|  |
| --- |
| Data on Demand |
| SSIS Migration Best Practices |
| This document is the compilation of best practices followed while migrating DTS Packages to SSIS Packages using DTS xChange tool |

|  |
| --- |
| Ramkumar Murugesan  9/19/2012 |

# DTSXChange Best Practices

While converting DTS Packages to SSIS using DTSXChange tool, the below migration settings help in optimum performance & lesser errors after converted into SSIS Packages. The below given practices can be set at DTSXChange tool while converting DTS packages to SSIS.

1. NULL Handling Setting
   1. Trim Extra Spaces for Boolean Values (if any)
   2. Generate NULL Handling only if Destination column allows NULLs
2. Connection Manager Consolidation Setting
   1. Do not consolidate flat file connections if they are used in source as well as destination inside a data pump task
   2. Do not consolidate connections that are used inside dynamic properties task
3. Handling Unsupported Data Providers Setting
   1. Select the “Check for 64bit Support – both x64 and IA64” option
   2. Ask DTSXChange to automatically replace the SQL OLEDB connections to SQL Native Client
4. Logging
   1. Log the Package execution results to SQL Server. Maintanance and Searching would be easier and faster
   2. If you do not have any specific requirement, choose only the following events to be logged for lesser network traffic
      1. OnTaskFailed
      2. OnError
      3. OnQueryCancel
5. Package Protection

While converting DTS Packages, ask DTSXChange to choose the following package protection level – “*DontSaveSensitive*”

1. Package Conversion Log

Save the Migration Log and the Application Log.

Before fully converting the Packages…

* Get the list of unconverted tasks from *Migration Log*.
* Get the list of unconverted Properties & indepth details from Application Log

Finally, When the Packages are fully converted, make sure the above errors/warnings are taken care.

# SSIS Package Best Practices

### Tasks & Properties Best Practices

1. Delay Validation Property needs to be set to *TRUE* for the below mentioned list to avoid build time errors …
   1. Connections to External Databases like MS Access
   2. Connections to Flat Files
   3. Execute SQL Tasks
2. “Keep Identity” property is to be set to TRUE for columns with identity values
3. For parallel tasks that involve access to external files, make sure you have a check to see if the file is not locked before accessing the file. Else the task may fail. The C# and VB.net code snippet is attached below…



1. In Script Tasks & Expressions, do not use GetDate() in expression for replacing Date function. Date function returns time field as 00.00.000.

In order to achieve the result, use the following code…

((DT\_WSTR,"23")(DT\_DBDATE)GETDATE())

1. In the destination file/database, Check if the NULL values are handled as NULLs (This is because SSIS handles NULL only if specified while conversion)
2. If there are any Fixed width text files in source/destination, Check if columns with just spaces should be handled as just spaces or NULL values (DTS considers spaces as NULLs, whereas SSIS does not)
3. Text files with “Fixed width” file format might be converted to “Ragged Right” by DTSXChange for performance reasons. It’s better to keep the Fixed Width format as it causes data issue
4. Connections with white spaces are not appropriately converted to SSIS. So remove the white spaces at the end of connection names
5. For Dataflow tasks with no transformation, always check if the column width is the same in “External Columns”, “Input Columns” & “Output Columns” properties
6. ActiveX Scripts found in Workflows (in DTS) are handled in SSIS by writing expressions in Precedence constraints
7. While Bulk copying packages from server to server, follow the below steps for easy migration…
   1. Open the excel, Choose the sheet – “Server to Server”. Make sure you have read/write access to both package store servers



* 1. Choose the source and destination of the packages. There are three options…
     1. Copy Packages from SQL Server to SQL Server
     2. Copy Packages from SQL Server to Share path
     3. Copy Packages from Share path to SQL Server
  2. Enter Server name and Sharepath details in the blue coloured cells
  3. Enter the Packages names in the A Column
  4. Copy the command from B Column
  5. Paste the copied content in a text file and rename the file as “.bat” file
  6. Execute the batch file

1. For Package Logging, Specify the server name, where the logs will reside, in a batch file. This simplifies the task of changing the batch file in case of executing the package in various environments like DEV, SIT, UAT & PROD.

The same can be followed for variables and other dynamic properties. Sample batch file content below…

|  |
| --- |
| DTEXEC /DTS "\MSDB\ASCS\CleanupSSISLogs"  /SERVER <Servername goes here>  /SET "\Package.Connections[<ConnectionName goes here>].Properties[ServerName]"; <ServerName goes here>  /SET "\Package.Connections[<ConnectionName goes here>].Properties[InitialCatalog]";<DatabaseName goes here> |

### Performance Best Practices

1. While converting Dynamic Property Task from DTS as Script Task in SSIS, comment/remove the unused connection properties
2. Use less number of variables as possible as this may affect performance
3. A simple “*select query without order by clause*” orders the rows differently in DTS and SSIS. Make sure you have an order by clause in those cases. This particularly applies for SQL Server Connections with the data access mode - “Table/View” or “Table/View Variable” but can occur in other cases where a simple select query without order by clause
4. While using SELECT queries, select only the columns you wanted
5. If SQL Server is used as Source Connection, use NOLOCK hints for SELECT queries
6. If possible, datetime conversions can be done at SQL Server Database engine side as SSIS consumes more time than Database engine
7. If migrating the ActiveX Script needs combination of SSIS Tasks and C# code to be created in SSIS 2008, then we would write C# code for SSIS Tasks too (The C# code will be part of a Script Task)

This VB.net code inside Script Task has significant performance benefits. Please see the below document for more details…

**