CORBA-as-Needed: A Technique to Construct High Performance CORBA Applications

Hui Dai¹, Shivakant Mishra¹, and Matti A. Hiltunen²

¹ Department of Computer Science, University of Colorado
Campus Box 0430, Boulder, CO 80309-0430
² AT&T Labs-Research
180 Park Avenue, Florham Park, NJ 07932

Abstract. This paper proposes a new optimization technique called CORBA-as-needed to improve the performance of distributed CORBA applications. This technique is based on the observation that in many cases the client and the server of a distributed application run on compatible computing platforms, and do not need the interoperability functionality of CORBA. CORBA-as-needed dynamically determines if the interoperability functionality is needed for a specific application invocation, and bypasses this functionality if it is not needed. Performance measurements from a prototype implementation in omniORB show that CORBA-as-needed achieves a very significant performance improvement.

1 Introduction

CORBA [3] has been successfully used in the development of a large number of distributed applications. However, it suffers from an important deficiency. There is a significant performance overhead in using CORBA to construct distributed object applications [4, 5]. Therefore, it is difficult to construct high performance distributed object applications using CORBA. The main source of this performance overhead is the large one-way communication delay incurred by CORBA communication methods, compared to the one-way communication delays of transport-level protocols such as UDP or TCP. We propose a technique called CORBA-as-needed that allows distributed applications to bypass CORBA interoperability functionality whenever the client and the server happen to run on compatible platforms. It allows client server applications to first detect if the interoperability functionality of CORBA is indeed needed for a particular invocation, and bypass it if this functionality is not necessary.

In this paper, we present the design, implementation, and performance evaluation of CORBA-as-needed technique in omniORB [1]. We explore four different design alternatives for incorporating CORBA-as-needed in the current CORBA architecture. These design alternatives are called service approach, integration approach, CORBA wrapper approach, and pluggable ORB module approach. These alternatives differ from one another in the exact layer of CORBA architecture where CORBA-as-needed is incorporated. We provide a thorough analysis...
of the four design alternatives, including their pros and cons. We have implemented a prototype of CORBA-as-needed using the pluggable ORB module approach in omniORB[1]. We describe this implementation and a detailed performance evaluation by comparing the performance of several distributed object applications implemented using our prototype and omniORB[1]. This performance comparison shows that CORBA-as-needed is a very useful technique that can improve the latency and scalability of distributed object applications by as much as 25% when clients and servers run on compatible computing platforms. Also, performance measurements show that this technique has insignificant performance overhead when the clients and servers run on incompatible platforms. CORBA-as-needed is a generic technique that can be used in association with other optimization techniques that researchers have used in the past to improve the performance of CORBA applications.

2 CORBA As Needed

While the ability to operate in a heterogeneous distributed computing environment is an important requirement for modern distributed applications, it is important to note that a majority of the applications do in fact use operating systems, network protocols, and hardware platforms that are compatible with one another. The main idea of this paper is to design a system that dynamically recognizes the situations when the extra support for interoperability provided by CORBA is not needed, and allows applications to simply bypass the interoperability functionality by using a standard TCP/UDP communication mechanism for those situations. In other words, interoperability support of CORBA is used by applications only when it is really needed.

The CORBA-as-needed technique identifies operating conditions when a client and a server are running on compatible computing platforms at the time when the client initiates a connection to the server. If the client and the server are determined to be running on compatible computing platforms, the ORB redirects all future communication requests to the lower transportation layer without passing through the rest of ORB core that implements interoperability functionality. If the client and the server are determined to be running on incompatible platforms, all future communication requests are directed through ORB core in the usual manner.

2.1 Design Alternatives

There are at least four different design alternatives to incorporate CORBA-as-needed technique in the current CORBA architecture. These are (1) service approach, (2) integration approach, (3) CORBA wrapper approach, and (4) pluggable ORB module approach.

In the service approach (Figure 1), CORBA-as-needed is implemented as a new CORBA service called bypass service. The bypass service uses a similar