



Published on System iNetwork (<http://systeminetwork.com>)

SOA Using CA Plex and WCF

By tzura

Created 11/09/2007 - 08:00

Service-Oriented Architecture (SOA) is becoming the backbone of enterprise IT architecture strategy. SOA-enabled organizations are reaping many benefits, most notably the ability to fully utilize existing IT assets by making them available across technologies and organizational boundaries. Among the different SOA platforms available, Microsoft's Windows Communication Foundation [1] (WCF) is one of the core SOA technologies. WCF is an excellent framework for integrating business services from Microsoft Server, the System i, and other enterprise computing platforms.

Beware of hyped-up SOA promises dealing with business agility and reusability. SOA architectures and technologies don't guarantee automatic success. In fact, many experts have pointed out that poorly implemented architectures can inhibit agility and provide a significant barrier to technical progress. True reuse takes both strong analysis and enterprise-class technology tools to bring the promise to reality.

How can an organization take advantage of SOA and WCF without succumbing to the pitfalls? One way is to use an architected rapid-application-development (ARAD) tool such as CA Plex [2] to help develop an enterprise SOA framework. CA Plex insulates an organization from underlying technology such as WCF and allows developers to remain focused at the business technology layer. CA Plex's pattern-modeling capability eases reuse and accelerates the development of an SOA gateway over both legacy applications and new systems. There is very little actual "code" involved with creating a WCF SOA gateway layer when you use CA Plex.

How Does CA Plex Support SOA?

CA Plex has a long history of support for SOA and components. CA Plex's model-based paradigm supports abstraction and reuse. It also supports COM and EJB as part of the core platform, and third-party patterns involving SOA have been available for some time from companies such as Websyidian.

Figure 1 [3] illustrates CA Plex SOA support and shows how you can use the SOA gateway to connect to business services on a variety of platforms. A key strength of CA Plex is its multiplatform capability. CA Plex generates both Java and C#, the de facto standards for developing enterprise web services on the System i and Microsoft Server platforms. CA Plex also generates other languages such as RPG and C++, which are useful for integrating with legacy

business logic.

In complex organizations, you often need to provide web services that pull from different systems implemented in different technologies. It is not uncommon for a web service layer to need to pull from System i-based RPG/COBOL, Java, and .NET-based systems simultaneously. With CA Plex, the implementation of a web services layer is transparent to the underlying technology, and you can generate business logic or services in the language that best makes sense.

With the planned release of CA Plex version 6.1, WCF support is being added to the core product. This gives another enterprise technology option for implementing SOA, one that is fully backed by Microsoft.

What Is WCF?

Windows Communication Foundation is the .NET communication subsystem that lets applications connected by a network communicate. The WCF programming model, as introduced with .NET 3.0, unifies Web Services, .NET Remoting, Distributed Transactions, and Message Queues into a single SOA programming model for distributed computing.

A WCF Service ([Figure 2](#) [4]) is the heart of the communication process. This kind of service is basically a component running within a host such as IIS or a custom executable. The host runs the service and directs calls to it over supported protocols such as http. External clients interact with the WCF service via a proxy that acts as a local instance of the service. The proxy takes the client request to call an operation on the service, packages it using the protocol that the service supports, and sends it to the service by way of the service host. The service then executes the operation and sends back any output to the proxy. The proxy in turn unpackages that information and returns it to the client.

An important concept is that the implementation of the WCF Service is protocol independent. A service invoked over the web may use http protocols although you may best implement a local service using MSMQ or .NET remoting. Regardless of the protocol, code changes are not required at the WCF service layer.

CA Plex takes this implementation independence one step further. You can automatically generate Web services modeled in Plex to WCF. The Plex developer doesn't need to have detailed knowledge of the WCF framework implementation details and can generally stay at the abstract model level.

The concept of WCF endpoints is shown in [Figure 3](#) [5]. As you can see from the lollipop-shaped figure, endpoints tell you the where, what, and how about the service. The address, typically a URL, tells where to find the service. The binding explains which communication protocols and security mechanisms are valid (i.e., SOAP over http or SOAP over MSMQ). The contract specifies the data requirements of the service. You can generate this information to WSDL and schema information for use with non-WCF SOA frameworks.

Plex and WCF

CA Plex already has an extensive generic component-modeling capability. WCF modeling (as well

as COM and EJB component modeling) fits nicely into this overall framework. [Figure 4](#) [6] shows how a CA Plex component model maps to WCF concepts. As you can see by the figure, a CA Plex "Code Library" sits at the highest level. The next level down is the "Package," which corresponds to a "Namespace." At the lowest level, a CA Plex Function corresponds to a WCF Data Contract.

CA Plex Development Workflow, Step by Step

The WCF development workflow does not vary significantly from a standard CA Plex modeling workflow. The standard flow is to model, generate, build, and deploy -- with a heavy emphasis on the first step, modeling. After you model the WCF SOA gateway, the subsequent generate, build, and deploy steps fall out more or less automatically.

First Step -- Model

The first step is to model the WCF interface and the underlying business logic. Plex provides an extensive pattern-based modeling capability, so this part is straightforward from a technology perspective. However, this step is important from an analysis perspective. You should do considerable planning to develop an agile framework that will be useful to the various consumers of the gateway services.

In Plex, you use the same basic techniques to model business logic regardless of the platform and target language. [Figure 5](#) [7] shows how the interface contract is modeled in CA Plex and how it relates to the generated code. In this example, a product-related namespace (or Package in Plex) contains a public interface named `IProduct` and an operation named "RunStockCheck." The CA Plex Object Browser depicts this relationship in an intuitive manner. [Figure 6](#) [8] shows how the data contract is modeled. As you can see, this is where details of the data passed to and from the service are specified. "StockCheckName" is a string element that will be used by the "RunStockCheck" service.

Second Step -- Generate

The next step is to generate the code from the model. For WCF SOA implementations, you generate the WCF logic as C#. Note that generating the WCF gateway as C# does not preclude the Plex developer from calling other Plex or external functions that may be deployed as Java, C++, or even RPG functions. These functions may access disparate enterprise databases such as SQL Server, Oracle, and DB2 running on a combination of System i, Unix, or Microsoft server platforms. Plex also generates the Microsoft Visual Studio project and the scripting information necessary for a seamless build and deploy of your WCF service.

Third Step -- Build

After you generate the code, the third step is to invoke the appropriate native source code compiler and IDE. Plex most commonly utilizes Microsoft Visual Studio and IBM Rational Application Developer for the build step, in general. However, for WCF and C# functions, you can use MCBuild directly.

Last Step -- Deploy

The last step in the development workflow is to deploy your WCF SOA gateway. CA Plex does a good bit of this automatically and packages up the runtime objects needed to run the Plex applications. CA Plex also provides a .NET management console for configuring and tuning the .NET application layer (Figure 7 [9]). In the management console, you can set application function location information and establish various database defaults. You can configure additional servers and environments, depending on production workloads.

Using Your WCF SOA Framework

After you have deployed your CA Plex-based WCF SOA framework, you can put it to productive use. External customers may need access to web services to automate common business interactions such as catalog requests and stock checks. Internal IT users can make use of services to create "mashup" presentations, dashboards, and workflow automation systems. An example of this type of integration is combining cartographic data from Microsoft Live with customer address data on the System i via a WCF service.

Here are some other case-study examples:

- CCH, a Wolters Kluwer business and a leading provider of tax, accounting, and audit solutions (salestax.com), relied on CA Plex as the primary tool to build its CertiTAX and ZipSales Returns solutions, which rely heavily on web services.
- TEEX, Texas A&M Extension Service, used CA Plex web services to integrate scanned registration data, government compliance requirements, and test scoring for a Department of Homeland Security program. This system relies heavily on both System i and Microsoft Server components to provide a complete solution.
- One insurance company used CA Plex to integrate with web services published by an insurance policy rendering package. The end result was a drastic reduction in the time needed to produce and print policies.

You and CA Plex

The strength of CA Plex is that it lets organizations model their SOA technology layer using abstract patterns and then generate the technical underpinnings that make it all possible. This particularly true of WCF -- CA Plex shields developers from technical details while at the same time provides all the power of native code generation utilizing WCF, .NET, and C#.

John Rhodes is the principal architect for ADC Austin, and you can reach him at jdrhodes@adcaustin.com [10]. Rob Layzell is a senior product architect for CA, and you can reach him at robert.layzell@ca.com.

Copyright © Penton Media

Source URL: <http://systeminetwork.com/node/26130>

Links:

- [1] <http://msdn2.microsoft.com/en-us/netframework/aa663324.aspx>
- [2] <http://ca.com/us/products/product.aspx?ID=258>
- [3] [http://pentontech.com/IBMContent/Images/article/55888_50489_Slide1\[1\].JPG](http://pentontech.com/IBMContent/Images/article/55888_50489_Slide1[1].JPG)
- [4] [http://pentontech.com/IBMContent/Images/article/55888_50490_Slide2\[1\].JPG](http://pentontech.com/IBMContent/Images/article/55888_50490_Slide2[1].JPG)

- [5] [http://pentontech.com/IBMContent/Images/article/55888_50491_Slide3\[1\].JPG](http://pentontech.com/IBMContent/Images/article/55888_50491_Slide3[1].JPG)
- [6] [http://pentontech.com/IBMContent/Images/article/55888_50492_Slide4\[1\].JPG](http://pentontech.com/IBMContent/Images/article/55888_50492_Slide4[1].JPG)
- [7] [http://pentontech.com/IBMContent/Images/article/55888_50493_Slide5\[1\].JPG](http://pentontech.com/IBMContent/Images/article/55888_50493_Slide5[1].JPG)
- [8] [http://pentontech.com/IBMContent/Images/article/55888_50494_Slide6\[1\].JPG](http://pentontech.com/IBMContent/Images/article/55888_50494_Slide6[1].JPG)
- [9] [http://pentontech.com/IBMContent/Images/article/55888_50495_Slide7\[1\].JPG](http://pentontech.com/IBMContent/Images/article/55888_50495_Slide7[1].JPG)
- [10] <mailto:jdrhodes@adcaustin.com>