http://www.sqlservercentral.com/articles/DTS/62469/
Printed 2008/04/10 05:56PM

Yet another way to include portability on DTS packages

By Alceu Rodrigues de Freitas Junior, 2008/03/05

Introduction

An issue that comes when using DTS packages in enterprise sized projects is the several environments that usually are used in: one for development, one for testing (sometimes called QA) and another one used in production.

The problem rises when it is necessary to move the DTS package developed in the development server to the other environments using the minimum effort necessary and avoiding new bugs by changing the packages during the transport.

Getting started

Initially it is necessary to make all configuration of the DTS package that will probably changed during moving from one environment to another to be dynamic: database connections, files paths and properties related to business rules enforced by the package. Those values could be fetched from somewhere like an INI file: anyway, using hard coded values is always a bad idea and DTS already has a good support for INI files.

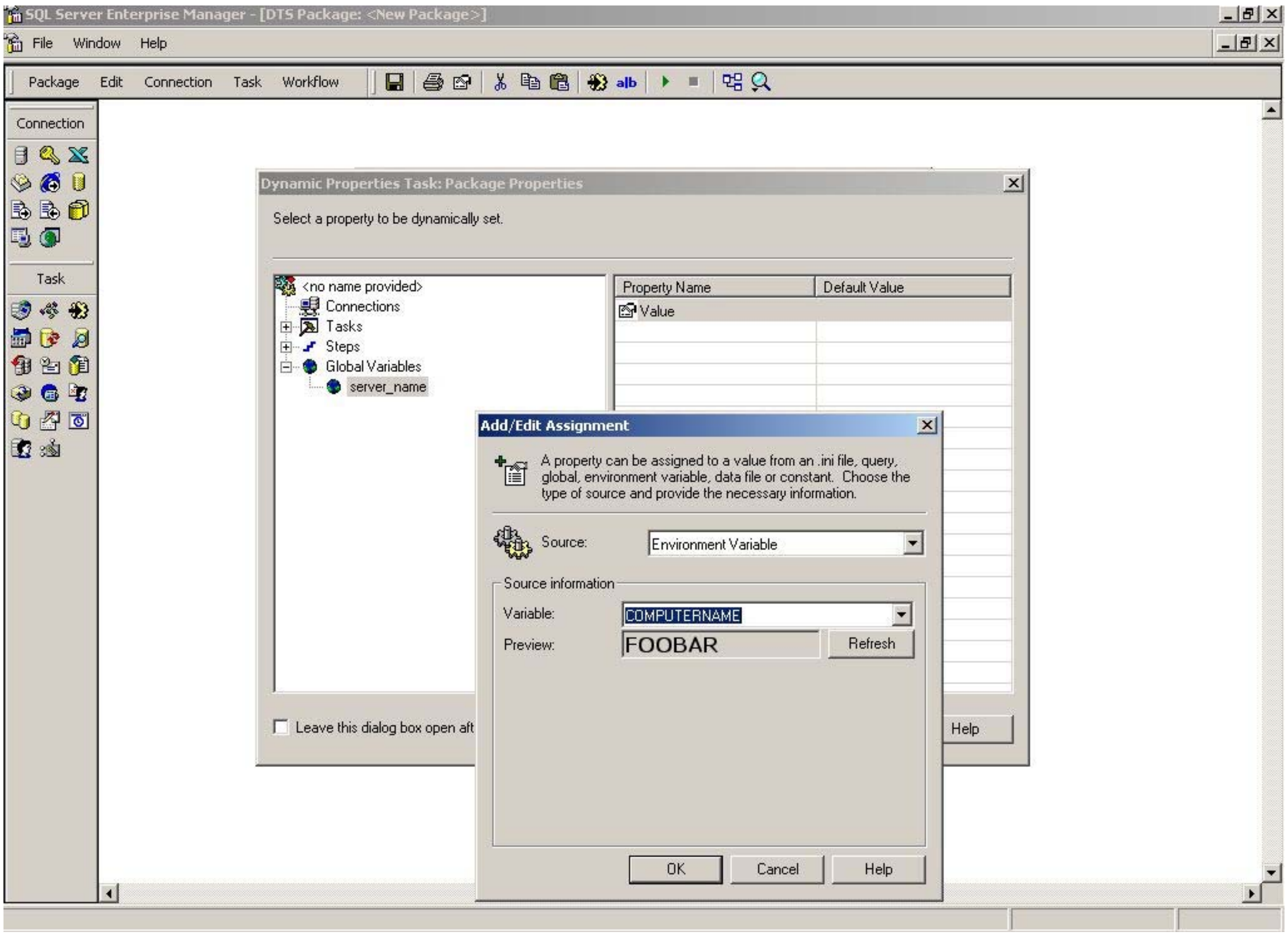
Once an INI is created and ready for use, then next step is to add some intelligence in the DTS package to fetch the correct INI file depending on the environment it is located. While changing manually a property to identify which INI files to read, there is always a chance to make something wrong since this procedure can be repeated many times.

There is a very simple solution for this case: all INI files could be located in a shared directory, using a defined pattern for the path name of the shared resource. The INI that the DTS must read should be fixed. The result should be something like:

\\<servername>\DTSConfig\dw_load.ini

where <servername> should be replaced by the MS SQL Server name where the DTS package is located, "DTSConfig" the shared directory and "dw_load.ini" the INI file. This value could be fetched easily by using a Dynamic Property Task with an assignment to a global variable using the value from the Environment Variable COMPUTERNAME.

An image should give a better idea of the concept:


SQL Server Enterprise Manager - [DTS Package: <New Package>]
File Window Help
Package Edit Connection Task Workflow
Connection
Task
Dynamic Properties Task: Package Properties
<no name provided>
Connections
Tasks
Steps
Global Variables
server_name
Property Name
Default Value
Value
Leave this dialog box open aft
Help
Add/Edit Assignment
Refresh
OK
Cancel
Help

Select a property to be dynamically set.

A property can be assigned to a value from an .ini file, query, global, environment variable, data file or constant. Choose the type of source and provide the necessary information.

Source: Environment Variable

Source information

Variable: COMPUTERNAME

Preview: FOOBAR

Of course, in each database server should be available a shared directory "DTSConfig" and the INI file should be available there as well, with the expected values for each environment.

Once the INI location is done, now the rest is read the INI file and set the necessary properties. The properties that must be changed will depend heavily on what the DTS package will do, but usually the following properties will demand some attention:

- Database connection properties (as Datasource, Catalog, Username and Password).
- Paths for logging activities of the package and datapumps.
- Profiles (and passwords) for SendMail tasks.
- Path and filenames for importing/exporting data from/to text files.

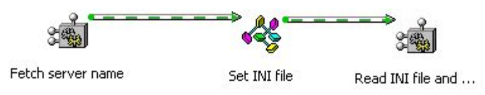
Probably you will be using a Dynamic Property Task to read the INI and set the properties as necessary. Once the task is configured by the first time in development server, the task will have all assignments with the property SourceIniFileFileName defined as \\devserver\DTSConfig\dw_load.ini. The "devserver" part must be replaced by the value fetched in the previous Dynamic Property Task ("Fetch server name" in the screenshot). To have some automatization on that some code is necessary. See below for an implementation example in VBscript:

```
Function Main()
    Dim oPackage
    Set oPackage = DTSGlobalVariables.Parent

    Dim sShareName
    sShareName = "DTSConfig"
    Dim sINI
    sINI = "dw_load.ini"

    Dim oCustomTask
    Dim oTask
    Dim oAssignment
    Set oTask = oPackage.Tasks("Tsk_ReadINIFile")
    Set oCustomTask = oTask.CustomTask

    Dim sPath
    sPath = "\\" & DTSGlobalVariables("computer_name").Value & "\" _
        & sShareName & "\" & sINIFilename
    Dim iCounter

    for iCounter = CInt( 1 ) to oCustomTask.Assignments.Count
        Set oAssignment = oCustomTask.Assignments( CInt ( iCounter ) )
        oAssignment.SourceIniFileFileName = sPath
    next

    Main = DTSTaskExecResult_Success

End Function
```

This code will loop over all assignments in the second Dynamic Property Task (called "Tsk_ReadINIFile" in this example) and change the property SourceIniFileFileName of each one of them. The must important part of the code is the line

Set oTask = oPackage.Tasks("Tsk_ReadINIFile")

If an invalid name or the name of other Task is passed as a parameter, the whole code will not work (and probably may cause other errors too). It is also a good idea to keep all reading of INI properties in a single Dynamic Property Task to simplify the automatic configuration. In the end, you will end if something like the screenshot below:



You just need to copy the same three steps above in each of the DTS packages that will be created, not forgetting to change:

1. The INI filename.
2. The share name (if changed).
3. The DynamicPropertyTask name.

But that's not all. There are some details that will need attention to avoid problems when moving the DTS package to other environments. That will be discussed in the next section.

Additional stuff

Some details still need to be changed to avoid problems that are usually difficult to detect when you migrate a DTS package using this configuration schema.

The first one, and probably the most important, is to avoid fully qualified name of tables when configuring a DTS Datapump Task that will execute some operation in a SQL database. In the cases that a datapump will fetch rows or populate a table, the properties SourceObjectName and DestinationObjectName will be populated by default with a value like "[Testing].[dbo].[CONTACT]". If the Catalog name of each server changed from one environment to another, that value must be changed to "CONTACT" only. Anyway, the configuration of the connections used by the datapump task will be (hopefully) correct configured and that should be sufficient to avoid problems about operations in the wrong table. Every time a new datapump is configured, the properties SourceObjectName and DestinationObjectName will used fully qualified values when pointing to a table in a database.

Another problem would be using Execute Package tasks: when such task is created and the PackageName property is set, the property PackageID is populated automatically with the DTS package ID that will be called by the task. When this property is populated, the package will be searched by its Package ID and not by its name. This is an problem, because the Package ID property will have a different value every time a new version of it is created in a database server. Using only the name would make possible to alter one or both packages (the caller and the called) in a server without breaking any configuration. Of course, this will only work if the DTS packages have unique names in each server.

Suggestions

Do not used mapped resources into drives: DTS supports reading directly from UNC shared resources without issues and that is much more flexible than having to map the same drive in each server (development, QA and production) to the proper shared directory.

Try to break the INI in two new files: one INI will hold infrastructure information (connections, users and passwords) and the other business related information. The INI with infrastructure information should be shared by all DTS packages while each DTS should have its exclusive INI for business information. By using configuration like that, you will be able to change business related information without needing to have access to user and passwords that will probably be hold by the IT team with care.

Conclusion

Although there are other methods described in SQL Server Central, this article combines using INI files for DTS configuration with a simple server name resolution. While is quite simple to use COMPUTERNAME as a point to change the configuration dynamically, it may be impossible to be used if the database server is configured in a cluster or there are security policies that do not allow shared directories to be available in the database server.

A minor issue also may be the need to copy the three initial steps in each DTS package. While it is easier to do during development, it could be a problem if there are many DTS packages that will need to be updated if something needs to be changed.

If those are not issues in your project, then you might be able to enjoy transporting very easily a DTS package from one server to another without having to worry (too much) about forgetting changing some tiny property somewhere.

Acknowledgments

I would like to thank you Guilherme Daniel Fernandes and Mario Tachibana for the first implementation of the code to change a Dynamic Property Task assignments and Fabio Carneiro Antunes for helping with the article revision.