants should apply to the appropriate ISSN center as determined by their country of publication. If multiple countries are involved, apply to the country considered the primary place of publication. If there is no ISSN center in the country of publication, apply to the ISSN International Centre. For serials from multinational publishers, ISSN assignment is governed by agreements between the national centers (see: http://www.issn.org/2-22649-Multinational-publishers.php for the list of agreements).
- An ISSN can usually be assigned prior to the publication of the first issue or any time thereafter. Ensure that publication information (e.g., title, imprint, projected publication date) is firm before submitting an application for a pre-publication ISSN.
- There is no charge to obtain an ISSN.
- A separate ISSN is required for each medium version of a title, e.g., an ISSN for the print version, a different ISSN for the online version, and another ISSN for the CD-ROM version.
- ISSN should be displayed prominently on the resource. If the publication is available in more than one version, e.g., print and online, both ISSN should be displayed on each version with appropriate labels so they can be distinguished, e.g., ISSN XXXX-XXXX (Print); ISSN YYYY-YYYY (Online).
- ISSN-L (the Linking ISSN) enables collocation or linking among the different medium versions of a continuing resource. The ISSN-L is one of the ISSN assigned to the different medium versions of a resource that has been designated to also group the medium-specific ISSN together. Only one ISSN-L is designated regardless of how many different medium versions of a continuing resource exist. A continuing resource is associated with only one ISSN-L. ISSN-L does not group earlier and later titles of the same resource. The relevant

ISSN-L can also be displayed on each medium version, along with the ISSN of each medium version, e.g., ISSN-L YYYY-YYYY (Linking ISSN).

- ISSN can be assigned to ceased titles whether they are in print or digital format. This enables ISSN to be available for use in archiving and digitization projects.
- ISSN can be requested either by the primary publisher or by other organizations needing a unique serial identifier for use in projects. Contact the appropriate ISSN center to discuss projects requiring large numbers of ISSN.
- If a title undergoes a major change as determined by the responsible ISSN center, then according to ISSN policy, a new ISSN is assigned. There are some minor changes to titles, such as changes in initial articles or prepositions, where no new ISSN are needed. Other changes to titles that might seem minor, such as changes to certain kinds of subtitles, can require new ISSN because library rules consider the changed words to be part of the main title. Publishers should contact their ISSN center in all cases for a determination of when a new ISSN is needed. Any time new ISSN are assigned for changed titles, current and former ISSN are linked in library and ISSN databases.
- All ISSN assignments are accompanied by metadata that includes the serial's dates of publication; place and publisher (current and former); variant, earlier, and later titles; and related medium versions.
- ISSN and ISSN metadata are included in the ISSN International Centre's ISSN Register, a database that is available as various products: the ISSN Portal, an online web subscription product, data files, an OAI-PMH web service, and other products.
- ISSN interoperates with many other metadata and identification standards. For example, an ISSN or ISSN-L can be embedded into a DOI or an OpenURL. See the ISSN international standard, ISO 3297, for examples.

C.2 CrossRef and DOI

CrossRef is an independent membership organization and Digital Object Identifier (DOI®) Registration Agency that enables collaborative services for scholarly publishers. CrossRef DOIs are at the heart of a cross-publisher citation linking network spanning a variety of content types including journals, books, conference proceedings, standards, and technical reports.

CrossRef is a member of the International DOI Foundation (IDF), a not-for-profit membership organization that is the governance and management body for the federation of Registration Agencies providing DOI services and registration, and is the registration authority for the ISO standard (ISO 26324) for the DOI system. (http://www.doi.org/) The other registration agencies include Airiti, Inc., China National Knowledge Infrastructure (CNKI), DataCite, Entertainment Identifier Registry (EIDR), The institute of Scientific and Technical Information of China (ISTIC), Japan Link Center (JaL C), Multilingual European DOI Registration Agency (mEDRA), and Publications Office of the European Union (OP).

For registration with CrossRef, member publishers submit basic citation metadata with each DOI they want to register. Each CrossRef DOI must link to a response page containing bibliographic information for the DOI being resolved. A CrossRef title-level DOI does not identify a journal as a whole. Instead, it identifies and creates a durable link to a journal title page, ensuring persistent access to information about the title.

For complete information about CrossRef and DOI, see the CrossRef website: http://crossref.org.

Title-Level DOI Best Practices

CrossRef DOIs have been registered for over 20,000 journals across many disciplines. CrossRef publishers are not required to create DOIs for journal titles but are strongly encouraged to do so. There are no deposit fees for title-level DOIs. Title-level DOI best practices are as follows:

- A distinct DOI should be created for each version of a title deposited with CrossRef. Any changes requiring a new ISSN should result in a new title-level DOI as well.
- A title-level DOI should resolve to a response page that displays the same title and ISSN recorded in the CrossRef database.
- Once assigned, a title-level DOI should be maintained.
- Responsibility for maintaining a title-level DOI transfers to the new owner when title ownership is transferred.

CrossRef Title and ISSN Management

Titles in the CrossRef system are created from publisher metadata with the first deposit of a journal. ISSN is used as the primary identifier of titles within the CrossRef system—the metadata submitted for a journal title DOI is limited to the journal title, title abbreviation(s), ISSN, and the journal URL. The publisher determines the exact title and ISSN included in the deposit. At least one ISSN is required for each journal deposited with CrossRef, but publishers are encouraged to deposit all ISSN available for a title. (CrossRef does not currently support the submission of ISSN-L in deposit metadata. ISSN-L can be added as "alternate ISSN" by CrossRef staff. Alternate ISSN are used to facilitate query matching but are not included in metadata requests.)

Title and ISSN combinations are not verified with an external agency. A check digit validation is performed on every ISSN submitted in a deposit. Once a title or ISSN is introduced into the CrossRef system, a new publication with the same title or ISSN cannot be created without CrossRef intervention. Publishers requesting significant changes to a title are instructed to request a new ISSN from the appropriate agency.

Appendix D: Related Standards and Recommended Practices

This section covers existing standards, best practices, and technologies that are related to or discussed in this Recommended Practice. The information covered in this section is by no means exhaustive.

D.1 File Formats

D.1.1 BagIt

BagIt is a file packaging format that uses hierarchical structures. The structures described by the BagIt specification can be used to create packages compliant with the practices outlined in this document.

The BagIt File Packaging Format (V0.97). IETF Internet-Draft. Internet Engineering Task Force, January 28, 2014. http://tools.ietf.org/html/draft-kunze-bagit-10

D.1.2 Tar

Tar was originally developed as a format for tape archiving (thus the name), but has since developed into a format for archiving and distributing multiple files, directories, and objects in a single file package.

TAR(5) manual [webpage]. https://github.com/libarchive/libarchive/wiki/ManPageTar5

D.1.3 ZIP

ZIP is both a file packaging format and a lossless compression program, which has been released into the public domain.

.ZIP File Format Specification. Application Note, version 6.3.3. PKWare, Inc., September 1, 2012. https://www.pkware.com/documents/casestudies/APPNOTE.TXT

D.2 File Transfer

Aspera [website]. http://asperasoft.com/

Open Archives Initiative Object Reuse and Exchange (OAI-ORE) Specification – Abstract Data Model. Open Archives Initiative, October 17, 2008. Available at: http://www.openarchives.org/ore/1.0/datamodel

Rsync [website]. http://rsync.samba.org/

D.3 Checksum

Microsoft File Checksum Integrity Verifier. http://www.microsoft.com/en-us/download/details.aspx?id=11533

MD5 & SHA Checksum Utility 2.0. http://raylin.wordpress.com/

D.4 Identifiers

D.4.1 DOI® (Digital Object Identifier)

A DOI is a unique, persistent, and actionable digital identifier for a content object. Underlying the DOI is an infrastructure "system" to ensure the persistence of the identifier. DOIs can be used as the serial item identifier in the file packages discussed in this Recommended Practice. See Appendix C.2 for more information on obtaining and using a DOI.

Syntax for the Digital Object Identifier. ANSI/NISO Z39.84-2005 (R2010). Bethesda, MD: National Information Standards Organization, September 30, 2005, reaffirmed May 13, 2010. http://www.niso.org/standards/z39-84-2005/

Information and documentation — Digital object identifier system. ISO 26324:2012. Geneva: International Organization for Standardization, April 23, 2012. http://www.iso.org/iso/catalogue_detail.htm?csnumber=43506

D.4.2 ISSN

The International Standard Serial Number (ISSN) uniquely identifies serials and is useful in identifying the particular serial to which the serial items being delivered belong. See Appendix C.1 for more information on obtaining and using an ISSN.

General information about ISSNs can be found at http://www.issn.org.

Information and documentation – International standard serial number (ISSN). ISO 3297:2007. Geneva: International Organization for Standardization, 2007. http://www.iso.org/iso/catalogue_detail?csnumber=39601

D.5 Metadata

The standards and specifications listed in this section are all methods that can be used to describe the serial metadata required for the various levels of conformance with the practices outlined in this document.

D.5.1 Dublin Core

Dublin Core Metadata Element Set. Version 1.1. Dublin Core Metadata Initiative, June 14, 2012. http://www.dublincore.org/documents/dces/
Also issued as ANSI/NISO Z39.85 (http://www.niso.org/standards/z39-85-2012) and ISO 15836 (http://www.iso.org/iso/catalogue_detail?csnumber=52142).

D.5.2 JATS

JATS: Journal Article Tag Suite. ANSI/NISO Z39.96-2012. Baltimore, MD: National Information Standards Organization, August 9, 2012. http://www.niso.org/standards/z39-96-2012/

JATS: Journal Article Tag Suite Supporting Documentation [website]. http://jats.nlm.nih.gov/

D.5.3 MARC

MARC Standards. Library of Congress. http://www.loc.gov/marc

D.5.4 Media Types

Media Type Specifications and Registration Procedures. RFC 6838. Internet Engineering Task Force, January 2013. http://www.rfc-editor.org/rfc/rfc6838.txt
Note: Previously referred to as MIME Type.

D.5.5 ONIX for Serials

ONIX for Serials. EDItEUR. http://www.editeur.org/17/ONIX-for-Serials/

D.6 Other Related Standards and Recommended Practices

Open Archival Information System (OAIS). Recommended Practice CCSDS 650.0-M-2. Washington, DC: Consultative Committee for Space Data Systems, June 2012. Available from: http://public.ccsds.org/publications/archive/650x0m2.pdf
Also issued as: ISO 14721:2012 (http://www.iso.org/iso/catalogue_detail.htm?csnumber=57284)

PIE-J: The Presentation & Identification of E-Journals. NISO RP-16-2013. Baltimore, MD: National Information Standards Organization, March 2013. Available from: http://www.niso.org/publications/rp/rp-16-2013

Recommended Format Specifications, Washington, DC: Library of Congress. Available from: http://www.loc.gov/preservation/resources/rfs/

Appendix E: FAQ

These are the Frequently Asked Questions (FAQ) that were available at the time of publication of the Recommended Practice. An up-to-date version of the FAQ can be found at http://www.niso.org/workrooms/pesc/.

1. **How do I indicate additional files, such as supplementary materials, in the manifest?**

   Supplementary material files would be listed alongside any other files in your manifest. For example, in the xml version of a Conformance Level 0 manifest you might have something like:

```
<container>
    <item>
        <file> folder1/folder2/article.xml</file>
        <file>folder1/folder2/article.pdf</file>
        <file>folder1/folder2/image1.tif</file>
        <file>folder1/folder2/image1.jpg</file>
        <file>folder1/folder2/image2.tif</file>
        <file>folder1/folder2/image2.jpg</file>
        <file>folder1/folder2/sup-mat-1.doc</file>
    </item>
</container>
```

   For a Conformance Level 1 manifest you would have the same basic structure except that the **<file>** element would contain children elements holding the file location and other required information (see the example Conformance Level 1 file in Appendix A).

2. **What if I need to have two streams of ingest for a single content item? For example, what if I want to send someone the full text and images of an article, but a supplementary data file might come to them from another source?**

   While no committee members have seen this use case at the time of writing, we believe that it is likely to turn up in the future. The PESC recommended practice should be able to handle this case; it would be a matter of making sure that the senders were using Conformance Level 1 or 2 to ensure that each individual item has a unique identifier. If that is the case, then if a second part of the content were to arrive in a separate package, the fact that it has the same identifier as a previous item would trigger the receiver to examine the package further, particularly the `<update_state>` element which would then indicate that the second arrival is meant as an addition to the first.

Appendix F:
XSD Schemas for each PESC Conformance Level

These schemas are provided for informational purposes only and are subject to change. When implementing PESC refer to the schemas, and sample files, available on line at http://www.niso.org/workrooms/pesc/.

F.1 Conformance Level 0

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="manifest">
        <xs:annotation>
            <xs:documentation>The &lt;manifest&gt; element contains the manifest for the
                package.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="package_info"/>
                <xs:element ref="container"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="package_info">
        <xs:annotation>
            <xs:documentation>The &lt;package_info&gt; element contains metadata that relates to the
                entire package being delivered.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="conformance"/>
                <xs:element ref="created"/>
                <xs:element ref="default_update_state"/>
                <xs:element ref="sender"/>
                <xs:element ref="recipient"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="conformance" type="xs:integer">
        <xs:annotation>
            <xs:documentation>The &lt;conformance&gt; element contains the level of conformance to the
                PESC recommended practice. When using this schema the value should be
                "0".</xs:documentation>
        </xs:annotation>
    </xs:element>

    <xs:element name="created" type="xs:date">
        <xs:annotation>
            <xs:documentation>The &lt;created&gt; element contains the data the package was created in the
                format YYY-MM-DD.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="sender" type="contact">
        <xs:annotation>
            <xs:documentation>The &lt;sender&gt; element contains contact information for the sender of
                the package.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="recipient" type="contact">
        <xs:annotation>
            <xs:documentation>The &lt;recipient&gt; element contains contact information for the recipient
                of the package.</xs:documentation>
        </xs:annotation>
```

```
</xs:element>
<xs:element name="default_update_state">
    <xs:annotation>
        <xs:documentation>The &lt;default_update_state&gt; element contains contains a value that
            indicates the nature of the package contents (new, replace, version, or delete). This value
            applies to all items in the package unless the element &lt;update_state&gt; gives a
            different value for particular &lt;item&gt; elements.</xs:documentation>
    </xs:annotation>
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:enumeration value="new"/>
            <xs:enumeration value="replace"/>
            <xs:enumeration value="version"/>
            <xs:enumeration value="delete"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
<xs:complexType name="contact">
    <xs:annotation>
        <xs:documentation>"contact" is a complex type that describes contact information for the
            sender and recipient of the package.</xs:documentation>
    </xs:annotation>
    <xs:sequence>
        <xs:element ref="name"/>
        <xs:element ref="email"/>
        <xs:element ref="organization"/>
    </xs:sequence>
</xs:complexType>
<xs:element name="name" type="xs:string">
    <xs:annotation>
        <xs:documentation>The &lt;name&gt; element contains the name of a contact. There are no
            constraints on how the name is formatted.</xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="email" type="email">
    <xs:annotation>
        <xs:documentation>The &lt;email&gt; element contains contains an email address of a
            contact.</xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="organization" type="xs:string">
    <xs:annotation>
        <xs:documentation>The &lt;organization&gt; element contains the name of the organization of
            the contact.</xs:documentation>
    </xs:annotation>
</xs:element>
<xs:simpleType name="email">
    <xs:restriction base="xs:string">
        <xs:pattern value="[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,6}"/>
    </xs:restriction>
</xs:simpleType>
<xs:element name="container">
    <xs:annotation>
        <xs:documentation>The &lt;container&gt; element may contain additional &lt;container&gt;
            elements or &lt;item&lt; elements. &lt;container&gt; represents a folder, or directory. By
            nesting &lt;container&gt; elements you can reproduce the nesting of
            volume/issue.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:choice>
            <xs:element maxOccurs="unbounded" ref="container"/>
            <xs:element maxOccurs="unbounded" ref="item"/>
        </xs:choice>
    </xs:complexType>
</xs:element>
<xs:element name="item">
```

```
        <xs:annotation>
            <xs:documentation>The &lt;item&gt; element contains references to the files that make up a
                single item, such as a journal article.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="update_state" minOccurs="0"/>
                <xs:element maxOccurs="unbounded" ref="file"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="update_state">
        <xs:annotation>
            <xs:documentation>The &lt;update_state&gt; element contains contains a value that indicates
                the nature of the package contents (new, replace, version, or delete). This element is
                optional; it should be used only when the &lt;update_state&gt; of an item is different than
                the &lt;default_update_state&gt; of the package.</xs:documentation>
        </xs:annotation>
        <xs:simpleType>
            <xs:restriction base="xs:string">
                <xs:enumeration value="new"/>
                <xs:enumeration value="replace"/>
                <xs:enumeration value="version"/>
                <xs:enumeration value="delete"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:element>
    <xs:element name="file" type="xs:anyURI">
        <xs:annotation>
            <xs:documentation>The &lt;file&gt; element contains the location of a particular file in the
                package.</xs:documentation>
        </xs:annotation>
    </xs:element>
</xs:schema>
```

F.2 Conformance Level 1

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="manifest">
        <xs:annotation>
            <xs:documentation>The &lt;manifest&gt; element contains the manifest for the
                package.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="package_info"/>
                <xs:element ref="container"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="package_info">
        <xs:annotation>
            <xs:documentation>The &lt;package_info&gt; element contains metadata that relates to
                the entire package being delivered.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="conformance"/>
                <xs:element ref="created"/>
                <xs:element ref="id"/>
                <xs:element ref="default_update_state"/>
                <xs:element ref="sender"/>
                <xs:element ref="recipient"/>
            </xs:sequence>
        </xs:complexType>
```

```
</xs:element>
<xs:element name="conformance" type="xs:integer">
    <xs:annotation>
        <xs:documentation>The &lt;conformance&gt; element contains the level of conformance
            to the PESC recommended practice. When using this schema the value should be
            "0".</xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="created" type="xs:date">
    <xs:annotation>
        <xs:documentation>The &lt;created&gt; element contains the data the package was
            created in the format YYY-MM-DD.</xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="id" type="xs:string">
    <xs:annotation>
        <xs:documentation>The &lt;id&gt; element contains an identifier for the
            package.</xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="sender" type="contact">
    <xs:annotation>
        <xs:documentation>The &lt;sender&gt; element contains contact information for the
            sender of the package.</xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="recipient" type="contact">
    <xs:annotation>
        <xs:documentation>The &lt;recipient&gt; element contains contact information for the
            recipient of the package.</xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="default_update_state">
    <xs:annotation>
        <xs:documentation>The &lt;default_update_state&gt; element contains contains a value
            that indicates the nature of the package contents (new, replace, version, or delete). This
            value applies to all items in the package unless the element &lt;update_state&gt; gives
            a different value for particular &lt;item&gt; elements.</xs:documentation>
    </xs:annotation>
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:enumeration value="new"/>
            <xs:enumeration value="replace"/>
            <xs:enumeration value="version"/>
            <xs:enumeration value="delete"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
<xs:complexType name="contact">
    <xs:annotation>
        <xs:documentation>"contact" is a complex type that describes contact information for the
            sender and recipient of the package.</xs:documentation>
    </xs:annotation>
    <xs:sequence>
        <xs:element ref="name"/>
        <xs:element ref="email"/>
        <xs:element ref="organization"/>
    </xs:sequence>
</xs:complexType>
<xs:element name="name" type="xs:string">
    <xs:annotation>
        <xs:documentation>The &lt;name&gt; element contains the name of a contact. There are
        no constraints on how the name is formatted.</xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="email" type="email">
```

```
    <xs:annotation>
        <xs:documentation>The &lt;email&gt; element contains contains an email address of a
            contact.</xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="organization" type="xs:string">
    <xs:annotation>
        <xs:documentation>The &lt;organization&gt; element contains the name of the
            organization of the contact.</xs:documentation>
    </xs:annotation>
</xs:element>
<xs:simpleType name="email">
    <xs:restriction base="xs:string">
        <xs:pattern value="[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,6}"/>
    </xs:restriction>
</xs:simpleType>
<xs:element name="container">
    <xs:annotation>
        <xs:documentation>The &lt;container&gt; element may contain additional
            &lt;container&gt; elements or &lt;item&lt; elements. &lt;container&gt; represents a
            folder, or directory. By nesting &lt;container&gt; elements you can reproduce the
            nesting of volume/issue.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:choice>
            <xs:element maxOccurs="unbounded" ref="container"/>
            <xs:element maxOccurs="unbounded" ref="item"/>
        </xs:choice>
    </xs:complexType>
</xs:element>
<xs:element name="item">
    <xs:annotation>
        <xs:documentation>The &lt;item&gt; element contains references to the files that make
            up a single item, such as a journal article.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="identifier"/>
            <xs:element ref="update_state" minOccurs="0"/>
            <xs:element maxOccurs="unbounded" ref="file"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="identifier">
    <xs:annotation>
        <xs:documentation>The &lt;identifier&gt; element contains an identifier for an item in a
            known system.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="type"/>
            <xs:element ref="value"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="type" type="xs:string">
    <xs:annotation>
        <xs:documentation>The &lt;type&gt; element holds the name of the system used for the
            identifier. Examples are "doi" or "pmid".</xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="value" type="xs:string">
    <xs:annotation>
        <xs:documentation>The &lt;value&gt; element holds the value of the
            identifier.</xs:documentation>
    </xs:annotation>
```

```
    </xs:element>
    <xs:element name="update_state">
        <xs:annotation>
            <xs:documentation>The &lt;update_state&gt; element contains contains a value that
                indicates the nature of the package contents (new, replace, version, or delete). This
                element is optional; it should be used only when the &lt;update_state&gt; of an item is
                different than the &lt;default_update_state&gt; of the package.</xs:documentation>
        </xs:annotation>
        <xs:simpleType>
            <xs:restriction base="xs:string">
                <xs:enumeration value="new"/>
                <xs:enumeration value="replace"/>
                <xs:enumeration value="version"/>
                <xs:enumeration value="delete"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:element>
    <xs:element name="file">
        <xs:annotation>
            <xs:documentation>The &lt;file&gt; element contains the location of a particular file in the
                package.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="loc"/>
                <xs:element ref="mime_type"/>
                <xs:element ref="role"/>
                <xs:element ref="checksum_type"/>
                <xs:element ref="checksum_value"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="loc" type="xs:anyURI">
        <xs:annotation>
            <xs:documentation>The &lt;loc&gt; element contains the location of the file in the
                package.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="mime_type" type="xs:string">
        <xs:annotation>
            <xs:documentation>The &lt;mime_type&gt; element contains the mime-type of the
                file.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="role" type="xs:string">
        <xs:annotation>
            <xs:documentation>The &lt;role&gt; element contextualizes the file in relation to other
                files in the package. Examples are "component: figure graphic web" or
                "component: figure graphic thumbnail".</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="checksum_type" type="xs:string">
        <xs:annotation>
            <xs:documentation>The &lt;checksum_type&gt; element contains the name of the
                algorithm used to generate the checksum.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="checksum_value" type="xs:string">
        <xs:annotation>
            <xs:documentation>The &lt;checksum_value&gt; element contains the value of the
                generated checksum for the file.</xs:documentation>
        </xs:annotation>
    </xs:element>
</xs:schema>
```

F.3 Conformance Level 2

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="manifest">
        <xs:annotation>
            <xs:documentation>The &lt;manifest&gt; element contains the manifest for the
                package.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="package_info"/>
                <xs:element ref="container"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="package_info">
        <xs:annotation>
            <xs:documentation>The &lt;package_info&gt; element contains metadata that relates to the
                entire package being delivered.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="conformance"/>
                <xs:element ref="created"/>
                <xs:element ref="id"/>
                <xs:element ref="default_update_state"/>
                <xs:element ref="sender"/>
                <xs:element ref="recipient"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="conformance" type="xs:integer">
        <xs:annotation>
            <xs:documentation>The &lt;conformance&gt; element contains the level of conformance to
                the PESC recommended practice. When using this schema the value should be
                "0".</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="created" type="xs:date">
        <xs:annotation>
            <xs:documentation>The &lt;created&gt; element contains the data the package was created in
                the format YYY-MM-DD.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="id" type="xs:string">
        <xs:annotation>
            <xs:documentation>The &lt;id&gt; element contains an identifier for the
                package.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="sender" type="contact">
        <xs:annotation>
            <xs:documentation>The &lt;sender&gt; element contains contact information for the sender of
                the package.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="recipient" type="contact">
        <xs:annotation>
            <xs:documentation>The &lt;recipient&gt; element contains contact information for the
                recipient of the package.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="default_update_state">
        <xs:annotation>
            <xs:documentation>The &lt;default_update_state&gt; element contains contains a value that
```

```
            indicates the nature of the package contents (new, replace, version, or delete). This value
            applies to all items in the package unless the element &lt;update_state&gt; gives a
            different value for particular &lt;item&gt; elements.</xs:documentation>
        </xs:annotation>
        <xs:simpleType>
            <xs:restriction base="xs:string">
                <xs:enumeration value="new"/>
                <xs:enumeration value="replace"/>
                <xs:enumeration value="version"/>
                <xs:enumeration value="delete"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:element>
    <xs:complexType name="contact">
        <xs:annotation>
            <xs:documentation>"contact" is a complex type that describes contact information for the
                sender and recipient of the package.</xs:documentation>
        </xs:annotation>
        <xs:sequence>
            <xs:element ref="name"/>
            <xs:element ref="email"/>
            <xs:element ref="organization"/>
        </xs:sequence>
    </xs:complexType>
    <xs:element name="name" type="xs:string">
        <xs:annotation>
            <xs:documentation>The &lt;name&gt; element contains the name of a contact. There are no
                constraints on how the name is formatted.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="email" type="email">
        <xs:annotation>
            <xs:documentation>The &lt;email&gt; element contains contains an email address of a
                contact.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="organization" type="xs:string">
        <xs:annotation>
            <xs:documentation>The &lt;organization&gt; element contains the name of the organization
                of the contact.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:simpleType name="email">
        <xs:restriction base="xs:string">
            <xs:pattern value="[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,6}"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:element name="container">
        <xs:annotation>
            <xs:documentation>The &lt;container&gt; element may contain additional &lt;container&gt;
                elements or &lt;item&lt; elements. &lt;container&gt; represents a folder, or directory. By
                nesting &lt;container&gt; elements you can reproduce the nesting of
                volume/issue.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:choice>
                <xs:element maxOccurs="unbounded" ref="container"/>
                <xs:element maxOccurs="unbounded" ref="item"/>
            </xs:choice>
        </xs:complexType>
    </xs:element>
    <xs:element name="item">
        <xs:annotation>
            <xs:documentation>The &lt;item&gt; element contains references to the files that make up a
                single item, such as a journal article.</xs:documentation>
        </xs:annotation>
```

```
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="identifier"/>
                <xs:element ref="update_state" minOccurs="0"/>
                <xs:element ref="metadata"/>
                <xs:element ref="publication"/>
                <xs:element maxOccurs="unbounded" ref="file"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="identifier">
        <xs:annotation>
            <xs:documentation>The &lt;identifier&gt; element contains an identifier for an item in a
                known system.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="type"/>
                <xs:element ref="value"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="type" type="xs:string">
        <xs:annotation>
            <xs:documentation>The &lt;type&gt; element holds the name of the system used for the
                identifier. Examples are "doi" or "pmid".</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="value" type="xs:string">
        <xs:annotation>
            <xs:documentation>The &lt;value&gt; element holds the value of the
                identifier.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="update_state">
        <xs:annotation>
            <xs:documentation>The &lt;update_state&gt; element contains contains a value that indicates
                the nature of the package contents (new, replace, version, or delete). This element is
                optional; it should be used only when the &lt;update_state&gt; of an item is different than
                the &lt;default_update_state&gt; of the package.</xs:documentation>
        </xs:annotation>
        <xs:simpleType>
            <xs:restriction base="xs:string">
                <xs:enumeration value="new"/>
                <xs:enumeration value="replace"/>
                <xs:enumeration value="version"/>
                <xs:enumeration value="delete"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:element>
    <xs:element name="file">
        <xs:annotation>
            <xs:documentation>The &lt;file&gt; element contains the location of a particular file in the
                package.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="loc"/>
                <xs:element ref="mime_type"/>
                <xs:element ref="role"/>
                <xs:element ref="checksum_type"/>
                <xs:element ref="checksum_value"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="loc" type="xs:anyURI">
```

```
        <xs:annotation>
            <xs:documentation>The &lt;loc&gt; element contains the location of the file in the
                package.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="mime_type" type="xs:string">
        <xs:annotation>
            <xs:documentation>The &lt;mime_type&gt; element contains the mime-type of the
                file.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="role" type="xs:string">
        <xs:annotation>
            <xs:documentation>The &lt;role&gt; element contextualizes the file in relation to other files
                in the package. Examples are "component: figure graphic web" or "component: figure
                graphic thumbnail".</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="checksum_type" type="xs:string">
        <xs:annotation>
            <xs:documentation>The &lt;checksum_type&gt; element contains the name of the algorithm
                used to generate the checksum.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="checksum_value" type="xs:string">
        <xs:annotation>
            <xs:documentation>The &lt;checksum_value&gt; element contains the value of the generated
                checksum for the file.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="metadata" type="xs:string">
        <xs:annotation>
            <xs:documentation>The &lt;metadata&gt; element contains the name of the file that holds
                metadata about the item.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="publication" type="xs:string">
        <xs:annotation>
            <xs:documentation>The &lt;publication&gt; element contains the name of the file that holds
                publication information about the item.</xs:documentation>
        </xs:annotation>
    </xs:element>
</xs:schema>
```