

ASP.NET as a Client for Business Intelligence

- By Akash Banerjee

Summary:

Business Intelligence (BI) over a period of time has emerged as one of the main technology drivers. With BI, IT managers and business decision makers are now taking business critical decisions on the move and every moment. There are lots of enabling technologies that can be taken as an alternative or as options to implement a BI solution. In this article I have selected SQL Server 2000 as a DB, for building BI solution I have used SQL Server 2000 Analysis Services and accessing it through a web presentation layer using ASP.NET.

Difficulty level: Intermediate.

Introduction:

“We are drowning in information and starving for knowledge”

- Rutherford D. Rogers, The New York Times, 25th February 1988.

In recent times Business Intelligence (BI) has gathered a significant momentum across organizations and industries. Organizations around the world need access to timely and accurate data to discover and distribute a useful business insight that in turn offers competitive advantage of better, faster and reliable business decisions to organizations. Using BI applications now it is possible to access and analyze information in almost every dimension of business. With recent improvements in information technology have made it faster to access large volumes of data and made it quite economical to distribute useful business insights from BI solution to more number of users who need it. Organizations use different BI tools and technology to develop different types of business intelligence applications. Some of the features that BI tools support are typically data analysis (both real time and historical data), web based interactive reporting, OLAP and alert messaging or notification services. Along with these set of features the tool should be easy to implement and manage and at the same time flexible enough to accommodate future changes and complexities of business.

In this article I will be using Microsoft SQL Server 2000 as database server, Microsoft SQL Server 2000 Analysis Services to build the BI implementation and ASP.NET as a web client to access this implementation and for presentation.

This article assumes you are comfortable using SQL Server 2000 Analysis Services, using and have a good understanding of programming ASP.NET with C# code behind.

Implementation

Requirements:

- SQL Server 2000
- SQL Server 2000 Analysis Services and MDX Sample Application.
- Visual Studio .NET (.NET Framework 1.1)
- IIS 5.0 or above.

Procedure Summary:

1. Build cube(s) from the warehouse and process it
2. Create MDX queries required for producing the required reports
3. Build ASP.NET page to provide interface for the report
4. Write code behind the ASP page to perform the following:
 - i. Connect to the database
 - ii. Execute the MDX query
 - iii. Show the results (as a table or using Pivot Table)

Building Cubes:

Modeling multidimensional data facilitates online business analysis and query performance. Analysis Manager provides facility to turn data stored in relational databases and warehouse into meaningful, easy-to-navigate business information by creating a data cube.

Analysis Services 2000 provides wizards to create cubes, their measures and the dimensions. Also cubes can be optimized to improve query performance and response time.

MDX Query:

Based on the reports that are required the MDX query should be build.

Example: Following MDX query measures for the stores summarized for the states of California and Washington.

```
SELECT Measures.MEMBERS ON COLUMNS,  
{[Store].[Store State].[CA].CHILDREN,  
 [Store].[Store State].[WA].CHILDREN} ON ROWS  
FROM [Sales]
```

This query could be tested using the sample MDX application that comes with the SQL Server 2000 Analysis Services. The results may look like this:

	Unit Sales	Store Cost	Store Sales	Sales Count	Store Sales I
Alameda					
Beverly Hills	21,333.00	18,266.44	\$45,750.24	6815	27,483.80
Los Angeles	25,663.00	21,771.54	\$54,545.28	8207	32,773.74
San Diego	25,635.00	21,713.53	\$54,431.14	8095	32,717.61
San Francisco	2,117.00	1,778.92	\$4,441.18	1325	2,662.26
Bellingham	2,237.00	1,896.62	\$4,739.23	1380	2,842.61
Bremerton	24,576.00	21,121.96	\$52,896.30	7876	31,774.34
Seattle	25,011.00	20,956.80	\$52,644.07	7956	31,687.27
Spokane	23,591.00	19,795.49	\$49,634.46	7397	29,838.97
Tacoma	35,257.00	29,959.28	\$74,843.96	11184	44,884.68
Walla Walla	2,203.00	1,880.34	\$4,705.97	1339	2,825.63
Yakima	11,491.00	9,713.81	\$24,329.23	3652	14,615.42

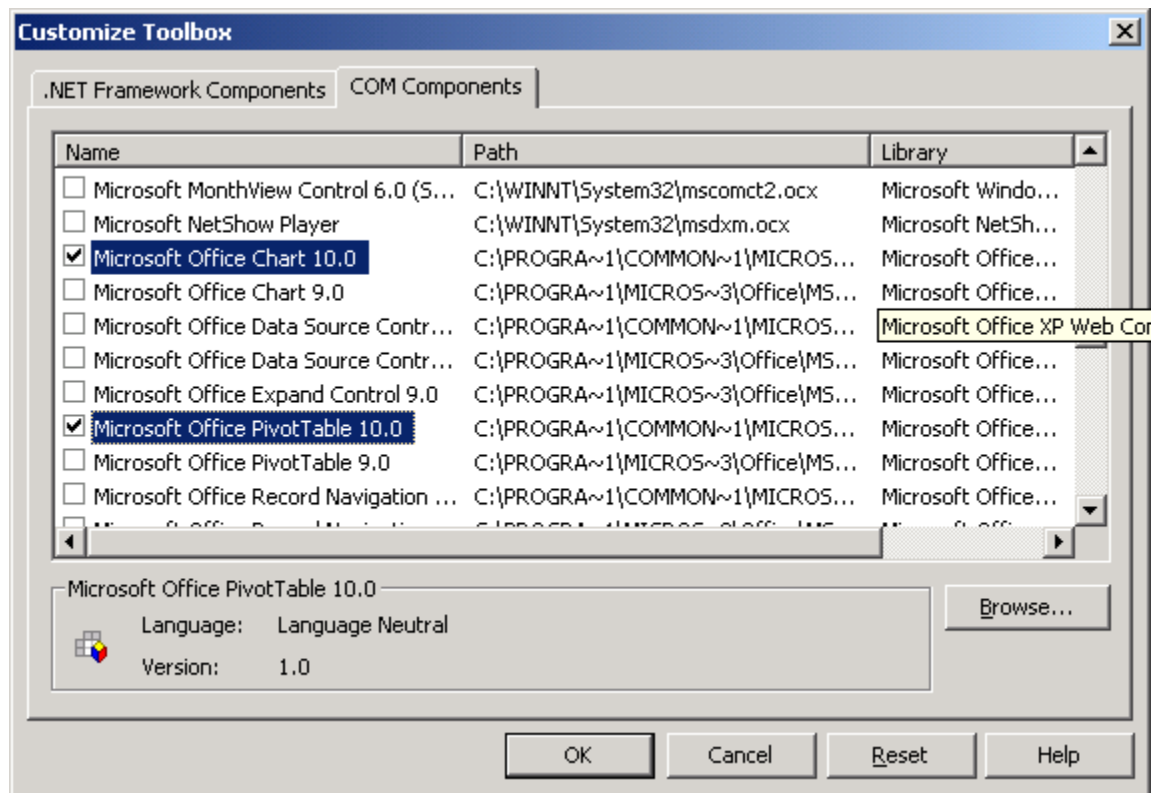
Building the ASP.NET web presentation interface:

There are two ways to build the interface for reports. They could either be developed using web components or by developing custom HTML interface. The usage of web components has a disadvantage that the component should be available on the client side browser (I developed this on IE 5.5).

Developing presentation interface using web component:

Office Web Components (OWC) are Microsoft ActiveX® controls that deliver the functionality of Excel spreadsheets, charts, and Microsoft PivotTable® and Microsoft PivotChart® technology in a Web environment. These controls can be embedded in Web pages and Web-based applications to facilitate flexible, customizable business intelligence solutions. Office Web Components could be used to publish interactive data to the intranet, perform sophisticated analysis in a Web browser, or retrieve data from the intranet for local use. The main feature of the OWC is that it has a highly customizable interface and contains a Chart and a Pivot table component that helps in performing adhoc analysis.

In an ASP.NET page, this component can be included by adding the component to the toolbox by right clicking the toolbox and choosing the 'Add/Remove Items' option. OWC is a COM component and can be found in the 'COM Components' tab.



Once the components are added, it appears on the toolbox. Now it could be dragged and dropped into the ASP.NET page. This will add an 'object' tag to the page and it includes

the component's CLSID. This component will be accessed in the client side, and it should be available in the client machine.

Since the component will be accessed from the client machine, it could be controlled only through the client side scripting.

Following script can be used to initialize the component with the MDX query. A pivot table with the name 'pvt' has been created. The MDX query here queries the measures for the stores summarized for the states of California and Washington.

```
<script language=javascript>

document.Form1.pvt.ConnectionString = "Provider=MSOLAP.2;Data
Source=10.200.240.9;Initial Catalog=FoodMart 2000;Client Cache Size=25;Auto Synch
Period=10000";

document.Form1.pvt.CommandText = "SELECT Measures.MEMBERS ON COLUMNS,
{[Store].[Store State].[CA].CHILDREN,
[Store].[StoreState].[WA].CHILDREN} ON ROWS
FROM [Sales]"

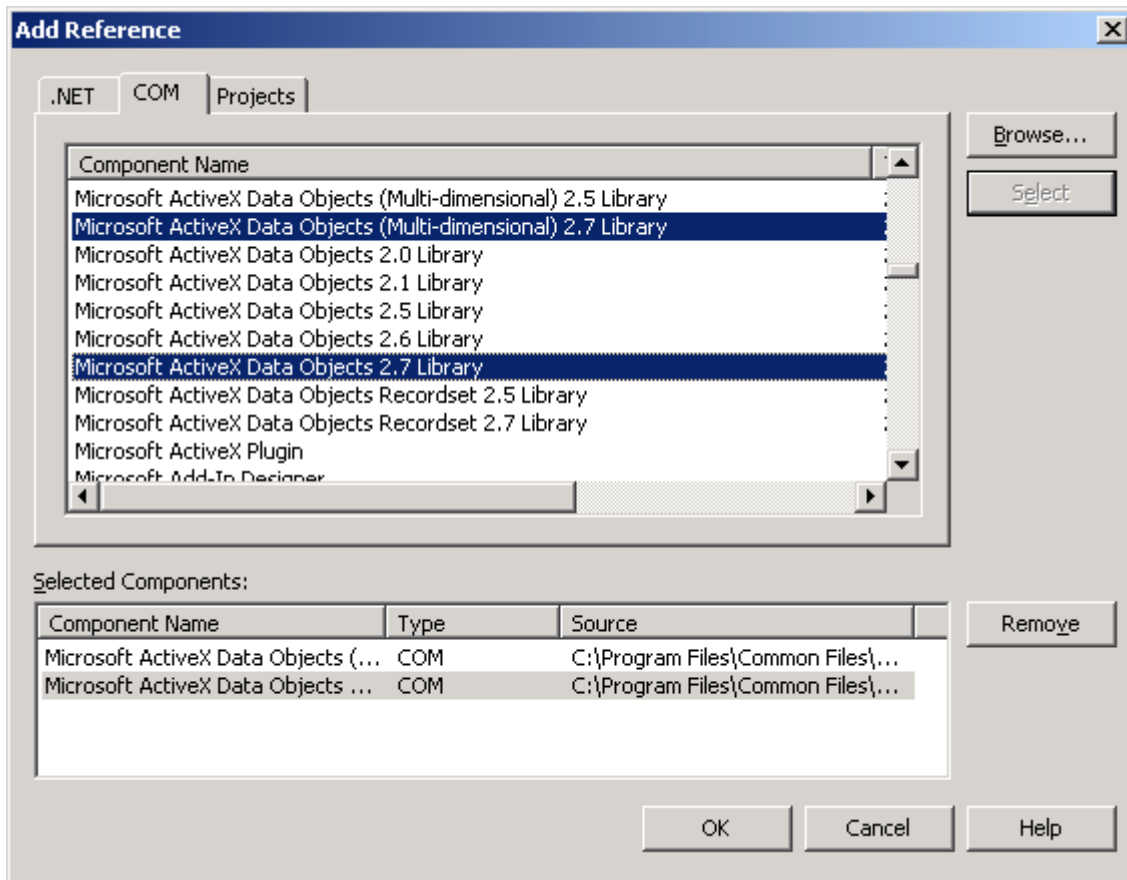
</script>
```

Developing presentation interface using ADOMD:

One of the major limitations of Office Web Component (OWC) is that they have to install on client side machine. To overcome this, the client can be provided with a simple HTML interface to view the reports.

ADOMD is a COM component that can be used to query and retrieve information from the multidimensional cubes. Both data and metadata could be retrieved from the cubes. When using ADOMD in ASP.NET, a reference should be added to those components. ADODB is also needed to establish connection with the server, which ADOMD uses to fetch the data. Both ADODB and ADOMD are COM Components and require a COM Interoperability to be used in .NET code. The reference to following COM components should be made.

1. ADOMD 2.7
2. ADODB 2.7



Following code snippets are used for explanation purpose:

In the code, their namespaces could be included as follows:

```
Using ADODB;
Using ADOMD;
```

The following code could be used to execute a MDX query using ADOMD:

```
ADODB.Connection con;    //To establish connection with database
ADOMD.Cellset cset;     //To hold the results of the MDX query

//Connection string to connect to Analysis Services 2000
string connString = "Datasource=" + BISConstants.CONF_SERVER_NAME + ";
Provider=msolap; Initial Catalog=FoodMart 2000;";
string query = "SELECT Measures.MEMBERS ON COLUMNS, {[Store].[Store
State].[CA].CHILDREN, [Store].[Store State].[WA].CHILDREN} ON ROWS FROM
[Sales]";

con = new ConnectionClass();
con.Open(connString,"", "", 0); //Open connection
cset = new CellsetClass();     //Cellset class to hold result
cset.ActiveConnection = con;
cset.Open(query, con);        //Execute the query
```

Now that the query is executed, the results can be accessed using the Cellset object.

Following code fetches the column headers:

```
int totColsLvl = 0; //total rows within column header (since cross joins may occur)

totColsLvl = cset.Axes[0].Positions[0].Members.Count;
//Axes[0] represent column headers. Axes[1] represent row headers.
//Positions[0] represents the first position within the columns

for (int i = 0; i < totColsLvl; i++) //For each row in the column header
{
    httpc.Response.Write("<TR>"); //Start a row
    //Get all the header of the current row

    foreach (ADOMD.Position pos in cset.Axes[0].Positions)
    {
        Response.Write(pos.Members[i].Caption);
        //Member[] holds the actual column data
        //Caption returns the header caption to be displayed
    }
    httpc.Response.Write("</TR>"); //Close the row
}
```

Following code is similar to the previous one, and it fetches the row headers:

```
int totRowsLvl = 0; //total rows within Rows header (since cross joins may occur)

totColsLvl = cset.Axes[1].Positions[0].Members.Count;
//Axes[1] is for Rows

for (int i = 0; i < totRowsLvl; i++) //For each row in the rows header
{
    httpc.Response.Write("<TR>"); //Start a row
    //Get all the header of the current row

    foreach (ADOMD.Position pos in cset.Axes[0].Positions)
    {
        Response.Write(pos.Members[i].Caption);
    }
    httpc.Response.Write("</TR>"); //Close the row
}
```

Finally, following code snippet fetches the data of the result:

```
totCols = cset.Axes[0].Positions.Count; //Total columns in the
result
totRows = cset.Axes[1].Positions.Count; //Total rows in the
result

object[] coords = new object[2];
//Objects to hold the coordinates of the data being accessed

for (int i = 0;i < totRows;i++) //For each row in the result
{
    httpc.Response.Write("<TR>"); //Start table row

    for ( int j = 0;j < totCol; j++) //In each row, for each
column
    {

        //Initialize the coordinates of the data to be fetched
        coords[0] = j; //column
        coords[1] = i; //row

        //get_Item fetches the data.
        cell = cset.get_Item(ref coords);

        dispvalue = cell.FormattedValue;

        if ( cell.FormattedValue != null)
        {
            //display data
            httpc.Response.Write("<TD>" + dispvalue + "</TD>/n");
        }

        httpc.Response.Write("</TR>"); //Close the row
    }
}
```

Note: When using other languages like VB.NET, the way the data is fetched is different. The 'get_Item' method of the 'cellset' object is used only when using C#. The code to access data using 'cellset' object in VB.NET will be:

```
Dim s As String
s = cs(i, j).Value
'i,j represents the coordinates
```

This method is not supported in C#.

Conclusion:

In response to competition and new emerging market opportunities, organizations are moving towards organization wide restructuring that helps managers to make rapid business critical decisions. This requires fast and reliable and easy access to information about the key factors driving the business. To meet this need, organizations are consolidating data from operational system to data warehouses and data marts, which are

structured to support decision making. On this front having a web interface to access BI data is of strategic importance, as this increases the reach of BI data to everyone in the organization – from senior management to production line executive to field salesmen, in a manner that supports the unique way they work with information.

In short, it is the combination of BI tools and a host of other supporting technologies that delivers strategic advantage on time and every time reliable business decisions.

About the author:

Akash Banerjee, working in Talent Transformation, Wipro Technologies as Specialist – Microsoft Solutions. His work profile involves higher end technology consulting in the field of various Microsoft technologies like .NET and SQL Server 2000 Administration, Analysis services and BI.