



Interoperating with .NET – Beyond Web Services

Jeroen Frijters

Sumatra Software b.v.

jeroen@sumatra.nl

<http://weblog.ikvm.net/>





Overview

- Full Disclosure
- Why Not Web Services?
- The Alternatives
- Demonstrations



Full Disclosure

- I'm the author of the Open Source IKVM.NET project. One of the solutions for integrating Java and .NET code.



Why Not Web Services?

- Networking overhead
 - Method call latency can be very significant for “chatty” interfaces.
- Not appropriate for client apps
- Deployment complexity



The Alternatives

- Unified Model
 - Compiling Java source on the CLR
 - Running Java Byte Code on the CLR
 - Running CIL on the Java Virtual Machine
- Proxy Based
 - Remoting
 - Hosting the JVM inside the CLR (or v.v.)



The Products

- Commercial
 - J-Integra for .NET (Ja.NET)
 - Codemesh JuggerNET
 - JNBridge
 - Microsoft Visual J#
 - Mainsoft Visual MainWin
 - Stryon iNET
- Open Source
 - Caffeine
 - IKVM.NET
 - IIOP.NET



Remoting

- Building on either Java RMI or .NET remoting, you can communicate between the JVM and CLR.
- Products
 - J-Integra for .NET (Ja.NET)
 - JNBridge
 - Caffeine
 - IIOP.NET

Compiling Java Source on the CLR

- Since the CLR supports multiple language, you could, theoretically, recompile your Java sources on the CLR.
 - In practice it isn't quite that easy. What about the Java libraries?
- Products
 - Microsoft Visual J#
 - Supports only JDK 1.1.4
 - Primarily intended as migration path for J++ users

Running Java Byte Code on the CLR

- Similar to the previous option. Instead of recompiling your Java sources, you convert Java Byte Code to CIL at runtime (or ahead of time).
 - Still need to find a way to have the Java libraries available.
- Products
 - IKVM.NET
 - Uses GNU Classpath to provide an implementation of the Java libraries.

Running CIL on the Java Virtual Machine

- Convert CIL to Java Byte Code (either at runtime or ahead of time).
 - What about the .NET Framework libraries?
- Products
 - Mainsoft Visual MainWin “GrassHopper”
 - Uses a modified version of (part of) the Mono implementation of .NET Framework libraries
 - Stryon iNET
 - Uses their own (partial) implementation of the .NET Framework libraries



Hosting the JVM Inside the CLR

- Using JNI host the JVM inside the CLR process.
 - Requires proxies on both sides and method marshalling is expensive.
- Products
 - Codemesh JuggerNET
 - Caffeine



Demonstrations

- Microsoft Visual J#
 - Converting a J++ application
- IKVM.NET
 - Running Eclipse on .NET
 - Using a Java library from .NET
 - Using a .NET library from Java
- Mainsoft Visual MainWin
 - Running an ASP.NET site on Tomcat



Mapping Object Models

- Java and .NET object models are very similar, but there are some differences.
- Wrapping vs. Mapping
 - `java.lang.Object`
 - `java.lang.String`
 - `java.lang.Throwable`
 - arrays
- In a “proxy” model, it is easier (for the developer of the interop product) to simply let the user deal with these issues.
- In a “unified” model, code has to run as is. So the interop product needs to do more work to make the two object model interoperate.



Future

- I will continue to develop IKVM.NET into the premiere Java .NET interoperability tool 😊
- The Portable.NET project would like to add Java Byte Code support to their runtime, but at this point it is unknown whether this will ever happen.
- A Universal Virtual Machine?



Questions





Resources

- <http://www.stryon.com/products.asp?s=4>
- <http://iiop-net.sourceforge.net/index.html>
- <http://j-integra.intrinsyc.com/net/info/>
- <http://www.codemesh.com/en/JuggerNETCurrentRelease.html?gadJNET>
- <http://www.jnbridge.com/>
- <http://caffeine.berlios.de/site/index.html>
- <http://msdn.microsoft.com/vjsharp/>
- <http://www.mainsoft.com/>
- <http://www.ikvm.net/>