

---

# EDRM XML Interchange Format Schema Documentation

## Table of Contents

1. Overview .....	1
2. EDRM XML File Format Reference .....	2
2.1. XML Prolog .....	3
2.2. Root Element .....	3
2.3. Batch Element .....	4
2.3.1. CustomBatchInfo Element .....	5
2.3.2. Documents Element .....	5
2.3.3. Folders Element .....	5
2.3.4. Relationships Element .....	5
2.4. Document Element .....	5
2.4.1. CustomDocumentInfo Element .....	5
2.4.2. Files Element .....	6
2.4.3. Locations Element .....	7
2.4.4. Reviews Element .....	7
2.4.5. Tags Element .....	8
2.5. Folder Element .....	8
2.5.1. Folder Document Element .....	9
2.6. Relationship Element .....	9

This document corresponds to EDRM XML XSD Version 1.0, **31-Oct-2007**.

## 1. Overview

Launched in May 2005, the Electronic Discovery Reference Model (EDRM) Project was created to address the lack of standards and guidelines in the electronic discovery market. The completed reference model provides a common, flexible and extensible framework for the development, selection, evaluation and use of electronic discovery products and services. The completed model was placed in the public domain in May 2006.

The goal of the EDRM XML project is to provide a standard, generally accepted XML schema to facilitate the movement of electronically stored information (ESI) from one step of the electronic discovery process to the next, from one software program to the next, and from one organization to the next. The ESI includes both underlying discovery materials (e.g., email messages and attachments, loose files, and databases) and information about those materials (e.g., the source of the underlying ESI, processing of that ESI, and production of that ESI).

If deployed properly, the schema should help:

- Reduce the costs of moving data from one step to the next, one program to the next, one organization to the next
- Reduce errors associated with that movement of data
- Reduce electronic discovery software development costs and shorten development cycles
- Increase the transparency of the entire electronic discovery process

## 2. EDRM XML File Format Reference

Figure 1, “ Sample XML File That Conforms to the EDRM Schema ” shows an example of an XML file that conforms to the schema discussed in the document. Specific element content has been removed from this figure to simplify viewing the structure of conformant files. This example file shows all elements and attributes that can be present in such a file—not all elements and attributes are mandatory.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Root RootFilePath="/temp/"
  MinorVersion="0" MajorVersion="1"
  Locale="US" Description="Test Case"
  DataInterchangeType="Update" CaseId="Case1">
  <Batch name="Sample Batch">
    <Documents>
      <Document MimeType="text/plain" DocType="Text File" DocID="1">
        </Document>
      <Document MimeType="text/plain" DocType="Text File 2" DocID="2">
        <Tags>
          <Tag TagValue="Tag Value??" TagName="Tag Name??"
            TagDataType="LongText" ModifiedBy="Jane Doe">
          </Tag>
        </Tags>
        <Files>
          <File FileType="7bit ASCII Doc">
            <ExternalFile MergeFileNum="0" MergeFileCount="0"
              Hash="MD5" FileSize="1000"
              FilePath="c:\" FileName="data.txt">
            </ExternalFile>
          </File>
        </Files>
        <Reviews>
          <Review ReviewId="1">
            <Tag TagValue="Tag Value??" TagName="Tag Name??"
              TagDataType="LongText" ModifiedBy="Jane Doe">
            </Tag>
          </Review>
        </Reviews>
        <Locations>
          <Location>
            <Custodian
              xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xmlns:xs="http://www.w3.org/2001/XMLSchema"
              xsi:type="xs:string">
              John Doe
```

```
        </Custodian>
        <LocationURL>ATL</LocationURI>
    </Location>
</Locations>
</Document>
</Documents>
<Relationships>
    <Relationship Type="NearDupe" ParentDocId="2" ChildDocId="1">
    </Relationship>
</Relationships>
<Folders>
    <Folder FolderParentName="" FolderName="SampleFolder">
        <Folder FolderParentName="SampleFolder" FolderName="SampleFolder2">
        </Folder>
        <Document DocId="1">
        </Document>
        <Document DocId="2">
        </Document>
        </Folder>
    </Folder>
</Folders>
</Batch>
</Root>
```

**Figure 1. Sample XML File That Conforms to the EDRM Schema**

Subsequent sections of this document describe each element and associated attributes in XML input files that conform to the EDRM XML Interchange Schema.

## 2.1. XML Prolog

This standard XML declaration identifies the file as an XML file that conforms to the XML Specification [<http://www.w3.org/TR/2000/REC-xml-20001006>]. The **version** attribute is the only mandatory attribute, and must contain the version of the XML specification to which the document conforms (currently, **1.0**). See the **XML specification** for information on other, optional attributes that are available.

## 2.2. Root Element

The **root** element is the primary node of a document in EDRM XML Interchange format. The root node is the parent of all documents being transmitted in this format.

The only valid child of the **Root** node is the **Batch** element.

Possible attributes of the root element are the following:

- **CaseId** - (optional, string) - A string used to identify the case with which the XML data is associated.
- **DataInterchangeType** - (required, action) - The type of data interchange being done between systems, and therefore the intended purpose of the XML file. Possible values are the following:

- **Delete** - used to delete documents and annotations already in the target product.
- **Update** - used to update documents and annotations that have already been imported into a target product, or to create new entries if they do not already exist. You can update existing native files, TIFF files, and text elements. Updating existing instances of any of these elements will over-write any current contents in the target product.

If this option is not specified, the default value used is **Update**.

- **Description** - (required, free form) - Text that describes the overall contents of the XML file that is being transmitted.
- **Locale** - (optional, string) - The locale for the data contained in the XML file. This should be a two-letter code that matches an ISO-3166 locale. The default is "US" (United States). The target product can use this format to determine the expected format of dates contained in the files referenced by the XML file.

For a complete list of valid ISO-3166 country codes, see the International Organization for Standards [<http://www.iso.org/iso/en/prods-services/iso3166ma/index.html>].

- **MajorVersion** - (string, optional) The major version of the EDRM Schema to which this XML file conforms. if this attribute is empty or not provided, the default value is **1**.
- **MinorVersion** - (string, optional) The minor version of the EDRM Schema to which this XML file conforms. if this attribute is empty or not provided, the default value is **0**.
- **RootFilePath** - (optional, string) The full pathname of the directory where any documents that need to be loaded using this XML file are located. If or not present, file locations are assumed to be relative to the directory in which the XML file is located.

## 2.3. Batch Element

The **Batch** element is a container node for a set of flat or hierarchical documents and related information about them. A single EDRM XML file can contain multiple **Batch** elements. Possible attributes of the Batch element are the following:

- **name** - (optional, string) a name to associate with this batch

Valid children of the **batch** node are the **CustomBatchInfo**, **Documents**, **Folders**, and **Relationships** elements.

### 2.3.1. CustomBatchInfo Element

The **CustomBatchInfo** element is an optional, free-form element that provides additional information about the documents contained in a **Batch** element.

A **CustomBatchInfo** node has no children and therefore cannot contain any other elements.

### 2.3.2. Documents Element

The **Documents** element is a container node for one or more **Document** elements. A **Documents** node has no children other than the **Document** element.

### 2.3.3. Folders Element

The **Folders** element is a container for one or more **Folder** elements. A **Folders** node has no children other than the **Folder** element.

### 2.3.4. Relationships Element

The **Relationships** element is a container for one or more **Relationship** elements. A **Relationships** node has no children other than the **Relationship** element.

## 2.4. Document Element

The **Document** element is a container node for various elements that provide information about each document that is being transmitted. Possible attributes of the **Document** element are the following:

- **DocID** - a unique identifier for a given document
- **DocType** - Identifies the type of a given document, either **Message** for message-type documents, or **File** for all others. The value of this attribute can be used to simplify parsing the EDRM-specific metadata tags that are specific to each **DocType** and are discussed in ???.
- **MimeType** - Identifies the MIME type of the document. See the MIME Reference [[http://www.w3schools.com/media/media\\_mimeref.asp](http://www.w3schools.com/media/media_mimeref.asp)] for more information about MIME types.

Valid children of the **Document** node are the **CustomDocumentInfo**, **Files**, **Locations**, **Reviews**, and **Tags** elements.

### 2.4.1. CustomDocumentInfo Element

The **CustomDocumentInfo** element is an optional, free-form element that provides

additional information about the document associated with a given Document element.

A **CustomDocumentInfo** node has no children and therefore cannot contain any other elements.

## 2.4.2. Files Element

The **Files** element is a container for one or more File elements. A **Files** node has no children other than the File element.

### 2.4.2.1. File Element

Each **File** element identifies a specific type of file associate with the current Document element. Possible attributes of the **File** element are the following:

- **FileType** - Identifies the type of a file. Valid values in EDRM 1.0 are **Image**, **Native**, **Redacted**, and **Text**.

Valid children of the **File** node are the ExternalFile and InlineContent elements.

#### 2.4.2.1.1. ExternalFile Element

The **ExternalFile** element provides information about a file that holds part of the content for a Document, but is stored in a separate file. Possible attributes of the **ExternalFile** element are the following:

- **FileName** - (required, string) - Identifies the name of the external file, as specified in the XML Schema [<http://www.w3.org/2001/XMLSchema>]
- **FilePath** - (optional, string) - Identifies the full or absolute path to the file, as specified in the XML Schema [<http://www.w3.org/2001/XMLSchema>]. If this attribute is not specified, the **FilePath** is the same as the path to the XML file.
- **FileSize** - (integer, optional) - the size of the file, in bytes.
- **Hash** - (string, required) - the hash of the file in hexadecimal string format, used for checksumming and to ensure that no changes have been made to the external file. The default hash type is MD5, which is the only hash algorithm that is currently supported.
- **MergeFileCount** - (integer, optional) - identifies the total number of external files that are referenced in an XML document. This enables multiple external files to be specified and combined into a single larger file.
- **MergeFileNum** - (integer, optional) - the sequence number of an external files. using this sequence number in conjunction with the **MergeFileNum** attribute enables multiple external files to be specified and combined into a single larger file. bytes.

An **ExternalFile** node has no children.

#### 2.4.2.1.2. InlineContent Element

The **InlineContent** element enables binary or text file content to be embedded directly within an XML file. For EDRM XML 1.0, this is only supported for files whose File-Type is **Text**.

An **InlineContent** node has no children or attributes.

### 2.4.3. Locations Element

The **Locations** element is a LocationType container for one or more Location elements. Every Document may contain location information identified by a LocationType. Location elements provide information about particular occurrences of a document in the original data source(s). A **Locations** node has no children other than the Location element.

#### 2.4.3.1. Location Element

Each **Location** element provides information about the location and ownership of a specific Document element. This element is optional.

Valid children of a **Location** node are the string LocationURI and Custodian elements.

##### 2.4.3.1.1. LocationURI Element

A string **Location** element provides a URI of an instance of the associated Document element. This element is required for each Location element.

##### 2.4.3.1.2. Custodian Element

A **Custodian** element is a free-form element that specifies the name of the custodian who owned the document at this particular location. This element is required for each Location element.

### 2.4.4. Reviews Element

The **Reviews** element is a container for one or more Review elements. A **Reviews** node has no children other than the Review element.

#### 2.4.4.1. Review Element

Each **Review** element provides a collection of information related to a user or group of users who have examined and marked up a document with annotations. This element is optional.

Possible attributes of the **Review** element are the following:

- **ReviewId** - (integer, required) a unique identifier for a specific **Review** element.

The only valid children of a **Review** element are one or more **Tag** elements.

## 2.4.5. Tags Element

The **Tags** element is a container for one or more **Tag** elements. A **Tags** node has no children other than the **Tag** element.

### 2.4.5.1. Tag Element

The **Tag** element provides a mechanism for associating comments or other external data with other elements. This element is optional.

Possible attributes of the **Tag** element are the following:

- **ModifiedBy** - (optional) - a string values that provides the name of the person who last modified the associated document
- **ModifiedOn** - (optional) - provides the date (in **datetime** format) on which the associated document was last modified
- **TagDataType** - (required) - specifies the data type of the associated **Tag** element. Possible values are **Boolean**, **DateTime**, **Decimal**, **Integer**, **LongInteger**, **LongText**, and **Text**.
- **TagName** - (required) - Used as an identifier for a given **Tag** element. This can either be an EDRM-compliant standard tag name (such as **#CC**) or a custom string. These names are case sensitive.
- **TagValue** - (required) - contains the actual tag value or the name of a file containing tag value information. The contents of this attribute must match the type specified in the **TagDataType** attribute.

## 2.5. Folder Element

The **Folder** element is used to hierarchically organize the documents contained in an EDRM import file. This element is optional. If this element is provided, the specified folder hierarchy must already exist in the target product. Possible attributes of the **Folder** element are the following:

- **FolderName** - (long, required) - a unique identifier for a given folder
- **FolderParentName** - (string, required) - the name of the folder in which the current folder is located.

Valid children of the **Folder** node are other Folder elements and the Folder Document element.

### 2.5.1. Folder Document Element

The **Folder Document** element identifies a document that is contained in a Folder hierarchy. This element is required at some level of a nested set of Folder elements. Possible attributes of the **Folder Document** element are the following:

- **DocumentID** - (string, required) - provides the name or ID of a Document element that is provided elsewhere in the EDRM interchange file.

## 2.6. Relationship Element

The **Relationship** element is used to provide information about the relationship between one document and others.

Possible attributes of the **Relationship** element are the following:

- **ChildDocID** - (string, required) -
- **ParentDocID** - (string, required) - a unique identifier for a given folder
- **Type** - (string, required) - This is the type of the relationship between this document and another. Valid values are **Attachment** (for email attachments), **Container** (for ZIP/PST and other archive-format files), **Discussion** (for threads), and **NearDupe**, for documents that have been identified as near duplicates of others.

The **Relationship** node cannot contain any other elements.

