Computer Engineering and Intelligent Systems
ISSN 2222-1719 (Paper) ISSN 2222-2863 (Online)
Vol 3, No.6, 2012

Integration of Java EE Applications on C – Based Implementations

RajaSekhar Kraleti
Sivani College of Engineering, Chilakapalem
Srikakulam , AndhraPradesh,India
Tel: +91-8860367494 E-mail: rajkraleti@ymail.com

Under Guidance of MVN Naidu (Asst. Proffesor),
The research is financed by my father.

Abstract

Sometimes we may encounter a scenario in which We need to Integrate the existing Java – Based applications With the Native formats of C/C++ Based implementations For example the applications of Defense, Aerospace weather Forecasting Applications contain complex algorithms and that might be too complex And too difficult or even some times implementing those again will include risqué as well As time consuming also in such cases to integrate such C based implementations with java programs We need an adapter in between those two language API 's . The Java Native Interface (JNI) will provide you the bridge for exchanging data between java and C,C++ API's this article describes the steps that will ease the integration of JAVA With C- Based Implementations.

Keywords: JNI : Java Native Interface
API : Application Programming Interface

1. Introduction

Though its normal for us to implement in general JNI but in large scale industrial perspective I have found the implementation with a large scale Software development tools such as IBM Websphere Integration Developer which will lets the user to work with the SOA where all the services will be saved in a WSRR (Websphere Service Repository Registry) and runs on WebSphere Application Server in the background though it is simple to develop with general programming but no real-time software development was processing on direct programming now a days that is the reason why I have selected this Integration Developer to run my Integrations in The Java™ Native Interface (JNI) which is a programming framework that enables Java code running in a Java Virtual Machine (JVM) to call and to be called by native applications and libraries written in other languages such as C, C++ and assembly. This article describes how developers can leverage this framework to integrate their J2EE™ applications deployed on IBM WebSphere Application Server with these 'C' libraries.

1.1 What are these Websphere technologies actually

Application Server is a Java EE 6 compatible, robust, and highly available middleware environment that provides a platform for hosting and managing a variety of enterprise applications. Some users may encounter scenarios in which they will need to integrate their Java-based applications we do not use Java programs directly for that there were different Software Development tools for that for a high scale organizations as they were written with the most efficient programming methodologies but with native C/C+ + based implementations in some traditional organizations which can be very difficult to reintegrate again and that are very difficult to communicate with Generic SOA (Service Oriented Architecture). For example, applications involving defense, aerospace, weather forecasting, and other scientific applications contain certain algorithms that can only be implemented using these native

languages and are either too complex, too difficult, or too risky to implement in Java. To integrate such C-based implementations with Java programs, you need to put some sort of bridge in place between the two programming language APIs

1.1.1 Implementation

Java Native Interface (JNI) provides this bridge to exchange data between Java and C/C ++ APIs. To achieve this solution, you need to define the interface in Java with the methods you wants to expose to other Java classes, generate a header file out of the compiled code of this interface, and then import the header file in native shared library modules, such as Dynamic Link Libraries (.dll) for Windows® and Shared Object (.so) libraries for UNIX®-based systems. These modules can be created through a variety of tools provided by a number of vendors. This article describes steps that will ease the integration of Java-based applications deployed on WebSphere Application Server with C-based implementations through Java Native Interfaces (JNI). At the end, you will be able to create, configure, and invoke shared libraries for WebSphere Application Server. The information included here applies to IBM WebSphere Application Server V6.1, V7, and V8 and assumes familiarity with the corresponding IBM Rational® tooling. Be aware that the examples presented here are very basic to illustrate the high level development procedure involved, and do not address data validation or other good practices that would otherwise ordinarily be included in typical application development.

1.1.2 My Idea of Development

Here is the way of developing JNI Codes with a WebSphere Integration Developer

Got to the Websphere Integration Developer and select a Java Project from the Integration Developper and after that Create a Java class with the methods that you want to associate with native methods The method declarations will be dependent on the signature of the C methods you want to invoke. In this case, the assumption is that a third party C API requires simple type and an array of data (in this case, a double array). (To invoke another native library, you need to associate it with your DLL by importing its header file during header Implementation.)

Here is the sample method invocation showing the java project Class such that we can write

```
public class InvokeNativeLanguages {
public Static native int sum (int a , int b);
public Static native double sum ( double a , double b);
public Static native double sum (double[ ] a);
    }
```

And after that Compile the Java Interface and run the command on the .class file Compile the Java Interface and run the .Class file generated from {APPSERVER_Rot}/Java/bin directory to generate respective C header files after that we need to implement the methods created in native library It is this library implementation that you can integrate with other C libraries during C API to C API calls. After that we have to configure the native library shared in the runtime environment for the Integration developer. In the path for that in the Application server will be as follows Server > ServerTypes > WebSphere Application Server > server1 > Java and Process Management > Process Definition > Java Virtual Machine > Custom Properties

References

IBM Info center for WebSphere Integration Developper

URL: http://publib.boulder.ibm.com/infocenter/dmndhelp/v6rxmx/index.jsp

Java Naming Interface Info Centre

URL:

http://docs.oracle.com/javase/6/docs/technotes/guides/jni

IBM Developer Works by Sandeep Kundra

URL:

http://www01.ibm.com/support/docview.wss?uid=swg21144595

Wiki

URL:

http://en.wikipedia.org/wiki/Java_Native_Interface

Notes

Note 1.

By encouraging this kind of work culture in the Enterprise work level will almost reduce 30% development costs that intern raises to software development a cheaper and efficient process that can reach many organizations which were suffering from this problem of having a old software in C, C++ based traditional language implementations and not yet supported by any enterprise level tools here i have taken IBM's Integration Developer as my tool on which i have a good command we can select any Integration tool like this even oracle BPM also

Note2

Many thanks provided to the IBM professionals who helped me in learning this proficient and strong technology as now a days SOA is stretching its arms towards perfection this is the right time to introduce this kind of technology