



Open Packaging Format (OPF) 2.0

INTERNAL WORKING DRAFT v0.7

December 17, 2006

Copyright © 2006 by International Digital Publishing Forum™.

All rights reserved. This work is protected under Title 17 of the United States Code. Reproduction and dissemination of this work with changes is prohibited except with the written permission of the Open eBook Forum.

TABLE OF CONTENTS

<INSERT TOC HERE>

1 Overview

1.1 Purpose and Scope

In order for electronic-book technology to achieve widespread success in the marketplace, Reading Systems must have convenient access to a large number and variety of titles. The Open Publication Structure (OPS) Specification describes a standard for representing the content of electronic publications and is meant to reduce barriers to the proliferation of content. Specifically, the specification is intended to:

- Give publication tool providers and content providers (e.g. publishers, authors, and others who have content to be displayed) and publication tool providers minimal and common guidelines that ensure fidelity, accuracy, accessibility, and adequate presentation of electronic content over various Reading Systems.
- Build on established content format standards.
- Define a standard means of content description in order for electronic books to move smoothly through the distribution chain.

This specification and the Open Packaging Format (OPF) Specification, defines the mechanism by which the various components of an OPS publication are tied together and provides additional structure and semantics to the electronic publication. Specifically, OPF:

- Describes and references all components of the electronic publication (e.g. markup files, images, navigation structures).
- Provides publication-level metadata.
- Specifies the linear reading-order of the publication.
- Provides fallback information to use when unsupported extensions to OPS are employed.

This OPF specification is separate from the OPS markup specification to modularize the described packaging methodology separate from the described content. This should help facilitate the use of this packaging technology by other standards bodies (e.g. Daisy) in non-OPS publications.

A third specification, the OEBPS Container Format (OCF) Specification, defines the standard mechanism by which all components of an electronic publication may be packaged together into a single archive for transmission, delivery and archival.

1.2 Definitions

CONTENT PROVIDER

A publisher, author, or other information provider who provides a publication to one or more Reading Systems in the form described in this specification.

DEPRECATED

A feature that is permitted, but not recommended, by this specification. Such features may be removed in future revisions. Conformant Reading Systems must support deprecated features.

INLINE XML ISLAND

An Inline XML Island is an XML document fragment using a Non-Preferred Vocabulary that exists within an XML document marked-up using a Preferred Vocabulary within an OPS publication.

OCF

The OEBPS Container Format defines a mechanism by which all components of an OPS Publication may be combined into a single file-system entity.

OEBPS

The Open eBook Publication Structure. Previous versions of this specification (OPF) and its related specification, OPS, were unified into the single OEBPS specification. For this version, OEBPS was broken into separate OPF and OPS specifications to aid modular adoption of the specifications. OEBPS 1.2 was the highest version of the previous unified specification.

OPF

The Open Packaging Format – this standard – defines the mechanism by which all components of a published work conforming to the OPS standard including metadata, reading order and navigational information are packaged into an OPS Publication.

OPF PACKAGE

An OPF Package Document that describes an OPS Publication and references all the files used by an OPS Publication. It identifies all other files in the publication and provides descriptive information about them. Defined by this specification.

OPF PACKAGE DOCUMENT

An XML file using the file extension *.opf*. The XML file may refer to other XML files via XML's general entity mechanism, but those files may not use the *.opf* file extension.

OPF PACKAGE MODULES

An XML file which does not use the extension *.opf*, nor is described in the Package, but is included into the Package using XML's general entity inclusion method. It is most often used to simplify the creation of Packages for very large documents.

OPS

The Open Publication Structure – the sister-standard to this standard – defines the markup required to construct OPS Content Documents.

OPS CONTENT DOCUMENT

An XHTML, DTBook, or out-of-line XML document that conforms to the OPS specification that may legally appear in the OPF Package **spine**.

OPS CORE MEDIA TYPE

A MIME media, defined in the OPS Specification, type that all Reading Systems must support.

OPS PUBLICATION

A collection of OPS Content Documents, an OPF Package file, and other files, typically in a variety of media types, including structured text and graphics, that constitute a cohesive unit for publication.

OUT-OF-LINE XML ISLAND

An Out-Of-Line XML Island is an XML document that exists within an OPS Publication and is not authored using a preferred vocabulary. It is an entirely separate, complete, and valid XML document.

PREFERRED VOCABULARY

XML consisting only of OPS-supported XHTML markup and/or DTBook markup.

READER

A person who reads a publication.

READING DEVICE

The physical platform (hardware and software) on which publications are rendered.

READING SYSTEM

A combination of hardware and/or software that accepts OPS Publications (likely packaged in an OCF Container) and makes them available to consumers of the content. Great variety is possible in the architecture of Reading Systems. A Reading System MAY be implemented entirely on one device, or it MAY be split among several computers. In particular, a Reading Device that is a component of a Reading System need not directly accept OPS Publications, but all Reading Systems MUST do so. Reading Systems MAY include additional processing functions, such as compression, indexing, encryption, rights management, and distribution.

XML DOCUMENT

An XML Document is a complete and valid XML document as defined in XML (<http://www.w3.org/TR/xml11/>).

XML DOCUMENT FRAGMENT

Referred to as either a document fragment or as an XML Document Fragment, as defined in Document Object Model Level 1 (<http://www.w3.org/TR/REC-DOM-Level-1/>) but with the additional requirement that they be well-formed.

XML NAMESPACES

Referred to as XML namespaces, or just namespaces, these must conform to XML Namespaces (<http://www.w3.org/TR/xml-names11/>).

XML ISLAND

An Inline XML Island or an Out-Of-Line XML island.

1.3 Relationship to Other Specifications

This specification combines subsets and applications of other specifications. Together, these facilitate the construction, organization, presentation, and unambiguous interchange of electronic documents:

1. XML 1.1 Extensible Markup Language specification (<http://www.w3.org/TR/xml11/>); and
2. XML 1.1 namespace specification (<http://www.w3.org/TR/xml-names11/>); and
3. The OPS Specification (); and,
4. XHTML 1.1 Extensible HyperText Markup Language specification

- (<http://www.w3.org/TR/xhtml11/>); and
5. Digital Talking Book (DTB) Specification (<http://www.niso.org/standards/resources/Z39-86-2005.html>); and
 6. Dublin Core metadata specification (<http://dublincore.org/documents/2004/12/20/dces/>) and the MARC relator code list (<http://www.loc.gov/marc/relators/>); and
 7. Unicode Standard, Version 4.0. Reading, Mass.: Addison-Wesley, 2003, as updated from time to time by the publication of new versions. (See <http://www.unicode.org/unicode/standard/versions> for the latest version and additional information on versions of the standard and of the Unicode Character Database); and
 8. Particular MIME media types (<http://www.ietf.org/rfc/rfc4288.txt> and <http://www.iana.org/assignments/media-types/index.html>); and
 9. Web Content Accessibility Guidelines 1.0 (<http://www.w3.org/TR/WCAG10/>); and
 10. *RFC 2119: Key words for use in RFCs to Indicate Requirement Levels.* (<http://www.ietf.org/rfc/rfc2119.txt>).

1.3.1 Relationship to XML

OPF is based on XML because of XML's generality and simplicity, and because XML documents are likely to adapt well to future technologies and uses. XML also provides well-defined rules for the syntax of documents, which decreases the cost to implementers and reduces incompatibility across systems. Further, XML is extensible: it is not tied to any particular set of element types, it supports internationalization, and it encourages document markup that can represent a document's internal parts more directly, making them amenable to automated formatting and other types of computer processing.

- Reading Systems **must** be XML processors as defined in XML 1.1. All OPF Packages **must** be valid XML documents according to the OPF package schema.

1.3.2 Relationship to XML Namespaces

Reading Systems **must** process XML namespaces according to the XML Namespaces Recommendation at <http://www.w3.org/TR/xml-names11/>.

Namespace prefixes distinguish identical names that are drawn from different XML vocabularies. An XML namespace declaration in an XML document associates a namespace prefix with a unique URI. The prefix can then be employed on element or attribute names in the document. Alternatively, a namespace declaration in an XML document may identify a URI as the default namespace, applicable to elements lacking a namespace prefix. The XML namespace prefix is separated from the suffix element or attribute name by a colon.

The namespace for the OPF Package file is <http://www.idpf.org/2007/opf>, and **must** be declared at the root of all package documents. In addition, a *version* attribute with a value of 2.0 **must** be specified on all **package** elements. A **package** element that omits the *version* attribute must be processed as an OEBPS 1.2 package.

Example:

```
<package version="2.0" xmlns="http://www.idpf.org/2007/opf">
    ...
</package>
```

1.3.3 XML Namespace Validation

Reading Systems are not required to validate according to XML Namespaces [\[LINK HERE\]](#), as the implementation details for namespace-level validation are unclear and are not supported in a uniform fashion by validation tools.

Reading Systems **must** validate the existence of the appropriate namespaces, as defined in the Relationship to XML Namespaces section, above.

1.3.4 Relationship to Dublin Core

The Dublin Core is designed to minimize the cataloging burden on authors and publishers, while providing enough metadata to be useful. This specification supports the set of Dublin Core 1.1 metadata elements (<http://dublincore.org/documents/2004/12/20/dces/>), supplemented with a small set of additional attributes addressing areas where more specific information may be useful. For example, the OPF *role* attribute added to the Dublin Core **contributor** element allows for much more detailed specification of contributors to a publication, including their roles expressed via relator codes.

Content providers **must** include a minimum set of a metadata elements, defined in section 2.2, and **should** incorporate additional metadata to enable readers to discover publications of interest.

1.3.5 Relationship to Unicode

OPF Packages **may** use the entire Unicode character set, in UTF-8 or UTF-16 encodings, as defined by Unicode (see <http://www.unicode.org/unicode/standard/versions>). The use of Unicode facilitates internationalization and multilingual documents. However, Reading Systems **are not required to** provide glyphs for all Unicode characters.

Reading Systems **must** parse all UTF-8 and UTF-16 characters properly (as required by XML). Reading Systems **may** decline to display some characters, but **must** be capable of signaling in some fashion that undisplayable characters are present. They **must not** display Unicode characters merely as if they were 8-bit characters. For example, the biohazard symbol (0x2623) need not be supported by including the correct glyph, but **must not** be parsed or displayed as if its component bytes were the two characters “&#” (0x0026 0x0023).

To aid Reading Systems in implementing consistent searching and sorting behavior it is recommended that Unicode Normalization Form C (NFC) be used (See <http://www.w3.org/TR/charmod-norm/>).

1.4 Conformance

The keywords "**must**", "**must not**", "**required**", "**shall**", "**shall not**", "**should**", "**recommended**", "**may**", and "**optional**" in this document **must** be interpreted as described in (<http://www.ietf.org/rfc/rfc2119.txt>).

This section defines conformance for OPF Package files, and Reading Systems that process those files.

1.4.1 Package Conformance

This specification defines conformance for both individual OPF Packages and for a collection of files including exactly one OPF Package Document and may include one or more OPF Package Modules. These are collectively referred to as an OPS Publication.

1.4.1.1 Package Conformance

Each conformant Package Document **must** meet these necessary conditions:

- i. it is a well-formed and valid XML document (as defined in XML 1.1);
- ii. it may consist of one or more XML files, but only one may have use the file extension *.opf*, being the OPF Package Document. Any other file used to define the Package is an OPF Package Module.

1.4.1.2 Publication Conformance

A collection of files is a conforming OPS Publication if and only if:

- i. it includes a single OPF Package Document that obeys the Package Document Requirements listed above; and
- ii. the OPF Package file includes one and only one manifest entry corresponding to each other file in the OPS Publication; and
- iii. the manifest entry for each file in the publication specifies a MIME media type for the file (see <http://www.ietf.org/rfc/rfc2046.txt>); and
- iv. each file whose manifest entry identifies it as being in one of the OPS Core Media Types conforms as defined for those MIME media types; and
- v. each file listed in the spine of the Package must conform to the OPS Content Document requirements defined in the OPS specification; and
- vi. if the publication contains one or more documents which are either DTBook or an Out-Of-Line XML Island, an NCX **must** be included; and
- vii. the **metadata** element or deprecated **dc-metadata** element contains at least one **Identifier** element, at least one **Title** element, and at least one **Language** element drawn from the Dublin Core tag set; and
- viii. the **unique-identifier** attribute of the **package** element is a correct XML IDREF to a Dublin Core **Identifier** element; and
- ix. any extended values specified for the Dublin Core Creator and **Contributor** elements' OPF *role* attribute **must** be taken from the registered MARC Relator Code list or **must** begin with "oth."; and
- x. any extended values specified for the **guide** element's *type* attribute begin with "other."; and
- xi. the *version* attribute of the **package** element is specified with a value of "2.0"; and
- xii. the *xmlns* attribute of the **package** element is specified with a value of "http://www.idpf.org/2007/opf".

1.4.2 Reading System Conformance

This specification defines conformance for a Reading System when presented with an OPS Publication. OPS Content documents have further conformance requirements that can be found in the OPS specification. A Reading System is conformant if and only if it processes documents as follows:

- A) When presented with an OPF Package file the Reading System
 - i. processes all elements and attributes as described in section 2 of this specification.
- B) When providing navigation via the OPF spine, the Reading System

- i. **must not** render content that is not an OPS Content Document.
- C) When presented with an OEBPS 1.2 Package file, the Reading System **must** process it as a conformant OEBPS 1.2 Reading System would. Note that only the Package **must** be processed as an OEBPS 1.2 Read System would, not the content documents referred to in the Package.
- D) When presented with an OEBPS 1.2 Publication, a Reading System **should** process it as a conformant OEBPS 1.2 Reading System would. Such a Reading System may claim an optional level of Reading System conformance, “Backward Compatibility Conformance”.

1.4.3 Compatibility of Version vNext

Version 2.0 of OPF is not meant to be a substantially “new” specification. However, version 2.0 does add one significant functional enhancement in addition to a number of other changes from OEBPS version 1.2. Specifically, the following are the most substantive additions:

- XML 1.1 is incorporated.
- XML namespace processing is now required.
- Support for Daisy’s Navigation Center eXtended (NCX) has been added to enhance ease of navigation and accessibility of publications.
- The leading character of Dublin Core element names **must** now be lower case to conform to Dublin Core’s XML implementation recommendations.
- The “tours” element has been deprecated.

While most changes from version OEBPS 1.2 to OPF 2.0 have been done via deprecation rather than removal of previous functionality, the OEBPS Package version 1.2 is not a fully compatible subset of OPF 2.0 (e.g. new namespace processing requirements).

1.5 Accessibility

This specification incorporates features that ensure content can be made accessible to, and usable by, persons with reading disabilities. Existing accessibility features developed by the World Wide Web Consortium (W3C) are incorporated into this specification.

Publications **should** be authored in accordance with the W3C Web Content Accessibility Guidelines 1.0 (<http://www.w3.org/TR/1999/WAI-WEBCONTENT-19990505/>) or, if it is released while the Working Group is active, the Web Content Accessibility Guidelines 2.0 (the current draft is available at <http://www.w3.org/TR/WCAG20/>) to ensure that the broadest possible set of users will have access to books delivered in this format.

In addition, recommendations from the W3C HTML 4.0 Guidelines for Mobile Access (<http://www.w3.org/TR/NOTE-html40-mobile/>) and the W3C Web Accessibility Initiative's proposed User Agent Guidelines (<http://www.w3.org/TR/WD-WAI-USERAGENT/>) should be reviewed and applied by OPF implementers to ensure that Reading Systems will be in conformance with accessibility requirements.

2 The OPF Package

A publication conforming to this specification **must** include exactly one OPF Package XML Document, which specifies the OPS Content Documents, images, and other objects that make

up the OPS Publication and how they relate to each other.

The package file **should** be named using the extension “.opf”, in order to make it readily identifiable within the group of files making up the publication. Package files are of MIME media type “application/oebps-package+xml”. This specification does not define means for physically bundling files together to make one data transfer object (such as using zip or tar); the OEBPS Container Format (OCF) specifies this functionality.

It is **not required** that the OPS Package schema be physically included in every publication. If included, it **should** be referenced by the manifest (as described below for other files).

The major parts of the OPF Package file are:

PACKAGE IDENTITY

A unique identifier for the OPS Publication as a whole.

METADATA

Publication metadata (title, author, publisher, etc.).

MANIFEST

A list of files (documents, images, style sheets, etc.) that make up the publication. The manifest also includes fallback declarations for files of types not supported by this specification.

SPINE

An arrangement of documents providing a linear reading order.

TOURS (DEPRECATED)

A set of alternate reading sequences through the publication, such as selective views for various reading purposes, reader expertise levels, etc.

GUIDE

A set of references to fundamental structural features of the publication, such as table of contents, foreword, bibliography, etc.

An OPF Package **must** be a valid XML document conforming to the OPF Package schema (Appendix A). An informal outline of the package is as follows:

```
<?xml version="1.0"?>
<!DOCTYPE package
  PUBLIC "-//ISBN 0-9673008-1-9//DTD OPF Package//EN"
  "http://www.idpf.org/dtds/2007/opf.dtd">
< version="2.0" xmlns="http://www.idpf.org/2007/opf">
  metadata
  manifest
  spine
  guide
</package>
```

[GC: Above public IDs may well need tuning.]

The following sections describe the parts of the OPF Package.

2.1 Package Identity

The **package** element is the root element in a package file; all other elements are nested within it.

The **package** must specify a value for its *unique-identifier* attribute. The *unique-identifier* attribute's value specifies which Dublin Core **identifier** element, described in section 2.2.10, provides the package's preferred, or primary, identifier. The package file's author is responsible for choosing a primary identifier that is unique to one and only one particular package (i.e., the set of files referenced from the package file's **manifest**).

Notwithstanding the requirement for uniqueness, Reading Systems **must not** fail catastrophically should they encounter two distinct packages with the same purportedly unique primary identifier.

2.2 Publication Metadata

The required **metadata** element is used to provide information about the publication as a whole. It **may** contain a Dublin Core metadata element directly or within a (now deprecated) **dc-metadata** sub-element. Supplemental metadata may also be specified directly or with a (now deprecated) **x-metadata** sub-element.

Reading Systems **must** allow the specification of the deprecated **dc-metadata** and **x-metadata** elements. Newly created OPS 2.0 packages **should not** create **dc-metadata** or **x-metadata** elements.

The required **metadata** or **dc-metadata** (deprecated) element contains specific publication-level metadata as defined by the Dublin Core Metadata Initiative (<http://dublincore.org/>). The descriptions below are included for convenience, and the Dublin Core's own definitions take precedence (see <http://dublincore.org/documents/2004/12/20/dces/>).

One or more optional instances of a **meta** element, analogous to the XHTML 1.1 **meta** element but applicable to the publication as a whole, may be placed within the **metadata** element or within the deprecated **x-metadata** element. This allows content providers to express arbitrary metadata beyond the data described by the Dublin Core specification. Individual OPS Content Documents may include the **meta** element directly (as in XHTML 1.1) for document-specific metadata. This specification uses the OPF Package file alone as the basis for expressing publication-level Dublin Core metadata.

For example:

```
<metadata xmlns:dc="http://purl.org/dc/elements/1.1/"
          xmlns:opf="http://www.idpf.org/2007/opf">
  <dc:title>Tale of Two Cities</dc:title>
  <dc:creator opf:role="aut">Charles Dickens</dc:creator>
  ...
  <meta name="price" content="USD 19.99" />
</metadata>
```

The XML namespace mechanism (see <http://www.w3.org/TR/REC-xml-names11/>) is used to identify the elements used for Dublin Core metadata without conflict. The **metadata** element **may** contain any number of instances of any Dublin Core elements. Dublin Core metadata elements may occur in any order; in fact, multiple instances of the same element type (multiple Dublin Core **creator** elements, for example) can be interspersed with other metadata elements without change of meaning.

Each Dublin Core field is represented by an element whose content is the field's value. At least one of each of Dublin Core **title**, **identifier** and **language** **must** be included in the **metadata** element. Dublin Core elements, like any other elements in the OPF Package file, **may** have an *id* attribute specified. At least one Dublin Core **identifier**, that which is referenced from the **package** *unique-identifier* attribute, **must** have an *id* specified.

Because the Dublin Core metadata fields for creator and contributor do not distinguish roles of specific contributors (such as author, editor, and illustrator), this specification adds an optional *role* attribute for this purpose. See section 2.2.6 for a discussion of *role*.

To facilitate machine processing of Dublin Core **creator** and **contributor** fields, this specification adds the optional *file-as* attribute for those elements. This attribute is used to specify a normalized form of the contents. See section 2.2.6 for a discussion of *file-as*.

This specification also adds a *scheme* attribute to the Dublin Core **identifier** element to provide a structural mechanism to separate an identifier value from the system or authority that generated or defined that identifier value. See section 2.2.10 for a discussion of *scheme*.

This specification also adds an *event* attribute to the Dublin Core **date** element to enable content providers to distinguish various publication specific dates (for example, creation, publication, modification). See section 2.2.7 for a discussion of *event*.

For example:

```
<package version="2.0" xmlns="http://www.idpf.org/2007/opf"
  unique-identifier="xyz">
  <metadata xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:opf="http://www.idpf.org/2007/opf">
    <dc:title>Alice in Wonderland</dc:title>
    <dc:language>en</dc:language>
    <dc:identifier id="xyz" opf:scheme="ISBN">
      123456789X
    </dc:identifier>
    <dc:creator opf:role="aut">Lewis Carroll</dc:creator>
  </metadata>
  ...
</package>
```

There are no attributes for the elements within **metadata** defined by Dublin Core – only the elements' contents are so defined.

The following subsections describe the individual Dublin Core metadata elements.

2.2.1 <title> </title>

The title of the publication. An OPF Package **must** include at least one instance of this element type, however multiple instances are permitted. Any Reading System that displays title metadata to the user **should** either use the first **title** only, or all **title** elements.

2.2.2 <creator> <creator>

A primary creator or author of the publication. Additional contributors whose contributions are secondary to those listed in **creator** elements **should** be named in **contributor** elements.

Publications with multiple co-authors **should** provide multiple **creator** elements, each containing one author. The order of **creator** elements is presumed to define the order in which the creators' names should be presented by the Reading System.

This specification recommends that the content of the **creator** elements hold the text for a single name as it would be presented to the Reader.

This specification adds to the **creator** element two optional attributes: *role* and *file-as*. The set of values for *role* are identical to those defined in section 2.2.6 for the **contributor** element. The *file-as* attribute **should** be used to specify a normalized form of the contents, suitable for machine processing. For example, one might find

```
<dc:creator opf:file-as="King, Martin Luther Jr." opf:role="aut">  
    Rev. Dr. Martin Luther King Jr.  
</dc:creator>
```

If a Reading System displays creator information, the Reading System **must** display the contents of all **creator** elements, in the order provided, with appropriate separating spacing and/or punctuation.

2.2.3 <subject> </subject>

Multiple instances of the **subject** element are supported, each including an arbitrary phrase or keyword. This specification makes no attempt to standardize subject naming schemes, such as the Library of Congress Subject Heading System.

2.2.4 <description> </description>

Description of the publication's content.

2.2.5 <publisher> </publisher>

The publisher as defined by the Dublin Core Metadata Element Set (<http://dublincore.org/documents/2004/12/20/dces/>).

2.2.6 <contributor> </contributor>

A party whose contribution to the publication is secondary to those named in **creator** elements.

Other than significance of contribution, the semantics of this element are identical to those of **creator**. Reading Systems are free to choose to display **creator** information without accompanying **contributor** information.

This specification adds to the **contributor** element two optional attributes: *role* and *file-as*. The *file-as* attribute is defined as for **creator**, and is documented in section 2.2.2.

The normative list of values used for the *role* attribute is defined by the MARC relator code list (<http://www.loc.gov/marc/relators/>). When roles are specified, the 3-character registered MARC values **must** be used when applicable. Although that list is extensive, other values **may** be added if a required role is not covered by those predefined values. Such values **must** begin with "oth.", and shall be considered subdivisions of the "other" relator code. Like other constructs in this specification, these values are case-sensitive and **must** be coded entirely in lower-case.

For convenience, some relator code values are listed here as examples. Consult the MARC code list cited above for the complete list.

Adapter [adp] Use for a person who 1) reworks a musical composition, usually for a different medium, or 2) rewrites novels or stories for motion pictures or other audiovisual medium.

- Annotator [ann]** Use for a person who writes manuscript annotations on a printed item.
- Arranger [arr]** Use for a person who transcribes a musical composition, usually for a different medium from that of the original; in an arrangement the musical substance remains essentially unchanged.
- Artist [art]** Use for a person (e.g., a painter) who conceives, and perhaps also implements, an original graphic design or work of art, if specific codes (e.g., [egr], [etr]) are not desired. For book illustrators, prefer *Illustrator* [ill].
- Associated name [asn]** Use as a general relator for a name associated with or found in an item or collection, or which cannot be determined to be that of a Former owner [fmo] or other designated relator indicative of provenance.
- Author [aut]** Use for a person or corporate body chiefly responsible for the intellectual or artistic content of a work. This term may also be used when more than one person or body bears such responsibility.
- Author in quotations or text extracts [aqt]** Use for a person whose work is largely quoted or extracted in a work to which he or she did not contribute directly. Such quotations are found particularly in exhibition catalogs, collections of photographs, etc.
- Author of afterword, colophon, etc. [aft]** Use for a person or corporate body responsible for an afterword, postface, colophon, etc. but who is not the chief author of a work.
- Author of introduction, etc. [aui]** Use for a person or corporate body responsible for an introduction, preface, foreword, or other critical matter, but who is not the chief author.
- Bibliographic antecedent [ant]** Use for the author responsible for a work upon which the work represented by the catalog record is based. This may be appropriate for adaptations, sequels, continuations, indexes, etc.
- Book producer [bkp]** Use for the person or firm responsible for the production of books and other print media, if specific codes (e.g., [bkd], [egr], [tyd], [prt]) are not desired.
- Collaborator [clb]** Use for a person or corporate body that takes a limited part in the elaboration of a work of another author or that brings complements (e.g., appendices, notes) to the work of another author.
- Commentator [cmm]** Use for a person who provides interpretation, analysis, or a discussion of the subject matter on a recording, motion picture, or other audiovisual medium.
- Compiler [com]** Use for a person who produces a work or publication by selecting and putting together material from the works of various persons or bodies.
- Designer [dsr]** Use for a person or organization responsible for design if specific codes (e.g., [bkd], [tyd]) are not desired.
- Editor [edt]** Use for a person who prepares for publication a work not primarily his/her own, such as by elucidating text, adding

introductory or other critical matter, or technically directing an editorial staff.

- Illustrator [ill]** Use for the person who conceives, and perhaps also implements, a design or illustration, usually to accompany a written text.
- Lyricist [lyr]** Use for the writer of the text of a song.
- Metadata contact [mdc]** Use for the person or organization primarily responsible for compiling and maintaining the original description of a metadata set (e.g., geospatial metadata set).
- Musician [mus]** Use for the person who performs music or contributes to the musical content of a work when it is not possible or desirable to identify the function more precisely.
- Narrator [nrt]** Use for the speaker who relates the particulars of an act, occurrence, or course of events.
- Other [oth]** Use for relator codes from other lists which have no equivalent in the MARC list or for terms which have not been assigned a code.
- Photographer [pht]** Use for the person or organization responsible for taking photographs, whether they are used in their original form or as reproductions.
- Printer [prt]** Use for the person or organization who prints texts, whether from type or plates.
- Redactor [red]** Use for a person who writes or develops the framework for an item without being intellectually responsible for its content.
- Reviewer [rev]** Use for a person or corporate body responsible for the review of book, motion picture, performance, etc.
- Sponsor [spn]** Use for the person or agency that issued a contract, or under whose auspices a work has been written, printed, published, etc.
- Thesis advisor [ths]** Use for the person under whose supervision a degree candidate develops and presents a thesis, memoir, or text of a dissertation.
- Transcriber [trc]** Use for a person who prepares a handwritten or typewritten copy from original material, including from dictated or orally recorded material.
- Translator [trl]** Use for a person who renders a text from one language into another, or from an older form of a language into the modern form.

2.2.7 <date> </date>

Date of publication, in the format defined by “Date and Time Formats” at <http://www.w3.org/TR/NOTE-datetime> and by ISO 8601 on which it is based. In particular, dates without times are represented in the form YYYY[-MM[-DD]]: a mandatory 4-digit year, an optional 2-digit month, and if the month is given, an optional 2-digit day of month.

The **date** element has one optional OPF *event* attribute. The set of values for *event* are not defined by this specification; possible values may include: **creation**, **publication**, and **modification**.

2.2.8 <type> </type>

Type includes terms describing general categories, functions, genres, or aggregation levels for

content. Recommended best practice is to select a value from a controlled vocabulary.

2.2.9 `<format>` `</format>`

The media type or dimensions of the resource. Best practice is to use a value from a controlled vocabulary (e.g. MIME media types).

2.2.10 `<identifier>` `</identifier>`

A string or number used to uniquely identify the resource. An OPF Package **must** include at least one instance of this element type, however multiple instances are permitted.

At least one **identifier** **must** have an *id* specified, so it can be referenced from the package **unique-identifier** attribute described in Section 2.1.

The **identifier** element has an optional OPF *scheme* attribute defined by this specification. The *scheme* attribute names the system or authority that generated or assigned the text contained within the **identifier** element, for example “ISBN” or “DOI.” The values of the *scheme* attribute are case sensitive.

This specification does not standardize or endorse any particular publication identifier scheme. Specific use of URLs or ISBNs is not yet addressed by this specification. Identifier schemes are not currently defined by Dublin Core.

2.2.11 `<source>` `</source>`

Information regarding a prior resource from which the publication was derived; see the Dublin Core Metadata Element Set (<http://dublincore.org/documents/2004/12/20/dces/>).

2.2.12 `<language>` `</language>`

Identifies a language of the intellectual content of the Publication. An OPF Package **must** include at least one instance of this element type, however multiple instances are permitted. The content of this element **must** comply with RFC 3066 (see <http://www.ietf.org/rfc/rfc3066.txt>), or its successor on the IETF Standards Track. The Dublin Core permits other descriptions as well; this specification does not.

2.2.13 `<relation>` `</relation>`

An identifier of an auxiliary resource and its relationship to the publication.

2.2.14 `<coverage>` `</coverage>`

The extent or scope of the publication’s content. Recommended best practice is to select a value from a controlled vocabulary; see the Dublin Core Metadata Element Set (<http://dublincore.org/documents/2004/12/20/dces/>).

2.2.15 `<rights>` `</rights>`

A statement about rights, or a reference to one. In this specification, the copyright notice and any further rights description **should** appear directly.

This specification does not address the manner in which a Content Provider specifies to a secure distributor any licensing terms under which readership rights or copies of the content may be sold.

2.3 Manifest

The required **manifest** provides a list of all the files that are parts of the publication. The **manifest** element **must** contain one or more **item** elements. Each **item** describes a document, an image file, a style sheet, or other component that is considered part of the publication. The manifest **must not** include **item** elements referring to the OPF Package Document or OPF Package Modules indirectly included through XML's general entity mechanism.

Each **item** element contained within a **manifest** element **must** have the attributes **id**, **href** (a URI; if relative, the URI is interpreted as relative to the package file itself), and **media-type** (specifying the item's MIME media type).

The order of **item** elements in the **manifest** is not significant.

For example,

```
<manifest>
  <item id="intro" href="introduction.html"
        media-type="application/xhtml+xml" />
  <item id="c1" href="chapter-1.html"
        media-type="application/xhtml+xml" />
  <item id="c2" href="chapter-2.html"
        media-type="application/xhtml+xml" />
  <item id="toc" href="contents.xml"
        media-type="application/xhtml+xml" />
  <item id="oview" href="arch.png"
        media-type="image/png" />
</manifest>
```

The URIs in **href** attributes of **item** elements in the **manifest** **must not** use fragment identifiers.

2.3.1 Fallback items

The OPS specification defines a set of OPS Core Media Types that all conforming Reading Systems **must** support (e.g. XHTML, PNG, SVG). For a publication that uses only those media types, the **manifest** merely lists the publication's component files directly. However, content providers **may** construct publications that reference items of additional media types. In order for such publications to be read by all conforming Reading Systems, content providers **must** provide alternative “fallback” items for each such item. For every item that is not an OPS Core Media Type, at least one of its associated fallback items **must** either be of a type drawn from the set of OPS Core Media Types or, in some cases, CSS styling **may** be provided for documents containing non-preferred XML vocabularies.

This specification and the OPS specification jointly define four different mechanisms for specifying OPS Core Media Type fallbacks. These are as follows:

1. For inline “replaced” resources referenced via the **object** element, this specification relies on that element's inherent replacement capabilities, described in section XXX of the OPS specification.
2. For inline “replaced” resources referenced via the **img** element, the text value of the *alt* attribute provides a valid fallback, described in section XXX of the OPS specification.
3. For Inline XML Islands a switch-based fallback mechanism is provided, described in section XXX of the OPS specification.

4. For non-inline destinations, whether referenced from a document or a package, and for inline “replaced” resources referenced via the **img** element, the various attributes of the package **item** element are used to provide fallback information. This is defined in this section of this specification.

For the purpose of fallback specification, the “text/xml” file that specifies the publication’s NCX (see below) should be considered a Core Media Type, thus fallback information **must not** be provided for this file.

There are two cases to be considered:

1. Non-OPS Core Media Types
2. Out-Of-Line XML Islands

2.3.1.1 *Items that are not OPS Core Media Types*

An item that specifies a resource that is not an OPS Core Media Type and is not an out-of-line XML island (e.g. a non-core image type, a text file, a PDF file). In this case, its fallback **must** be identified with a *fallback* attribute pointing to another item.

This second case is simpler. An **item** identifies a fallback item using its *fallback* attribute, which **must** specify the ID of the **item** element that identifies the fallback. Items referenced from *fallback* attributes may each specify a *fallback* attribute in turn, forming a multi-level fallback chain. For example,

```
<manifest>
  <item id="item1"
        href="FunDoc.txt"
        media-type="text/plain"
        fallback="fall1" />
  <item id="fall1" fallback="fall2"
        href="FunDoc.pdf"
        media-type="application/pdf" />
  <item id="fall2"
        href="FunDoc.html"
        media-type="application/html+xml" />
  <item ...>
</manifest>
```

If a *fallback* attribute points to an **item** that also has a *fallback* attribute, a Reading System **must** continue down the fallback chain until it reaches a reference to an **item** with a media type it can display (or as specified below, it reaches an item with an *fallback-style* attribute). A Reading System **may** continue further, and **may** display any item from the chain. In the absence of element-specific (i.e. **img** and **object**) fallback information, every item in a publication that does not have one of the OPS Core Media Types **must**, directly or indirectly, specify a fallback to an item that does have one of the OPS Core Media Types.

Fallback chains **must** terminate; circular references are not permitted. Nevertheless, Reading Systems **should not** fail catastrophically if they encounter such a loop.

2.3.1.2 *Items that are Out-Of-Line XML Islands*

An **item** that specifies a resource that is an Out-Of-Line XML Island (an XML document that is not authored in a Preferred Vocabulary). In this case, the namespace of the XML Island **must** be specified with the *island-type* attribute and its fallback **must** be identified with either a *fallback* attribute pointing to another **item** or by providing CSS styling that may be used to render the island via the *fallback-style* attribute.

In the case of an Out-Of-Line XML Island, more information is needed, and more freedom is

provided, for fallback determination. Such a document is an XML document with a media-type that is neither `application/xhtml+xml`, `application/dtbresource+xml` nor, the deprecated, `text/x-eob1-document`. The item specifying such a document must include an `fallback-style` attribute with a value containing the document's XML namespace. The fallback information for such an item **must** be specified by using a `fallback` attribute pointing to another **item** or by CSS styling that may be used to render the island via the `fallback-style` attribute.

If the `fallback` attribute is specified, Reading System processing is identical to the Non-OPS Core Media Types handling described above.

If the `fallback-style` attribute is specified, a Reading System **may** choose to process the Out-Of-Line XML Island (even though it can not natively process the vocabulary) using the stylesheet specified by the `fallback-style` attribute's value which **must** contain a reference to the id of the **item** containing and href to the island's stylesheet.

Both `fallback` and `fallback-style` attributes **may** be specified, in which case the Reading System **may** choose to either follow the fallback chain or to process the Out-Of-Line XML Island with the supplied CSS stylesheet.

Most importantly, a Reading System that can natively process the non-preferred vocabulary used for an Out-Of-Line XML Island **may** choose to use its integral understanding to natively process the document. However, fallback information **must** be provided for Reading Systems that do not have such native processing ability.

For example:

```
<manifest>
  <item id="item1"
        href="Doc1.hpy"
        media-type="text/happy+xml"
        island-type="http://happy.com/ns/happy1/"
        fallback="item2" />
  <item id="item2"
        href="Doc1.less-hpy"
        media-type="text/less-happy+xml"
        island-type=http://happy.com/ns/happy2/
        fallback="item3"
        fallback-style="css1" />
  <item id="item3"
        href="Doc1.dtb"
        media-type="application/dtbresource+xml" />
  <item id="item4"
        href="Doc2.hpy"
        media-type="text/happy+xml"
        island-type="http://happy.com/ns/happy1/"
        fallback-style="css1" />
  <item id="css1"
        href="happy.css"
        media-type="text/css" />
</manifest>
```

In the above example when processing “item1” a Reading System may choose to render item1 natively, item2 natively, item2 with only styling from css1, or item3 natively. When processing “item4” a Reading System may choose to render item4 natively or item4 with only styling from css1.

2.4 Spine

Following the **manifest**, there **must** be one **spine** element, which defines the primary linear reading order of the publication. It specifies an ordered list of one or more OPS Content Documents drawn from the manifest, using **itemref** elements contained within the **spine** element.

A publication **must** specify exactly one **spine**. Reading Systems **must** treat the file named in the first **itemref** element within the **spine** as the first file to be rendered in the reading of the publication. The successive files named in successive **itemref** elements are those that are to be rendered using “next-page”-type functionality that may be available in the Reading System.

The **spine must** refer only to **item** elements with media types of `application/xhtml+xml`, `application/dtbresource+xml`, the deprecated, `text/x-eob1-document`, or an Out-Of-Line XML Island (with required fallback). Content with other media types **may** be referenced via OPS Content Documents, which **should** provide text alternates and other information to enhance accessibility as appropriate.

The **spine** need not include references to every one of the manifest’s **item** elements that reference OPS Content Documents, because there are means other than the **spine** for accessing documents in the publication. For example, hypertext links **may** provide access to documents not in the **spine**, as **may** tours and guides (see below).

For example,

```
<manifest>
  <item id="toc"
        href="contents.html"
        media-type="application/html+xml" />
  <item id="c1"
        href="chap1.html"
        media-type="application/html+xml" />
  <item id="c2"
        href="chap2.dtb"
        media-type="application/dtbresource+xml" />
  <item id="c3"
        href="chap3.html"
        media-type="application/html+xml" />
  <item id="footnotes"
        href="footnotes.html"
        media-type="text/x-eob1-document" />
  <item id="f1" href="fig1.jpg" media-type="image/jpeg" />
  <item id="f2" href="fig2.jpg" media-type="image/jpeg" />
  <item id="f3" href="fig3.jpg" media-type="image/jpeg" />
  <item id="ncx" href="toc.ncx" media-type="text/xml" />
</manifest>

<spine toc="ncx">
  <itemref idref="toc" />
  <itemref idref="c1" />
  <itemref idref="c2" />
  <itemref idref="c3" />
</spine>
```

In the above example, suppose the document referenced by ID “c1” is being viewed by a reader. When the end of the document is reached, the next document in linear order would be the

document referenced by ID “c2”. Document “c1” might also have hypertext links to locations in other files such as the “footnotes” file. Such a file **must** be listed in the **manifest**, but **need not** be named by any **itemref** of the **spine**. If a reader follows the hyperlink in “c1” to “footnotes”, proceeds to the bottom of the footnotes file so that the end of that file is reached, then no successor in linear order is defined by this specification.

The NCX section below specifies the requirements for the navigation control file and the **spine toc** attribute.

All content reachable in the publication **must** be referenced with either an **itemref** element. (Content is considered reachable if it is referenced directly or indirectly through hyperlinks or navigation structure (NCX). Referencing content through **img** or **object** elements does not make an item reachable.) Each item may only be referenced by a single element (**itemref**). Content referenced by an **itemref** element which omits the optional *linear="no"* attribute is called primary content. Content referenced by an **itemref** element which include the optional *linear="no"* attribute is referred to as auxiliary content.

Reading Systems **must** display all primary (and optionally auxiliary) content items in a way that allows linear (forward and backward) navigation through all the items in the order in which they are listed in the spine. The decision on whether to include auxiliary content in the linear navigation order should be based on the Reading System design and device capabilities. This specification neither encourages nor discourages such inclusion. Reading Systems **must** display in some fashion all auxiliary content items if such items can be reached through hyperlinks, even if the auxiliary content items are not included in the linear navigation order.

For example:

```
<spine toc="ncx">
  <itemref idref="intro" />
  <itemref idref="chapter1" />
  <itemref idref="chapter1_answerkey" linear="no" />
  <itemref idref="chapter2" />
  <itemref idref="chapter2_answerkey" linear="no" />
  <itemref idref="conclusion" />
  <itemref idref="about_this_book" linear="no" />
</spine>
```

Note that this design is backward compatible with OEBPS 1.2, assuming that the OEBPS 1.2 system ignores unknown attributes.

2.4.1 Declarative Table of Contents – the NCX

In order to enable ease of navigation and provide greater accessibility, the OPF Package supports a “Navigation Center eXtended,” the NCX. This is a concept and implementation has been standardized by the DAISY Consortium.

This specification uses the NCX defined in the DAISY/NISO Standard, formally the ANSI/NISO Z39.86-2005, Specifications for the Digital Talking Book. The NCX is a portion (Section 8) of this comprehensive multimedia standard. The DAISY Consortium maintains the NCX DTD. No modifications to the DTD are necessary for use with OPF. In the future the DAISY/NISO Advisory Committee will consider modularizing the NCX and changing terminology to be more in line with eBooks, Multimedia publications and other electronic document usage.

Some optional elements and metadata items are not needed to implement the NCX for this specification. The sections below have been changed to normatively reference the DAISY/NISO standard for the NCX rather than duplicating it here. All “exceptions” are described in Section 2.4.2, below.

2.4.1.1 Introduction

The Navigation Control file for XML applications (NCX) exposes the hierarchical structure of an eBook to allow the user to navigate through it. The NCX is similar to a table of contents in that it enables the reader to jump directly to any of the major structural elements of the document, i.e. part, chapter, or section, but it will often contain more elements of the document than the publisher chooses to include in the original print table of contents. It can be visualized as a collapsible tree familiar to PC users. Its development was motivated by the need to provide quick access to the main structural elements of the document without the need to parse the entire marked-up text file. Other elements such as pages, footnotes, figures, tables, etc. can be included in separate, nonhierarchical lists and can be accessed by the user as well.

It is important to emphasize that these navigation features are intended as a convenience for users who want them, and not a burden to those who do not. The alternative **guide** to the book **may** be provided for those users not requiring the navigation features of the NCX.

2.4.1.2 Key NCX Requirements

Reading Systems **must** support NCX.

Publications containing only XHTML OPS Content Documents **should** include an NCX (this “**should**” may transition to a “**must**” in future versions of this specification).

Publications containing DTBook or Out-Of-Line XML Island OPS Content Documents **must** include an NCX.

Like all other publication components, the NCX must be included as an **item** in the **manifest**. The NCX-referencing **item must not** contain any fallback information (*island-type*, *fallback* or *fallback-style* attributes).

If a publication includes an NCX, the **item** that describes the NCX must be referenced by the **spine toc** attribute.

The NCX file must be a valid XML document conforming to `ncx-2005-1.dtd` [\[Add link\]](#) and comply with the additional normative requirements defined in <http://www.niso.org/standards/resources/Z39-86-2002.html>. The **version** and *xmlns* attributes on the **ncx** element must be explicitly specified in the document instance, using values drawn from the above-named DTD.

[JMR: We need an example of an NCX.]

2.4.2 NCX Exceptions in Usage for eBooks

The NCX as defined in the ANSI/NISO Z39.86-2005 Standard Section 8 is ideal for OPS Publication applications, however some exceptions should be noted. In the standard, the links from the NCX to the publication point to SMIL documents. For OPS Publications, the link will point to an XML element in the source OPS Content Document. This difference causes the following exceptions to be noted from Section 8 in that standard:

- `<smilCustomTest>` is not used in the OPF application of the NCX;
- `<audio>` is not used in the OPS application of the NCX;
- `clipBegin (%SMILtimeVal, REQUIRED)`: is used for identifying the start of an audio segment, and hence not used in the OPF application of the NCX;

- clipEnd (%SMILtimeVal, REQUIRED): is used for identifying the end of an audio segment, and hence not used in the OPF application of the NCX;
- “<content> Description: Pointer into SMIL file to beginning of the item referenced by the navPoint or navTarget.” Note: In the OPF application of the NCX the pointer is to an XML element and not a SMIL location.
- DTBs Spanning Multiple Media Units are not relevant in the eBook context because OPS Publications are significantly smaller than multimedia files.
- Examples show links to SMIL files, but in the OPF application of NCX the links will be to XML elements. Also, the examples show audio references with clipBegin and clipEnd, which are not used in the OPF application of the NCX.

2.4.3 XML Islands in the Spine

XML Islands [\[LINK HERE\]](#) may be referenced from the spine. In the event that a Reading System cannot display the XML Island correctly, then the standard fallback methodology defined in the Open Document Structure [\[LINK HERE\]](#) must be used.

In short, the Reading System must display the first renderable fallback for an XML Island in the event that the island itself cannot be displayed.

2.5 Tours *[Deprecated]*

Much as a tour-guide might assemble points of interest into a set of sightseers’ tours, a content provider may assemble selected parts of a publication into a set of tours to enable convenient navigation.

An OPS Package **may, but need not**, contain one **tours** element, which in turn contains one or more **tour** elements. Each **tour** **must** have a *title* attribute, intended for presentation to the user. Reading Systems may use tours to provide various access sequences to parts of the publication, such as selective views for various reading purposes, reader expertise levels, etc. Because Reading Systems are not required to implement tour support, content providers **should** also provide other means of accessing content referenced from tours.

Each **tour** element contains one or more **site** elements, each of which **must** have an *href* attribute and a *title* attribute. The *href* attribute **must** refer to an OPS Content Document included in the manifest, and **may** include a fragment identifier as defined in section 4.1 of RFC 2396 (see <http://www.ietf.org/rfc/rfc2396.txt>). Each **site** element specifies a starting point from which the reader may explore freely. Reading Systems **may** use the bounds of the referenced element to determine the scope of the site. If a fragment identifier is not used, the scope is considered to be the entire document. This specification does not require Reading Systems to mark or otherwise identify the entire scope of a referenced element. The order of **site** elements is presumed to be significant, and **should** be used by Reading Systems to aid navigation.

Example:

```

<tours>
  <tour id="tour1" title="Chicken Recipes">
    <site title="Chicken Fingers"
      href="appetizers.html#r3" />
    <site title="Chicken a la King"
      href="entrees.html#r5" />
  </tour>
  <tour id="tour2" title="Vegan Recipes">
    <site title="Hummus" href="appetizer.html#r6" />
    <site title="Lentil Casserole" href="lentils.html" />
  </tour>
</tours>

```

2.6 Guide

Within the **package** there **may** be one **guide** element, containing one or more **reference** elements. The **guide** element identifies fundamental structural components of the publication, to enable Reading Systems to provide convenient access to them.

Example:

```

<guide>
  <reference type="toc" title="Table of Contents"
    href="toc.html" />
  <reference type="loi" title="List Of Illustrations"
    href="toc.html#figures" />
  <reference type="other.intro" title="Introduction"
    href="intro.html" />
</guide>

```

The structural components of the books are listed in **reference** elements contained within the **guide** element. These components may refer to the table of contents, list of illustrations, foreword, bibliography, and many other standard parts of the book. Reading Systems are **not required** to use the **guide** element in any way.

Each reference **must** have an *href* attribute referring to an OPS Content Document included in the **manifest**, and which **may** include a fragment identifier as defined in section 4.1 of RFC 2396 (see <http://www.ietf.org/rfc/rfc2396.txt>). Reading Systems **may** use the bounds of the referenced element to determine the scope of the reference. If a fragment identifier is not used, the scope is considered to be the entire document. This specification **does not require** Reading Systems to mark or otherwise identify the entire scope of a referenced element.

The required *type* attribute describes the publication component referenced by the *href* attribute. The values for the *type* attributes **must** be selected from the list defined below when applicable. Other types **may** be used when none of the predefined types are applicable; their names **must** begin with the string “other.”. The value for the *type* attribute is case-sensitive.

The following list of *type* values is derived from the 13th edition of the *Chicago Manual of Style*:

cover	the book cover(s), jacket information, etc.
title-page	page with possibly title, author, publisher, and other metadata
toc	table of contents

index back-of-book style index
glossary glossary
acknowledgements
bibliography
colophon
copyright-page
dedication
epigraph
foreword
loi list of illustrations
lot list of tables
notes
preface
text First “real” page of content (e.g. “Chapter 1”)

APPENDIX A: THE OPF PACKAGE SCHEMA

TBD

APPENDIX B: Differences Between the OPF 2.0 Package and the OEBPS 1.2 Package

[Covered at a high level in introductory sections. Details here or omit?]

APPENDIX C: CONTRIBUTORS

This specification has been developed through a cooperative effort, bringing together publishers, Reading System vendors, software developers, and experts in the relevant standards.

Version 2.0 of this specification was prepared by the Open eBook Publication Structure Working Group. Active members of the working group at the time of publication of revision 2.0 were:

TBD

Contributors to the Version 2.0 effort who were no longer active working group members at the time of publication include:

TBD

Inclusion of 1.0.x authors, TBD.