Integrating With .NET

Patterns for Leveraging SAS in Microsoft's Strategic Architecture

Dan Jahn
SAS
What Is .NET

- .NET Framework
  - Visual Studio .NET
  - ASP.NET
- .NET Servers
  - BizTalk
  - SharePoint Portal Server, SharePoint Team Services
  - OS – Windows .NET Server
  - Commerce Server, Application Center, Content Management Server, Exchange Server, Host Integration Server, Internet Security and Acceleration Server, Microsoft Operations Manager, Microsoft Operations Manager, Mobile Information Server, SQL Server
## Terminology

<table>
<thead>
<tr>
<th>SAS</th>
<th>.NET</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset</td>
<td>Table</td>
</tr>
<tr>
<td>Library</td>
<td>Dataset</td>
</tr>
<tr>
<td>Observation</td>
<td>Record</td>
</tr>
<tr>
<td>Variable</td>
<td>Column</td>
</tr>
<tr>
<td>XML</td>
<td>XML</td>
</tr>
</tbody>
</table>
Using the .NET Framework With SAS

- COM Interop
  - Tightly integrated with .NET
- IOM Workspace
  - Toplevel SAS Object
- Workspace Manager
  - Creates connections to SAS Workspace
The Workspace Hierarchy

- SAS Workspace
  - DataService
    - Libref
  - FileService
    - Fileref
  - Utilities
    - FormatService
    - ResultPackageService
    - OptionService
    - HostSystem
  - LanguageService
    - StoredProcessService
  - ADO/OLE DB or JDBC
.Net to Enterprise Servers

Windows Client

OS/390

UNIX

Windows NT/2000 Server

IOM Bridge for COM

Local to Client

VB Program

COM

SAS

COM

DCOM
XML

- SAS
  - XML Libname Engine – Can read or write XML
  - XMLMAP – Maps XML on the way in to SAS
  - IOMProvider – Populate a .NET Dataset

- .NET
  - .NET DataSet – can read and write XML
    - Is Disconnected; contains all data
  - OleDBDataAdapter – uses Ole/DB but can populate a .NET DataSet
Web Services

- Based on SOAP
  - XML on the wire
- Can be hosted on any Web Server
  - JSP or ASP
- Stateless
  - Different from COM
ASP.NET

- Integrated with IIS
- Uses compiled code with real VB
  - Scripto no longer needed
- Demo Application
BizTalk

- EAI – Enterprise Application Integration
- Business Process Automation
- XLANG
- Ports
  - File
  - Message Queue
  - HTTP
  - COM
WebDAV

- Web Distributing Authoring and Versioning
- Supported by IIS 5.0, Apache
- Saves Properties with the content
  - Metadata for the document and the document are stored together
SharePoint Integration

- SharePoint Portal Server AND SharePoint Team Services
  - STS uses SQL Server
    - Designed for small teams
  - SPS uses Web Storage System (same as Exchange)
    - Designed for the enterprise
    - Search and browse WebDAV properties
Interoperability Summary

- COM, BizTalk, XLANG, MSMQ, Active Directory, LDAP, ASP.NET, VB, C#, XML, OLE/DB, OleDBAdapter, Dataset, ADO.NET, Web Services, SharePoint, WebDAV, IIS...
More Information

- SAS Integration Technologies Homepage
  http://www.sas.com/rnd/itech
  ASP Sample Library - Samples

- Microsoft .NET Homepage
  http://www.microsoft.com/net/

- Dan.Jahn@sas.com
semiconductor

Documents

<No Title>
index.html

Modified: 6/14/2001

jahn

AMAT
WORK.A.AMAT.GIF
Proc reg plot for AMAT

Modified: 6/14/2001

jahn

TER
WORK.A.TER.GIF
Proc reg plot for TER

Modified: 6/14/2001

jahn
Published with webdav

close = 86721 -11.465 d +0.0004 dsq

N 251
Rsg 0.8350
AdjRsg 0.8337
RMSE 8.6316

Discussion about http://localhost/Stocks/semiconductor/index.html
Subject: I think it's at a low...buy, buy, buy!!!
From: SUGI2763/MyUser

3/19/2002 5:43 PM
/// <summary>
/// Call the SAS web service, telling it to run Proc Corr
/// Note that we have the nice error handling here
/// </summary>
private void btnRunProcCorr_Click(object sender, System.EventArgs e)
{
    localhost.Service1 svc = new localhost.Service1();
    try
    {
        DataSet newds = svc.RunStoredProcedure("ProcCorr", m_DS);
        dgrResults.DataSource = newds;
        dgrResults.DataBind();
        Label1.Text = ";
    }
    catch(Exception ex)
    {
        Label1.Text = "You need to start sas before doing this.";
    }
}
### Dataset

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Delete</td>
<td>Edit</td>
<td>1</td>
</tr>
<tr>
<td>Delete</td>
<td>Edit</td>
<td>2</td>
</tr>
<tr>
<td>Delete</td>
<td>Edit</td>
<td>3</td>
</tr>
</tbody>
</table>

### Statistics

<table>
<thead>
<tr>
<th>TYPE</th>
<th>NAME</th>
<th>Y</th>
<th>X</th>
</tr>
</thead>
<tbody>
<tr>
<td>MEAN</td>
<td></td>
<td>15</td>
<td>2</td>
</tr>
<tr>
<td>STD</td>
<td></td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>N</td>
<td></td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>CORR</td>
<td>Y</td>
<td>1</td>
<td>0.5</td>
</tr>
<tr>
<td>CORR</td>
<td>X</td>
<td>0.5</td>
<td>1</td>
</tr>
</tbody>
</table>
' This combines the SendDataToSAS and GetDataFromSAS methods to take an input DataSet and return the computed Dataset.

<WebMethod()> Public Function RunStoredProcedure(ByVal storedProcedureName As String, ByVal inputDataset As DataSet) As DataSet

    Dim obSAS As SAS.Workspace = GetSAS()

    ' Send the data to SAS; create the WebSvc libname
    SendData(obSAS, inputDataset)
' Run the requested stored procedure

obSASEvents = obSAS.LanguageService
obSASEvents.Async = True
obSASEvents.StoredProcessService.Repository = "file:c:\SASRepository"

' Parameters to a StoredProcess are name=value space-separated pairs
Dim params As String

' "inData=WebSvc.a outData=Work.out"
params = "inData=WebSvc." & inputDataset.Tables(0).TableName & " outData=work.out"

obSASEvents.StoredProcessService.Execute(storedProcedureName, params)
' We're using events to detect errors in the submitted SAS code. So, we'll
' wait here until the SAS code has finished running, and see if we did get any errors.
m_EventComplete.WaitOne(20000, True)
If (m_ErrorCount > 0) Then
    ' Web service exceptions can be caught be the caller
    m_ErrorCount = 0
    Throw New SystemException("The submit has failed:" & storedProcedureName)
End If

' This method copies the named SAS dataset into a .NET dataset
RunStoredProcedure = GetData(obSAS, "work.out")
DoneWithSAS(obSAS)
End Function
%LET inData=sashelp.class; * The name of the dataset to read*

%LET outData=work.out;   * The name of the dataset to create*

*ProcessBody;*

proc means data=&inData;
    output out=&outData;
run;