Cookbook for using SQL Server DTS 2000 with .NET

http://SQLDev.Net/DTS/DotNETCookBook.htm

Version: 1.0 revision 15
Last updated: Tuesday, July 23, 2002
Author: Gert E.R. Drapers (GertD@SQLDev.Net)

Table of content:

Introduction:

This cookbook describes how you can use SQL Server 2000 DTS in a .NET environment, using Visual Basic .NET or C#. It provides a detailed step-by-step check list for the following three scenarios:

1. Execution of existing DTS packages
2. Creation of DTS packages
3. Creation of Custom Tasks

Each part can be read independently so you will notice duplications of activities/steps between the sections, but the goal of this cookbook is to provide you with a complete and working checklist for each scenario. Each scenario will be shown both using Visual Basic .NET and using C#. The use of Visual Studio is optional, so steps will be explained using the .NET Framework SDK and alternative steps will be provides if different when using Visual Studio .NET

Requirements:
The cookbook assumes that you have installed the following software:

1. SQL Server 2000 SP2 or better
2. .NET Frameworks SDK 1.0 (SP1 is optional but advised)
   See http://msdn.microsoft.com/netframework/ and follow the download links.
3. Optionally: Visual Studio .NET

Initial (required) steps:

This section describes a set of steps that you have to perform before you can start coding, which are independent of the programming language that you have chosen to use.

NOTE: ***These steps only have to be executed once.***

1. Create a Strong Name Key Pair file

   In order to make the assemblies uniquely identifiable, we need to create a strong name key pair file. This key file is later used to create assemblies that are uniquely identifiable. The key file is created using the SN.EXE utility located in the .NET Framework SDK directory.

   When you only have the .NET Framework SDK installed the default location for this directory is:
   ```
   ”C:\Program Files\Microsoft.NET\FrameworkSDK\Bin”
   ```

   When Visual Studio is installed the default location for the directory is:
   ```
   “C:\Program Files\Microsoft Visual Studio .NET\FrameworkSDK\Bin”
   ```

   To generate a key file use:
   ```
   sn.exe –k C:\DTS.KEY
   ```

   NOTE: If you installed the .NET Framework SDK (first), the directory location containing SN.EXE and other SDK tools is on the PATH, but when you installed Visual Studio .NET this location is ***not*** on the PATH

2. Create Runtime Callable Wrapper (RCW)

   Create an assembly to interact between SQL Server DTS and .NET, which is called a Runtime Callable Wrapper, RCW for short.
   This is done using the utility called TLBIMP.EXE located in the .NET Framework SDK directory.

   In order to interact with DTS Packages (execution, creation and enumeration) or create DTS Custom Tasks you need to interact with the DTS Package COM object, which is an in-process COM server, hosted in :
   "C:\Program Files\Microsoft SQL Server\80\Tools\Binn\dtspkg.dll"

   To create the RCW use:
   ```
   tlbimp.exe "C:\Program Files\Microsoft SQL
   Server\80\Tools\Binn\dtspkg.dll"
   /out:C:\Microsoft.SqlServer.DTSPkg80.dll /keyfile:C:\DTS.KEY
   ```

   NOTE: This is one continuous command line!

3. Register the RCW in the GAC (Global Assembly Cache)

   Because the RCW assembly is going to be used by a COM based application/ environment which will probably not run in the same directory, you need to install the RCW in the GAC (Global Assembly Cache). This makes it possible to locate the assembly even when it is not in your own directory.

   This prevents you for having to have multiple identical copies of this assembly on your machine.

   To install the RCW assembly in the GAC use:
   ```
   gacutil.exe -i C:\Microsoft.SqlServer.DTSPkg80.dll
   ```

After these three steps you are ready to start.

Cookbook using Visual Basic .NET

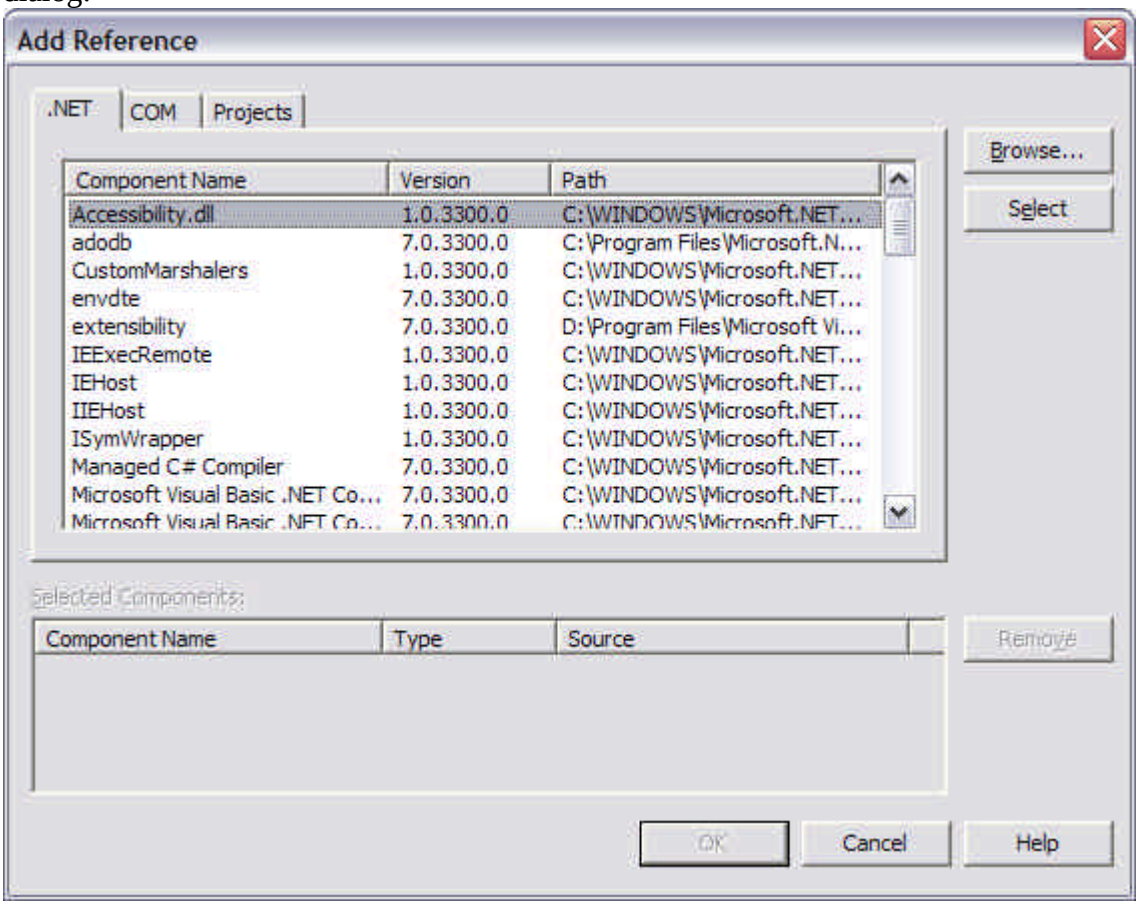
This part describes how to use Visual Basic .NET to drive SQL Server 2000 DTS.

General steps using Visual Basic .NET:

1. Perform all the step lined out in "Initial required steps" first

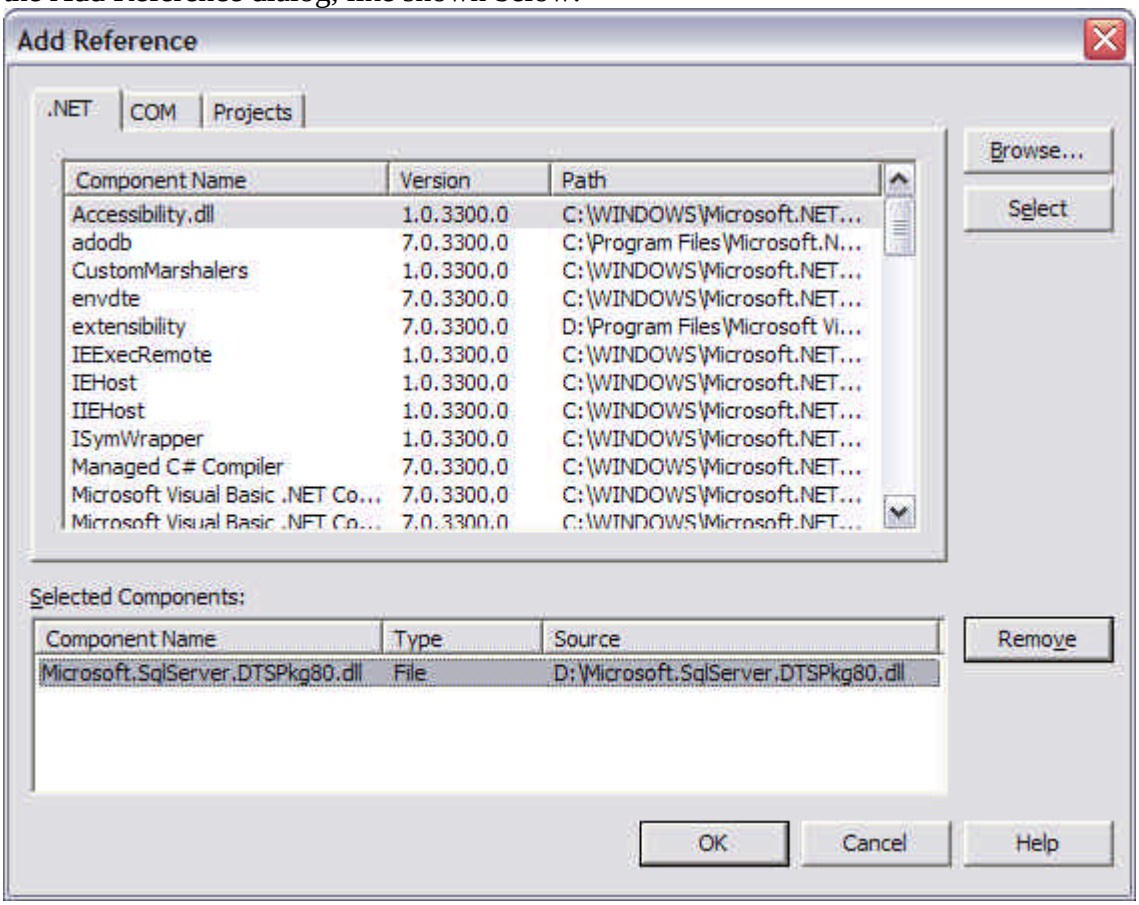
2. If you are using Visual Studio .NET add a reference to your project to the RCW assembly you created, in this cookbook this is:
   `C:\Microsoft.SqlServer.DTSPkg80.dll`

   To accomplish this perform the following steps:
   a. Create a new Command Line project
   b. Open the Solution Explorer, right mouse click on the Reference node, choose the option Add Reference. This action will present the following dialog:



   c. Click on the Browse... button, which will present a common File Open dialog. Use this to navigate to the location where you stored the RCW assembly named "Microsoft.SqlServer.DTSPkg80.dll" and select the file.

d. The RCW assembly file name should now be shown in the bottom part of the Add Reference dialog, like shown below:



e. If this is the case, select OK and the references has been added to the Visual Studio project.

Execution of existing DTS packages:

1. Perform all the steps lined out in "General steps using Visual Basic .NET" first

2. Next step is to execute an existing DTS package named DTSTestPackage, stored in the file c:\test1.dts.

   Create a file named ExecPkg.vb and add the following code:

```
' Import the RCW assembly and assign and alias
Imports DTS = Microsoft.SqlServer.DTSPkg80

Module ExecPkg
    Sub Main()

        ' Create an instance of a new Package object
        Dim package As DTS.Package2Class
        package = New DTS.Package2Class()

        Dim filename As String
```

```
        Dim password As String
        Dim packageID As String
        Dim versionID As String
        Dim name As String
        Dim pVarPersustStgOfHost As Object

        filename = "c:\test1.dts"
        password = ""
        packageID = ""
        versionID = ""
        name = "DTSTestPackage"
        pVarPersustStgOfHost = Nothing

        ' Load the package from file
        package.LoadFromStorageFile(filename, _
            password, packageID, versionID, _
            name, pVarPersustStgOfHost)

        ' Execute the package
        package.Execute()

        ' Destroy the execution state
        package.UnInitialize()

        package = Nothing
    End Sub

End Module
```

3. Compile the code using:
   ```
   vbc ExecPkg.vb /target:exe /out:C:\ExecPkg.exe /main:ExecPkg
   /reference:C:\Microsoft.SQLServer.DTSPkg80.dll /debug:full
   ```

4. Now you should have an executable name ExecPkg.exe that you can execute. The result of executing ExecPkg.exe is a message box with the text “Hello DTS”, like

Creation of DTS packages:

1. Perform all the steps lined out in “General steps using Visual Basic .NET” first
2. Next steps...

Creation of Custom Tasks:

1. Perform all the steps lined out in “General steps using Visual Basic .NET” first
2. Next steps...

Cookbook using Visual C#

This part describes how to use Visual C# to drive SQL Server 2000 DTS.

General steps using Visual C#:

1. Perform all the step lined out in "Initial required steps" first

2. If you are using Visual Studio .NET add a reference to your project to the RCW assembly you created, in this cookbook this is:
   `C:\Microsoft.SqlServer.DTSPkg80.dll`

   NOTE: See "General steps using Visual Basic .NET" for a detailed description of the steps).

Execution of existing DTS packages

1. Perform all the steps lined out in "General steps using Visual C#" first

2. Next step is to execute an existing DTS package named DTSTestPackage, stored in the file c:\test1.dts.

   Create a file named ExecPkg.cs and add the following code:

```
using System;
using DTS = Microsoft.SqlServer.DTSPkg80;

namespace ExecPkg
{
	/// <summary>
	/// Summary description for App.
	/// </summary>
	class App
	{
		/// <summary>
		/// The main entry point for the application.
		/// </summary>
		[MTAThread]
		static void Main(string[] args)
		{
			DTS.Package2Class package = new DTS.Package2Class();

			string filename = @"c:\test1.dts";
			string password = null;
			string packageID = null;
			string versionID = null;
			string name = "DTSTestPackage";
			object pVarPersistStfOfHost = null;

			package.LoadFromStorageFile(
				filename,
				password,
				packageID,
				versionID,
				name, ref
				pVarPersistStfOfHost);

			package.Execute();
```

```
                package.UnInitialize();

                package = null;
            }
        }
}
```

3. Compile the code using:
   ```
   csc /target:exe /out:C:\ExecPkg.exe /debug:full
   /reference:C:\Microsoft.SQLServer.DTSPkg80.dll ExecPkg.cs
   ```

4. Now you should have an executable name ExecPkg.exe that you can execute. The result of executing ExecPkg.exe is a message box with the text “Hello DTS”, like

Creation of DTS packages

1. Perform all the steps lined out in “General steps using Visual C#” first
2. Next steps...

Creation of Custom Tasks

1. Perform all the steps lined out in “General steps using Visual C#” first
2. Next steps...