INTRODUCTION

Since its introduction in 1991, the Common Object Request Broker Architecture (CORBA) standard, defined by the Object Management Group (OMG, 2004b), has undergone several major revisions and has spread far throughout the domain of object-oriented and distributed systems. It not only brings about independence of computer architectures, operating systems, and programming languages, but also ensures freedom of choice with respect to Object Request Broker (ORB) product vendors. The latter benefit was the result of the introduction of a globally unique object reference, the Interoperable Object Reference (IOR), and a standard transmission protocol, the Internet Inter-ORB Protocol (IIOP), in CORBA 2.0.

CORBA uses an Interface Definition Language (IDL) to specify the interfaces that objects present to clients in order to offer their services. IDL is a purely declarative language—that is, it is used to describe the data types and interfaces in terms of their attributes, operations, and exceptions, but not to define implementation algorithms for their operations. IDL forms the foundation of CORBA’s programming language independence, and language-specific IDL compilers must be used to translate IDL interface definitions into concrete programming languages. Besides the language mappings defined in the OMG standard (i.e., mappings from IDL to Ada, C, C++, COBOL, Java, Lisp, PL/I, Python, and Smalltalk), there are also non-standard language mappings to programming languages like Eiffel, Objective C, and Perl, which are exclusively implemented in certain ORB products.

While CORBA has been very successful in the domain of enterprise computing, its adoption for mobile devices is obstructed by a central problem: the limited resources of such devices. If standard-compliant CORBA-based applications are to be executed on mobile devices, storage requirements, for example, represent a major bottleneck. But for all that, several research groups have made an effort over the past few years to establish the CORBA standard in the domain of mobile devices. The existing approaches can be divided into three categories:

1. approaches that are restricted in the sense that they use an implementation of the IIOP protocol only,
2. approaches that build on the minimum CORBA specification, and
3. approaches that rely on other ways to reduce the memory footprint of a CORBA implementation.

In the following sections of this chapter, these approaches will be discussed in detail.

THE PROTOCOL APPROACH

The first alternative to realize a CORBA infrastructure on mobile devices can be described as the “protocol approach.” Instead of providing an implementation of the complete CORBA specification, only the IIOP protocol is implemented in this solution.

As already mentioned, the CORBA 2.0 standard for the first time introduced the definition of a protocol for the communication between different ORBs. On an abstract level, the General Inter-ORB Protocol (GIOP) was defined, which specifies a standardized transmission syntax, the common data representation (CDR), and several message formats. Among the characteristics of CDR is a complete mapping of all data types defined in IDL and the support of different byte orders. With CORBA 2.1, the set of possible GIOP message formats was extended by “Fragment” messages. The bi-directional variant of GIOP permits both the client and the server to act as the initiator of a message for all possible kinds of messages. Since GIOP is an abstract protocol, the actual communication is routed via the Internet Inter-ORB Protocol, which provides a mapping between GIOP messages and the Transmission Control Protocol/Internet Protocol (TCP/IP) layer used in the Internet. Apart from IIOP, so-called Environment-specific Inter-ORB Protocols (ESIOPs) can be used. Currently,
CORBA on Mobile Devices

four different IIOP versions (1.0, 1.1, 1.2, and 1.3) can be encountered. To avoid interoperability problems that might occur when ORBs implementing different protocol versions need to communicate, the OMG specifies requirements as to which protocol has to be supported in which context (cf. OMG, 2004b, pp. 15-51).

In the context of mobile devices, a challenge lies in the realization of GIOP over wireless networks. Current CORBA implementations typically use IIOP to guarantee interoperability between CORBA-based applications. However, TCP/IP is not a suitable transport layer for wireless communication (Amir, Balakrishnan, Seshan, & Katz, 1995), so better alternatives like the Mobility Layer (Haahr, Cunningham, & Cahill, 1999, 2000) or WAP (Ruggaber, Schiller, & Seitz, 1999; Ruggaber & Seitz, 2000) were developed for that purpose. In order to provide a standardized solution for this aspect too, the OMG has adopted the wireless access and terminal mobility in CORBA specification (OMG, 2005). Furthermore, to account for the fact that the Bluetooth protocol has gained increasing popularity in the area of mobile devices, the OMG has issued the GIOP Tunneling Over Bluetooth specification (OMG, 2003).

The protocol approach was implemented in the context of several projects. For example, the work described by Haahr et al. (1999) is one of the first solutions belonging to that category. Moreover, BASE (Becker, Schiele, Gubbel, & Rothermel, 2003) and LegORB (Roman, Mickunas, Kon, & Campbell, 2000) are representatives of the protocol approach. Among the defining characteristics of BASE are, according to Becker et al. (2003), the uniform access to remote services and device-specific capabilities, the decoupling of the application communication model and the underlying interoperability protocols, and its dynamic extensibility supporting the whole range of devices from simple sensors to high-capacity workstations. There are two ways to generate invocations in BASE: they can be either generated by proxies representing services, or they are encoded analogous to a CORBA DII invocation by the application developers. The “micro-broker” coming with BASE only necessitates a plugin to transport (marshal and send) an operation invocation. The return values an invocation might possibly construct may be accepted by an additional transport plug-in.

LegORB implements a microkernel-type architecture. Its core only contains components for low-level services. Application developers have to implement specific policies or simply select them suitably, whenever they are packaged with the ORB. LegORB’s core consists of three customized components: the LegORB configurator, the client-side configurator, and the server-side configurator. They provide the glue necessary to put all the components together. LegORB itself is an assembly of components with different functional scopes with duties concerning network, marshaling, demarshaling, and so forth. The actual service capability as well as the size of LegORB is determined by the number and type of components that are composed in a concrete development project.

On the one hand, the protocol approach is sufficient to enable communication between different mobile as well as stationary CORBA-based applications, but on the other hand, it has considerable limitations. The first restriction pertains to a lack of source code portability. Since the presented solutions are specifically designed to address the communication aspect, this approach requires conceptual rethinking with respect to the way the mobile CORBA applications have to be developed. This not only holds for the migration of existing CORBA-based applications to mobile devices. Moreover, modifications to the “traditional” development process used for conventional CORBA-based applications are needed to meet the changed conditions in a mobile applications setting. The conventional development process usually starts with the specification of the IDL interfaces required in the application, if static operation invocations are intended, which is the normal case. These IDL definitions are then translated using an IDL compiler. Subsequently, the developers use different standardized CORBA classes and interfaces for ORB initialization or object activation. However, the protocol-based approaches often do not provide IDL support and do not necessarily allow for the use of the stipulated classes and interfaces for routine CORBA tasks, so that they often lead to considerable learning curves, even for experienced CORBA developers, in order to adapt to the changed programming conditions.

**APPROACHES BASED ON THE MINIMUMCORBA STANDARD**

The first dedicated OMG specification targeted on the reduction of the footprint of CORBA-based solutions was the minimumCORBA specification (OMG, 2002). In this specification, the OMG identified parts of the full CORBA standard that might be dispensable under certain circumstances like in the case of the limited resources available on mobile devices:

The features of CORBA omitted by this profile clearly have value in mainstream CORBA applications. However, they are provided at some cost, in terms of resources, and there is a significant class of applications for which that cost cannot be justified.

The omissions and reductions adopted by the OMG mainly concern the following points:

- **Omission of the Dynamic Invocation Interface:** The Dynamic Invocation Interface (DII) provides functionality that enables a client application to invoke operations of objects and to receive the returned results...
Related Content

Professional Development with and for Emerging Technologies: A Case Study with Asian Languages and Cultural Studies Teachers in Australia
Ria Hanewald (2013). *Pedagogical Applications and Social Effects of Mobile Technology Integration* (pp. 175-192).
[www.igi-global.com/chapter/professional-development-emerging-technologies/74911?camid=4v1](www.igi-global.com/chapter/professional-development-emerging-technologies/74911?camid=4v1)

Virtual EZ Grid: A Volunteer Computing Infrastructure for Scientific Medical Applications
[www.igi-global.com/article/virtual-grid-volunteer-computing-infrastructure/64366?camid=4v1](www.igi-global.com/article/virtual-grid-volunteer-computing-infrastructure/64366?camid=4v1)

Attacks and Authentication for Mobile Banking
[www.igi-global.com/chapter/attacks-and-authentication-for-mobile-banking/188297?camid=4v1](www.igi-global.com/chapter/attacks-and-authentication-for-mobile-banking/188297?camid=4v1)

Improving the Security of Storage Systems: Bahrain Case Study
[www.igi-global.com/article/improving-the-security-of-storage-systems/130482?camid=4v1](www.igi-global.com/article/improving-the-security-of-storage-systems/130482?camid=4v1)