Unloading Master Data from SAP BI 7.0 using XMLA and C#

April 2008

Author

Hermann Daeubler, Senior Program Manager, Microsoft
Juergen Daiberl, Technical Evangelist, Microsoft

Contents

Summary

This paper describes how to unload data from SAP NetWeaver BI using C# and the SAP NetWeaver BI XMLA Web service. The paper contains a walk-through of how to use the XMLA Web service within a C# console program in order to unload master data from SAP NetWeaver BI without the need of any additional driver or library installation.

Applies to

- Microsoft .NET
- Microsoft Business Intelligence (Microsoft BI)
- Microsoft SQL Server Integration Services
- Microsoft SQL Server Reporting Services
- SAP NetWeaver 04s
- SAP NetWeaver BI
- XMLA
- C#

Keywords

SAP NetWeaver, SAP NetWeaver BI, XMLA, C#, Microsoft BI, Microsoft SQL Server, SSRS, SSIS

Audience

IT Management, Technical Architects, Technical Consultants, Developers

Introduction

In certain customer scenarios it's necessary to unload Master Data from SAP BI in order to integrate with a Microsoft BI solution. The key question is how could or should it be implemented?

Here is a list which shows five different approaches:

- Direct SAP NetWeaver BI database table access (not recommended)
- Using one of the SAP NetWeaver BI reporting APIs (e.g. XMLA or Ole DB for OLAP)
- 3rd-party tool
- Microsoft BizTalk Server Adapter
- Upcoming SSIS 2008 certification for SAP BI 7.0

This paper will focus on using the XMLA interface in SAP NetWeaver BI. It's a reporting API which was built to allow 3rd-party vendors to connect their reporting tools like Microsoft SQL Server Reporting Services (Microsoft SSRS) to SAP NetWeaver BI. The so-called 'Open Hub Service' is the official interface in SAP NetWeaver BI to unload data into external targets like Microsoft SQL Server Analysis Services (Microsoft SSAS). It's still an option though to use a reporting API for this purpose as long as the amount of data is limited or the unload of data won't happen too often which is usually true for master data.

The XMLA Web service has to be configured and activated on the SAP side. Once this is done it's possible to include it in a Visual S Studio C# project via 'Add Web Reference'. As the communication works over HTTP/SOAP there is no need for any additional driver or library installation.

Only objects on the "InfoProvider" level can be used for reporting within SAP NetWeaver BI. So-called 'characteristics" are 'InfoObjects" which keep master data. They are not on the 'InfoProvider' level by default. This has to be configured first. Then it makes sense to define a SAP NetWeaver BI query using the SAP Query Designer. Doing it via a query gives a lot of flexibility in specifying which data should be unloaded. A basic concept within SAP NetWeaver BI is to present an existing query as a virtual cube to the outside world via the reporting APIs. An external program sending a MDX query to SAP NetWeaver BI will in fact trigger the execution of the existing SAP query and retrieve the result set from it as it would be a MOLAP cube.

The same methodology could be used to unload subsets from an 'InfoCube'.

IMPORTANT: Please keep in mind that 'UNLOADING' data from SAP BI requires additional SAP licensing. The customer has to be aware of this and we recommend that the customer checks this with SAP.

Sample

The sample gives an overview about what's necessary as an absolute minimum to retrieve some master data (for the test case just 24 generated data rows loaded from a flat file) from SAP NetWeaver BI by using C# and XMLA. The focus is on extracting data. There are no special considerations of security aspects described, for example the SAP login is just hard-coded in the sample. Here is a short list of the key steps:

- Activate the XMLA service on the SAP side (transaction shortcut /nSICF)
- Test the service by a right-click on the service -> 'test service'. The output in the browser should look like on the screenshot in the walk-through section further down
- Select the 'characteristic' (InfoObject) keeping the master data which should be unloaded and declare it as an 'InfoProvider' in the modeling screen of the warehouse work bench (transaction shortcut /nRSA1)
- Now use the SAP Query Designer and define a query on the corresponding 'characteristic'. It's important to set the 'external access' flag in the query properties
- You can test the newly created query in the SAP NetWeaver BI Query Monitor. The little C# console sample program should return the same result as it can be seen in the SAP GUI (transaction shortcut /nRSRT2)
- Now we are ready to open a C# console program in VS. The first thing to do is to add the XMLA service as a web reference. Just enter the URL which was used for the browser test. It should then tell that it found a 'Discover' and an 'Execute' function. The description should be : 'MsXmlAnalysis'
- Unfortunately there is a little issue with the code which is generated by VS for the XMLA web service. It's necessary to add something to the properties parameter definition for the Execute command. In Reference.cs you will find the following definition of the parameter : 'PropertiesType Properties' at the Execute command code. You should add the following before 'Properties Type' : [XmlElement (typeof(DiscoverProperties))]
- Within the C# program you have to set the appropriate properties before it's possible to execute a command. The sample code is shown further down. The SAP login has to be specified as well as some XMLA specific properties. Details about the XMLA part can be found in the official XMLA spec.
- XMLA works with MDX as the query language. In order to specify the MDX command which will retrieve the data it's recommended to develop it within the 'mdxtest' transaction of SAP NetWeaver BI. The transaction shortcut 'mdxtest' leads to a screen in the SAP GUI which allows to create and test MDX queries. Here you can select the SAP query which was created before and is now presented as a virtual cube. The result of the MDX query should be the same as with the query test in transaction RSRT2. Once this works just copy the statement into the C# console program.
- The next hurdle is the question how to process the result set which the Execute method returns. One option is to add the Analysis Services ADOMD

Client as a reference. This will introduce a ‘CellSet’. Use a XmlNodeReader to convert the result set into a ‘CellSet’. Then you can use all the CellSet related methods to process the result set.

- Running the C# console test program should finally deliver the same output as seen before in SAP NetWeaver BI

The nice thing about this approach is that it doesn’t require any additional driver, library or 3rd-party tool. As it’s using an official SAP NetWeaver BI interface it also doesn’t matter on which platform (OS and DB) SAP NetWeaver BI is running.

Walk-Through

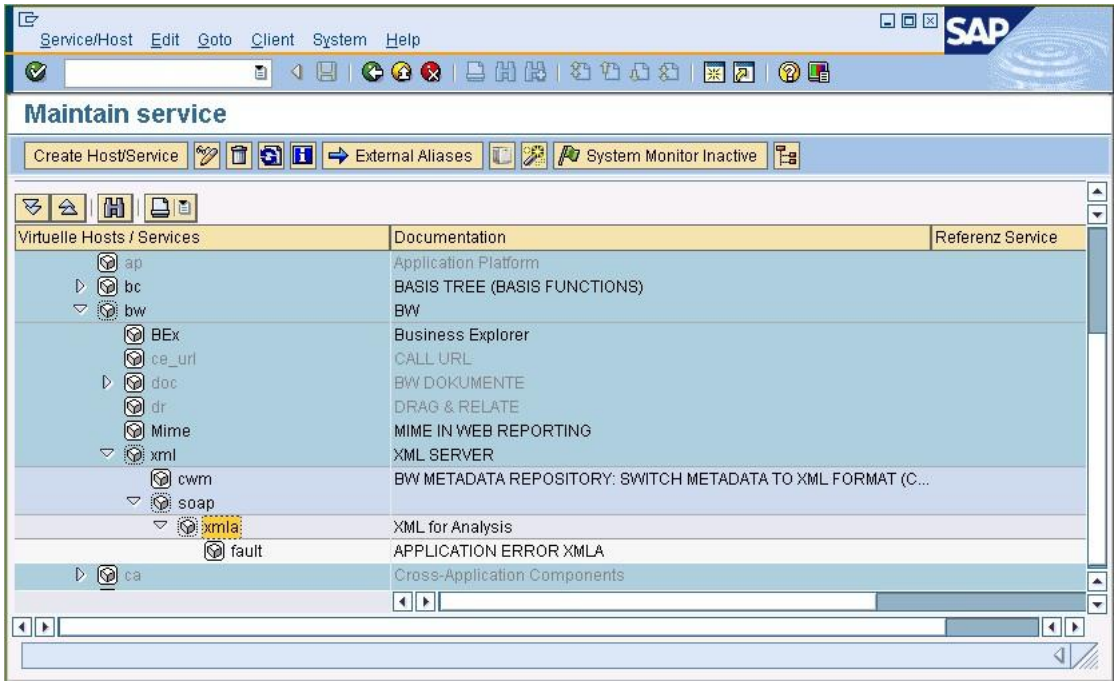

Figure 1 Activate the XMLA service on the SAP NetWeaver BI side

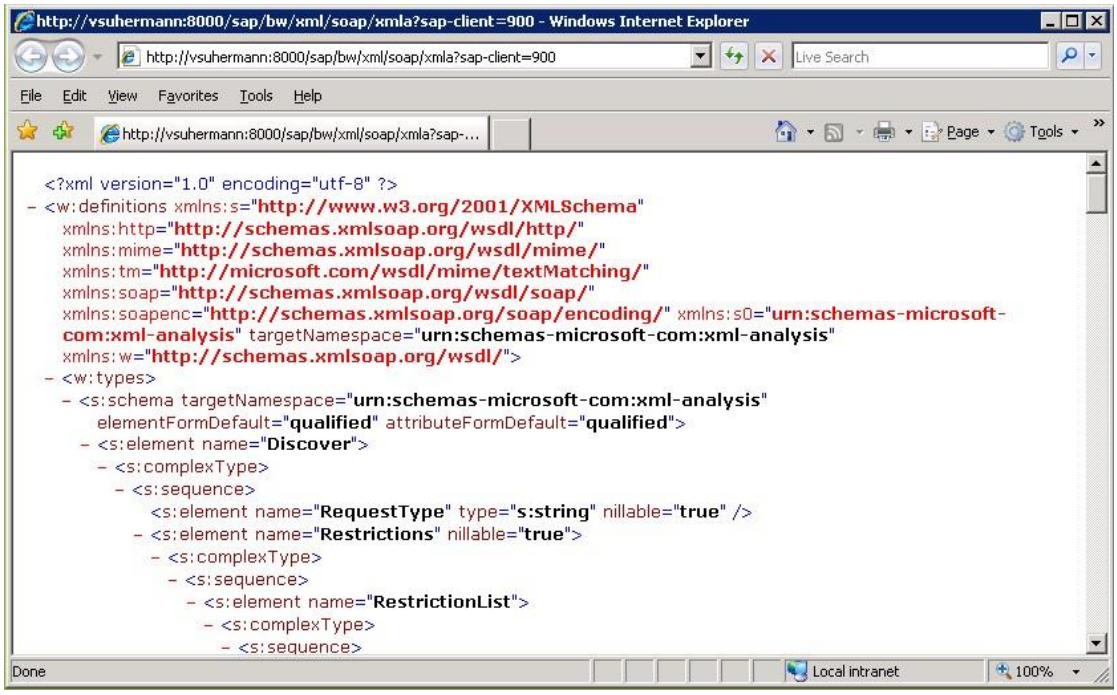

Figure 2 Test of the XMLA web service via browser. The correct URL should return the same XML document as seen on this screenshot

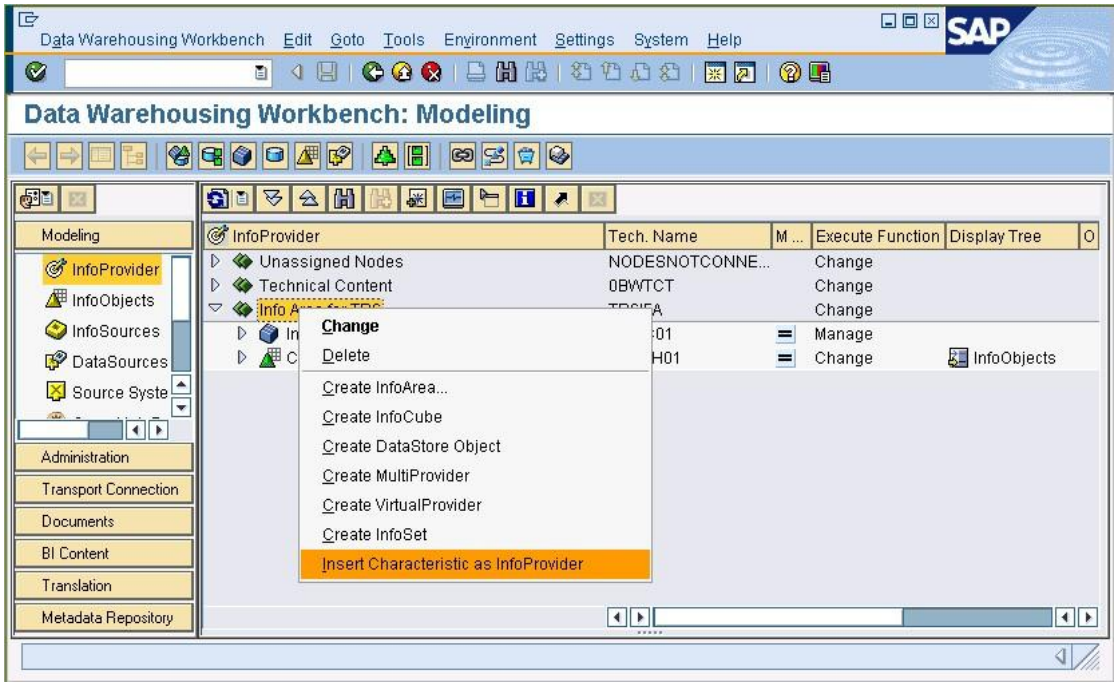

Figure 3 Declare the 'characteristic' as an 'InfoProvider' to allow reporting

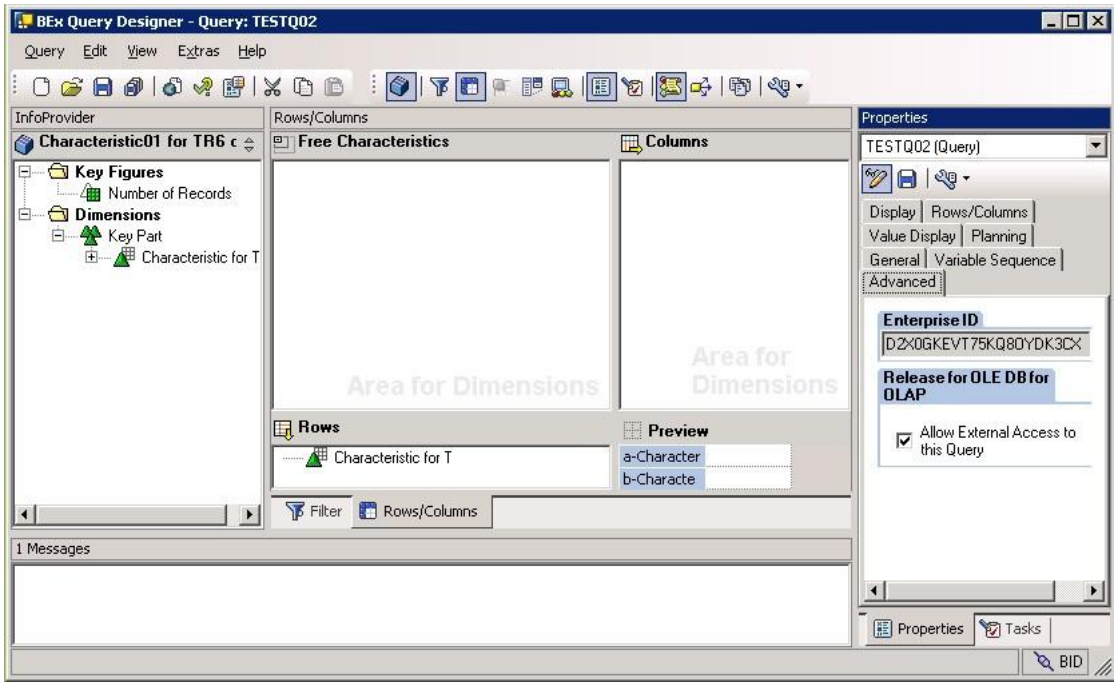

Figure 4 Define a query on the 'characteristic' InfoProvider using the SAP Query Designer. IMPORTANT : set the 'External Access' flag in the query properties

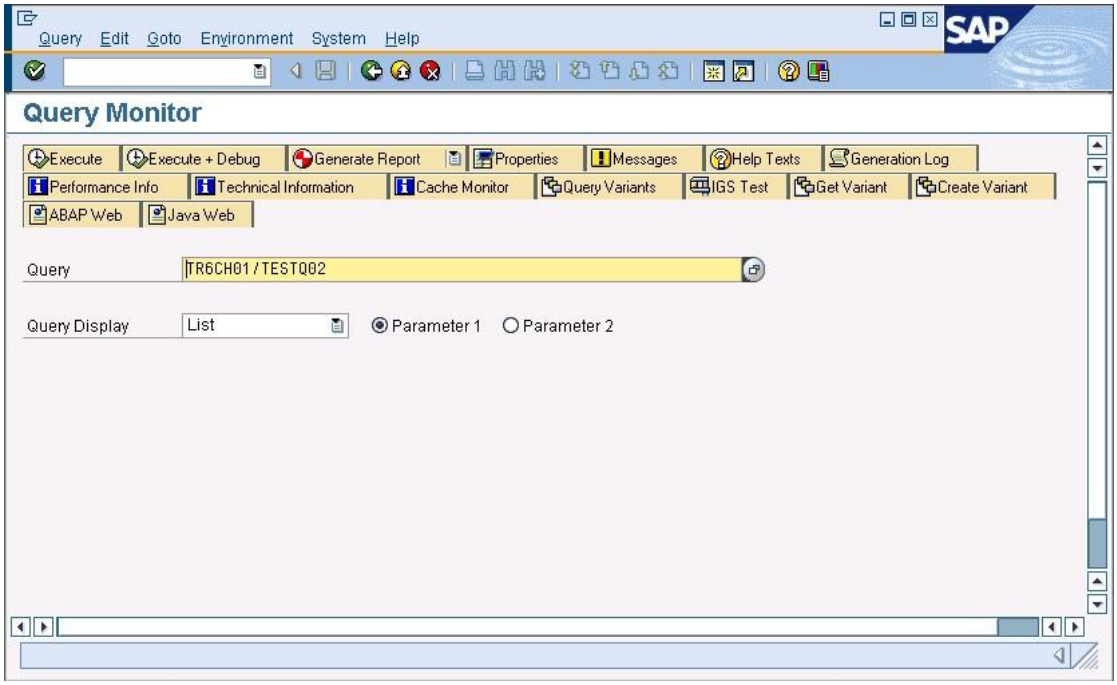

Figure 5 Look for the just saved BI query in the Query Monitor (shortcut /nRSRT2)

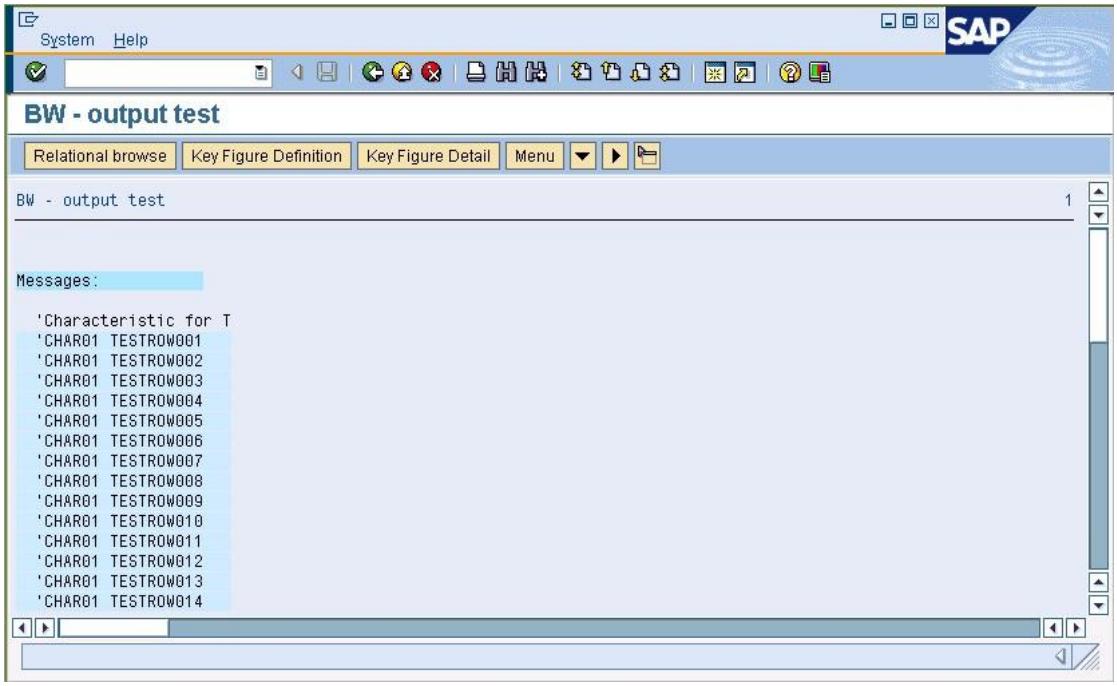

Figure 6 The output of the BI query are 24 test data rows which were loaded before

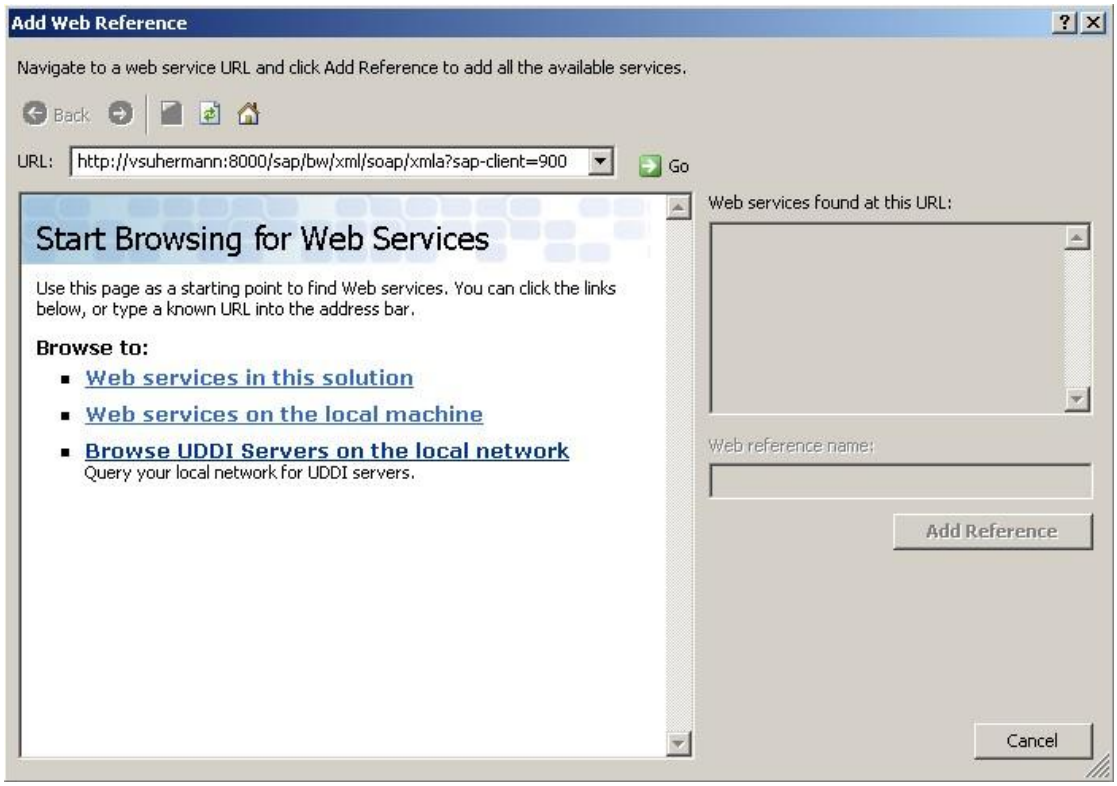

Figure 7 Add the XMLA web service as a 'web reference' to the VS C# console program by using the same URL as for the XMLA browser test

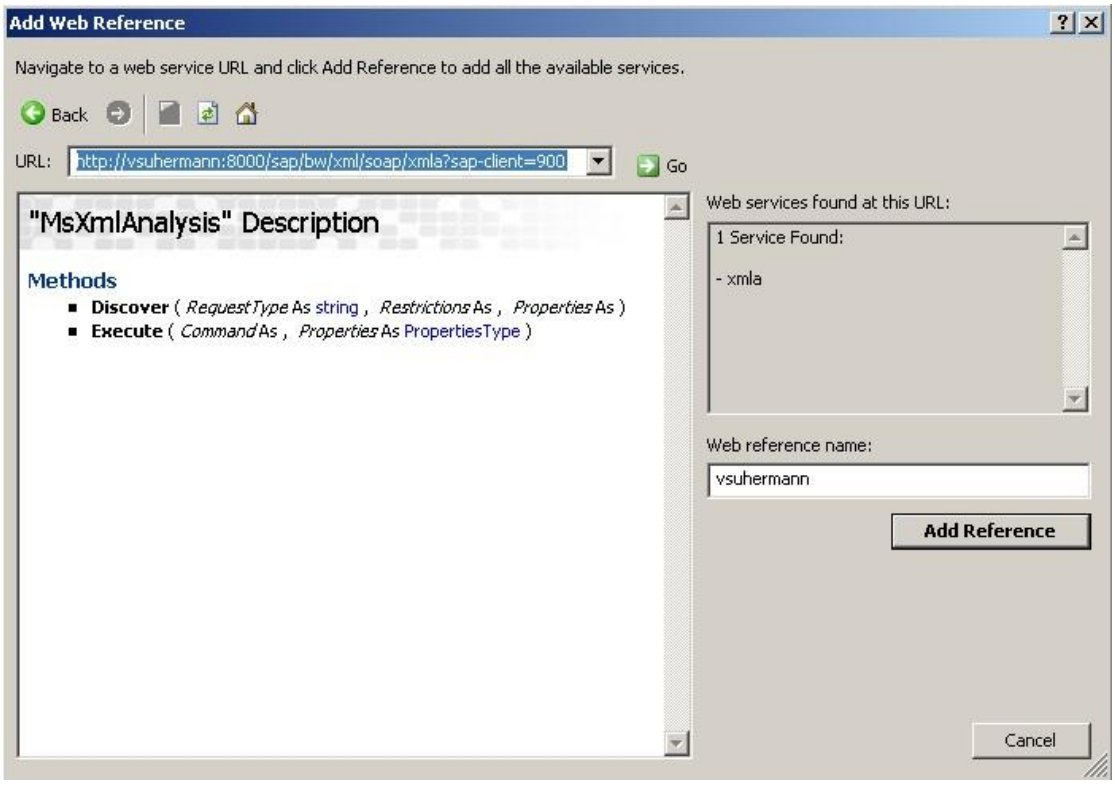

Figure 8 It should detect two methods : Discover, Execute and it's called 'MsXmlAnalysis'

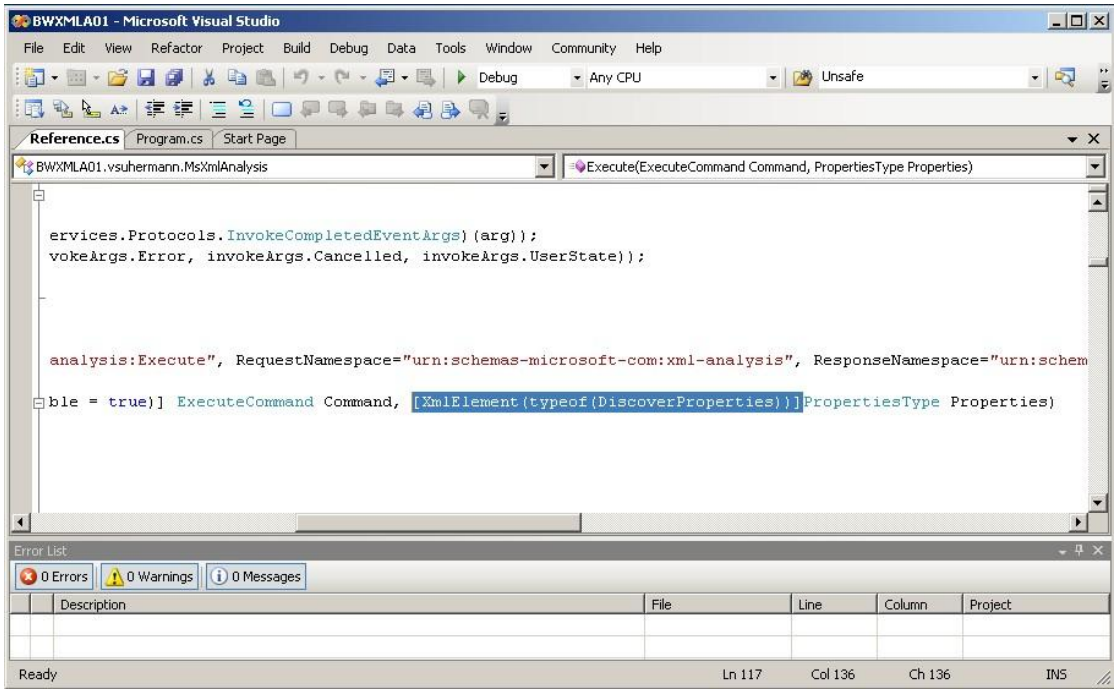

Figure 9 It's necessary to fix the generated properties parameter definition for the Execute command in the Reference.cs file

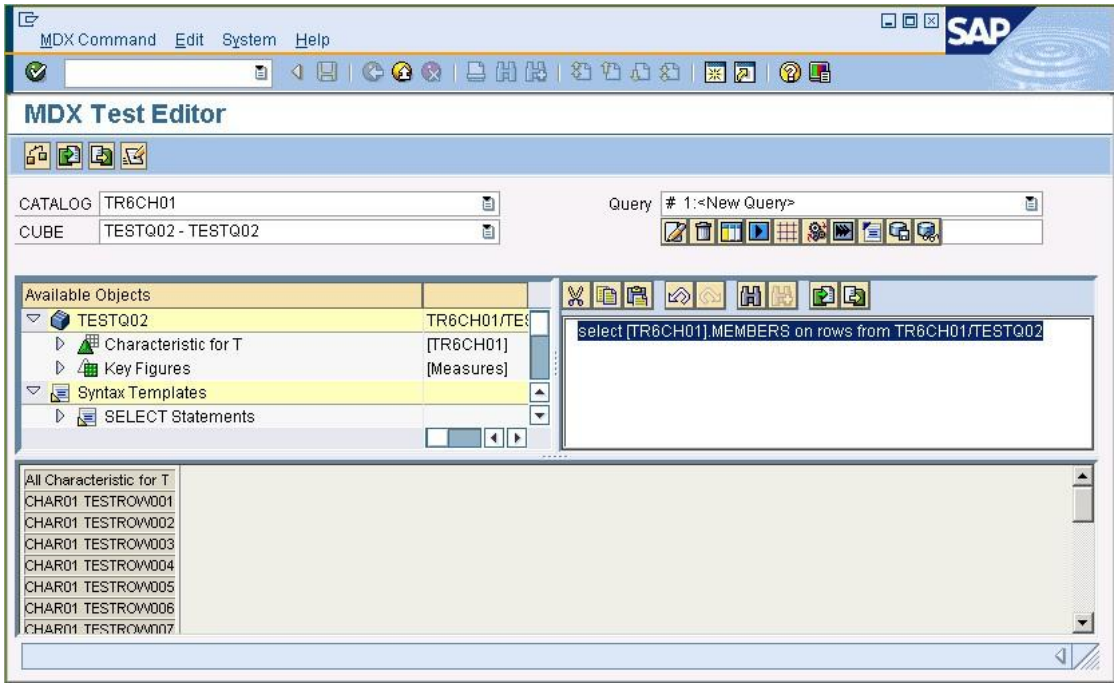

Figure 10 Create and test the MDX command to retrieve the data in the SAP MDX Test Editor. Once it works correct – just copy it over to the VS C# program

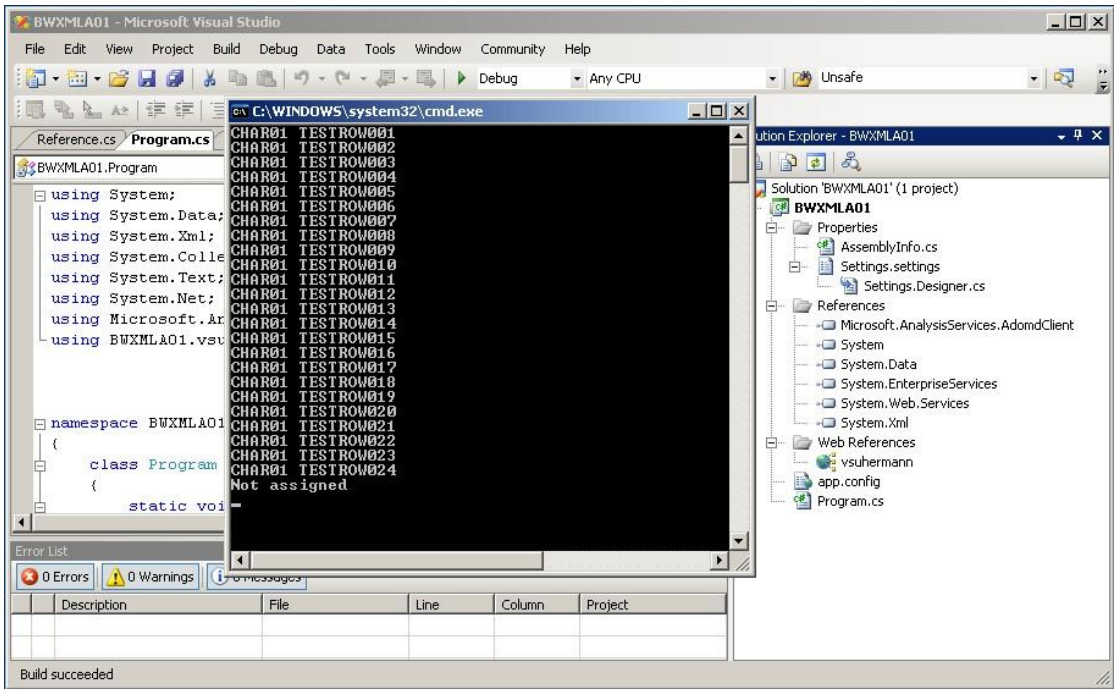

Figure 11 Result

Sample code

```
using System;
using System.Data;
using System.Xml;
using System.Collections.Generic;
using System.Text;
using System.Net;
using Microsoft.AnalysisServices.AdomdClient;
using BWXMLA01.vsuhermann;

namespace BWXMLA01
{
class Program
  {
  static void Main(string[] args)
  {
    MsXmlAnalysis         BWQuery;
    ExecuteCommand     Command;
    XmlElement             result;
    XmlNodeReader       convert_result;
    DiscoverProperties   properties;
    NetworkCredential    login;
    CellSet                  cellset_result;
    TupleCollection        tuples_on_columns;

    int i;
    int nr_tuples_on_columns;

    BWQuery                    = new MsXmlAnalysis();
    Command                  = new ExecuteCommand();
    properties                 = new DiscoverProperties();
    properties.PropertyList  = new DiscoverPropertiesPropertyList();
    login                        = new NetworkCredential();

    login.UserName = "<SAP User Name>";
    login.Password   = "<SAP Password>";

    BWQuery.Credentials     = login;
    BWQuery.PreAuthenticate  = true;

    properties.PropertyList.Content            = "SchemaData";
    properties.PropertyList.Format             = "Multidimensional";
    properties.PropertyList.DataSourceInfo = "default";

     Command.Statement =
     "select [TR6CH01].MEMBERS on rows from TR6CH01/TESTQ02";

     result             = BWQuery.Execute(Command,properties);
     convert_result = new XmlNodeReader(result);
     cellset_result   = CellSet.LoadXml(convert_result);

     nr_tuples_on_columns = cellset_result.Axes[0].Set.Tuples.Count;
     tuples_on_columns    = cellset_result.Axes[0].Set.Tuples;

     for (i = 1; i <nr_tuples_on_columns; i++)
     {
       Console.WriteLine(tuples_on_columns[i].Members[0].Caption);
```

```
    }
    Console.ReadLine();

    }

  }

}
```

References

SAP Help Portal

http://help.sap.com

Microsoft SAP Customer Information Center

http://www.microsoft.com/sap

Microsoft SQL Server BI

http://www.microsoft.com/sql/solutions/bi/default.mspx