[MS-DPIS]:

Integration Services Data Portability Overview

Revision Summary

Date	Revision History	Revision Class	Comments
2/9/2011	0.1	New	Released new document.
7/7/2011	1.0	Major	Updated and revised the technical content.
11/3/2011	1.0	None	No changes to the meaning, language, or formatting of the technical content.
1/19/2012	1.0	None	No changes to the meaning, language, or formatting of the technical content.
2/23/2012	1.0	None	No changes to the meaning, language, or formatting of the technical content.
3/27/2012	1.0	None	No changes to the meaning, language, or formatting of the technical content.
5/24/2012	1.0	None	No changes to the meaning, language, or formatting of the technical content.
6/29/2012	1.0	None	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	1.0	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	1.0	None	No changes to the meaning, language, or formatting of the technical content.
10/23/2012	1.0	None	No changes to the meaning, language, or formatting of the technical content.
3/26/2013	1.0	None	No changes to the meaning, language, or formatting of the technical content.
6/11/2013	1.0	None	No changes to the meaning, language, or formatting of the technical content.
8/8/2013	1.0	None	No changes to the meaning, language, or formatting of the technical content.
12/5/2013	1.0	None	No changes to the meaning, language, or formatting of the technical content.
2/11/2014	1.0	None	No changes to the meaning, language, or formatting of the technical content.
5/20/2014	1.0	None	No changes to the meaning, language, or formatting of the technical content.
5/10/2016	1.0	None	No changes to the meaning, language, or formatting of the technical content.
8/16/2017	1.0	None	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction 5
1.1 Glossary 5
1.2 References 6

2 Data Portability Scenarios 7
2.1 Third-Party Integration Tool or Platform Consuming Integration Services Packages from MSDB Repository 7
2.1.1 Data Description 7
2.1.2 Format and Protocol Summary 7
2.1.3 Data Portability Methodology 7
2.1.3.1 Preconditions 8
2.1.3.2 Versioning 8
2.1.3.3 Error Handling 8
2.1.3.4 Coherency Requirements 8
2.1.3.5 Additional Considerations 8
2.2 Third-Party Integration Platform or Tool Consuming Integration Services Packages in the File System 8
2.2.1 Data Description 8
2.2.2 Format and Protocol Summary 8
2.2.3 Data Portability Methodology 8
2.2.3.1 Preconditions 8
2.2.3.2 Versioning 9
2.2.3.3 Error Handling 9
2.2.3.4 Coherency Requirements 9
2.2.3.5 Additional Considerations 9
2.3 Third-Party Integration Tool or Platform Consuming DTS Packages from MSDB Repository 9
2.3.1 Data Description 9
2.3.2 Format and Protocol Summary 9
2.3.3 Data Portability Methodology 9
2.3.3.1 Preconditions 10
2.3.3.2 Versioning 10
2.3.3.3 Error Handling 10
2.3.3.4 Coherency Requirements 10
2.3.3.5 Additional Considerations 10
2.4 Third-Party Integration Platform or Tool Consuming DTS Packages in the File System 10
2.4.1 Data Description 10
2.4.2 Format and Protocol Summary 10
2.4.3 Data Portability Methodology 11
2.4.3.1 Preconditions 11
2.4.3.2 Versioning 11
2.4.3.3 Error Handling 11
2.4.3.4 Coherency Requirements 11
2.4.3.5 Additional Considerations 11
2.5 Third-Party Integration Tool or Platform Consuming Integration Services Project Deployment Files from SSISDB Repository 11
2.5.1 Data Description 11
2.5.2 Format and Protocol Summary 11
2.5.3 Data Portability Methodology 12
2.5.3.1 Preconditions 12
2.5.3.2 Versioning 12
2.5.3.3 Error Handling 12
2.5.3.4 Coherency Requirements 12
2.5.3.5 Additional Considerations 12

2.6 Third-Party Integration Platform or Tool Consuming Integration Services Project Deployment Files Packages in the File System 13
2.6.1 Data Description 13
2.6.2 Format and Protocol Summary 13
2.6.3 Data Portability Methodology 13
2.6.3.1 Preconditions 13
2.6.3.2 Versioning 13
2.6.3.3 Error Handling 13
2.6.3.4 Coherency Requirements 13
2.6.3.5 Additional Considerations 13
3 Change Tracking 14
4 Index 15

1 Introduction

The Integration Services Data Portability Overview document provides an overview of data portability for SQL Server Integration Services (SSIS). SSIS includes a repository for data integration artifacts, including Integration Services packages, Data Transformation Services (DTS) packages, and Integration Services project deployment files. This repository includes a set of Microsoft SQL Server tables, views, and **stored procedures**, depending on the format that is used.

The following table describes the formats that can be used.

Artifact	Customary file system extension	Documented in	SQL Server object	SQL Server database
Data Transformation Services Package XML package	.dtsx	[MS-DTSX] and [MS-DTSX2]	**sysssispackages** (table)	**msdb**
Data Transformation Services package	.dts	[MS-DTS]	**sysdtspackages** (table)	**msdb**
Integration Services project deployment file	.ispac	[MS-ISPAC]	**catalog.projects** (view) **catalog.get_project** (stored procedure)	**SSISDB**

Data integration artifacts are set or retrieved in this database by using SQL statements that are transmitted over the SQL Server **Tabular Data Stream (TDS)** protocol [MS-TDS]. Or, the artifacts are opened or saved in the file system, where they exist as files that are named with an extension that is based on the customary file system extension, as described in the preceding table.

1.1 Glossary

This document uses the following terms:

connection manager: A component that is referenced by an SSIS package. A connection manager stores the information necessary to establish connections to external resources and establishes and provides these connections, on demand, to other components within the SSIS package.

Open Database Connectivity (ODBC): A standard software API method for accessing data that is stored in a variety of proprietary personal computer, minicomputer, and mainframe databases. It is an implementation of [ISO/IEC9075-3:2008] and provides extensions to that standard.

SQL statement: A character string expression in a language that the server understands.

stored procedure: A precompiled collection of SQL statements and, optionally, control-of-flow statements that are stored under a name and processed as a unit. They are stored in a SQL database and can be run with one call from an application. Stored procedures return an integer return code and can additionally return one or more result sets. Also referred to as sproc.

Tabular Data Stream (TDS): An application-level protocol that is used by SQL Server to facilitate requests and responses between a database server and client as specified in [MS-TDS].

XML: The Extensible Markup Language, as described in [XML1.0].

1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.

[MS-DTSX2] Microsoft Corporation, "Data Transformation Services Package XMLVersion 2 File Format".

[MS-DTSX] Microsoft Corporation, "Data Transformation Services Package XML File Format".

[MS-DTS] Microsoft Corporation, "Data Transformation Services Package File Format".

[MS-ISPAC] Microsoft Corporation, "Integration Services Project Deployment File Format".

[MS-TDS] Microsoft Corporation, "Tabular Data Stream Protocol".

2 Data Portability Scenarios

2.1 Third-Party Integration Tool or Platform Consuming Integration Services Packages from MSDB Repository

2.1.1 Data Description

The DTSX documentation [MS-DTSX] and [MS-DTSX2] contain the definition of a package, which includes information about configured **connection managers**, data sources, destinations, and transformations that are to be applied to data, as well as the ordering of various tasks that are involved in an extraction, transformation, and loading (ETL) package. This data is represented in the DTSX document as **XML**, as documented in [MS-DTSX] and [MS-DTSX2].

This DTSX data is used to effect a set of data movements and transformations, typically from one or more sources to one or more destinations as configured in the package. This DTSX data is created by using the Business Intelligence Design Studio that is included with Microsoft SQL Server or by using the **Microsoft.SqlServer.Dts** object model.

DTSX data can be stored as a file on the file system or in the **msdb** repository as a row in the **sysssispackages** table.

2.1.2 Format and Protocol Summary

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

Format or protocol name	Description	Short name
TDS	This protocol is used to communicate with SQL Server to execute **SQL statements** and retrieve results.	[MS-TDS]

2.1.3 Data Portability Methodology

For this scenario, the documents that contain the DTSX data are extracted from the **msdb** repository one by one and stored in a file on the file system. The method of extracting the DTSX data from the **msdb** repository for use in a third-party integration tool or platform in this scenario is to use the **TDS** protocol that is provided by the SQL Server instance.

To extract the data, follow these steps:

1. Create a folder on the client machine for storing the retrieved DTSX documents.
2. Connect and authenticate to the desired server by using an **ODBC**, OLEDB, or ADO.NET provider or any other TDS implementation.
3. Issue the following **SQL statement**:

```
SELECT name, packagedata from msdb..sysssispackages
```

4. Store the data. To do this, follow these steps for each row that is returned in step 3:
    1. Create a new file in the folder that was created in step 1.
    2. Save the contents of the **packagedata** field to the new file.

5. Use the DTSX documentation [MS-DTSX] or [MS-DTSX2] to interpret the DTSX data that was retrieved in the previous step for use in the third-party integration tool or platform.

2.1.3.1 Preconditions

Ensure that the SQL Server service is started on the server. Grant the appropriate permissions to the user to access the **sysssispackages** table.

2.1.3.2 Versioning

None.

2.1.3.3 Error Handling

None.

2.1.3.4 Coherency Requirements

This data portability scenario has no special coherency requirements.

2.1.3.5 Additional Considerations

There are no additional considerations.

2.2 Third-Party Integration Platform or Tool Consuming Integration Services Packages in the File System

2.2.1 Data Description

The DTSX documentation [MS-DTSX] and [MS-DTSX2] contain the definition of a package, which includes information about configured **connection managers**, data sources, destinations, and transformations that are to be applied to data, and the ordering of various tasks that are involved in an extraction, transformation, and loading (ETL) package. This data is represented in the DTSX document as **XML**, as documented in [MS-DTSX] and [MS-DTSX2].

This DTSX data is used to effect a set of data movements and transformations, typically from one or more sources to one or more destinations as configured in the package. This data is created by using the Business Intelligence Design Studio that is included with SQL Server or by using the **Microsoft.SqlServer.Dts** object model.

2.2.2 Format and Protocol Summary

No formats or protocols are used in this scenario.

2.2.3 Data Portability Methodology

In this scenario, the DTSX data is stored in the file system as packages (*.dtsx files). These files can be stored at any location on a system. Use the DTSX documentation [MS-DTSX] or [MS-DTSX2] to interpret the DTSX data in these files.

2.2.3.1 Preconditions

None.

2.2.3.2 Versioning

None.

2.2.3.3 Error Handling

None.

2.2.3.4 Coherency Requirements

This data portability scenario has no special coherency requirements.

2.2.3.5 Additional Considerations

There are no additional considerations.

2.3 Third-Party Integration Tool or Platform Consuming DTS Packages from MSDB Repository

2.3.1 Data Description

The DTS [MS-DTS] document contains the definition of a DTS package, which includes information about the configuration and order of tasks and data pumps that are applied to data in a DTS package. This data is represented in the DTS document as a binary file, as documented in [MS-DTS].

This DTS data is used to effect a set of data movements and transformations, typically from one or more sources to one or more destinations as configured in the package. This data is created by using the Business Intelligence Design Studio that is included with SQL Server or by using SQL Server Enterprise Manager in Microsoft SQL Server 2000.

DTS data can be stored as a file on the file system or in the repository as a row in the **sysdtspackages** table in the **msdb** repository.

2.3.2 Format and Protocol Summary

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

Format or protocol name	Description	Short name
TDS	This protocol is used to communicate with SQL Server to execute **SQL statements** and retrieve results.	[MS-TDS]

2.3.3 Data Portability Methodology

For this scenario, the documents that contain the DTS data are extracted from the **msdb** repository one by one and stored in a file on the file system. The method of extracting the DTS data from the **msdb** repository for use in a third-party integration tool or platform in this scenario is to use the **TDS** protocol that is provided by the SQL Server instance.

To extract the data, follow these steps:

1. Create a folder on the client machine for storing the retrieved DTS documents.

2. Connect and authenticate to the desired server by using an **ODBC**, OLEDB, or ADO.NET provider or any other TDS implementation.

3. Issue the following **SQL statement**:

   "SELECT name, packagedata from msdb..sysdtspackages"

4. Store the data. To do this, follow these steps for each returned row in step 3:

   1. Create a new file in the folder that was created in step 1.

   2. Save the contents of the **packagedata** field to the new file.

5. Use the DTS documentation [MS-DTS] to interpret the DTS data that was retrieved in the previous step for use in the third-party integration tool or platform.

2.3.3.1 Preconditions

Ensure that the SQL Server service is started on the server. Grant the appropriate permissions to the user to access the **sysdtspackages** table.

2.3.3.2 Versioning

Each DTS document contains multiple versions of the DTS package, as documented in [MS-DTS].

2.3.3.3 Error Handling

None.

2.3.3.4 Coherency Requirements

This data portability scenario has no special coherency requirements.

2.3.3.5 Additional Considerations

There are no additional considerations.

2.4 Third-Party Integration Platform or Tool Consuming DTS Packages in the File System

2.4.1 Data Description

The DTS [MS-DTS] document contains the definition of a DTS package, which includes information about the configuration and order of tasks and data pumps that are applied to data in a DTS package. This data is represented in the DTS document as a binary file, as documented in [MS-DTS].

This DTS data is used to effect a set of data movements and transformations, typically from one or more sources to one or more destinations as configured in the package. This data is created by using the Business Intelligence Design Studio that is included with SQL Server or by using SQL Server Enterprise Manager in SQL Server 2000.

2.4.2 Format and Protocol Summary

No formats or protocols are used in this scenario.

2.4.3 Data Portability Methodology

In this scenario, the DTS data is stored in the file system as packages (*.dts files). These files can be stored at any location on a system. Use the DTS documentation [MS-DTS] to interpret the DTS data in these files.

2.4.3.1 Preconditions

None.

2.4.3.2 Versioning

Each DTS document contains multiple versions of the DTS package as documented in [MS-DTS].

2.4.3.3 Error Handling

None.

2.4.3.4 Coherency Requirements

This data portability scenario has no special coherency requirements.

2.4.3.5 Additional Considerations

There are no additional considerations.

2.5 Third-Party Integration Tool or Platform Consuming Integration Services Project Deployment Files from SSISDB Repository

2.5.1 Data Description

An ISPAC document contains the definition of an Integration Services project deployment file, which includes the packaged metadata of a data integration project. This data is represented in the ISPAC document, as documented in [MS-ISPAC].

This ISPAC data is used to package a set of interrelated metadata that describes one or more data integration processes. This data is created by using the Business Intelligence Design Studio that is included with SQL Server or by using the **Microsoft.SqlServer.Dts** object model.

ISPAC data can be stored as a file on the file system or in the repository as a row that is accessible through the catalog.projects view in the **SSISDB** database.

2.5.2 Format and Protocol Summary

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

Format or protocol name	Description	Short name
TDS	This protocol is used to communicate with SQL Server to execute **SQL statements** and retrieve results.	[MS-TDS]

2.5.3 Data Portability Methodology

For this scenario, the documents containing the ISPAC data are extracted from the **SSISDB** database one by one and stored in a file on the file system. The method of extracting the ISPAC data from the **SSISDB** database for use in a third-party integration tool or platform in this scenario is to use the **TDS** protocol that is provided by the SQL Server instance.

To extract the data, follow these steps:

1. Create a folder on the client machine for storing the retrieved ISPAC documents.
2. Connect and authenticate to the desired server by using an **ODBC**, OLEDB, or ADO.NET provider or any other TDS implementation.
3. Issue the following **SQL statement**:

   ```
   SELECT P.name as project_name, F.name as folder_name from ssisdb.catalog.projects P
   INNER JOIN ssisdb.catalog.folders F on F.folder_id=P.folder_id
   ```

4. Store the data. To do this, follow these steps for each returned row in step 3:
   1. Create a new file in the folder that was created in step 1.
   2. Invoke the **catalog.get_project** stored function, passing the value of the **project_name** and **folder_name** fields into the *@project_name* and *@folder_name* parameters, respectively.
   3. Save the contents of the return value from **catalog.get_project** to the new file.
5. Use the ISPAC documentation [MS-ISPAC] to interpret the ISPAC data that was retrieved in the previous step for use in the third-party integration tool or platform.

2.5.3.1 Preconditions

Ensure that the SQL Server service is started on the server. Grant the appropriate permissions to the user to access the **SSISDB** catalog views.

2.5.3.2 Versioning

None.

2.5.3.3 Error Handling

None.

2.5.3.4 Coherency Requirements

This data portability scenario has no special coherency requirements.

2.5.3.5 Additional Considerations

There are no additional considerations.

2.6 Third-Party Integration Platform or Tool Consuming Integration Services Project Deployment Files Packages in the File System

2.6.1 Data Description

An ISPAC document contains the definition of an Integration Services project deployment file, which includes the packaged metadata of a data integration project. This data is represented in the ISPAC document, as documented in [MS-ISPAC].

This ISPAC data is used to package a set of interrelated metadata that describes one or more data integration processes. This data is created by using the Business Intelligence Design Studio that is included with SQL Server or by using the **Microsoft.SqlServer.Dts** object model.

ISPAC data can be stored as a file on the file system or in the repository as a row that is accessible through the **catalog.projects** view in the **SSISDB** database.

2.6.2 Format and Protocol Summary

No formats or protocols are used in this scenario.

2.6.3 Data Portability Methodology

In this scenario, the ISPAC data is stored in the file system as packages (*.ispac files). These files can be stored at any location on a system. Use the ISPAC documentation [MS-ISPAC] to interpret the ISPAC data in these files.

2.6.3.1 Preconditions

None.

2.6.3.2 Versioning

None.

2.6.3.3 Error Handling

None.

2.6.3.4 Coherency Requirements

This data portability scenario has no special coherency requirements.

2.6.3.5 Additional Considerations

There are no additional considerations.

3 Change Tracking

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

4 Index

C

Change tracking 14

G

Glossary 5

I

Informative references 6
Introduction 5

R

References 6

T

Tracking changes 14