Open Document Format for Office Applications (OpenDocument)v1.2

Part 3: Packages

Pre-Draft6, 12 Sep 2007

Specification URIs:

This Version:
OpenDocument-package-v1.2-draft6.odt at http://www.oasis-open.org/committees/office

Previous Version:
http://docs.oasis-open.org/office/v1.1/OS/OpenDocument-v1.1.odt
http://docs.oasis-open.org/office/v1.1/OS/OpenDocument-v1.1.pdf
http://docs.oasis-open.org/office/v1.1/OS/OpenDocument-v1.1.html.zip

Latest Version:
http://www.oasis-open.org/committees/office

Latest Approved Version:
http://docs.oasis-open.org/office/v1.1/OS/OpenDocument-v1.1.odt
http://docs.oasis-open.org/office/v1.1/OS/OpenDocument-v1.1.pdf
http://docs.oasis-open.org/office/v1.1/OS/OpenDocument-v1.1.html.zip

Technical Committee:
OASIS Open Document Format for Office Applications (OpenDocument) TC

Chair:
Michael Brauer, Sun Microsystems, Inc.

Editors:
Patrick Durusau, Individual
Michael Brauer, Sun Microsystems, Inc.

Related Work:
This specification supersedes OASIS OpenDocument v1.1.

Declared XML Namespaces:
A list of XML namespaces declared by this specification is available in section 1.3.

Abstract:
This is the specification of the Open Document Format for Office Applications (OpenDocument) format, an open, XML-based file format for office applications, based on OpenOffice.org XML [OOo].

This specification has three parts.

Part 1 defines an XML schema for office applications and its semantics. The schema is suitable for office documents, including text documents, spreadsheets, charts and graphical documents like drawings or presentations, but is not restricted to these kinds of documents.

Part 2 (this document) defines a formula language to be used in OpenDocument documents.

Part 3 defines a package format to be used for OpenDocument documents.

Status:

This document is a preliminary draft. Its outline is incomplete and does not provide an outlook to the topics addressed by OpenDocument v1.2.

This document was last revised or approved by the OASIS Open Document Format for Office Applications (OpenDocument) Technical Committee on the above date. The level of approval is also listed above. Check the current location noted above for possible later revisions of this document. This document is updated periodically on no particular schedule.

Technical Committee members should send comments on this specification to the Technical Committee's email list. Others should send comments to the Technical Committee by using the "Send A Comment" button on the Technical Committee's web page at

www.oasis-open.org/committees/office.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page

(www.oasis-open.org/committees/office/ipr.php.

The non-normative errata page for this specification is located at www.oasis-open.org/committees/office.

Notices

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full Policy may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification.

OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this specification by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification. OASIS may include such claims on its website, but disclaims any obligation to do so.

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS' procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Committee Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

The names "OASIS", “OpenDocument”, “Open Document Format” and “ODF” are trademarks of OASIS, the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use

of, specifications, while reserving the right to enforce its marks against misleading uses. Please see http://www.oasis-open.org/who/trademark.php for above guidance.

Table of Contents

1 Introduction

1.1 Introduction

This is the specification of the Open Document Format for Office Applications (OpenDocument) format, an open, XML-based file format for office applications.

This specification has three parts.

Part 1 defines an XML schema for office applications and its semantics. The schema is suitable for office documents, including text documents, spreadsheets, charts and graphical documents like drawings or presentations, but is not restricted to these kinds of documents.

Part 2 defines a formula language to be used in OpenDocument documents.

Part 3 (this document) defines a package format to be used for OpenDocument documents.

1.2 Notation

Within this specification, the key words "**shall**", "**shall not**", "**should**", "**should not**" and "**may**" are to be interpreted as described in Annex H of [ISO/IEC Directives] if they appear in bold letters.

1.3 Namespaces

Table 1 lists the namespaces that are defined by schemas defined in the OpenDocument specification part 3 and their default prefixes. For more information about XML namespaces, please refer to the *Namespaces in XML* specification [xml-names].

Table 1 - XML Namespaces defined by the OpenDocument specification part 3

Prefix	***Description***	***Namespace***
manifest	Elements and attribute contained in the package manifest.	urn:oasis:names:tc:opendocument:xmlns:manifest:1.0

Table 2 lists the namespaces that are imported into schemas defined by the OpenDocument specification part 3 and their default prefixes.

Table 2 - XML Namespaces used by the OpenDocument specification part 3

Prefix	***Description***	***Namespace***

2 Packages

This chapter describes the package format that optionally can be used in OpenDocument. It contains the following sections:

- Introduction
- Zip File Structure
- Encryption
- Preview Image
- Manifest File

2.1 Introduction

As XML has no native support for binary objects such as images, [OLE] objects, or other media types, and because uncompressed XML files can get very large, OpenDocument uses a package file to store the XML content of a document together with its associated binary data, and to optionally compress the XML content. This package is a standard Zip file, whose structure is discussed below.

Information about the files contained in the package is stored in an XML file called the manifest file. The manifest file is always stored at the pathname META-INF/manifest.xml. The main pieces of information stored in the manifest are as follows:

- A list of all of the files in the package.
- The media type of each file in the package.
- If a file stored in the package is encrypted, the information required to decrypt the file is stored in the manifest.

2.2 Zip File Structure

A Zip file starts with a sequence of files, each of which can be compressed or stored in raw format. Each file has a local header immediately before its data, which contains most of the information about the file, including time-stamps, compression method and file name. The compressed file contents immediately follow, and are terminated by an optional data descriptor. The data descriptor contains the CRC and compressed size of the file, which are frequently not available when writing the local file header. If these details were included, the data descriptor can be skipped.

Each file in the archive is laid down sequentially in this format, followed by a central directory at the end of the Zip archive. The central directory is a contiguous set of directory entries, each of which contains all the information in the local file header, plus extras such as file comments and attributes. Most importantly, the central directory contains pointers to the position of each file in the archive for navigation of the Zip file.

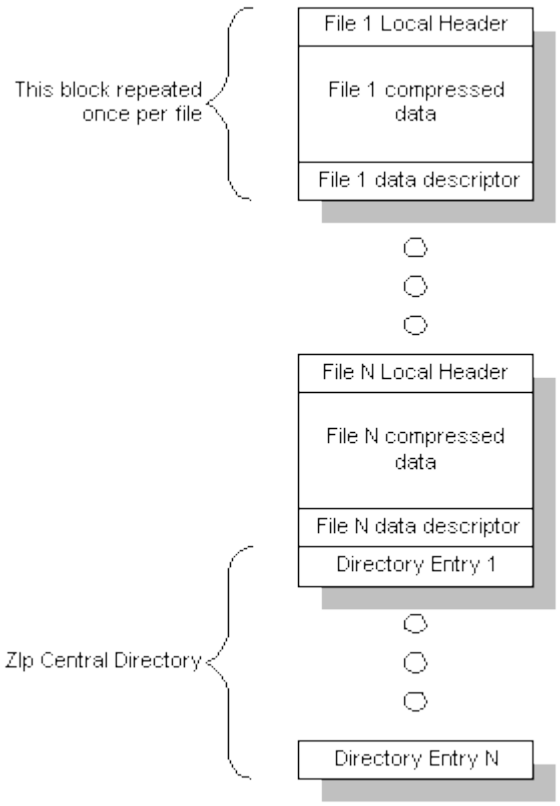

Figure 1 - Zip file structure

For more details about the Zip file format, see [ZIP].

2.3 Encryption

The encryption process takes place in the following multiple stages:

1. A 20-byte SHA1 digest of the user entered password is created and passed to the package component.
2. The package component initializes a random number generator with the current time.
3. The random number generator is used to generate a random 8-byte initialization vector and 16-byte salt for each file.
4. This salt is used together with the 20-byte SHA1 digest of the password to derive a unique 128-bit key for each file. The algorithm used to derive the key is PBKDF2 using HMAC-SHA-1 (see [RFC2898]) with an iteration count of 1024.
5. The derived key is used together with the initialization vector to encrypt the file using the Blowfish algorithm in cipher-feedback (CFB) mode.

Each file that is encrypted is compressed before being encrypted. To allow the contents of the package file to be verified, it is necessary that encrypted files are flagged as 'STORED' rather than 'DEFLATED'. As entries which are 'STORED' must have their size equal to the compressed size, it is necessary to store the uncompressed size in the manifest. The compressed size is stored in both the local file header and central directory record of the Zip file.

2.4 Digital Signatures

Files within a package may have a digital signature applied. Digital signatures are stored in one or more files within the META-INF folder. The names of these files **shall** contain the term "signatures". Each of these files contains a `<dsig:document-signatures>` root element that serves as a container for an arbitrary `<Signature>` element as defined by the [xml-dsig] specification. If the `<dsig:document-signatures>` element contains multiple `<Signature>` elements, then there **should** be a relation between the digital signatures they define, for instance, they may all apply to the same set of files.

Applications **may** require that a digital signature includes a certain set of files. That is, they **may** consider a digital signature to be valid if, and only if,

- the digital signature itself is valid, and
- if the `<Reference>` child elements of the `<Signature>` element reference a certain set of files.

In particular, application **may** require that a digital signature references all files contained in a package.

The schema for digital signatures is:

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
OASIS OpenDocument v1.2
Draft1, 19 Oct 2006
Relax-NG Manifest Schema

$Id$

© 2002-2005 OASIS Open
-->

<grammar
    xmlns="http://relaxng.org/ns/structure/1.0"

    datatypeLibrary="http://www.w3.org/2001/XMLSchema-datatypes"

    xmlns:dsig="urn:oasis:names:tc:opendocument:xmlns:digitalsignature:1.0"
    xmlns:xmldsig="http://www.w3.org/2000/09/xmldsig#">

<define name="dsig-document-signatures">
    <element name="dsig:document-signatures">
        <oneOrMore>
            <ref name="xmldsig-signature"/>
        </oneOrMore>
    </element>
</define>

<start>
    <choice>
        <ref name="dsig-document-signatures"/>
    </choice>
</start>

<define name="xmldsig-signature">
    <element name="xmldsig:Signature">
        <ref name="xmldsigMarkup"/>
    </element>
</define>
```

```
<!-- To avoid inclusion of the complete XMLDSIG schema, anything -->
<!-- is allowed within a xmldsig:Signature element               -->
<define name="xmldsigMarkup">
    <zeroOrMore>
        <choice>
            <attribute>
                <anyName/>
            </attribute>
            <text/>
            <element>
                <anyName/>
                <ref name="xmldsigMarkup"/>
            </element>
        </choice>
    </zeroOrMore>
</define>
</grammar>
```

2.5 MIME Type Stream

If a MIME type for a document that makes use of packages exists, then the package **should** contain a stream called "mimetype". This stream **should** be first stream of the package's zip file, it **shall not** be compressed, and it **shall not** use an 'extra field' in its header. See [ZIP].

The purpose is to allow packaged files to be identified through 'magic number' mechanisms, such as Unix's file/magic utility. If a ZIP file contains a stream at the beginning of the file that is uncompressed, and has no extra data in the header, then the stream name and the stream content can be found at fixed positions. More specifically, one will find:

- a string 'PK' at position 0 of all zip files
- a string 'mimetype' at position 30 of all such package files
- the mimetype itself at position 38 of such a package.

2.6 Usage of IRIs Within Packages

Within a file that is contained in a package, relative IRIs are used to reference other sub files of the package, but can also be used to reference files within the file system.

The following restrictions exist for IRIs that are used within a package:

- only sub files within the same package and files outside the package can be referenced.
- IRIs that reference a sub file of a package **shall** be relative, and they **shall not** contain paths that are not within the package. This especially means that sub files of a package **shall not** be referenced by an absolute IRI.
- sub file of a package can not be referenced from outside the package, for instance from the file system or another package.

A relative-path reference (as described in §6.5 of [RFC3987]) that occurs in a file that is contained in a package has to be resolved exactly as it would be resolved if the whole package gets unzipped into a directory at its current location. The base IRI for resolving relative-path references is the one that has to be used to retrieve the (unzipped) file that contains the relative-path reference.

All other kinds of IRI references, namely the ones that start with a protocol (like http:), an authority (i.e., //) or an absolute-path (i.e., /) do not need any special processing. This means that absolute-

paths do not reference files inside the package, but within the hierarchy the package is contained in, for instance the file system. IRI references inside a package may leave the package, but once they have left the package, they never can return into the package or another one.

2.7 Preview Image

A thumbnail representation of a document should be generated by default when the file is saved. It should be a representation of the first page, first sheet, etc. of the document. For maximum reusability of the thumbnails they have to be generated without any effects, surrounding frames, or borders. Such effects might interfere with effects added to the thumbnails by the different file system explorers or may not be desired at all for certain use cases.

The thumbnail **shall** be saved as “thumbnail.png” in a separate folder named “Thumbnails”.

Ed. Note: The above shall was a simple must. Same below.

Thumbnails **shall** be saved in PNG format.

Note: Current desktops display thumbnail images within squares of up to 256 pixel width and height, and 24 bit per pixel. While this specification does not define upper or lower limits for thumbnail image sizes, implementations should only use image sizes that are displayed with a reasonable quality if scaled to fit into 256x256 pixel square.

The “Thumbnails” folder **shall not** get a media type in the manifest.xml file, since it is not actually part of the document.

Encrypted files are intended to be unreadable for unauthorized users and a thumbnail for such files must not be generated. Instead of saving a thumbnail of the first page a replacement representation that doesn't depend on the contents of the document is saved for encrypted files which makes obvious that the corresponding file is encrypted.

2.8 Manifest File

The elements and attributes in the manifest file are in the namespace: urn:oasis:names:tc:opendocument:xmlns:manifest:1.0.

2.8.1 Relax-NG Schema

The normative XML Schema for OpenDocument Manifest files is embedded within this specification. It can be obtained from the specification document by concatenating all schema fragments contained in this chapters. All schema fragments have a gray background color and line numbers.

The schema language used within this specification is Relax-NG. See [RNG].

Prefix for the normative Relax-NG Manifest schema:

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <!--
3      OASIS OpenDocument v1.2
4      Draft6, 12 Sep 2007
5      Relax-NG Manifest Schema
6
7      $Id$
8
9      © 2002-2005 OASIS Open
10     © 1999-2005 Sun Microsystems, Inc.
11 -->
12
```

```
<grammar
    xmlns="http://relaxng.org/ns/structure/1.0"

    datatypeLibrary="http://www.w3.org/2001/XMLSchema-datatypes"

    xmlns:manifest="urn:oasis:names:tc:opendocument:xmlns:manifest:1.0">
```

2.8.2 Manifest Root Element

The root element of the manifest is the `<manifest>` element. The `<manifest>` element contains one fixed attribute which specifies the namespace as described above and multiple `<manifest:file-entry>` elements, each of which describes a single file in the package.

```
<define name="manifest">
    <element name="manifest:manifest">
        <oneOrMore>
            <ref name="file-entry"/>
        </oneOrMore>
    </element>
</define>

<start>
    <choice>
        <ref name="manifest"/>
    </choice>
</start>
```

2.8.3 File Entry

The `<manifest:file-entry>` element represents a single file within the package, and stores the files location in the package, the mime-type of the file and optionally the data required to decrypt this file.

Directories only receive `<manifest:file-entry>` entries if they have inherent semantics. For example, a directory that constitutes a sub-document referenced as an object from within the main document would contain a `<manifest:file-entry>` with a suitable media type. A directory for administrative or convenience purposes, such as a directory that contains various image files, would not receive an entry in the manifest file.

```
<define name="file-entry">
    <element name="manifest:file-entry">
        <ref name="file-entry-attlist"/>
        <optional>
            <ref name="encryption-data"/>
        </optional>
    </element>
</define>
```

The attributes associated with a `<manifest:file-entry>` are as follows:

- Full path
- Size
- Media type

2.8.3.1 Full Path

The `manifest:full-path` attribute describes the location of the file within the package.

```
<define name="file-entry-attlist" combine="interleave">
    <attribute name="manifest:full-path">
        <data type="string"/>
    </attribute>
</define>
```

2.8.3.2 Size

The `manifest:size` attribute is only present if the file is stored in an encrypted format. See section 2.3. This attribute is only used for encrypted files.

```
<define name="file-entry-attlist" combine="interleave">
    <optional>
        <attribute name="manifest:size">
            <data type="nonNegativeInteger"/>
        </attribute>
    </optional>
</define>
```

2.8.3.3 Media Type

The `manifest:media-type` attribute specifies the mime type of the specified file. For a full list of mime types see http://www.isi.edu/in-notes/iana/assignments/media-types/media-types. All XML streams have the media type "text/xml".

```
<define name="file-entry-attlist" combine="interleave">
    <attribute name="manifest:media-type">
        <data type="string"/>
    </attribute>
</define>
```

2.8.4 Encryption Data

The `<manifest:encryption-data>` element contains all of the information required to decrypt the file.

```
<define name="encryption-data">
    <element name="manifest:encryption-data">
        <ref name="encryption-data-attlist"/>
        <ref name="algorithm"/>
        <ref name="key-derivation"/>
    </element>
</define>
```

The `<manifest:encryption-data>` element contains the following elements:

- Algorithm
- Key Derivation

2.8.4.1 Checksum Type

The `manifest:checksum-type` attribute specifies the name of digest algorithm that can be used to check password correctness. Currently, the only supported digest algorithm is SHA1.

```
<define name="encryption-data-attlist" combine="interleave">
    <attribute name="manifest:checksum-type">
        <data type="string"/>
    </attribute>
```

```
</define>
```

2.8.4.2 Checksum

The `manifest:checksum` attribute specifies the digest in BASE64 encoding (see [RFC2045]) that can be used to detect password correctness as specified within `manifest:checksum-type` attribute.

```
<define name="encryption-data-attlist" combine="interleave">
    <attribute name="manifest:checksum">
        <data type="base64Binary"/>
    </attribute>
</define>
```

2.8.5 Algorithm

The `<manifest:algorithm>` element contains information about the algorithm used to encrypt the data.

```
<define name="algorithm">
    <element name="manifest:algorithm">
        <ref name="algorithm-attlist"/>
        <empty/>
    </element>
</define>
```

The attributes associated with `<manifest:algorithm>` are as follows:

- Algorithm name
- Initialization vector

2.8.5.1 Algorithm Name

The `manifest:algorithm-name` attribute specifies the name of the algorithm used to encrypt the file, and also specifies in which mode this algorithm was used. Currently, the only supports algorithm is the `Blowfish` algorithm in `CFB` mode.

```
<define name="algorithm-attlist" combine="interleave">
    <attribute name="manifest:algorithm-name">
        <data type="string"/>
    </attribute>
</define>
```

2.8.5.2 Initialization Vector

The `manifest:initialisation-vector` attribute specifies the 8 bytes used as an initialization vector to the stream cipher. The initialization vector is an 8 byte binary sequence, and so is encoded in BASE64 (see [RFC2045]) when written to the manifest file.

```
<define name="algorithm-attlist" combine="interleave">
    <attribute name="manifest:initialisation-vector">
        <data type="base64Binary"/>
    </attribute>
</define>
```

2.8.6 Key Derivation

The `<manifest:key-derivation>` element contains the information that was used to derive the encryption key for this file from the user specified password.

```
<define name="key-derivation">
    <element name="manifest:key-derivation">
        <ref name="key-derivation-attlist"/>
        <empty/>
    </element>
</define>
```

The attributes associated with the `<manifest:key-derivation>` element are as follows:

- Key derivation name
- Salt
- Iteration count

2.8.6.1 Key Derivation Name

The `manifest:key-derivation-name` attribute specifies the name of the algorithm used to derive the name. At this time, the packages only supports the use of the PBKDF2 key derivation method. See [RFC2898].

```
<define name="key-derivation-attlist" combine="interleave">
    <attribute name="manifest:key-derivation-name">
        <data type="string"/>
    </attribute>
</define>
```

2.8.6.2 Salt

The `manifest:salt` attribute specifies the 16-byte sequence used as the 'salt' by the key derivation algorithm. The salt is a 16-byte binary sequence, and thus is encoded in BASE64 (as defined in [RFC2045]) before being written to the manifest file.

```
<define name="key-derivation-attlist" combine="interleave">
    <attribute name="manifest:salt">
        <data type="base64Binary"/>
    </attribute>
</define>
```

2.8.6.3 Iteration Count

The `manifest:iteration-count` attribute specifies the number of iterations used by the key derivation algorithm to derive the key.

```
<define name="key-derivation-attlist" combine="interleave">
    <attribute name="manifest:iteration-count">
            <data type="nonNegativeInteger"/>
        </attribute>
</define>
```

Example: Sample Manifest

```
<manifest:manifest
    xmlns:manifest="urn:oasis:names:tc:opendocument:xmlns:manifest:1.0">
    <manifest:file-entry
        manifest:media-type="application/vnd.oasis.opendocument.text"
```

```
        manifest:full-path="/"/>
    <manifest:file-entry manifest:media-type="image/jpeg"
        manifest:full-path="Pictures/10000000000003200000258912EB1C3.jpg"
        manifest:size="66704">
        <manifest:encryption-data>
            <manifest:algorithm manifest:algorithm-name="Blowfish CFB"
                manifest:initialisation-vector="T+miu403484="/>
            <manifest:key-derivation manifest:key-derivation-name="PBKDF2"
                manifest:iteration-count="1024"
                manifest:salt="aNYdmqv4cObAJSJjm4RzqA=="/>
        </manifest:encryption-data>
    </manifest:file-entry>
    <manifest:file-entry
        manifest:media-type="text/xml" manifest:full-path="content.xml"
        manifest:size="3143">
        <manifest:encryption-data>
            <manifest:algorithm manifest:algorithm-name="Blowfish CFB"
                manifest:initialisation-vector="T+miu403484="/>
            <manifest:key-derivation manifest:key-derivation-name="PBKDF2"
                manifest:iteration-count="1024"
                manifest:salt="aNYdmqv4cObAJSJjm4RzqA=="/>
        </manifest:encryption-data>
    </manifest:file-entry>
    <manifest:file-entry manifest:media-type="text/xml"
        manifest:full-path="styles.xml" manifest:size="5159">
        <manifest:encryption-data>
            <manifest:algorithm manifest:algorithm-name="Blowfish CFB"
                manifest:initialisation-vector="bChL2No5I+A="/>
            <manifest:key-derivation manifest:key-derivation-name="PBKDF2"
                manifest:iteration-count="1024"
                manifest:salt="/kfasyu7X0Ae+1uopdeCtA=="/>
        </manifest:encryption-data>
    </manifest:file-entry>
    <manifest:file-entry
        manifest:media-type="text/xml" manifest:full-path="meta.xml"/>
    <manifest:file-entry
        manifest:media-type="text/xml"
        manifest:full-path="settings.xml" manifest:size="5317">
        <manifest:encryption-data>
            <manifest:algorithm manifest:algorithm-name="Blowfish CFB"
                manifest:initialisation-vector="JQxEm6rD+4c="/>
            <manifest:key-derivation manifest:key-derivation-name="PBKDF2"
                manifest:iteration-count="1024"
                manifest:salt="PlpDaxloh4KUKx+v1g4V9g=="/>
        </manifest:encryption-data>
    </manifest:file-entry>
</manifest:manifest>
```

2.8.7 Relax-NG Schema Suffix

Suffix for the normative Relax-NG Manifest schema:

111
```
</grammar>
```

Appendix A.References

[**ISO/IEC Directives**] ISO/IEC Directives, Part 2 *Rules for the structure and drafting of International Standards*, 2004

[**OLE**] Kraig Brockschmidt, Inside OLE, Microsoft Press, 1995, ISBN: 1-55615-843-2

[**OOo**] , *OpenOffice.org XML File Format 1.0 Technical Reference Manual*, http://xml.openoffice.org/xml_specification.pdf, Sun Microsystems, Inc., 2002.

[**RFC2045**] N. Freed and N. Borenstein, *Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies*, http://www.ietf.org/rfc/rfc2045.txt, IETF, 1996.

[**RFC2898**] B. Kaliski, *PKCS #5: Password-Based Cryptography Specification Version 2.0*, http://www.ietf.org/rfc/rfc2898, IETF, 2000.

[**RFC3987**] M. Duerst, M. Suignard, *Internationalized Resource Identifiers (IRIs)*, http://www.ietf.org/rfc/rfc3987.txt, IETF, 2005.

[**RNG**] ISO/IEC 19757-2 *Document Schema Definition Language (DSDL) -- Part 2: Regular-grammar-based validation -- RELAX NG*, 2003

[**xml-names**] Tim Bray, Dave Hollander, Andrew Layman, *Namespaces in XML*, http://www.w3.org/TR/REC-xml-names/, W3C, 1999.

[**ZIP**] *Info-ZIP Application Note 970311*, ftp://ftp.uu.net/pub/archiving/zip/doc/appnote-970311-iz.zip, 1997

Appendix B.Changes From Previous Specification Versions (Non Normative)

B.1. Changes from “Open Document Format for Office Applications (OpenDocument) v1.1”

Appendix C.Acknowledgments (Non Normative)

Current Contributors:

Daniel Brotsky, Adobe Systems
Jerome Dumonteil, Ars Aperta
Charles Schulz, Ars Aperta
Jerry Berrier, BayState Council of the Blind (BSCB)
Donglin Wang, Beijing Sursen International Information Technology Co., Ltd.
Rui Zhao, Changfeng Open Standards Platform Software Alliance
Stephen Noble, Design Science, Inc.
John Madden, Duke University
Chieko Asakawa, IBM
Nathaniel Borenstein, IBM
Pete Brunet, IBM
Yue Ma, IBM
Richard Schwerdtfeger, IBM
Robert Weir, IBM
Zhi Yu Yue, IBM
John Barstow, Individual
Patrick Durusau, Individual
Michael Paciello, Individual
Janina Sajka, Individual
David Clark, Institute for Community Inclusion
Waldo Bastian, Intel Corporation
James Mason, ISO/IEC JTC1/SC34
David Faure, KDE e.V
Jody Goldberg, Novell
David Pawson, Royal National Institute for the Blind
Michael Brauer, Sun Microsystems, Inc.
Peter Korn, Sun Microsystems, Inc.
Lars Oppermann, Sun Microsystems, Inc.
Eike Rathke, Sun Microsystems, Inc.
Svante Schubert, Sun Microsystems, Inc.
Frank Stecher, Sun Microsystems, Inc.
Malte Timmermann, Sun Microsystems, Inc.
Daniel Bricklin, The OpenDocument Foundation, Inc.
Daniel Carrera, The OpenDocument Foundation, Inc.
Bruce D'Arcus, The OpenDocument Foundation, Inc.
Gary Edwards, The OpenDocument Foundation, Inc.
Elmar Geese, The OpenDocument Foundation, Inc.
Sam Hiser, The OpenDocument Foundation, Inc.
Michael Kleinhenz, The OpenDocument Foundation, Inc.
Tomas Mecir, The OpenDocument Foundation, Inc.
Thomas Metcalf, The OpenDocument Foundation, Inc.
Stefan Nikolaus, The OpenDocument Foundation, Inc.
Florian Reuter, The OpenDocument Foundation, Inc.
Daniel Vogelheim, The OpenDocument Foundation, Inc.
David A. Wheeler, The OpenDocument Foundation, Inc.
Chris Nokleberg, Tonic Systems, Inc.

Previous Contributors:

Paul Grosso, Arbortext
Tom Magliery, Blast Radius
Doug Alberg, Boeing

Paul Langille, Corel
John Chelsom, CSW Informatics
Monica Martin, Drake Certivo
Jason Harrop, Individual
Uche Ogbuji, Individual
Lauren Wood, Individual
Simon Davis, National Archive of Australia
Mark Heller, New York State Office of the Attorney General
Phil Boutros, Stellent