Accessing OPC from inside T-SQL (ODSOLE), using QuickOPC-COM

Sometimes you may want to access OPC data from inside a Transact-SQL code, i.e. not from an external program, but directly by the SQL Server. This comes useful if you already have lots of database code, and need to access some OPC data e.g. to log their status, create a report, or download a recipe to the PLC.

The code below shows the basics of such approach. From inside a Microsoft SQL Server, it reads a single OPC item value, and prints it out. We are using the SQL Server's ODSOLE facility for this (Open Data Services Object Linking and Embedding).

If you prefer .NET, similar thing can also be done using QuickOPC and CLR integration. We will be posting an example of that to the Knowledge Base soon as well.

```
-- ReadAndDisplayValue.sql: Reads and displays an OPC item value.


-- The 'sp_configure' calls below update the configuration options in order to
prevent following error:
-- Msg 15281, Level 16, State 1, Procedure sp_OACreate, Line 1
-- SQL Server blocked access to procedure 'sys.sp_OACreate' of component 'Ole
Automation Procedures' because this component
-- is turned off as part of the security configuration for this server. A system
administrator can enable the use of 'Ole
-- Automation Procedures' by using sp_configure. For more information about enabling
'Ole Automation Procedures', see "Surface
-- Area Configuration" in SQL Server Books Online.
EXEC sp_configure 'show advanced options', 1;
GO
RECONFIGURE;
EXEC sp_configure 'Ole Automation Procedures', 1;
GO
RECONFIGURE;
GO


-- Stop the OLE Automation execution environment. This is useful if you are
experimenting with the registration or security,
-- so that the modified settings always take effect before attempting to execute our
code.
EXEC sp_OAStop;
GO


-- The actual example starts here.
DECLARE @hr int;
DECLARE @src varchar(255), @desc varchar(255);
```

```
-- Create EasyOPC-DA component
DECLARE @easyDAClient int;
EXEC @hr = sp_OACreate 'OPCLabs.EasyDAClient.5.1', @easyDAClient OUT, 4;
IF @hr <> 0
BEGIN
   EXEC sp_OAGetErrorInfo @easyDAClient, @src OUT, @desc OUT;
   raiserror('Error Creating COM Component 0x%x, %s, %s', 16, 1, @hr, @src, @desc);
   RETURN;
END;

-- Read item value and display it
DECLARE @value float;
EXEC @hr = sp_OAMethod @easyDAClient, 'ReadItemValue', @value OUT, '',
'OPCLabs.KitServer.2', 'Demo.Single';
IF @hr <> 0
BEGIN
   EXEC sp_OAGetErrorInfo @easyDAClient, @src OUT, @desc OUT;
   raiserror('Error Calling COM Method 0x%x, %s, %s', 16, 1, @hr, @src, @desc);
   -- If you get following error: "Error Calling COM Method 0x80070005,
OPCLabs.EasyDAClient.5.1, Access is denied.",
   -- start DCOMCNFG, and for applications "EasyOPC-COM 5.1" and "OPCKitServer",
configure their security for the SQL Server
   -- process identity. Typically, this means adding Network Service or Local Service
(depending on the account the SQL Server
   -- uses) to "Launch and Activation Permissions -> Local Launch, Local Activation",
and to "Access Permissions -> Local
   -- Access".
   RETURN;
END;
PRINT @value;

GO
```

This example will be included with the product starting with build 5.12.1305.1. Please use the code from the product itself for the most up-to-date version of the example.

General information about integration of QuickOPC and Microsoft SQL Server is here.