

CRN TEST CENTER

Review: Code In .Net, Run On J2EE

By Mark Spiwak

Mainssoft has one up on the competition: It's the only place to go for all enterprise Java and .Net mixed mode solutions.

San Jose, Calif.-based Mainssoft's Visual MainWin For J2EE product suite can convert code from .Net to Java, enabling .Net and J2EE to work together, the CRN Test Center finds.

Visual MainWin uses MSIL binaries from C#, ASP.Net and VB.Net code compilations and cross-compiles them into Java byte code. The output Java then can be deployed to Tomcat or an application server. Therefore, runtime code is in pure Java, and source code is maintained in .Net languages.

Microsoft's grip on the presentation layer makes ASP.Net developers more desirable than JSP counterparts. ASP.Net developers are easier to find because not as much expertise is required to build presentation code. What's more, junior ASP.Net developers are profitable and can be productive in a relatively short time, due to Microsoft's simple frameworks and highly productive Visual Studio development suite.

However, in Java shops where core assets are built with EJBs, getting both technology stacks to work together isn't that simple. A typical workaround is to consume EJBs with Web services and then consume that Web service within .Net applications at the presentation tier. Combining .Net and J2EE applications makes it difficult for operations to isolate problems.

While runtime complexities alone make it unappealing to mix technologies, testing creates even more stress for developers. Two different sets of tests must be conducted, with boundaries set at the presentation layer, between ASP.Net and Web services and between Web services and EJBs. That's the only way it can be done

without Visual MainWin.

Even with a Web services architecture in place, interoperability between SOAP servers can be a big issue during development because different platforms can interpret data structures differently. Since data structures are usually tightly bound to environments, developers with domain knowledge of both sides are needed to orchestrate correct solutions between different groups without encountering problems.

For instance, by exposing an ADO.Net's DataSet from a database table as a Web service, any presentation tier written in Java will break because it can't read .Net artifacts. These data-type limitations between .Net and Java can be worked out using Visual MainWin. Mainssoft provides instructions on how to serialize and deserialize data so that Java Web services can consume .Net directly.

Solving interoperability with third-party bridging technologies can alleviate some of the complexities. But since most of these enterprise-bus products use proprietary solutions to optimize communication, applications become intricately dependent on them. Integrators must be aware that by not locking low-level access to specific technologies, they are essentially shifting to old models often used by EAI vendors. Adhering to standards should take precedence whenever possible, so enterprise applications aren't bound to proprietary solutions.

Visual MainWin solves all of those issues because it provides an open-ended architectural solution. It doesn't tie architects to any vendor solution. Porting code also can solve all interoperability problems, but that's extremely expensive for development. Web services offer a simpler alternative, but Test Center engineers believe that Visual MainWin can make it cost-effective.

Visual MainWin covers C#, ASP.Net, VB.Net and most of

the .Net framework. WinForms for building Windows forms, though, can't be converted because there are no direct alternatives on other platforms such as Linux. When using third-party components, developers must recompile them into Java so that .Net applications can use them on the Java side.

If source code from third-party software isn't available, customers have to buy a Java implementation of that software or abstract the services provided by the software through code. The correct way to work with many third-party applications is to abstract services from those products. If that practice isn't put in place, porting code then becomes extremely difficult and Mainsoft's solution won't work.

The new Visual MainWin 1.8 Portal Edition promotes what it calls .Net extensions to work with the IBM

WebSphere portal software. These extensions are ASP.Net code running natively inside the WebSphere portal. The Visual MainWin enterprise version works with WebSphere, JBoss and BEA WebLogic and includes multi-CPU capabilities. It also can consume EJBs. Visual MainWin, too, is now able to work with Remoting and .Net System.DirectoryServices on J2EE.

Visual MainWin costs \$5,000 per developer seat and \$2,500 per CPU. Mainsoft provides in-person technical training and offers on-site technical support, including phone support. The company offers an average margin of 30 percent, and solution providers also can earn revenue by providing complete support when porting .Net applications. Mainsoft funds 50 percent of joint marketing and lead generation for up to \$5,000 per year.