The Design of a Real-time Java, Real-time CORBA ORB

Douglas C. Schmidt, Mayur Deshpande and Carlos O'Ryan

{schmidt,deshpanm,coryan}@uci.edu
Department of Electrical and Computer Engineering
University of California, Irvine [*]

Abstract

Implementing distributed, real-time applications is a daunting challenge, even for the most seasoned teams, implementing those systems on time, and under budget is even more challenging. This situation is worsened as the cost of finding and retaining experienced real-time developers keeps increasing.

Currently several technologies are available to mitigate those problems: CORBA, and more recently RT-CORBA, has proven to be applicable for many real-time distributed applications. Unfortunately, most QoS-enabled implementations of CORBA are only available in C++ or ADA, finding and retaining developers trained on these languages is becoming too costly for many projects. The Java programming language appears as an attractive alternative, this language is increasingly taught in universities and easily learned by C++ or Ada developers, more interestingly the developer base is reported to be over 100,000. Moreover, the adoption of the Real-time Specification for Java (RTSJ) has solved many of the problems that made Java unattractive for real-time applications. However, RTSJ does not include any facilities for implementing distributed applications, in this paper we present a design of Real-time Java, RT-CORBA ORB, we believe that this two technologies will represent a giant step forward in the effort to reduce the costs of developing and maintaining distributed applications with stringent QoS requirements.

This paper presents several contributions to the design of QoS-enabled middleware. First it presents an patterns-based architecture for a CORBA ORB, this architecture eliminates the most common sources of overhead and non-determinism in ORB implementations. Second, it illustrates how the ORB can be designed to support multiple transport protocols, enabling application developers to re-target the ORB for specialized environments and to take advantage of more efficient communication infrastructures. Third it shows how to use design patterns to automatically select the minimum set of components used by an application. Fourth it presents a small number of extensions to the RT-CORBA programming model, these extensions will allow application developers to more effectively control their CPU and memory resources.

[*] This work was funded in part by