[MS-DPEDM]:
Entity Data Model Data Portability Overview

This document provides an overview for data portability in the Conceptual Schema Definition Language (CSDL), Store Schema Definition Language (SSDL), and Mapping Specification Language (MSL) file formats. It is intended for use in conjunction with the Microsoft protocol technical specifications, publicly available standard specifications, network programming art, and Microsoft Windows distributed systems concepts. It assumes that the reader is either familiar with the aforementioned material or has immediate access to it.

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
09/03/2010	0.1	New	Released new document.
02/09/2011	0.2	Minor	Clarified the meaning of the technical content.
07/07/2011	0.2	No change	No changes to the meaning, language, or formatting of the technical content.
11/03/2011	0.2	No change	No changes to the meaning, language, or formatting of the technical content.

Contents

1 Introduction

The Entity Data Model (EDM) is an **entity**-relationship model that is used to define an application's data model. The data model is represented by using entities and the relationships (that is, **associations**) between those entities.

Three specifications are pertinent to EDM:

- [MC-CSDL]: Conceptual Schema Definition File Format. Conceptual Schema Definition Language (CSDL) is a specification language that is used for defining the neutral conceptual mode, which is independent of the storage model used for persisting application data. Because the conceptual model is neutral and independent, application implementers can build a conceptual model in a way that is independent of persistence concerns.
- [MS-SSDL]: Store Schema Definition File Format. Store Schema Definition Language (SSDL) is a specification language that is used to formally describe the database schema used for persisting the data of an application built on top of EDM.

  The SSDL definition is similar to a CSDL definition; the primary difference between them is that the data types used in SSDL declarations are the actual types supported by a particular database product engine.
- [MS-MSL]: Mapping Specification File Format. Mapping Specification Language (MSL) is a specification language that is used to connect the types that are declared in CSDL to store the schema definition that is declared in SSDL.

1.1 Glossary

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

association
entity

1.2 References

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

[MS-MSL] Microsoft Corporation, "Mapping Specification Format".

[MS-SSDL] Microsoft Corporation, "Store Schema Definition File Format".

2 Data Portability Scenarios

2.1 Exposing an Instance of a Data Model by Using CSDL

A third-party product or service can use CSDL to expose a data model so that the data model uses the concepts of EDM and CSDL to describe the structure and some of the semantics that pertain to the data model. Consumers of this data model can deduce facts about it and use it for various scenarios at a higher level of abstraction without needing to know where the data is physically stored.

Examples of this scenario include the following:

- A customer relationship management (CRM) or enterprise resource planning (ERP) product uses CSDL to describe a data model as a way to express the structure of entities and their relationships. Implementers who are building solutions on top of this product can use the schema to deduce facts about this data model without having to concern themselves about its physical schema.
- EDM is used as the schema format for enterprise scenarios such as reporting, business intelligence, synchronization, and so on. Building these scenarios on top of EDM eliminates the need to be concerned about the physical store of the data so that the user can focus on the conceptual/domain rules that govern the data model.

2.1.1 Data Description

The data in this scenario is the CSDL metadata that is used to represent the data model. MSL and SSDL are not part of this data portability scenario.

2.1.2 Format and Protocol Summary

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

Protocol or format name	Description	Reference
Conceptual Schema Definition Language	Metadata definition language that is used to describe the entities and associations in terms of the business/conceptual domain.	[MC-CSDL]

2.1.3 Reverse Engineering a Database and Creating an EDM Model

A schema from a relational database, such as Microsoft® SQL Server®, can be reverse-engineered into an EDM by using the EDMGen.exe tool that is included in the Microsoft® .Net Framework 3.5 Service Pack 1 (SP1) and in the Microsoft® .NET Framework 4.0. Alternatively, Microsoft® Visual Studio® 2008 with Service Pack 1 (SP1) and Microsoft® Visual Studio® 2010 can be used.

This data portability scenario describes how to use the EDMGen.exe tool to create an EDM.

To reverse-engineer a database and to create an EDM

- From a command prompt, run EDMGen.exe by pointing it to the database that you want to reverse engineer into an EDM.

For example, the following command-line command uses the Northwind database schema that is installed on a local instance of Microsoft® SQL Server® Express. This command reverse engineers an EDM model (CSDL file) along with the corresponding SSDL schema and MSL mapping metadata:

```
edmgen.exe /mode:FullGeneration /connectionString:"server=.\sqlexpress;integrated
security=true;database=Northwind" /project:Northwind /namespace:Northwind
/EntityContainer:NorthwindEntitites
```

2.1.3.1 Preconditions

The database being reverse engineered must exist. Additionally, Microsoft® Visual Studio® 2008, Microsoft® Visual Studio® 2010, and EDMGen.exe require an Entity Framework–enabled ADO.NET provider for connectivity to the database. Database product vendors and third parties offer database providers that enable ADO.NET-based connectivity to relational databases.

2.1.3.2 Versioning

Versioning considerations for CSDL, SSDL, and MSL are documented in [MC-CSDL], [MS-SSDL], and [MS-MSL].

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 Implementing Visual Designers and Tooling for CSDL, SSDL, and MSL

A third party can build a visual design tool that enables its users to build customized, EDM-based models. The tool can support various aspects, such as reverse engineering an existing database, that enable users to create a new model or support various code generation strategies based on the model that is being created.

2.2.1 Data Description

The data in this scenario consists of the CDSL, SSDL, and MSL metadata that is generated by the design tool.

2.2.2 Format and Protocol Summary

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

Protocol or format name	Description	References
Conceptual Schema Definition Language	Metadata definition language that is used to describe the entities and associations in terms of the business/conceptual domain.	[MC-CSDL]
Store Schema Definition Language	Metadata definition language that is used to describe the physical store schema used for persisting the data in a relational database.	[MS-SSDL]
Mapping Specification Language	Mapping language that is used to map the concepts in the store schema to the concepts in the conceptual schema.	[MS-MSL]

2.2.3 Generating and Persisting an EDM–Based Metadata to Disk

In this scenario, the design tool is responsible for the following functionalities:

- Enabling users to build CSDL, SSDL, and MSL metadata.

  The tool should adhere to the syntactic and semantic rules for generating CSDL, SSDL, and MSL metadata so that the schema can then be used and deployed in a variety of scenarios.

- Enabling the data portability scenario described in section 2.1.3
- Performing the functions that are done by the EDM designer in Visual Studio or EDMGen.exe.

There are no prescriptive requirements for how a design tool should connect to the database, generate metadata, or persist XML for the metadata on disk.

The rules for generating metadata that are described in [MC-CSDL], [MS-SSDL], and [MS-MSL] should be followed.

2.2.3.1 Preconditions

None.

2.2.3.2 Versioning

Versioning considerations for CSDL, SSDL, and MSL are documented in [MC-CSDL], [MS-SSDL], and [MS-MSL].

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 Page

No table of changes is available. The document is either new or has had no changes since its last release.

4 Index