Implementing and Invoking a Remote Object Calling Native Methods via RMI-IIOP and JNI

Minglong QI, Qingping GUO, Luo ZHONG
School of Computer Science, Wuhan University of Technology
Ma Fang San Campus, 430070 Wuhan, China
Email: minglongqi@sina.com; **Tel.:** +86-(0)27-87292452

ABSTRACT

In this article, we discuss how to combine one of the distributed computing technologies created by Sun Microsystems, called RMI over IIOP, and its other famous technology, Java Native Interface JNI. Sometimes in a distributed computing application written entirely in Java, client side application need to invoke native methods specific to a platform and encoded in a another programming language (for example in C or C++) .To do this, we must make a distributed computing solution that can integrate native applications in Java. We think that Sun's RMI over IIOP and JNI is a good choice. The first reason is that, RMI over IIOP takes advantage of both RMI (easy to use and encoding uniquely in Java), and IIOP (interoperability with another CORBA ORB products). In addition to this, we can switch transport protocols from JRMP (Java Remote Method Protocol) to IIOP (Internet Inter-ORB). The second reason is that, JNI cannot only integrate native applications in Java, but can also embed JVM implementation in a native application. To demonstrate the combination between RMI-IIOP and JNI, we have developed a typical example: on the server side, implementation of a remote interface method create an instance of a Java class and called its native method, whose implementation use JNI and Borland InterBase C API to extract a database table. On the client side it invoked this remote object and displays the database table.

Keywords: RMI-IIOP, JNI, InterBase API, CORBA, ORB.