Oriental Journal of Computer Science & Technology Vol. **3**(1), 215-217 (2010)

Distributed remote architectures CORBA

HASRAT JAHAN[1] and MANMOHAN SINGH[2]

Department of Biotechnology, Bhopal Degree College Bhopal, (India).
Department of Computer Science, BIST, Bhopal (India).

(Received: April 12, 2010; Accepted: June 04, 2010)

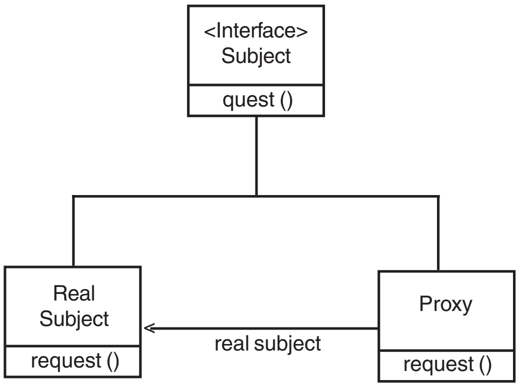
ABSTRACT

An emerging trend in the Signal and Image Processing (SIP) community is the appearance of middleware and middleware standards that can be readily exploited for distributed computing applications by the SIP community. High performance computing and High Performance Embedded Computing (HPEC) applications will benefit significantly from highly efficient & portable computational middleware for signal & image processing. Open middleware standards such as VSIPL, MPI, CORBA, encoding and SOAP –based messaging protocols. More specifically, we will be focused on the appropriate use of such technologies for implementing new SIP applications, or extending legacy applications through the use of these technologies. The three middleware standards we have selected all have certain commonalities. All are based around the concept of a client application using the services available on a remote machine, or server. A remote executable object that implements one or more exposed interfaces provides these services. The object's interface represents a contract between the client and the server. This interface is written as a Java interface for Java RMI, in IDL for CORBA, and in WSDL for web services. In the latter two cases, the more generic descriptions can be translated intospecific language imp lementations,

Keywords: Remote Architectures, CORBA.

INTRODUCTION

Transport Layer makes the connection between JVM's. All connections are stream-based network connections that use TCP/IP. Even if two JVM's are running on the same physical computer, they connect through their host computer's TCP/IP network protocol stack.



Naming Remote Objects Now you'll find the answer to that question. Clients find remote services by using a naming or directory service. This may seem like circular logic. How can a client locate a service by using a service? In fact, that is exactly the case.

Distributed Garbage

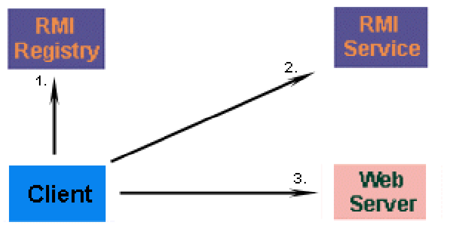
One of the joys of programming for the Java platform is not worrying about memory allocation. The JVM has an automatic garbage collector that will reclaim the memory from any object that has been discarded by the running program.

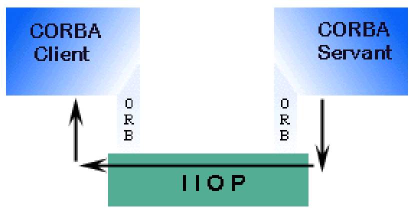
Serializing Remote Objects

When designing a system using RMI, there are times when you would like to have the flexibility to control where a remote object runs. Today, when a remote object is brought to life on a particular JVM, it will remain on that JVM. You cannot "send"

Mobile Agent Architectures

The solution to the mobile computing agent using RMI is, at best, a work-around. Other distributed architectures have been designed to address this issue and others. These are collectively called *mobile agent architectures*.





RMI pros and cons

Remote method invocation has significant features that CORBA doesn't possess - most notably the ability to send new objects (code and data) across a network, and for foreign virtual machines to seamlessly handle the new objects Remote method invocation has been available since JDK,

Remote method invocation

Pros	Cons
Portable across many platforms	Tied only to platforms with Java support
Can introduce new code to foreign JVMs	Security threats with remote code execution, and limitations on functionality enforced by security restrictions
Java developers may already have experience with RMI (available since	Learning curve for developers that have no RMI experience is comparable with CORBA

CORBA pros and cons

CORBA is gaining strong support from developers, because of its ease of use, functionality, and portability across language and platform. CORBA is particularly important in large organizations, where many systems must interact with each other,

Common Object Request Broker Architecture

Pros	**Cons**
Services can be written in many different languages, executed on many different platforms, and accessed by any language with an interface definition language (IDL) mapping.	Describing services require the use of an interface definition language (IDL) which must be learned. Implementing or using services require an IDL mapping to your required language - writing one for a language that isn't supported would take a large amount of work.
With IDL, the interface is clearly separated from implementation, and developers can create different implementations based on the same interface.	IDL to language mapping tools create code stubs based on the interface - some tools may not integrate new changes with existing code.
CORBA supports primitive data types, and a wide range of data structures, as parameters	CORBA does not support the transfer of objects, or code.
CORBA is ideally suited to use with legacy systems, and to ensure that applications written now will be accessible in the future.	The future is uncertain - if CORBA fails to achieve sufficient adoption by industry, then CORBA implementations become the legacy systems.

Creating 3-Tier Distributed Applications with RMI

The 2-tier model for applications is the most common model in use today. Many application designers think only in terms of the database and the application.

The availability of 2-tier application builders has helped perpetuate this philosophy. The 2-tier model is not a “bad thing,” but there are cases in which the 3-tier model would be a better choice.

CONCLUSIONS

CORBA doesn’t reveal an optimum solution - one is not “better” than the other. The properties of these two technologies lend themselves to different situations. A comparison of RMI and CORBA helps to highlight individual strengths and weaknesses, but the applicability of one technology over the other depends largely on the purposes for which it is to be used, the experience of the developers who will design, implement and maintain the distributed system, and whether non-Java systems are intended to access the system now or in the future.

REFERENCES

1. Elfwing, R., Paulsson, U., and Lundberg, L, Performance of SOAP in Web Service Environment Compared to CORBA, In Proceedings of the Ninth Asia- Pacific Software Engineering Conference, IEEE, (2002).
2. Davis, D., and Parashar, M., Latency Performance of SOAP Implementations, In Proceedings of 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid, IEEE, (2002).
3. Gorton, I., Liu, A., and Brebner, P., “Rigorous evaluation of COTS middleware technology”, Computer, IEEE, pp. 50-55 (2003).
6. Juric, M.B., Rozman, I., and Hericko,M.,Performance Comparison of CORBA and RMI, Information and Software Technology 42, pp 915-933, (2000).
7. Breese, J. S., Heckerman, D., Kadie, C. Empirical analysis of predictive algorithms for collaborative filtering. In Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI-98), COOPER, G. F. and MORAL, S., Eds. Morgan Kaufmann, Sa Francisco. 43-52 (1998).
8. Canny, J. Collaborative Filtering with Privacy via Factor Analysis. (2002).
9. Baeza-yates, R., Ribiero-neto, B. ACM Press / Addison Wesley, New York. Bailey, B. P., Gurak, L. J., Konstan, J. A. 2001. An Examination of Trust Production in Computer-Mediated Exchange. In Proceedingsof the 7th Conference on Human Factors and the Web, July 2001 (1999).
10. Balabanovíc, M., Shoham, Y. Fab: Content-Based, Collaborative Recommendation. Communications of the ACM 40, 66-72. BASU, C., HIRSH, H., COHEN, W. W. 1998. Recommendation as Classification: Using Social and Content-Based Information (1997).
11. Billsus, D., Pazzani, M. J. Learning collaborative information filters. In Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98), RICH, C. and MOSTOW, J., Eds. AAAI Press, Menlo Park, CA. 46-53 (1998).
12. Breese, J. S., Heckerman, D., Kadie, C. Empirical analysis of predictive algorithms for collaborative filtering (1998).
13. M. C. Hung, S. Q. Weng, J. Wu, and D. L. Yang, “Efficient Mining of Association Rules Using Merged Transactions.