This is clause 8.5 from Murata-san’s draft circulated on May 8.

## Relationships

### General (informative)

Parts may contain references to other parts in the package and to resources outside of the package. These references are represented inside the referring part in ways that are specific to the media type of the part; that is, in arbitrary markup or an application-defined encoding. This effectively hides the links between parts from consumers that do not understand the media types of the parts containing such references.

The package model introduces a higher-level mechanism to describe references from parts to other parts or external resources, namely, relationships. Relationships represent connections from a source part to a target part or target resource. Relationships make the connection directly discoverable without looking at the part contents, so they are independent of content-specific schemas and are quick to resolve.

Relationships have a second important function: to provide additional information about parts without modifying their content. [Note: Some scenarios require information to be attached to an existing part without modifying that part, for example, because the part is encrypted and cannot be decrypted, or because it is digitally signed and changing it would invalidate the signature. end note]

Relationships may have identifiers. Source parts may specify these identifiers at certain locations as references to relationships, but are not required to do so.

Relationships have absolute URIs as types. They specify roles of relationships.

Perhaps: Relationships are categorized via types that identify roles for relationships. The types are expressed using absolute URIs [or IRIs]. [Examples might be useful] Apart from the relationship types defined in Annex E of this standard, relationship types are implementation-dependent.

Relationships are represented in XML in a Relationships part. Each part in the package that is the source of one or more relationships has an associated Relationships part. This part holds the list of relationships for the source part. For more information on the Relationships namespace and relationship types specified in this part of this standard, see Annex E.

### Relationships Part

|  |  |
| --- | --- |
| media Type: | application/vnd.openxmlformats-package.relationships+xml |
| Root Namespace: | http://schemas.openxmlformats.org/package/2006/relationships |

A Relationships Part is a container of relationships.

Each elationships part in a package shall have a source. A source is either a non-relationships part in the same package or the package itself. A relationships part shall not be the source of another relationships part.

There is a naming convention for associating relationships parts with sources. If the source of a relationships part is a package, the part name of this relationships part shall be “/\_rels/.rels”. If the source of a relationship part is a non-relationship part, the part name of this relationship part shall be constructed from the part name of the source part, by first inserting “\_rels/” immediately before the last I18N segment in ps and then appending “.rels”.

[Example: If a source part name is “/foo”, the last I18N segment is “foo” and the relationship part name is “/\_rels/foo.rels”. If a source part name is “/foo/bar.xml”, the last I18N segment is “bar.xml” and the relationship part name is “/foo/\_rels/bar.xml.rels”. end example]

### Relationship Markup

#### General

The content of a relationships part shall be an XML document. It consists of a single Relationships element containing zero or more Relationship elements. These elements are defined in the Relationships namespace, as specified in **Error! Reference source not found.**. The W3C XML Schema for relationships is described in Annex **Error! Reference source not found.**.

After the removal of any extensions by an MCE processor as specified in ISO/IEC 29500-3, a Relationships part shall be a schema-valid XML document against opc-relationships.xsd.

The xml:base attribute shall not be used to specify a base URI for relationship XML content.

#### Relationships Element

|  |
| --- |
| Parent Elements |
| Root element of Relationships part |

|  |  |
| --- | --- |
| Child Elements | Subclause |
| Relationship | §**Error! Reference source not found.** |

#### Relationship Element

A Relationship element shall represent a relationship, a connection from a source (a part or the package as a whole) to a target (another part or an external resource). The source of a relationship shall be the source of the relationships part containing this Relationship element. (Note: The target of a relationship is specified by the attributes of the Relationship element.]

A package relationship shall be a relationship having a package as the source. A part relationship shall be a relationship having a part as the source.

|  |
| --- |
| Parent Elements |
| Relationships |

|  |  |
| --- | --- |
| Attributes | Description |
| TargetMode | This attribute specifies whether or not the target describes a part inside the package or a resource outside the package.  Permissible values of the TargetMode attribute are Internal and External.  This attribute is optional and the default value is Internal.  The possible values for this attribute are defined by the ST\_TargetMode simple type (§**Error! Reference source not found.**). |
| Target | This attribute specifies the target of a relationship.  If the value of the TargetMode attribute is Internal, the Target attribute shall be a relative reference to a part. If the value of the TargetMode attribute is External, the Target attribute shall be a relative reference or an absolute IRI, which may contain fragment identifiers. Base IRIs for resolving relative references are defined in 8.3.6.  [Editor's note: In 8.3.6, we have to define base IRIs]  Required  The possible values for this attribute are defined by the xsd:anyURI simple type of the W3C Recommendation “XML Schema Part 2: Datatypes.”. |
| Type | This attribute specifies the role of a relationship, relationship type.  Relationship types can be compared to determine whether two Relationship elements are of the same type. This comparison is conducted in the same way as when comparing URIs that identify XML namespaces: the two URIs are treated as strings and considered identical if and only if the strings have the same sequence of characters. The comparison is case-sensitive and no escaping is done or undone.  Required. Real examples would be nice.  The possible values for this attribute are defined by the xsd:anyURI simple type of the W3C Recommendation “XML Schema Part 2: Datatypes.”. |
| Id | This attribute specifies the identifier of a relationship. The value of the Id attribute shall be unique within the Relationships part.  Required  The possible values for this attribute are defined by the xsd:ID simple type of the W3C Recommendation “XML Schema Part 2: Datatypes.”. |

### Examples of relationship markup (non-normative)

#### Relationship part “/\_rels/.rels”

Consider a package, available at <http://www.example.com/ex.opc>. Suppose that the package contains a relationship part “/\_rels/.rels”. The source for this relationship part is the package.

Also suppose that the content of this relationship part is the XML document shown below:

<Relationships

xmlns="http://schemas.openxmlformats.org/package/2006/relationships">

<Relationship

Target="./a.xml"

Id="IDI1"

Type="http://example.com/relTypeInt1"/>

<Relationship

Target="./a.xml"

TargetMode="External"

Id="IDE1"

Type="http://example.com/relTypeExt1"/>

</Relationships>

There are two relationships in this relationship part. The source of both relationships is the package.

The mode of the first relationship is Internal (default). Thus, the base IRI for resolving "./a.xml" is the pack URI (pack://http%3c,,www.example.com,ex.opc) created from the IRI of the package (http://www.example.com/ex.opc). By resolving "./a.xml", we have pack://http%3c,,www.example.com,ex.opc/a.xml". The target of this relationship is thus the part "/a.xml" in this package. The relationship type of this relationship is "http://example.com/relTypeInt1". The identifier of this relationship is "IDI1".

The mode of the second relationship is External. Thus, the base IRI for resolving "./a.xml" is the IRI (http://www.example.com/ex.opc) of the package. The target of this relationship is thus the resource at http://www.example.com/a.xml. The relationship type of this relationship is "http://example.com/relTypeExt1". The identifier of this relationship is "IDE1".

#### Relationships part “/foo\_rels/test.xml.rels”

Consider a package, availalbe at <http://www.example.com/ex.opc>. Suppose that the package contains a relationship part“/foo\_rels/test.xml.rels”, the source of which is a part “/foo/test.xml”.

Also suppose that the content of this relationship part is an XML document shown below:

<Relationships

xmlns="http://schemas.openxmlformats.org/package/2006/relationships">

<Relationship

Target="./b.xml"

Id="IDI2"

Type="http://example.com/relTypeInt2"/>

<Relationship

Target="./b.xml"

TargetMode="External"

Id="IDE2"

Type="http://example.com/relTypeExt2"/>

</Relationships>

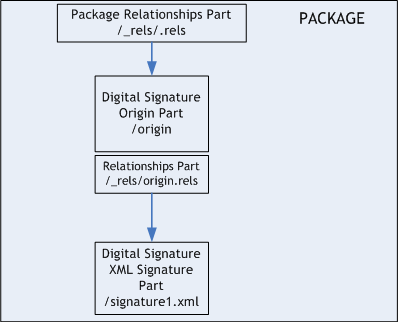
There are two relationships in this relationship part. The source of both relationships is the part “/foo/test.xml”.

The mode of the first relationship is Internal (default). Thus, the base IRI (“pack://http%3c,,www.example.com,ex.opc/foo/test.xml”) for resolving "./b.xml" is the pack URI created from the IRI (http://www.example.com/ex.opc) of the package and the part name “/foo/test.xml”. By resolving "./b.xml", we have pack://http%3c,,www.example.com,ex.opc/foo/b.xml”. The target of this relationship is thus the part "/foo/b.xml" in this package. The relationship type of this relationship is "http://example.com/relTypeInt2". The identifier of this relationship is "IDI2".

The mode of the second relationship is External. Thus, the base IRI for resolving "./b.xml" is the IRI (http://www.example.com/ex.opc) of the package. The target of this relationship is thus the resource at http://www.example.com/b.xml. The relationship type of this relationship is "http://example.com/relTypeExt2". The identifier of this relationship is "IDE2".

#### Digital Signature

The figure below shows a Digital Signature Origin part and a Digital Signature XML Signature part. The Digital Signature Origin part is targeted by a package relationship. The connection from the Digital Signature Origin to the Digital Signature XML Signature part is represented by a relationship.



The relationship targeting the Digital Signature Origin part is stored in /\_rels/.rels and the relationship for the Digital Signature XML Signature part is stored in /\_rels/origin.rels.

The Relationships part associated with the Digital Signature Origin contains a relationship that connects the Digital Signature Origin part to the Digital Signature XML Signature part. This relationship is expressed as follows:

<Relationships

xmlns="http://schemas.openxmlformats.org/package/2006/relationships">

<Relationship

Target="./Signature.xml"

Id="A5FFC797514BC"

Type="http://schemas.openxmlformats.org/package/2006/relationships/

digital-signature/signature"/>

</Relationships>

end example]

#### External resources

Relationships can target resources outside of the package at an absolute location and resources located relative to the current location of the package. The following Relationships part specifies relationships that connect a part to pic1.jpg at an external absolute location, and to my\_house.jpg at an external location relative to the location of the package:

<Relationships

xmlns="http://schemas.openxmlformats.org/package/2006/relationships"

<Relationship

TargetMode="External"

Id="A9EFC627517BC"

Target="http://www.custom.com/images/pic1.jpg"

Type="http://www.custom.com/external-resource"/>

<Relationship

TargetMode="External"

Id="A5EFC797514BC"

Target="./images/my\_house.jpg"

Type="http://www.custom.com/external-resource"/>

</Relationships>

#### Multiple relationships to the same target

The following Relationships part contains two relationships, each with a unique Id value. The relationships share the same Target, but have different relationship types.

<Relationships

xmlns="http://schemas.openxmlformats.org/package/2006/relationships">

<Relationship

Target="./Signature.xml"

Id="A5FFC797514BC"

Type="http://schemas.openxmlformats.org/package/2006/

relationships/digital-signature/signature"/>

<Relationship

Target="./Signature.xml"

Id="B5F32797CC4B7"

Type="http://www.custom.com/internal-resource"/>

</Relationships>

end example]

### Support for Versioning and Extensibility

Producers might generate relationship markup that uses the versioning and extensibility mechanisms defined in Part 3 to incorporate elements and attributes drawn from other XML namespaces. [O1.7]

Consumers shall process relationship markup in a manner that conforms to Part 3. [M1.31]