A Cutting Edge Approach in Bridging between Microsoft .NET Framework and SAP R/3

Anupam Das
Student of M. Tech (CSE)
JIS College of Engineering

Avick Kumar Dey
Student of MCA
DSMS Business School

Baishakhi Karmakar
Student of MCA
DSMS Business School

Anirban Das
Research Scholar
Department of Mathematics

ABSTRACT
The approach here is to create a connectivity path between Microsoft .NET and SAP R/3 software. Enterprises of all ends are trying to integrate and manage all data within a single sign on system. The mission is appropriate as because they store their assets i.e. data in innumerable of incongruent systems - relational databases, different operating systems, mainframes and so on. Managing their own data in an efficient, integrated, and single login approach has led the enterprises in a new height of flourishing.

SAP, in the domain of enterprise software, is the leader with thousands of installations so far. Specifically the EAI and EII product vendors eye on SAP R/3 Enterprise as a data source to make it needful for integration. Microsoft .NET as a platform for developing enterprise software is emerging for integration with SAP R/3 Enterprise.

Keywords
Data Warehousing, SOAP, HTTP, CLR, WSDL, UDDI, RFC, DLL.

1. INTRODUCTION
Before getting introduced we will simply have a glance on SAP AG and Microsoft dot net along with their approaches. In 1972, SAP AG was founded in Germany by five engineers who had a mission to develop an integrated business application for enterprises. SAP stands for *Systeme, Anwendunger, und Prudukte in Datenverarbettung* i.e. Systems, Applications and Products in Data Processing. The leading ERP package vendor SAP launched its first application software named R/2 in early 80's.

The main feature of R/2 was like centralized database. In 1992 SAP launched another application software R/3 which captured the most ERP software market and it came to dominate in market. And gradually SAP AG became the 3rd largest independent software vendor in 1999. SAP extended in different modules like Customer Relationship Management (CRM), Supply Chain Management (SCM) and Data Warehousing in the same year. The business portal of SAP around the Internet is *mySAP.com*

This portal is dedicated to the users in various roles in enterprise & in E-Commerce also.

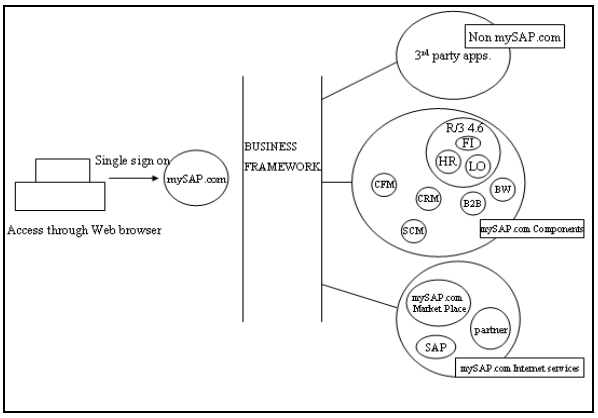

Fig1: SAP Approaches: Modified from the SAP Business portal: mySAP.com (Source SAP, 12/99)

NET is a "software platform", made by Microsoft. It's a language neutral environment for writing programs that can easily securely interoperate. Rather than targeting a particular hardware/operating System combination, programs will instead target ".NET" and will run wherever .NET is implemented. Dot net framework is of two sub parts:

i) CLR- Common Language Runtime (Heart of .NET)

ii) Hierarchical set of class libraries (DLLs i.e. Dynamic Link Libraries)

As dot net framework is a widespread environment for building, deploying, and running web services as well as web applications, in this experiment we will use ASP.NET (updated version) as front end and C# as code behind.

2. LITERATURE REVIEW
Ulf Helmke and Matthias Hansgen opines about integration of external systems with SAP in their article like 'Building Bridges between the SAP Environment and Legacy Systems'. In accordance with them the integration of external systems into SAP environment is possible with SAP's exchange infrastructure generally called as XI. The external systems are connected to these XI through technical adapters. These legacy adapters are responsible in integrating legacy applications along with data structures into SAP based environment.

3. BRIDGING BETWEEN SAP & MICROSOFT .NET–A BROADER LOOK

Gone are the days of stand-alone technologies. Now the era comes to communicate with different technologies because the necessity demands.

Let's say for instance, two organizations are there orgA and orgB. Now orgA implements its total domain in SAP. On the other end, orgB manages its departments through application software made in Microsoft .NET. If orgB becomes the client or sister concern of orgA or, orgB is sold to orgA, then the utility arises to communicate with both he organizations. As orgA follows SAP, it will try to connect with the system of orgB i.e. in Microsoft .NET.

Two ways we will show through which the communication is possible:

i) Communication through Webservices

ii) Communication through SAP.NET connector.

3.1 COMMUNICATION THROUGH WEB-SERVICES

Webservices are nothing but a service where we can guide various web methods. In those web methods, we can place our data. These web methods are consumed by web application and the data, written in web-methods, is fetched.

Here we can create a web-service in Microsoft .NET. We follow Microsoft Visual Studio .NET (version 2005 & onwards).

Route to create Web-service in Microsoft .NET:-

a. We need to install Microsoft Visual Studio. NET(version 2005 & onwards).

b. Open the Visual Studio & there will be options. We select ASP.NET Web-service.

c. Now it is needed to give a valid name and path of the service and click "OK" button.

d. The window opens with the code as:

```
//[Import Assemblies]
Using System; // import namespaces
Using System. Web; // import namespaces
Using System.Web.Services; // import namespaces
Using System.Web.Services.Protocols; // import namespaces
[Webservice (Namespace="http://tempuri.org/")]
[WebserviceBinding
(ConformsTo=WsiProfiles.BasicProfile1_1)]
//[Code for webservices]
public class Service: System.Web.Services.Webservice
{
public Service()
        {
        }
[WebMethod]
public string HelloWorld()
{
        return "Hello World";
}
----------
----------
}
```

So many web methods we can write at the [WebMethod] section.

e. Now switch to solution explorer and click to *Add Web Reference.* One window will open to select browsing webservice.

f. Browse the webservice named Service .

g. The window will show the service with its path:

http://localhost:[port name]/---
/[application_name]/Service.asmx

[In case of local server only]

Under the said path, the web method will be shown as a link.

As on the example it will be "*HelloWorld*" .

h. Click on Add Reference on the same window.

i. In Solution Explorer, we can see the files added already:

Service.asmx

Service. disco

Service.discomap

Service.wsdl

j. Finally go for Build menu and build solution.

k. While Build is succeeded Run it.

l. In Browser which comes after running the solution, click Service.asmx. And, it will return the web method.

m. By clicking the web method link here, HelloWorld, a window comes with Invoke button.

n. Invoking it one XML file will pen stating the string value of the web method i.e. "Hello World" will be returned.

So, webservice is ready in .NET (version 2005 & onwards).

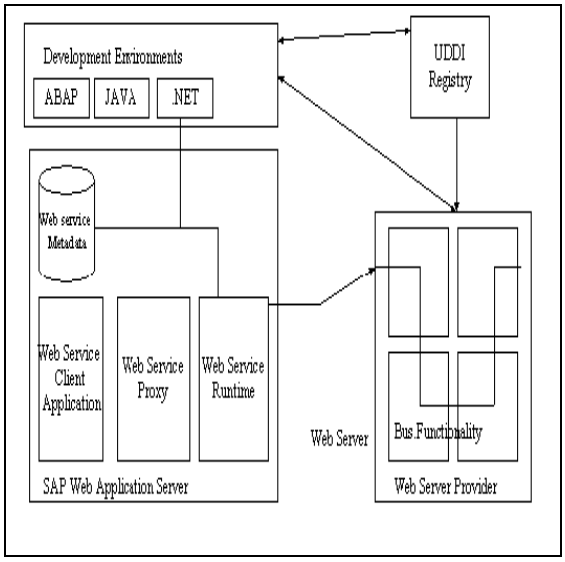

Fig 2: Connection to SAP through dotnet Webservice [Source: Modified from SAP AG 2005, SAP TechEd '05/CD153/23]

3.2 COMMUNICATION THROUGH SAP.NET CONNECTOR

Visual studio .NET platform provides a programming environment between .NET and SAP called *SAP.NET connector*. It enables communication by supporting .NET web-services and Remote Function Call (RFC) of SAP systems. All common language runtime programming languages like Visual Basic .NET, C# etc can be used.

We will discuss some features of SAP.NET Connector along with its architecture.

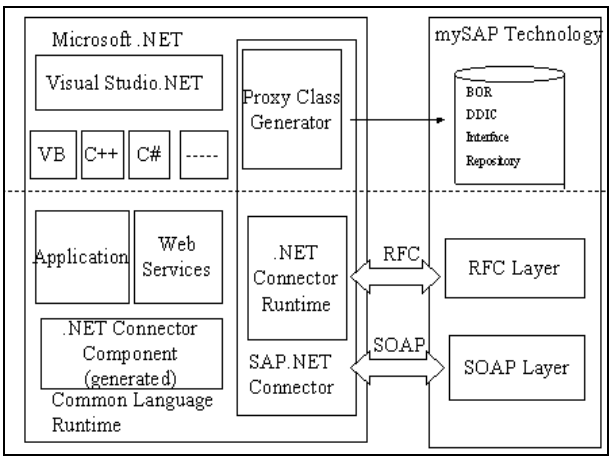

Fig 3: Connection to SAP & dotnet -upper portion of the dotted divider is representing the design time

The SAP.NET connector with in Visual Studio includes various designer components to develop applications of SAP including SAP Table component and destination components.

SAP Tables are nothing but a datatype as they generally keep the results of RFC call. As we get graphical interfaces (GUI) in Visual Studio for SAP proxies, we can customize in our own way, which lead to a better understanding while doing programs to connect SAP system and .NET.

For the interaction with SAP proxies, .NET developers can choose any language of CLR. We prefer here C#. SAP proxies are self-generated in C#.

The SAP.NET connector facilitates a custom tool by which without having to do changes manually or rerunning proxy generation wizard, SAP proxies can be updated and customized automatically.

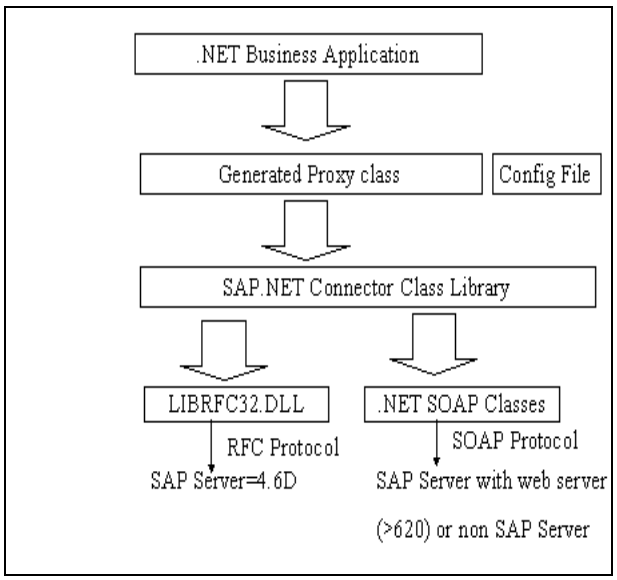

Fig 4: Runtime Architecture

SAP proxies communicate with SAP systems in two ways:

1. Through SOAP (Simple Object Access Protocol)
2. By SAP Remote Function Call protocol (*librfc32.dll*)

Up to the release of 4.6D, SAP doesn't have SOAP support. But from 6.20 release supports either RFC or SOAP. Via SOAP non-SAP systems can be connected.

We derive SAPClient solution from *SoapHttpClientProtocol* class of Microsoft. This class specifies class proxies at the time we use SOAP.

The SAP.NET connector supports all types of Visual Studios .NET solution. We can easily write RFC Server application which run in .NET platform and ca be implemented in SAP. For Server (SAP) client applications be written through RFCs or SOAP/ HTTP/XML. We can write .NET windows and web forms that are accessible to SAP RFC.

Now we come to create an ASP.NET web application using SAP.NET connector, which will present a clear picture of communication of SAP and .NET.

In the example here we will try to connect with SAP system from .NET and fetch data from there.

Walkthrough to make the connection:-

A. Go for Visual Studio .NET. Select a new C# web form.

B. From File menu select New → New Project → Visual C# projects → ASP.NET web applications

C. A web form will come while we go for ADD New Item in Solution Explorer and select web form.

D. Give the name say, webformSAPNET.aspx.

E. We need to have some web controls now. Let us have a TextBox, a Button and one GridView control.

F. To connect with SAP server we need to add Proxy classes:

i) Go to Solution Explorer and right clicking the solution get Add → Add New Item

ii) Choose Web Project Items→ SAP connector class. Now click on 'Open'. Now SAP.NET Connector wizard will open now.

iii) Proxies can be generated from several places:

a. SAP server

b. WSDL (Web Services Description Language) files which are in interface repository (IFR)

c. Standard WSDL files.

We are to take decision where from we want to create proxies.

iv) Choose client proxy object type.

v) Choose the RFM (Remote Function Modules) we use in the proxy object. In this example, enter 'RFC_CUST*' as search argument in Name-Filter and choose RFC_CUSTOMER_GET.

vi) We need to add the modules to proxy object. Then go for 'Next'. The proxy classes are added to the project.

G. Go to Build menu and select Build Solution.

H. For authenticating username and password create an SAP login page.

i) In Solution Explorer Add→Add New Item.

ii) Select Web Project Items→ SAP login form.

Do give the name as SAPLogin.aspx.

I. In Solution Explorer window find SAPLogin.aspx and make it present in design view.

Search the destination1 component and click on it to set the properties to connect to SAP system.

Now, System connection information is destination object is set.

J. Do databind to Gridview BRFCKNA1Table, which is a parameter of RFC_CUSTOMER_GET that reflects the customers' list.

i) Go to SAP Table wizard from SAP Proxy toolbox and drag & drop it. Select BRFCKNA1Table.

ii) In Gridview properties make the datasource to BRFCKNA1Table.

K. Now we have to select connect code from SAP proxy toolbox. In the eventhandler drag& drop the same.

It connects with SAP server using proxy wizard.

The code will be like:---

```
private void Button1_Click(object sender, System.EventArgs e)
{
// object of SAPProxy1 class creation
  SAPProxy1 prox=new SAPProxy1();
try
{
prox.Connection=SAP.Connector.SAPLoginProvider.GetSAPConnection(this);
//method is called.
prox.Rfc_Customer_Get("",TextBox1.Text,ref brfcknA1Table1);
//finally data bind.
this.DataBind();
}
Catch(Exception exp)
{
Response.write(exp.Message);
}
}
```

L. Build Menu→ Build Solution.

Run the Application.

Browser window will show SAPLogin.aspx page.

M. Insert the required fields.

Insert a search argument in TextBox & go for the Button for Search.

N. This will make a bridge to SAP system. Populate the GridView with the desired search results.

This is the way to connect with SAP systems from Microsoft .NET.

4. CONCLUSION

An important part of integration and bridging between Microsoft .NET framework and SAP R/3 software is the dynamic communication between not only two different software and also two different platforms of business entities. This paper presents a comprehensive framework to integrate Microsoft .NET and SAP R/3 software communication via Webservices and SAP .NET connector also. This paper presents an approach to connect with SAP & Non SAP Systems which will be effective to merge up two or more various SAP enabled organizations and organizations which follow non-SAP support.

5. REFERENCES

[1] Alice Hsu, Manage your Way to ERP Implementation Success!

[2] Haim Mendelson, ERP Overview, January 2000, Graduate School of Business, Stanford University, Stanford, CA 94305-5015.

[3] Ulf Helmke and Matthias Hansgen, October 2005, Building Bridges between the SAP Environment and Legacy Systems

[4] SAP . NET Connector, version 1.0, November 2002.

[5] SAP AG 2005, SAP TechEd '05/CD153/23

[6] SAP AG 2005, SAP TechEd '05/CD153/24

[7] The SAP Business Portal: mySAP.com(SAP, 12/99)

[8] Sapr3abap's Weblog, available at http://sapr3abap.Wordpress.com/2008/04/09consume-web-services-in-abap-and-how-to-connect-sap-to-net/

[9] SAP Community Network Forums, available at https://sdn.sap.com/irj/sdn/thread?messageID=4109146

[10] The Code Project, available at http://www.codeproject.com/KB/mobile/SAP_connector_for_NET.aspx

[11] The Code Project, available at http://www.codeproject.com/KB/dotnet/SAP_NET_Connector.aspx

[12] www.cio.com, ABC: An Introduction to ERP- Thomus Wailgum, CIO, March '07.