# Energistics Packaging Conventions Specification

For use with Energistics data-exchange standards

| | |
|---|---|
| **EPC Overview** | To address the challenges of the multi-file data sets used in upstream oil and gas, Energistics and its members have developed file packaging conventions based on the Open Packaging Conventions (OPC), a widely used container-file technology that allows multiple types of files to be bundled together into a single package. The Energistics Packaging Convention (EPC) is intended for use with all Energistics standards. |
| **Version of Standard** | V1.0 |
| Abstract: | This document defines and describes the details of the Energistics Packaging Conventions (EPC). |
| Prepared by: | Energistics and the RESQML SIG |
| Date published: | September 2014 |
| Document type: | Specification |
| Keywords: | standards, energy, data, information, process |

| Document Information | |
|---|---|
| **DOCUMENT VERSION** | 1.0 |
| **Date** | September 2014 |
| **Language** | U.S. English |

**Usage, Intellectual Property Rights, and Copyright**

This document was developed using the Energistics Standards Procedures. These procedures help implement Energistics' requirements for consensus building and openness. Questions concerning the meaning of the contents of this document or comments about the standards procedures may be sent to Energistics at info@energistics.org.

The material described in this document was developed by and is the intellectual property of Energistics. Energistics develops material for open, public use so that the material is accessible and can be of maximum value to everyone.

Use of the material in this document is governed by the Energistics Intellectual Property Policy document and the Product Licensing Agreement, both of which can be found on the Energistics website, http://www.energistics.org/legal-policies.

All Energistics published materials are freely available for public comment and use. Anyone may copy and share the materials but must always acknowledge Energistics as the source. No one may restrict use or dissemination of Energistics materials in any way.

**Trademarks**

Energistics®, WITSML™, PRODML™, RESQML™, Upstream Standards. Bottom Line Results.®, The Energy Standards Resource Centre™ and their logos are trademarks or registered trademarks of Energistics in the United States. Access, receipt, and/or use of these documents and all Energistics materials are generally available to the public and are specifically governed by the Energistics Product Licensing Agreement (http://www.energistics.org/product-license-agreement).

Other company, product, or service names may be trademarks or service marks of others.

| Amendment History | | | |
|---|---|---|---|
| **Version** | **Date** | **Comment** | **By** |
| 1.0 | September 2014 | First publication. | Energistics and the RESQML SIG |

# Table of Contents

# 1 Introduction

## 1.1 What is Energistics Packaging Conventions (EPC)?

### 1.1.1 Challenge: Complex, Multi-File Datasets

With the complexity of oil and gas exploration and production (E&P), petro-technical professionals find themselves dealing with large, complex data sets, which may include various types of data files, reports, images, and more.

When exchanging that data during the course of normal E&P workflows and through the use of related software applications, users need to know that they do indeed have "all the pieces" that make up a particular data set, and they need to understand what those pieces are and how they are related to one another.

### 1.1.2 Solution: Standards for Packaging Together Files

To address these challenges, Energistics and its members have developed a file packaging convention based on the Open Packaging Conventions (OPC), a widely used container-file technology that allows multiple types of files to be bundled together into a single package. Built on the widely used ZIP file structure and originally created by Microsoft, OPC is now an open standard supported by these standards organizations:

- Ecma International (http://www.ecma-international.org/publications/standards/Ecma-376.htm)
- *ISO/IEC 29500-2:2012*, which has 4 parts, which are all freely available at this link (near bottom of the page) (http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html).

Energistics has adopted all of these practices and adapted them to meet the needs of its data-exchange standards, beginning with RESQML V2.0.

- This tailoring of OPC is referred to as the Energistics Packaging Conventions (EPC).
- The resulting package that contains all of the component files is referred to as an Energistics package.

**IMPORTANT!** IT professionals implementing EPC MUST use either OPC libraries or the OPC specification for the majority of their coding. Use this EPC specification to implement the EPC adaptations, which are explained in Chapter 3 (page 10).

### 1.1.3 Why OPC?

Energistics chose to base EPC on OPC for several key reasons, which include:

- OPC is an existing, widely supported standard.
- It addresses the use cases and requirements of Energistics, as explained above. Namely, it specifically supports the loose coupling of multiple document parts into a coherent whole. It has the ability to appear as folder and file structure as well as a single file archive.
- It supports a rich and extensible mechanism for describing relationships between the package and individual parts or files within the package and resources external to the package.
- Various programming languages already provide wide support (read-write libraries) for OPC. These languages include: the Microsoft.NET framework and Java. Energistics SIG members have tested various tool sets, and several vendors, whose applications rely on very large 3D models and digital media, have field-tested it. These libraries handle the parts and relationships of the package.
- It is based on the commonly used ZIP file structure. In the absence of explicit support for EPC, developers can at least open the package and see the list of files using commonly available zip tools. (To open with a ZIP tool: Right-click the file; from the menu, choose "Open with"; and select or navigate to your ZIP tool.)

## 1.2 Audience, Purpose, and Scope

This document is intended for use by information technology (IT) professionals (e.g., software developers, architects, etc.) to help them understand, create, and read an Energistics package.

The scope of this document is to:

- Provide a high-level overview of OPC (the standard on which EPC is based).
- Specify how EPC differs from OPC.

### 1.2.1 EPC Scope

As of this publication, the scope of an Energistics package is:

- **Always transient**. Energistics and its members are discussing use of EPC to support the notion of projects and archives. However, for this version, the focus is only on using EPC to group together files for data exchange.
- **Use with RESQML V2.0.** The current version of an Energistics package may contain files from: RESQML V2.0 and V1.1 and WITSML 1.4.1 (and later versions of these standards). However, currently only RESQML V2.0 has the technology to actually write and read an Energistics package.

## 1.3 Documentation Conventions

This documentation observes the conventions listed in below.

| | Document/Resource | Description |
|---|---|---|
| 1. | Document Hyperlinks: Internal | Though no special text-formatting convention is used:<br>All section, page, and figure numbers in this and all Energistics documents are hyperlinks. The table of contents is also hyperlinked. |

## 1.4 Resource Set

Energistics provides the following resources to help understand and implement EPC.

| Document / Resource | Description |
|---|---|
| *Energistics Packaging Conventions Specification* (this document) | Rules and guidelines for implementing EPC and a brief overview of OPC, on which EPC is based. |

For information about metadata standards used by Energistics, see the following:

| Document/Resource | Description |
|---|---|
| *Energy Industry Profile of ISO 19115-1 (EIP)*<br>Link to Web page:<br>http://www.energistics.org/asset-data-management/energy-industry-profile-standard | An open, non-proprietary exchange standard for metadata used to document information resources, and in particular resources referenced to a geographic location, e.g., geospatial datasets and web services, physical resources with associated location, or mapping, interpretation, and modeling datasets.<br>It is an ISO Conformance Level 1 profile of the published international standard ISO 19115-1:2014, which is the latest version of the mature conceptual specification ISO 19115:2003. |

### 1.4.1 OPC Resources

The table below contains links to the OPC specifications published by Ecma International and ISO.

| | Document / Resource | Description |
|---|---|---|
| 1. | *Standard ECMA-376*<br>*Office Open XML File Formats* | http://www.ecma-international.org/publications/standards/Ecma-376.htm |
| 2. | *ISO/IEC 29500-2:2012*<br>*Information technology -- Document description and processing languages -- Office Open XML File Formats -- Part 2: Open Packaging Conventions* | http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html<br>(All four parts of the standard are freely available at this link; scroll to bottom of the page.) |

# 2 OPC Overview

Because EPC is based on OPC, this chapter has been provided to explain some general concepts about how OPC works.

## 2.1 Basic Characteristics

The basic characteristics of an OPC package are:

- It is a ZIP archive containing various files called *parts* with different (informed) content.
- It may be documented by means of metadata called *core properties.*
- Relationships between parts are formally described.

## 2.2 OPC Key Concepts

This section provides an overview of key OPC concepts that are important to understanding how an OPC package is structured and how it works. **Figure 1** shows an example of an OPC package. The table below lists and describes the key concepts and includes references to more information available in this document.

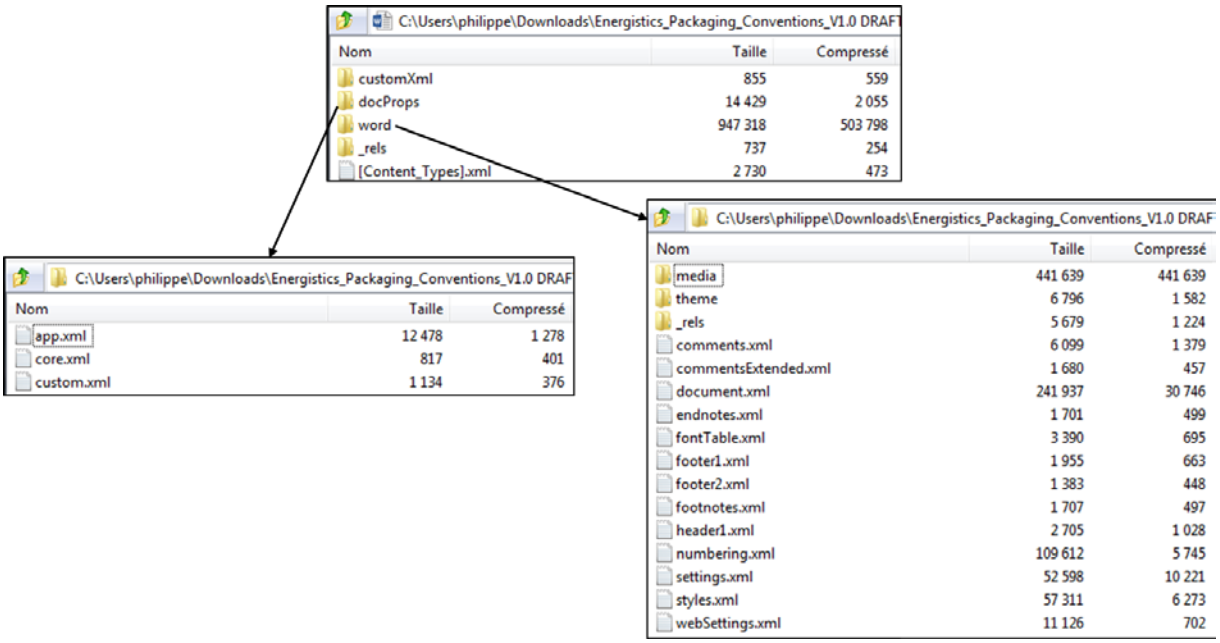


**Figure 1—An example OPC file. Microsoft uses OPC to group together the parts that make up documents (.docx files). This example is the OPC file for the specification document you are currently reading.**

| OPC Concept | Definition, Purpose, and Requirements |
|---|---|
| [Content_Types].xml | This file contains the content type, a description of the type of content of a part, for each part in the package.<br>This file name is a reserved name in OPC.<br>For:<br>• General OPC information, see Section 2.2.1 (below).<br>• EPC-specific information, see Section 3.3 (page 12). |
| Relationship folders and files (_rels) | Parts may contain references to other parts in the package and to resources outside of the package.<br>Uses _rels folders and .rels extensions, which are reserved for this purpose.<br>For:<br>• General OPC information, see Section 2.2.2 (below).<br>• EPC-specific information, see Section 3.4 (page 13). |
| Core Properties | The identifying information of a package, which includes key metadata such as the creator of the package and the date it was created.<br>Core properties are now optional in OPC, but are required for EPC.<br>For EPC-specific information, see Section 3.5 (page 14). |

### 2.2.1 Content Type

The root folder of an OPC package contains a special file with the reserved name *[Content_Types].xml*; this file must always be stored in the root. The purpose of this file is to define a specific mime content type for each part in the package. For more information on how this is used in EPC, see Section 3.3 (page 12).

The mime-type mapping allows a level of indirection from file extensions (which can have different meanings in different systems) and an explicit type that software packages must support. It can also be useful in the case of non-XML content (such as PDF or application files, such as .DOCX) because it provides a hint to the operating system about which viewers and editors support the content.

### 2.2.2 Relationships

OPC provides mechanisms for defining relationships between the package parts, and external resources. The mechanism can be used for both internal parts and external resources, meaning that relationships can actually point outside the package, which includes a file on a Web or file server, a Web service address, or almost anything that can be described by a URI.

Each folder of the package (including the root), has a special folder with the reserved name "_rels". This folder contains files whose names are made up of any file name in the parent folder, with the reserved extension ".rels" appended to it.

The relationships for the package as a whole are defined in the file **/**_rels/.rels. (This is commonly where the location of the core properties file is stored.) The following table explains the key components of the .rels file.

| Attribute | Definition / Purpose |
|---|---|
| Id | A unique name for the relationship. It must be unique within the context of the overall package |
| Target | The file or resource to which the relationship refers. It can point to a part inside the package or to an external resource. |
| Type | Indicates the kind of relationship that exists. |

# 3 EPC-Specific Content and Structure

An Energistics package is an OPC package; this means first and foremost that any standard OPC reader or library should be able to read an Energistics package. This chapter explains the EPC-specific differences that developers must address when implementing EPC.

**IMPORTANT!** IT professionals implementing EPC must use either OPC libraries or the OPC specification for the majority of their coding. Use this EPC specification to implement the EPC adaptations, which are explained in this chapter.

**Figure 2** shows an example Energistics package.

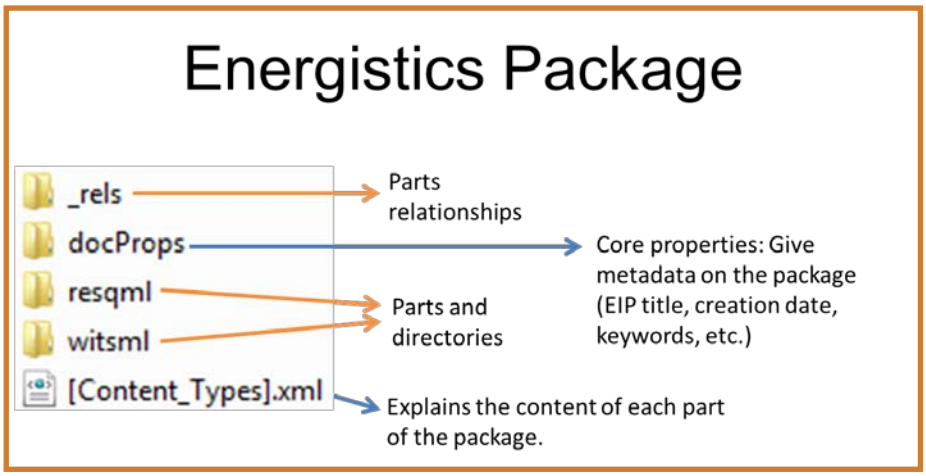


**Figure 2—An example Energistics package. The basic package structure is consistent with OPC; that is; the root folder must contain at least the *_rels* folder and the *[Content_Types].xml* document. An Energistics package must include core properties, which may be stored in a docProps folder. This example includes the data being exchanged in folders labeled *resqml* and *witsml*. All of these *parts* are described below.**

## 3.1 Package Naming Convention

The Energistics package shall have the file extension: ***.epc***.

For example: *MyEarthModel.epc.*

## 3.2 The EPC Parts

The table below lists the additional parts (in addition to the standard OPC parts) used in an Energistics package.

| EPC Part | Definition, Purpose, and Requirements |
|---|---|
| XML data-objects | XML data-objects defined in RESQML V2.0 or data-objects from other Energistics data-exchange standards, such as WITSML or older versions of RESQML, may be included. In Figure 2 above, these parts are stored in the *resqml* and *witsml* folders. |
| EpcExternalPartReference | Any file that is part of an Energistics package but stored externally to the package must have an external part reference that points to the external part.<br>For example, because an HDF5 file is designed for random access (not streaming) and can already compress its data sets, an HDF5 file(s) may be stored outside of an Energistics package. However, to accurately maintain all relationships, the package requires use of an external reference to the HDF5 file.<br>If stored externally to an Energistics package, an HDF5 file(s) must:<br>• Be referenced in the package through an EPC external part reference |

| EPC Part | Definition, Purpose, and Requirements |
|---|---|
| | part.<br>• Have the file extension *.h5*.<br>For more information on HDF5 files, see Section 3.2.3 (page 11). |
| Other types of files | Optionally, other files that contain additional, informal information relevant to the contents of the package, such as these listed below. As a guideline these files are stored in a folder named "media"; for more information, see Section 3.6 (page 15).<br>• PDF<br>• Video<br>• SEGY<br>• Images<br>• Microsoft Word documents |

### 3.2.1 Part Naming Conventions and Content Types

Parts in an EPC package must follow the OPC specification on part names. EPC XML data-objects have additional requirements on their names:

- The name of an EPC XML data-object part shall be composed of the object type and the UUID of the object as in: <objectType>_<uuid>.xml. Examples:
  - obj_IJKGridRepresentation_a8f023cf-de55-47ac-bfb1-b81626f47f6c.xml
  - obj_global2dCRS_afd24701-17fc-4e15-8d88-adc537ead8f7.xml.

The object type of an object is defined to be the XSD complexType of the root element of the data-object.

Following OPC requirements, each part in the EPC must have a mime content type. The EPC requirement is that, in addition to the object type, the content type of each XML data-object contains the name and version of the standard. For a description of the format for the content type, see Section 3.3 (page 12).

### 3.2.2 XML Data-Objects

XML data-objects define the actual data that is being exchanged. These data-objects may be from RESQML V2.0 or from other Energistics data-exchange standards, such as WITSML or older versions of RESQML.

XML data-objects have the following mandatory elements:

- UUID. This is a unique ID scoped to the creator of the package. This means that the ID should be unique for all packages generated by the same creator. The format of the UUID is described in ISO/IEC 9834-8:2008 (which is freely available at http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html) and IETF RFC 4122. NOTE: For UUIDs, RESQML is case-insensitive.
- Metadata citations group including the title, a human readable identification of the data-object. For more information about metadata citations in Energistics standards, see the *Energy Industry Profile of ISO 19115-1 (EIP)* (link to this document available in Section 1.4 (page 6)).

NOTE: A data-object may optionally NOT contain any data. That is, standards like RESQML V2.0 allow reference to data-objects to establish or maintain accurate context for a model, but the actual data may have been transferred previously or may be transferred in the future.

### 3.2.3 EPC External Part References and HDF5 Files

In some cases, it may be desirable to store parts of an Energistics package externally to the EPC file, for example, for HDF5 files.

Hierarchical Data Format (HDF) is a data model, library, and file format for storing and managing data. It supports an unlimited variety of data types, and is designed for flexible and efficient I/O and for high volume and complex data—particularly when compared to XML. HDF version 5 is part of the Energistics Common Technical Architecture and is used in RESQMLV2.0 and will be used in other Energistics standards.

Because an HDF5 file is designed for random access (not streaming) and can already compress its data sets, an HDF5 file(s) may be stored outside of an Energistics package. However, to accurately maintain all relationships, the package requires use of an external reference to the HDF5 file.

The following items describe how to store and reference HDF files in the context of an Energistics package.

- HDF5 files may be stored inside or outside the Energistics package.
- When HDF5 files are used with Energistics standards they shall use the file extension: *.h5*
- If stored inside the Energistics package, the mime type for an HDF5 file is: application/x-hdf5
- If stored outside, it is recommended that the HDF5 files are stored in the same location as the Energistics package.
- If stored outside, each HDF5 file must have an EPC external part reference, which is a proxy that points to the HDF5 file. This reference must be stored inside the Energistics package.
  - The HDF5 external part reference must have a relationship file that defines the actual location of the physical HDF5 file.
  - The corresponding entry in the relationship file must have a type attribute set to: http://schemas.energistics.org/package/2012/relationships/externalResource
  - As an XML data-object, an external part reference must have a UUID, which must be included as an attribute of the physical HDF5 file to allow cross validation.
  - The format of the UUID in the HDF5 file must be of data type *c_s1* (from HDF5 documentation) in its canonical format (lower case letters only).
- Multiple HDF files can be used to describe the array data of multiple Energistics parts. Although not recommended, one XML data-object can reference multiple EPC external part references. However one EPC external part reference can be associated with only one HDF5 file.
  - The XML data-object must reference the HDF external part reference.
  - The corresponding entry in the relationship file must have a type attribute set to: http://schemas.energistics.org/package/2012/relationships/mlToExternalPartProxy
  - The corresponding backward entry (in the rel file of the proxy) must have a type attribute set to: http://schemas.energistics.org/package/2012/relationships/externalPartProxyToMl

## 3.3 Content Types

The file named [Content_Types].xml is used to associate file name extensions used in the package with specific mime content type. For general information about Content Types in OPC, see Section 2.2.1 (page 9).

The following code example shows examples of content types specified for an Energistics package.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Types xmlns="http://schemas.openxmlformats.org/package/2006/content-types">
    <Default
        Extension="rels"
        ContentType="application/vnd.openxmlformats-package.relationships+xml" />
    <Override
        PartName="/docProps/core.xml"
        ContentType="application/vnd.openxmlformats-package.core-properties+xml"/>
    <Override
        PartName="/EpcExternalPartReference_63f74a6b-6ac5-405c-843a-b82ef8f89b33.xml"
        ContentType="application/x-resqml+xml;version=2.0;type=EpcExternalPartReference"/>
    <Override
```

```
        PartName="/obj_Global2dCrs_cea79118-0e71-4cea-b570-69c321d98646.xml"
        ContentType="application/x-resqml+xml;version=2.0;type=obj_Global2dCrs"/>
</Types>
```

Use these rules to define content in the Energistics package:

- [ContentTypes].xml is mandatory in OPC, so is mandatory in EPC.
- An optional media folder may contain any type of media file—no restrictions. For example, it can be used to store any type of supporting documents you might want to include in an Energistics package, such as graphics, videos, reports, etc.).
- Each XML data-object part must have a describing 'contentType'. The content type must follow rfc 2616: http://www.w3.org/Protocols/rfc2616/rfc2616-sec3.html#sec3.7 and must have the following format:

  application/x-**<energisticsStandard>**+xml;version=**<versionNumber>**;type=**<objectType>**

  where:

  - energisticsStandard is the name of the standard used, e.g. “resqml”
  - versionNumber is the major schema version number used, e.g. “2.0”
  - objectType is the XML Schema type of the root element of the XML data-object

  Examples are:

```
"application/x-resqml+xml;version=2.0;type=obj_EpcExternalPartReference"
"application/x-resqml+xml;version=2.0;type=obj_FaultInterpretation"
"application/x-resqml+xml;version=2.0;type=obj_Global1dCrs"
"application/x-resqml+xml;version=2.0;type=obj_TriangulatedSetRepresentation"
```

## 3.4 Relationships

EPC depends strongly on the OPC mechanism for defining relationships between parts. It is primarily used to store the relationships between various XML data-objects and between data and proxy objects. For general information about specifying relationships in OPC, see Section 2.2.2 (page 9).

This section provides rules and guidelines for specifying relationships in an Energistics package. The following code is an example _rels file, showing the relationship between a fault interpretation and a local 3D CRS, plus one external relationship to an HDF5 file.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships">
        <Relationship
                Id="_f99e7434-3a4d-4a65-a15c-9bad78889b2f"
                Type="http://schemas.energistics.org/package/2012/relationships/destinationObject"
                Target="obj_FaultInterpretation_f99e7434-3a4d-4a65-a15c-9bad78889b2f.xml"/>
        <Relationship
                Id="_6f178884-210a-4b27-b56d-f1b2826dc20b"
                Type="http://schemas.energistics.org/package/2012/relationships/destinationObject"
                Target="obj_Local3dCrs_6f178884-210a-4b27-b56d-f1b2826dc20b.xml"/>
        <Relationship
                Id="_ffbd74d6-9e41-4428-9189-810d32c1b2a0"

        Type="http://schemas.energistics.org/package/2012/relationships/mlToExternalPartProxy"
                Target="obj_EpcExternalPartReference_ffbd74d6-9e41-4428-9189-810d32c1b2a0.xml"/>
</Relationships>
```

### 3.4.1 Rules for Specifying Relationships

Observe the following rules and guidelines when specifying relationships in EPC:

- If a data-object is broken into many files, every file (part) must be linked together with _rels. Example: WITSML contains very large log files, which are often chunked into several smaller files.
- Every XML data-object that contains an element of type DataObjectReference is required to have that relationship documented within the OPC relationship system.

- The relationship file must define all the forward and backward relationships of the XML data-object. So for every relationship between two XML data-objects there are two .rels files; one for the source object and one for the destination object.
- Relationships to non-XML data-objects can be documented but are optional.
- If an XML element contained in an Energistics package is involved in one conceptual relationship, then the Energistics package part must have an associated rels file. These XML files can be from any of the Energistics data-exchange standards including: RESQML, WITSML, and PRODML. Rels files for other Energistics package parts (jpg, pdf, etc.) are optional.
- The ID of the relationships follows the OPC format: it is an XML id. It must start with a letter or an underscore (not a digit). It is acceptable for the ID to always start with an underscore.
- The Type of a relationship shall be one of the Energistics defined types as given in Section 3.4.2 (below).

### 3.4.2 Relationship Types and Namespaces

The relationship type defines the role of the relationship. Energistics defines the following valid types.

The fully qualified relationship type is defined as:

http://schemas.energistics.org/package/2012/relationships/<Type>

where <Type> is one of the listed types in the following table.

| Type | Role |
|---|---|
| destinationObject | The object in Target is the destination of the relationship. |
| sourceObject | The current object is the source in the relationship with the target object. |
| mlToExternalPartProxy | The target object is a proxy object for an external data object (HDF5 file). |
| externalPartProxyToMl | The current object is used as a proxy object by the target object. |
| externalResource | The target is a resource outside of the EPC package. Note that TargetMode should be “External” for this relationship. |
| destinationMedia | The object in Target is a media representation for the current object. As a guideline, media files should be stored in a "media" folder in the root of the package. |
| sourceMedia | The current object is a media representation for the object in Target. |
| chunkedPart | The target is part of a larger data-object that has been chunked into several smaller files. |

## 3.5 Core Properties

### 3.5.1 Existence

Contrary to the OPC standard, the Core Properties part is mandatory in an Energistics package. The Core Properties part contains key identification metadata.

### 3.5.2 Location

The location of the Core Properties part is specified in the OPC standards in chapter 11 as being the target of a well-defined package relationship. This means that the /_rels/.rels file in an EPC package must contain at least the following relationship:

```
<Relationship
Id="<unique id>"
Type="http://schemas.openxmlformats.org/package/2006/relationships/metadata/core-properties"
Target="<location>"/>
```

Where:

- <unique id> is the identifier of the relationship (only needs to be unique within the relationship file).

- <location> is the location of the core properties part within the Energistics package. A suggested location is in the “docProps” folder.
- As specified in the OPC specifications, the content type of a Core Properties part is: application/vnd.openxmlformats-package.core-properties+xml

### 3.5.3 Contents

For an Energistics package, the Core Properties part should contain:

- creator (mandatory). Free text.
- created (mandatory). This field must correspond to the package creation time, given in the W3CDTF format.
- version (mandatory). The EPC specification version that the package is based on (currently 1.0).
- description (optional). Free text.
- identifier (optional). URN with UUID.
  - See http://www.ietf.org/rfc/rfc2141.txt
  - Example : `urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6`
- keywords (optional). Free keyword.
  - If various keywords are given in one unique string (possibly caused by the mixed attribute being set to "true" in the complex type “CT_Keywords”), then it is assumed that the delimiter between keywords is a semicolon. In this situation, trailing and leading spaces are not allowed between the key words.
- title (optional). Free text.
- All other core properties specified in OPC are assumed NOT to be given.

## 3.6 Directory Layout

This EPC specification imposes no additional rules for directory layout. However, as a guideline, directory layouts should be clearly organized and named for easy reading. For example:

- “Media” directory for media files.
- “RESQML” directory for RESQML files.
- “WITSML” directory for WITSML files.
- Etc.