ASYNCHRONOUS RMI

SUMMARY

Java RMI's synchronous invocation model may cause scalability challenges when long duration invocations are targeted. One way of overcoming this difficulty is adopting an *asynchronous* mode of operation. An asynchronous invocation allows the client to continue with its computation after dispatching a call, thus eliminating the need to wait idle while its request is being process by a remote server. In this study we define and implement an execution framework which extends Java RMI functionality with asynchrony.

From out of the box point of view, there is a java client program and a server program both written in java programming language and living in different virtual machines that can be in different physical locations and using Java's RMI (Remote Method Invocation) technology to communicate. Considering previous work carried on this topic to achieve asynchrony on Java's RMI special preprocessing of the client stub, modification of the remote objects or stubs or modification of the Sun's Java VM (virtual machine) is required which is not straight forward since Sun does not provide built-in mechanisms for those operations.

In this study, we implemented an asynchronous RMI execution framework on the top of RMI technology, using the thread pooling capability and the reflection mechanism of Java. It differs from previous work as it does not require any external tool, preprocessor, or compiler and it may be integrated with previously developed software, as no modification of target remote objects is necessary. User of the framework is provided a simple jar packet with couple of basic classes to make asynchronous calls to remote objects that (s)he develops or had been developed.

Brief code examples with real life problems will be given with the usage of the asynchronous RMI framework, obvious performance gain over standard RMI calls will be shown and the techniques which were used to optimize Java's multi threading and reflection in this study will be explained briefly with comparisons and results.