Energistics Identifier Specification v5.0

Energistics Identifier Overview	In the context of digital workflows in upstream oil and gas, the concept and ability to identify specific business and data objects is crucial to the accuracy of analysis and results.
Version of Standard	5.0
Version of Document	1.0
Date published	May 16, 2022
Prepared by	Energistics Technical Architecture Team
Abstract	This document describes the syntax and semantics of data object and dataspace identifiers as used within the Energistics family of data transfer standards and the Energistics Transfer Protocol (ETP). It also defines formats for data object references and data object component references. It provides the relevant definitions and rules that MUST be observed beginning with the specified version of the Energistics standards listed in Chapter 1 of this document.
Document type	Specification
Language	U.S. English
Keywords:	standards, energy, data, information, process

ENERGISTICS
ADOPT > ADVANCE > ACCELERATE

<table>
<tr><th colspan="4">Amendment History</th></tr>
<tr><th>Standard Version</th><th>Document Version</th><th>Date</th><th>Comment</th></tr>
<tr><td>5.0</td><td>1.0</td><td>May 16, 2022</td><td>Changes made to this Specification address changes required/initiated by design requirements of Energistics Transfer Protocol v1.2 (published Sept 2021).
Note that the ETP v1.2 Specification contains an Appendix named Energistics Identifiers. That appendix was a precursor to this document and is the basis for this document. While no intentional behavioral changes have been made from that appendix to this document, some clarifications and correction to wording have been made.
Summary of changes to this version of the specification from the previously published Energistics Identifier Specification :
• The contentType has been replaced with qualifiedType for both URIs and data object references.
• Significant clarification of allowed URI formats, specification of canonical Energistics URIs, and specification of regular expressions (regX) for each.
• For data object references, additional attributes have been specified.
• A new mechanism, data object component reference is defined.</td></tr>
</table>

Table of Contents

1 Introduction

This document:

- Defines key terminology and concepts that are important to the identification of business objects, data objects, and dataspaces when used in Energistics standards.
- Specifies the syntax and semantics of data object and dataspace identifiers—which are defined as canonical Energistics URIs in Chapter 3—as used in the Energistics Transfer Protocol (ETP).
- Specifies the syntax and semantics of data object references (DORs), an Energistics mechanism that allows an Energistics data object to reference other Energistics data objects and of data object component references (DOCRs), an Energistics mechanism that allows a data object to a reference a particular component(s) in another data object.
- Provides the relevant definitions and rules that MUST be observed beginning with the specified version of these Energistics standards:
  - Energistics Transfer Protocol (ETP) v1.2
  - RESQML v2.2
  - WITSML v2.1
  - PRODML v2.2
  - Energistics *common* v2.3 (namespace EML), a defined set of data objects shared by the three domain standards listed above.

1.1 Audience, Purpose and Scope

This document is intended for developers implementing any of the Energistics standards listed above. It defines key terminology and Energistics requirements, guidelines, and rules for:

- Use of universally unique identifiers (UUIDs).
- Construction and use of uniform resource identifiers (URIs) to identify Energistics data objects and dataspaces (if used) when using the Energistics Transfer Protocol (ETP) v1.2.
- Construction and use of data object references (DORs) and data object component references (DOCRs).

1.2 Documentation

Energistics is committed to providing quality documentation to help people understand, adopt, and implement its standards. As uptake of the standards increases, lessons learned, best practices, and other relevant information will be captured and incorporated into the documentation. Updated versions of the documentation will be published as they become available.

1.2.1 Conventions

This document uses the conventions listed in the following table.

	Document/Resource	Description
1.	Key words	The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY" and "OPTIONAL" in this document are to be interpreted as described in RFC 2119. (http://www.ietf.org/rfc/rfc2119.txt).
2.	Business Rules	Some mandatory behaviors cannot be implemented in schemas and are specified as business rules as shown in the example below. BUSINESS RULE: Array length is the number of cells in the grid or the blocked well.

	Document/Resource	Description
3.	Document Hyperlinks: Document-Internal	In general, though no special text-formatting convention is used, all section, page and figure numbers in Energistics documents are hyperlinks. The table of contents is also hyperlinked. The ETP Specification v1.2 is the ONLY Energistics document that DOES use formatted text to indicate links.

2 Definitions and Key Concepts

This chapter provides definitions of basic terms and key concepts for understanding how to construct Energistics identifiers, data object references, and data object component references.

2.1 Definitions

Term	Definition
business object	A physical or conceptual object worth identifying. In Energistics a business object is identified by a UUID. For information about UUIDs, see Section 2.3.1.
data object	A digital representation of a business object. A data object for a given business object can have multiple versions, multiple formats, and exist in multiple locations. Energistics domain data model specifications—WITSML, RESQML, PRODML and EML (i.e., Energistics *common*, which is shared by the other 3 MLs)—define data objects, which represent real-world business objects such as wellbores, logs, channels, earth models, faults, production reports, and PVT data, to name a few. For more information on: • Energistics data objects, see Section 2.1.1. • How to identify data objects, see Section 3.4.2.
data object reference (DOR)	In upstream oil and gas workflows, the relationship(s) between various data objects is crucial for business and technical workflows. Energistics has defined a standard mechanism to identify these relationships. For information on how to specify a DOR, see Section 2.4. NOTE: Energistics data models are defined as graphs, where the graph nodes represent data objects, and the line between the nodes represent the relationships between the data objects, many of which are defined by an Energistics DOR. For more information on data models as graphs, see the *Energistics Transfer Protocol (ETP) Specification v1.2* and the *Energistics Common Technical Architecture Overview Guide v2.3*.
data object component reference (DOCR)	A DOR that additionally references one or more components in the data object being referenced. This is new in Energistics *common* v2.3. For information on how to specify a DOCR, see Section 2.5.
dataspace	A dataspace is an abstract concept representing a distinct group of data objects. Dataspaces have been kept as general as possible to support a variety of use cases. Energistics standards do not assign a specific meaning to dataspaces, but different use cases may use a dataspace to represent a project on disk, a specific database, a specific tenant in a multi-tenant store, a specific back-end data store, etc. **EXAMPLE:** When working on a large oil and gas asset, it is common to organize work into projects. Subsequently, the data may also be organized and stored as projects. In this type of organization, it is possible for the same data object to be worked on by multiple project teams and exist in multiple locations (data stores), and as multiple versions. In these situations, users keep track of which projects they are working on and which data store the project is stored in. ETP dataspaces may be used to represent these different data stores and the projects in them. URIs for data objects within the projects will be prefixed with the project's dataspace URI. Use of dataspaces is optional and implementation-dependent. When dataspaces are used, they are specified in an Energistics URI. For more information on specifying dataspaces in Energistics URIs, see Section 3.4.1 and Section 3.4.2.1.

2.1.1 Energistics Data Objects

A **data object** defined by an Energistics specification is a valid document of the specified format (XML, JSON, other), which conforms to one of the schemas specified in the Energistics namespace and inherits from AbstractObject, which is defined in Energistics *common*.

Energistics has these broad categories of data objects, each of which has some specific considerations when being operated upon by the various sub-protocols that make up ETP:

- **"static" data objects**. These are informally referred to as "static" (compared to "growing"; see below) because they change only when people, process, and/or software change them. Additionally, they may have a "main" object (sometimes called a "header" object) and associated arrays of numeric data.
- **"growing" data objects**. Objects that change inherently over time by adding to them. These objects typically exist in the drilling domain and are defined in WITSML, such as trajectories (grows as new trajectory stations are added), and "mud logs" (now called wellbore geology) which grow in several ways with the evaluation and recordings at different intervals for geological cuttings samples, lithology sequences along the length of the wellbore, and interpretations of the quality of hydrocarbon shows along the wellbore).
- **channel data objects**. A channel is a series of values, usually measured or calculated, that are referenced to one or more indexes, usually time or depth. Groups of channels are informally called “logs” and individual channels are sometimes referred to as “curves”. Channels are similar to growing data objects, but they are important enough and different enough to be treated as a distinct type of object. In ETP, the channel protocols are dedicated to handling channel data.
- **"contained" and "container" data object.** A contained data object refers to a data object that is contained by another data object (the container) with a ByValue reference (and ONLY a ByValue reference; i.e., relationships specified by an Energistics Data Object Reference (DOR) do not result in container/contained objects). An Energistics data object MAY be included in one or more container data objects.

  One of the best-known examples comes from WITSML where:

  - One or more Channel data objects can be contained in one or more ChannelSet data objects. In this example, the Channels are the "contained" data objects and the Channel Set is the "container".
  - One or more ChannelSet data objects can be contained in one or more Log data objects. In this example, the ChannelSets are the "contained" data objects and the Log is the "container".

    NOTE: Individual data objects that may be containers or contained data objects are listed in the relevant ML's ETP implementation specification (which is a companion document to the *ETP Specification*). For example, Channel, Channel Set and other contained data objects defined in WITSML are listed in the *ETP v1.2 for WITSML v2.0 Implementation Specification*.

2.2 Mechanism for Identification: URIs

An Energistics uniform resource identifier (URI) provides a mechanism to identify dataspaces and data objects. Energistics URIs are formatted according to RFC 3986 (Uniform Resource Identifier (URI): Generic Syntax (https://tools.ietf.org/html/rfc3986)), and, beginning with ETP v1.2, Energistics URIs are based on OData URI syntax (http://docs.oasis-open.org/odata/odata/v4.01/odata-v4.01-part2-url-conventions.html) with some tailoring as described in Chapter 3.

- The components that are used to construct Energistics URIs are defined in Section 2.3.
- The details of how to use these components to construct URIs are specified in Chapter 3.

2.3 Components of Energistics Identifiers

Energistics URIs are constructed from the components identified and defined below. For additional related definitions, see Section 2.1.

For details of how to use these components to construct URIs for dataspaces and data objects, see Chapter 3.

2.3.1 UUID

A unique instances of **business object** defined by Energistics data models must be identified with a **UUID** as defined by RFC 4122 (https://tools.ietf.org/html/rfc4122) and ISO/IEC 9834-8:2008 (which is freely available at http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html). A UUID is an array of 16 unsigned bytes (or a single 128-bit unsigned integer), and can be printed and serialized in various ways. Additionally, implementations of Energistics standards must follow these rules:

- For case sensitivity of UUIDs, Energistics standards follow the IETF which states:
  The hexadecimal values "a" through "f" are output as lower case characters and are case insensitive on input.
  RECOMMENDATION: When creating a string representation of a UUID for use with Energistics standards, the hexadecimal values "a" through "f" SHOULD be lower case when they appear in the UUID. When comparing the string representation of two UUIDs, the comparison MUST be case insensitive.
- For use in Energistics domain standards, for string representation of a data object, a UUID MUST be serialized using Microsoft Registry Format; that is, with dashes inside the UUID and without curly braces. **EXAMPLE:** 00112233-4455-6677-8899-aabbccddeeff
- For use in ETP messages–with the exception of string representation of data objects that may be conveyed with a message (as described in the previous bullet) –ETP uses the ***Uuid*** datatype (Section 23.6 in the ETP v1.2 Specification) to send UUIDs. The ***Uuid*** data type is encoded as an array of 16 bytes in big-endian format.
  EXAMPLE: The UUID “00112233-4455-6677-8899-aabbccddeeff” is encoded as the byte array **[0x00, 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77, 0x88, 0x99, 0xaa, 0xbb, 0xcc, 0xdd, 0xee, 0xff]** in the ***Uuid*** data type.
- The application that first creates a data object of a unique business object assigns its UUID. If during data transfer an application changes the UUID of a data object, that application MUST preserve the original UUID as an alias and it is up to the application to change the authority.

2.3.2 Object Version

The object version is the version of a data object. It appears as an attribute (objectVersion) on the AbstractDataObject schema as part of Energistics *common*. (For more information, see the *Energistics common Technical Reference Guide v2.3.*)

The object version is used to indicate a data object has been changed. The updated version makes it possible for reading applications to determine that a data object has changed without having to do a full read and compare. When a static data object is modified, it is recommended that its version is modified.

2.3.3 Data Object Type

In ETP, a data object type (dataObjectType) is the semantic equivalent of a qualifiedEntityType in OData. (In Energistics documentation, the terms "data object type", "qualified entity type", and "qualified type" are semantically equivalent.)

The newest format of the Energistics data object reference (DOR) also uses a qualified type, which replaces the previously used ContentType; for more information, see Section 2.4.

The data object type is composed of:

- The Energistics domain standard or Energistics *common* (designated by *eml*) and version where the data object type is defined.

- The data object type name as defined by its schema.

Examples:

- witsml20.Well
- resqml20.obj_UnstructuredGridRepresentation (NOTE: Use of this obj_ convention has been discontinued as part of the Energistics v2 architecture; RESQML v2.0.1 is the last published Energistics domain standard to use it.)
- prodml20.ProductVolume
- eml21.DataAssuranceRecord

You MUST follow these rules to create a valid data object type:

1. The Energistics domain standard MUST be one of the three Energistics domain standards or *eml* (for data object types that are defined in Energistics *common* for use by the domain standards), all in lower case concatenated with the first 2 digits of its version. See the examples in the list above.
2. The data object type name MUST be the schema name of the data object as defined in the Energistics standard. It IS case sensitive.

2.4 Data Object Reference

The data object reference (DOR) (**Figure 2-1**) (DataObjectReference, in the ObjectReference folder of Energistics *common* schemas) provides a way for an Energistics data object to reference other Energistics data objects. For more information about how DORs work, see the *Energistics common Technical Architecture Overview Guide v2.3.*

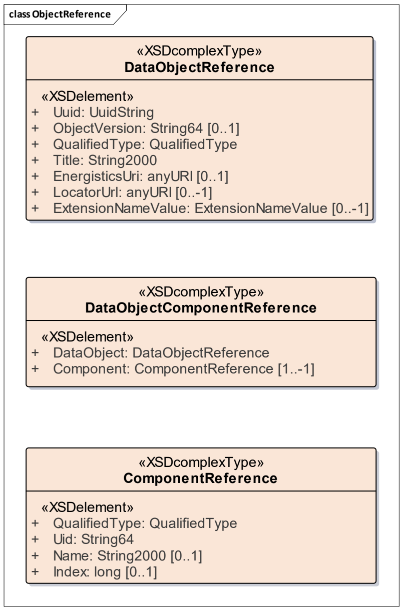
class ObjectReference
«XSDcomplexType»
DataObjectReference
«XSDelement»
+ Uuid: UuidString
+ ObjectVersion: String64 [0..1]
+ QualifiedType: QualifiedType
+ Title: String2000
+ EnergisticsUri: anyURI [0..1]
+ LocatorUrl: anyURI [0..-1]
+ ExtensionNameValue: ExtensionNameValue [0..-1]
«XSDcomplexType»
DataObjectComponentReference
«XSDelement»
+ DataObject: DataObjectReference
+ Component: ComponentReference [1..-1]
«XSDcomplexType»
ComponentReference
«XSDelement»
+ QualifiedType: QualifiedType
+ Uid: String64
+ Name: String2000 [0..1]
+ Index: long [0..1]

Figure 2-1. The DataObjectReference (DOR) and DataObjectComponentReference (DOCR) from the ObjectReference UML package and diagram.

NOTE: Version 2.3 of Energistics *common* also provides a new mechanism called data object component reference (DataObjectComponentReference); for more information, see Section 2.5.

Examples of usage of data object reference:

- In WITSML, it is used to create the well/wellbore/<object> hierarchy, where <object> can be a log, wellbore geology, or trajectory station.

- In RESQML, it is used to create relationships between earth modeling data objects--faults, horizons, geobodies, grids, properties, etc.--to define models, not just independent objects.

2.4.1 Two Versions of Data Object Reference

With the publication of Energistics *common* v2.3, the format of the data object reference has changed. The main reason for the changes were due to design changes required by the publication of the Energistics Transfer Protocol (ETP) v1.2 and its increasing use of URIs.

Summary of changes in v2.3:

- The ContentType has been replaced with QualifiedType (for consistency with ETP v1.2).
- Addition of an optional URI to specify a reference to a data object in an ETP server.
- Addition of other optional elements as specified in Section 2.4.2.

2.4.1.1 Usage Rules for versions of DORs

With the publication of *common* v2.3, there are now two formats to specify DORs—the format specified in v2.3 and the format specified "pre-v2.3" (**NOTE:** The format of DOR in common v2.0, v2.1, and v2.2 is the same.) An implementations MUST observe these rules for the version of DOR it uses:

- An Implementation MUST use the DOR for the version of the ML/common in which it is creating the DOR. **NOTE:** This rule is enforced through schema validation.
  EXAMPLES:
  - RESQML v2.0.1 was published with Energistics *common* v2.0, so it must use the DOR published in that version of *common*.
  - WITSML v2.0 was published with Energistics *common* v2.1, so it must use the DOR published in that version of *common*.
  - RESQML v2.2, WITSML v2.1 and PRODML v2.2 will be published with Energistics *common* v2.3, so they must use the new version of the DOR described here.
- Any v2.x data object in an ML can point to any v2.x data object in another ML.
  EXAMPLES:
  - RESQML v2.0.1 can point to a well or wellbore in WITSML v2.0 or WITSML v2.1.
  - RESQML v2.2 can point to a well or wellbore in WITSML v2.0 or WITSML v2.1.

2.4.2 DOR Format for common v2.3

The table below lists the parts of a data object reference (DOR) in Energistics *common* v2.3 and specifies if the part is optional or required. The format and optionality are also enforced with the DataObjectReference schema found in the ObjectReference folder of Energistics *common* v2.3.

Attribute	Description/Reference	Optional/ Required
Uuid	Reference to a data object using its universally unique identifier. For information about format requirements, see Section 2.3.1.	Required
ObjectVersion	For the definition, see Section 2.3.2. If the objectVersion is not present, then the latest version of the data object is assumed. For information, see Section 3.4.2.1.	Optional
QualifiedType	For the definition and format, see Section 2.3.3.	Required
Title	This is the Title from the Citation element on AbstractObject. For more information, see the *Energistics common v2.3 Technical Reference Guide*. It is used as a hint for human readers; it is not enforced to match the Title of the referenced object.	Required

Attribute	Description/Reference	Optional/ Required
EnergisticsUri	The canonical URI of a referenced data object. Use of this element is intended for use with the Energistics Transfer Protocol (ETP). Do not use this element to store the path and file names of an external data object. Optionally use one or more LocatorUrl elements to provide hints on how to resolve the URI into a data object.	Optional
LocatorUrl	An optional location to help in resolving the correct referenced data object. (It's a hint.)	Optional
ExtensionNameValue	A standard Energistics extension mechanism to add custom data using name:value pairs.	Optional

2.5 Data Object Component Reference

The data object component reference (DOCR) (Figure 2-1, above) (DataObjectComponentReference, in the ObjectReference folder of Energistics *common* v2.3 schemas) provides a way for a data object to reference a specific component in another data object; **EXAMPLE:** A data object could reference a specific trajectory station (TrajectoryStation) in a WITSML Trajectory data object.

A DOCR is a DataObjectReference with one or more components (ComponentReference in Figure 2-1, whose details are listed in the table below). For details of the DOR format, see Section 2.4. The ETP Specification describes required behavior for navigating DORs in a data model.

To find the specific component specified by a DOCR requires use of an ETP query protocol or getting the data object and inspecting the XML (or JSON) to understand which specific component is being referred to. If more than one component is listed, simply work through the list in the specified order.

Attribute	Description/Reference	Optional/ Required
QualifiedType	For the definition and format, see Section 2.3.3.	Required
Uid	The UID of the referenced component.	Required
Name	The human-readable name of the referenced component.	Optional
Index	The numerical (i.e., NOT time or depth) index of the referenced component.	Optional

3 Energistics URIs

An Energistics uniform resource identifier (URI) provides a mechanism to identify dataspaces and data objects. Energistics URIs are formatted according to RFC 3986 (Uniform Resource Identifier (URI): Generic Syntax (https://tools.ietf.org/html/rfc3986)), and, beginning with ETP v1.2, Energistics URIs are based on OData URI syntax (http://docs.oasis-open.org/odata/odata/v4.01/odata-v4.01-part2-url-conventions.html) with some tailoring as described here.

IMPORTANT: The URI formats specified here are for use with the versions of Energistics standards specified on page 5. Previous versions of Energistics standards used a different URI format (e.g., ETP v1.1) or did not use URIs at all.

This section:

- Defines the term canonical URI.
- Specifies the form of canonical Energistics URIs for dataspaces, data objects, and queries that will match a collection of data objects within a dataspace and provides examples.
- Defines so-called "alternate" URIs and their usage.
- Provides regular expressions (REGEXes) and examples of each.

3.1 Requirements for Supporting URIs

An endpoint in an ETP session:

- MUST support the canonical Energistics URIs.
- MUST support **eml:///**, which is the URI for the default dataspace, which may or may not be empty.
- MAY support alternate URI formats, which are explained in Section 3.4.4.

3.2 Overview of Energistics URIs

Energistics URIs provide a flexible way to identify dataspaces and data objects within dataspaces. By building on OData URI syntax, Energistics URIs can represent:

- individual dataspaces and data objects
- hierarchical relationships between objects
- sub-elements within data objects
- queries for collections of objects

ETP v1.2+ uses a subset of Energistics URIs to identify:

- Dataspaces
- Individual data objects within a dataspace
- A query that will match a collection of data objects within a dataspace

When both ETP endpoints in a session can support them, ETP v1.2+ allows optional use of other forms of Energistics URIs in some protocol messages, which may have application-specific meaning. For more information on optional use of other URI forms, see Sections 3.4 and Section 3.4.4.

3.3 URI Notation

When describing the form of URIs in this document:

- { } indicate a parameter that is substituted with an actual value
- [ ] indicate an optional portion of the URI, which may be omitted

3.4 Canonical Energistics URIs

Because of the flexibility of Energistics URIs, many URIs can be semantically equivalent—that is, they identify the same unique dataspace or data object or they represent the same query.

A **canonical URI** is the preferred, and often shortest, URI out of a set of semantically equivalent URIs.

In ETP, observe these rules about use of canonical URIs:

- ETP endpoints MUST support canonical Energistics URIs. In some ETP messages, their use is always required. In other messages, their use is required unless both ETP endpoints in the session support alternate forms of Energistics URIs.
- Even when the use of canonical URIs is optional, use of canonical URIs MUST always be supported.

This section defines the form of canonical Energistics URIs for ETP v1.2+.

3.4.1 Dataspace URIs

Dataspace URIs identify individual dataspaces, which contain zero or more data objects. ETP supports named dataspaces, which use a path as a name, and the default dataspace, which has no name.

- The canonical form of the default dataspace URI is:
  eml:///
- The canonical form for named dataspace URIs is:
  eml:///dataspace('{path}')
- An example dataspace URI is:
  eml:///dataspace('/folder-name/project-name')
- For named dataspaces, the path may be a relative path. For example:
  eml:///dataspace('rdms-db')

Observe these rules for dataspace URIs:

- In addition to named dataspaces, all ETP stores and producers MUST support the default, nameless dataspace, which is identified by the empty string.
  - While the default dataspace MUST be supported, it MAY be empty; that is, it may not have any data objects in it.

IMPORTANT: The default dataspace is NOT an alias for a named dataspace. It is a simplification for ETP stores and producers that do not need to support named dataspaces.

3.4.2 Data Object URIs

A data object URI is one that provides direct reference to a single data object in a dataspace contained behind an ETP endpoint. A data object URI may optionally refer to a specific version of a data object.

- The canonical form of a data object URI without a version is:
  eml:///[dataspace('{path}')/]{DataObjectType}({uuid})
- The canonical form of a data object URI with a version is:
  eml:///[dataspace('{path}')/]{DataObjectType}(uuid={uuid},version='{version}')

Example data object URIs are:

- eml:///witsml20.ChannelSet(2c0f6ef2-cc54-4104-8523-0f0fbaba3661)
- eml:///dataspace('rdms-db')/resqml20.obj_HorizonInterpretation(uuid=421a7a05-033a-450d-bcef-051352023578,version='2.0')

3.4.2.1 Rules for Using Dataspaces and Version in Data Object URIs

Observe these rules for using dataspaces and version in data object URIs:

- If the dataspace is the default dataspace, then the dataspace segment MUST be omitted from the canonical URI.

- Data objects identified by Energistics URIs are always in a dataspace, so the data object URI is prefixed with the relevant dataspace.
  - If the dataspace is omitted from the URI, then the URI implicitly refers to the default dataspace.
- If version is omitted and there are multiple versions of a data object behind an ETP endpoint, then the URI implicitly refers to the most recent version.
  - ETP 1.2 does not provide rules that define which of two versions of a data object is the most recent version. The data object version that is most recent is ETP-endpoint-dependent.
  - If the intent is to refer to the most recent version of the data object, then the version segment SHOULD be omitted from the canonical URI.

The data object URI uses these conventions from OData:

- The data object type in a data object URI is semantically equivalent to an OData qualifiedEntityType; for example: witsml20.Well (as described in Section 2.3.3).
- The specification of the uuid and the optional version are semantically equivalent to keys in OData collections.

3.4.3 Data Object Query URIs

A data object query URI is one that refers to a collection of data objects in a dataspace contained behind an ETP endpoint.

The canonical form of a data object query URI MUST be one of the following:

- **{DataObjectUri}/{DataObjectType}[?{query}]**
- **eml:///[dataspace('{path}')]/{DataObjectType}[?{query}]**
- **{DataObjectUri}?{query}**

Example data object query URIs are:

- **eml:///dataspace('rdms-db')/resqml20.obj_HorizonInterpretation**
- **eml:///witsml20.Well(uuid=ec8c3f16-1454-4f36-ae10-27d2a2680cf2,version='1.0')/witsml20.Wellbore**
- e**ml:///witsml20.Channel?$filter=ChannelClass/Title eq 'Gamma'&$top=300**

3.4.3.1 Rules for Using Dataspaces and Version in Data Object Query URIs

Observe these rules for using dataspaces and version in data object query URIs:

- If the dataspace is the default dataspace, then the dataspace segment MUST be omitted from the canonical URI.
- If the intent is to refer to the most recent version of the data object, then the version segment SHOULD be omitted from the canonical URI.
- A data object query URI MAY specify an OData Entity Collection. That is, a data object type without associated uuid or version. This represents a query for objects of the specified type.
- When a data object query URI includes a specific data object uuid, the query operates on data objects that have some relationship to the data object specified by the uuid.
- Whether the relationship is primary or secondary or goes from sources to targets or targets to sources depends on other contextual information where the URI is used. **EXAMPLE:** In DiscoveryQuery, the *context* and *scope* fields on the ***FindResources*** message will provide this information.
- A data object query URI MAY also include a URI query string (for details, see Chapter **Error! Reference source not found.**).
  - When the URI path ends with an OData Entity Collection, the query string is optional (because the OData Entity Collection represents an implicit query).
  - When the URI path ends with a specific data object, the query string is required.

3.4.4 Alternate URIs

In some situations, ETP v1.2+ also allows applications to use so-called **alternate URIs**. These URIs MUST be valid Energistics URIs, but they need not be canonical URIs. Alternate URIs may have application-specific meaning.

NOTE: For alternate URIs to be used in an ETP session, the store MUST return the allowed alternate formats in Discovery (Protocol 3). For more information, see Chapter 8 of the *ETP v1.2 Specification*.

Here is a non-exhaustive list of alternate URI forms and examples of them:

1. URIs prefixed with eml:/ instead of eml:///
2. Hierarchical data object URIs where more than one path segment uniquely identifies a data object and the path indicates a navigable relationship between the objects.
   eml:///witsml20.Well(ec8c3f16-1454-4f36-ae10-27d2a2680cf2)/witsml20.Wellbore(81bb7920-fa42-48cb-b9ac-38031e2703a8)
3. Template URIs where multiple path segments specify a data object type.
   eml:///witsml20.Well/witsml20.Wellbore
4. URIs with hash segments.
   eml:///resqml20.obj_HorizonInterpretation(421a7a05-033a-450d-bcef-051352023578)#hash
5. Dataspace URIs with query segments.
   eml:///dataspace('rdms-db')?$filter=Name eq 'mydb'
6. URIs with path segments that address elements within data objects.
   eml:///witsml20.Channel(53b3bf2b-3aa3-458d-b40c-9a4cb754210e)/ChannelClass/Title
7. URIs that include the primary key name:
   eml:///witsml20.Well(uuid=ec8c3f16-1454-4f36-ae10-27d2a2680cf2)
8. URIs that use alternate OData keys:
   eml:///witsml20.ChannelSet(2c0f6ef2-cc54-4104-8523-0f0fbaba3661)/witsml20.Channel(Mnemonic='HKLD')
9. URIs that support earlier ML versions:
   eml:///witsml14.well(uid='abc')/witsml14.wellbore(uid='def')

Observe these rules for using alternate URIs:

- To use alternate URIs in an ETP session, BOTH ETP endpoints MUST have set the SupportsAlternateRequestUris endpoint capability to "true" in the ***RequestSession*** and ***OpenSession*** messages (in their respective *endpointCapabilities* fields) that were exchanged to establish the ETP session.
- ETP does not provide functionality for an endpoint to advertise all possible alternate URIs it supports.
  - An ETP endpoint that wants to use alternate URIs in requests SHOULD assume the other endpoint in the session supports only alternate URIs it has explicitly received in response to previous requests.
  - Even if an endpoint indicates it supports alternate URIs, it is NOT required or guaranteed that all possible forms of alternate URIs are supported.

3.4.5 Regular Expressions for Validating Canonical Energistics URIs

The following regular expressions can be used to validate canonical Energistics URIs. These regular expressions use ECMAScript regular expression syntax.

- **Canonical Dataspace URIs:**

  ```
  ^eml:\/\/\/(?:dataspace\('(?<dataspace>[^']*?(?:''[^']*?)*)'\))?$
  ```

  EXAMPLES:

  - eml:///
  - eml:///dataspace('/folder-name/project-name')
  - eml:///dataspace('rdms-db')

- **Canonical Data Object URIs:**

  ^eml:\/\/\/(?:dataspace\('(?<dataspace>[^']*?(?:''[^']*?)*)'\)\/)?(?<domain>witsml|resqml|prodml|eml)(?<domainVersion>[1-9]\d)\.(?<objectType>\w+)\((?:(?<uuid>[0-9a-fA-F]{8}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{12})|uuid=(?<uuid2>[0-9a-fA-F]{8}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{12}),version='(?<version>[^']*?(?:''[^']*?)*)')\)$

  EXAMPLES:

  - eml:///witsml20.Well(ec8c3f16-1454-4f36-ae10-27d2a2680cf2)
  - eml:///witsml20.Well(uuid=ec8c3f16-1454-4f36-ae10-27d2a2680cf2,version='1.0')
  - eml:///dataspace('/folder-name/project-name')/resqml20.obj_HorizonInterpretation(uuid=421a7a05-033a-450d-bcef-051352023578,version='2.0')

- **Canonical Data Object Query URIs with both Data Object and OData Entity Collection:**

  ^eml:\/\/\/(?:dataspace\('(?<dataspace>[^']*?(?:''[^']*?)*)'\)\/)?(?<domain>witsml|resqml|prodml|eml)(?<domainVersion>[1-9]\d)\.(?<objectType>\w+)\((?:(?<uuid>[0-9a-fA-F]{8}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{12})|uuid=(?<uuid2>[0-9a-fA-F]{8}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{12}),version='(?<version>[^']*?(?:''[^']*?)*)')\)\/(?<collectionDomain>witsml|resqml|prodml|eml)(?<collectionDomainVersion>[1-9]\d)\.(?<collectionType>\w+)(?:\?(?<query>[^#]+))?$

  EXAMPLES:

  - eml:///witsml20.Well(ec8c3f16-1454-4f36-ae10-27d2a2680cf2)/witsml20.Wellbore?query
  - eml:///witsml20.Well(uuid=ec8c3f16-1454-4f36-ae10-27d2a2680cf2,version='1.0')/witsml20.Wellbore?query
  - eml:///dataspace('/folder-name/project-name')/witsml20.Well(uuid=ec8c3f16-1454-4f36-ae10-27d2a2680cf2,version='1.0')/witsml20.Wellbore?query

- **Canonical Data Object Query URIs with OData Entity Collection but no Data Object:**

  ^eml:\/\/\/(?:dataspace\('(?<dataspace>[^']*?(?:''[^']*?)*)'\)\/)?(?<collectionDomain>witsml|resqml|prodml|eml)(?<collectionDomainVersion>[1-9]\d)\.(?<collectionType>\w+)(?:\?(?<query>[^#]+))?$

  EXAMPLES:

  - eml:///witsml20.Well?query
  - eml:///dataspace('/folder-name/project-name')/resqml20.obj_HorizonInterpretation?query

- **Canonical Data Object Query URIs with Data Object but no OData Entity Collection:**

  ^eml:\/\/\/(?:dataspace\('(?<dataspace>[^']*?(?:''[^']*?)*)'\)\/)?(?<domain>witsml|resqml|prodml|eml)(?<domainVersion>[1-9]\d)\.(?<objectType>\w+)\((?:(?<uuid>[0-9a-fA-F]{8}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{12})|uuid=(?<uuid2>[0-9a-fA-F]{8}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{12}),version='(?<version>[^']*?(?:''[^']*?)*)')\)\?(?<query>[^#]+)$

  EXAMPLES:

  - eml:///witsml20.Well(ec8c3f16-1454-4f36-ae10-27d2a2680cf2)?query
  - eml:///witsml20.Well(uuid=ec8c3f16-1454-4f36-ae10-27d2a2680cf2,version='1.0')?query
  - eml:///dataspace('/folder-name/project-name')/resqml20.obj_HorizonInterpretation(uuid=421a7a05-033a-450d-bcef-051352023578,version='2.0')?query

- ^(?<domain>witsml|resqml|prodml|eml|custom)(?<domainVersion>[1-9]\d)\.(?<objectType>\w+)$