[MS-DPEDMX]:
Entity Data Model for Data Services Packaging Format Data Portability Overview

---

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft Open Specification Promise or the Community Promise. If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.
- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications do not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments you are free to take advantage of them. Certain Open Specifications are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

Revision Summary

Date	Revision History	Revision Class	Comments
06/04/2010	0.1	Major	First release.
09/03/2010	0.1	No change	No changes to the meaning, language, or formatting of the technical content.
02/09/2011	0.1	No change	No changes to the meaning, language, or formatting of the technical content.
07/07/2011	0.1.1	Minor	Clarified the meaning of the technical content.

Contents

1 Introduction 4
1.1 Glossary 4
1.2 References 4

2 Data Portability Scenarios 6
2.1 Exporting the Schema for Vendor Consumption 6
2.1.1 Data Description 6
2.1.2 Format and Protocol Summary 6
2.1.3 Data Portability Methodology 6
2.1.3.1 Preconditions 7
2.1.3.2 Versioning 7
2.1.3.3 Error Handling 7
2.1.3.4 Coherency Requirements 7
2.1.3.5 Additional Considerations 7
2.2 Using the Data for Client Code Generation 7
2.2.1 Data Description 7
2.2.2 Format and Protocol Summary 7
2.2.3 Data Portability Methodology 8
2.2.3.1 Preconditions 8
2.2.3.2 Versioning 8
2.2.3.3 Error Handling 8
2.2.3.4 Coherency Requirements 8
2.2.3.5 Additional Considerations 8

3 Change Tracking 9

4 Index 11

1 Introduction

An Entity Data Model for Data Services Packaging Format (EDMX) document is an XML-based file format that serves as the packaging format for the service **schema** of a **data service**. As specified in [MC-APDSU], clients can obtain the service schema for a data service from the $metadata URI that has the following signature:

http://<host>/<prefix>/<service path>/$metadata

The data service returns the schema packaged in an EDMX document. The root of an EDMX document is an <edmx:Edmx> element, which contains exactly one <edmx:DataServices> subelement. The <edmx:DataServices> subelement contains zero or more <Schema> subelements, which specify Entity Data Model (EDM) conceptual schemas. These **EDM** conceptual schemas are annotated as specified in [MC-APDSU].

Conceptually, the structure of an EDMX document is similar to the following example.

```
<edmx:Edmx>
  <edmx:DataServices>
    <!-- Entity Data Model conceptual schemas, as specified in
         [MC-CSDL] and annotated as specified in [MC-APDSU]-->
    <Schema>
    </Schema>
    <!--
       Additional Entity Data Model conceptual schemas as
       specified in [MC-CSDL] and annotated as specified in [MC-APDSU]
    -->
  </edmx:DataServices>
</edmx:Edmx>
```

The contents of an EDMX document are determined by the data service in question and vary depending on the data service, as specified in [MC-APDSU].

1.1 Glossary

The following terms are defined in [MC-CSDL]:

EDM
schema

The following terms are defined in [MC-APDSU]:

data service
link
service operation

The following terms are defined in [MC-EDMX]:

<edmx:Edmx>
<edmx:DataServices>

1.2 References

[MS-ODATA] Microsoft Corporation, "Open Data Protocol (OData) Specification", February 2009.

[MC-CSDL] Microsoft Corporation, "Conceptual Schema Definition File Format".

[MC-EDMX] Microsoft Corporation, "Entity Data Model for Data Services Packaging Format".

2 Data Portability Scenarios

2.1 Exporting the Schema for Vendor Consumption

The schema export scenario describes exporting the data service schema from a data service to an .xml file to be consumed by a vendor's application. This section provides a step-by-step description and references for exporting a schema to an .edmx file.

2.1.1 Data Description

The EDMX document contains the schema of a data service as an .xml file. The EDMX document includes information about entities that are exposed by the data service, constraints on those entities, and **links** between those types. The document also includes descriptions of any **service operation** that is exposed by the data service.

The EDMX document is used to describe the metadata; the EDMX document is used by a data service client application to perform code generation and query generation. The metadata is always generated automatically by the data service.

2.1.2 Format and Protocol Summary

The following table provides a comprehensive list of the formats and protocols used in this scenario.

Protocol or format name	Description	Reference
Entity Data Model for Data Services Packaging Format	The Entity Data Model for Data Services Packing Format document contains the schema of a data service as an .xml file.	[MC-EDMX]

2.1.3 Data Portability Methodology

For this scenario, the EDMX document is extracted from the data service and stored in an .xml file. This is done by using any HTTP-capable application to query the $metadata endpoint that is exposed by the data service. The $metadata endpoint always returns the entire EDMX document for the data service.

To extract the EDMX document from the data service

1. Create an .edmx file on the client machine to store the document in.
2. Using any HTTP-enabled client application (such as a Web browser or Fiddler2) or an HTTP request generation API (such as HTTPWebRequest), execute an HTTP GET request to the $metadata endpoint at the root of the data service.

   For example, if the root of the data service is "http://myhost/myservice.svc", execute an HTTP GET request to the following URI:

   http://myhost/myservice.svc/$metadata

   Note If using a browser, the view window for the browser may add additional HTML markup that is not in the original EDMX document. In this case, use the **View Source** feature to get the original .edmx data file.

3. Copy the response to the HTTP GET request. For example, copy the text in the viewing pane from a browser, or copy the response stream from an HTTPWebRequest.

4. Write the response to the .edmx file that was created in step 1.

2.1.3.1 Preconditions

The data service must be running and reachable from the client.

2.1.3.2 Versioning

This version of export data scenario is applicable to ADO.NET Data Services in the Microsoft® .Net Framework 3.5 Service Pack 1 (SP1).

2.1.3.3 Error Handling

None.

2.1.3.4 Coherency Requirements

There are no special coherency requirements.

2.1.3.5 Additional Considerations

There are no additional considerations.

2.2 Using the Data for Client Code Generation

This scenario describes using the vendor's data that is contained in the .edmx file to generate client proxy classes that can interact with a data service. The vendor can produce an .edmx file that conforms to the format that is defined in [MC-EDMX], and can use the data for generating client proxy classes.

2.2.1 Data Description

The EDMX document contains the schema of a data service as an .xml file. The document includes information about entities that are exposed by the data service, constraints on those entities, and links between those types. The document also includes descriptions of any service operation that is exposed by the data service.

The EDMX data is used to describe the data and is used by a data service client application to perform code generation and query generation. The EDMX data is always generated automatically by the data service.

2.2.2 Format and Protocol Summary

The following table provides a comprehensive list of the formats and protocols used in this scenario.

Protocol or format name	Description	Reference
Entity Data Model for Data Services Packaging Format	The EDMX document contains the schema of a data service as an .xml file.	[MC-EDMX]

2.2.3 Data Portability Methodology

The approach in this section is to use the DataSvcUtil.exe utility that is provided with Microsoft® Visual Studio® 2008 with Service Pack 1 (SP1) to generate C# or Microsoft® Visual Basic® classes based on the .edmx file (stored as an .edmx file).

To perform client proxy code generation

1. Open a Visual Studio 2008 with SP1 command prompt on the machine.
2. Change to the directory that contains the .edmx file.
3. Use the DataSvcUtil.exe command-line utility to generate client classes from the .edmx file. To do this, follow these steps:

    1. Type datasvcutil /in:filename.edmx /out:outputfilename.cs
    2. Press ENTER.

The .cs file that is created contains the proxy classes.

2.2.3.1 Preconditions

Microsoft® Visual Studio® 2008 with Service Pack 1 (SP1) must be installed.

2.2.3.2 Versioning

This version of import data scenario is applicable to Microsoft® Visual Studio® 2008 with Service Pack 1 (SP1) and Microsoft® .Net Framework 3.5 Service Pack 1 (SP1).

2.2.3.3 Error Handling

None.

2.2.3.4 Coherency Requirements

There are no special coherency requirements.

2.2.3.5 Additional Considerations

There are no additional considerations.

3 Change Tracking

This section identifies changes that were made to the [MS-DPEDMX] protocol document between the February 2011 and July 2011 releases. Changes are classified as New, Major, Minor, Editorial, or No change.

The revision class **New** means that a new document is being released.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements or functionality.
- An extensive rewrite, addition, or deletion of major portions of content.
- The removal of a document from the documentation set.
- Changes made for template compliance.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **Editorial** means that the language and formatting in the technical content was changed. Editorial changes apply to grammatical, formatting, and style issues.

The revision class **No change** means that no new technical or language changes were introduced. The technical content of the document is identical to the last released version, but minor editorial and formatting changes, as well as updates to the header and footer information, and to the revision summary, may have been made.

Major and minor changes can be described further using the following change types:

- New content added.
- Content updated.
- Content removed.
- New product behavior note added.
- Product behavior note updated.
- Product behavior note removed.
- New protocol syntax added.
- Protocol syntax updated.
- Protocol syntax removed.
- New content added due to protocol revision.
- Content updated due to protocol revision.
- Content removed due to protocol revision.
- New protocol syntax added due to protocol revision.

- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.
- New content added for template compliance.
- Content updated for template compliance.
- Content removed for template compliance.
- Obsolete document removed.

Editorial changes are always classified with the change type **Editorially updated.**

Some important terms used in the change type descriptions are defined as follows:

- **Protocol syntax** refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.
- **Protocol revision** refers to changes made to a protocol that affect the bits that are sent over the wire.

The changes made to this document are listed in the following table. For more information, please contact protocol@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
1.2 References	Updated reference citation of [MC-APDSU] to reflect updated name: [MS-ODATA]: Open Data Protocol (OData) Specification.	N	Content updated.

4 Index

C

Change tracking 9

G

Glossary 4

T

Tracking changes 9