

Using WAP for Wireless CORBA

Shahzad Malik (219762)
95.590X Course Report
Carleton University
April 11, 2001

smalik@chat.carleton.ca

ABSTRACT

As CORBA is currently the most popular middleware technology for distributed applications, it is desirable to offer it in the current mobile environment. However, the standard approach when deploying CORBA over a wired network is via TCP/IP, which experiences poor performance and reliability over current wireless links. This paper describes a proposal in the current literature which attempts to offer wireless CORBA via the Wireless Application Protocol (WAP), in order to provide a transparent CORBA implementation that addresses many of the shortcomings of TCP/IP-based approaches.

General Terms

Performance, Reliability, Experimentation.

Keywords

WAP, CORBA, wireless, middleware, client-server.

1. INTRODUCTION

With the explosive growth in the area of wireless communications technology and devices, as well as the increase in demand for distributed applications, it is only logical to combine these two technologies together. For distributed applications, Common Object Request Broker Architecture (CORBA) is currently the most popular middleware technology. However, blindly using CORBA in the current mobile and wireless environment will result in poor performance, largely due to the fact that CORBA relies on the TCP/IP communication protocol which is well-suited to the wired network environment. Thus issues such as sudden loss of connection, low bandwidth, and high error rates must be dealt with accordingly. This research paper will describe a proposed solution from the current literature which allows CORBA to be efficiently used over today's wireless links via the Wireless Application Protocol (WAP).

1.1 CORBA Overview

Proposed by the Object Management Group (OMG), CORBA is an open systems standard which enables

communication between distributed objects. The key component that allows this is the Object Request Broker (ORB), which provides transparent client/server relationships between objects. The power in the technology stems from the fact that a client object doesn't need to know where a server object is located, what platform it is running on, what communication protocol is needed to reach it, or what programming language it is implemented in [OMG95].

Before discussing CORBA's bottlenecks when it comes to wireless networks, it is worthwhile to describe the CORBA technology briefly. Figure 1 shows the basic outline of CORBA's client/server architecture.

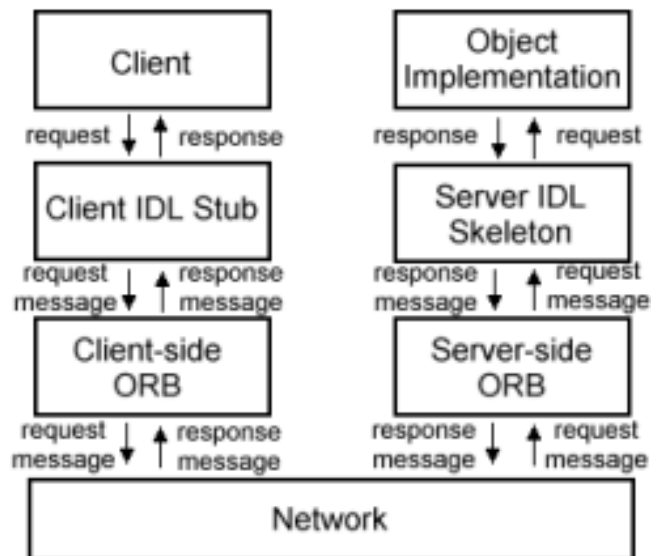


Figure 1 – CORBA architecture

An object is first written in C, C++, Ada 95, or Java. Then a server object's interface is written in an Interface Definition Language (IDL), which is independent of any particular programming language and is defined separately from the implementation. This IDL specification is then translated into the object's implementation language, using standard OMG IDL compilers. This generates client-side stubs

and server-side skeletons. The client-side stub creates a request and sends it on behalf of a client object, while a server-side skeleton receives a request and delivers it to the CORBA object implementation. The stub programs are responsible for packaging and unpacking high-level data types such that they can be sent and received over the network properly (eg. As byte streams). When a response is generated, the server forwards the result to the server-side skeleton, which in turn forwards it to the server-side ORB and over the network to the client side, where it is eventually unpacked and interpreted by the client.

Since the full details, feature set, and services of CORBA are beyond the scope of this paper, we will instead focus on CORBA's communication infrastructure, and how it relates to wireless and mobile environments.

1.2 CORBA Communication

The General Inter-ORB Protocol (GIOP) defines a standard for the exchange of data between ORBs, which assumes reliable, connection-oriented transport. The GIOP specification outlines the common data representations and the GIOP message formats [RUGG99]. The GIOP has thus been adapted for Internet use via the Internet Inter-ORB Protocol (IIOP), which defines a mapping from the GIOP to TCP/IP. The IIOP, while ideal for the wired network environment, doesn't lend itself well to the wireless environment.

As we already know, the wireless link exhibits high bit error rates and sudden connection losses. Additionally, today's more popular wireless link layers such as GSM provide high transmission delays coupled with relatively low bandwidths [RUGG00].

Now when we consider how TCP/IP handles errors, we can see why CORBA over IIOP is not well suited to the wireless environment. The TCP/IP sliding window protocol treats timed-out packets as congestion, which in turn causes the protocol to transfer fewer packets per unit time. However, packet loss in the wireless environment is usually more often due to the high bit error rate. Thus slowing down the rate at which TCP/IP transmits packets is not the proper solution. The desired approach would be to re-transmit the packet as quickly as possible to improve the chance that the packet successfully reaches its destination.

[RUGG00] also points out that in the wireless environment, a sudden connection loss from the perspective of a CORBA application will result in an exception being thrown to the ORB. This in turn will result in the application receiving an exception, which in most cases will cause the application to terminate. However, in the wireless environment, this situation can occur frequently, due to the properties of the wireless medium and also because mobile nodes expect to smoothly and transparently switch (handover) from one base station to another.

2. WIRELESS CORBA

In order to handle CORBA efficiently over a wireless link, many proposals have been made in the current literature. In this section, an approach based on the Wireless Application Protocol (WAP) is described. For all wireless CORBA approaches, [RUGG00] outlines the following requirements that should be met:

- Efficient operation over low-bandwidth wireless connections
- Transparent to CORBA applications
- Address the problem of sudden disconnections and handover

As we will see, the WAP-based approach to wireless CORBA fulfills these requirements.

2.1 The Wireless Application Protocol

Figure 2 shows the basic WAP protocol layers. The goal of WAP is to provide a protocol suite which enables efficient wireless communication across different wireless bearer services. WAP does not specify these bearer services, but instead uses already existing data services and plans to integrate future services as they become available.

Above the bearer service level, the following layers have been defined, each one specially-tailored for efficient operation over wireless links [WAP98]:

Wireless Datagram Protocol (WDP)

This provides a bearer-independent, consistent datagram-oriented service to higher layers of WAP. This layer is very similar to the UDP layer in the TCP/IP stack in terms of functionality.

asynchronous (ie. oneway) invocations, [RUGG99] suggests using class 0 or class 1 transactions based on whether reliability is desired.

In both cases, the CORBA-related request and reply messages are encapsulated in the data field of WTP messages. Thus WTP efficiently handles the issues associated with the segmentation and reassembly of large messages, much like TCP/IP handles it in the case of the IIOP.

2.2.2 Sudden Disconnection Handling

The use of the WSP almost automatically gives our wireless CORBA applications the ability to handle temporary disconnections, since it provides a *logical* connection between a client and a server. However, [RUGG00] notes that this requires a client ORB to implement *timeouts* in order to provide a consistent view of a connection (ie. deciding whether a disconnection is temporary or permanent). Thus the ORB timeout should be configured such that it is much longer than the WSP timeout, which in turn should be longer than the transport connection timeout. Therefore, if a temporary disruption occurs, the WSP session will maintain the connection information such that a CORBA application doesn't generate premature exceptions.

2.2.3 Handover Handling

The handover procedure for mobile CORBA nodes is relatively straightforward if we assume that all access nodes are configured with the WIOP gateway from Figure 3 [RUGG00]. For *proactive* handovers (where the mobile node decides to connect via a new access node), the mobile node can simply disconnect from the current access node and reconnect to the new one via the WSP facilities. Similarly, for *reactive* handovers (where the mobile node is disconnected automatically for some reason), the mobile node can simply reconnect to the same access node using the session information from WSP, or connect to a new access node via a new WSP session.

Thus the only major issue that hasn't been considered is the situation where an outstanding CORBA request is not completed before a disconnection occurs. While a handover mechanism similar to that of Mobile IP could be implemented, [RUGG00] suggests a more efficient approach. Figure 4 shows a request from a client to a CORBA server through WIOP gateway A. Before the reply is received, a disconnection occurs and the client reconnects to the network via WIOP gateway B. At this point, the associated response will

either already be stored at gateway A, or else it will be on its way to gateway A. In either case, the client knows it still hasn't received a response for its previous request, so it makes the same request to gateway B but exchanges the object reference to the CORBA server with a reference to gateway A. Additionally, it includes the WSP transaction identification information for the original request so that gateway A can identify the result. Thus gateway B knows to forward this request to gateway A instead of to the CORBA server. If the result has arrived at gateway A, the current request and the stored CORBA response are matched together and thus delivered to the client via gateway B. In the case where the result still hasn't arrived at gateway A, the reply back to the client is simply delayed until the reply from the CORBA server is received.

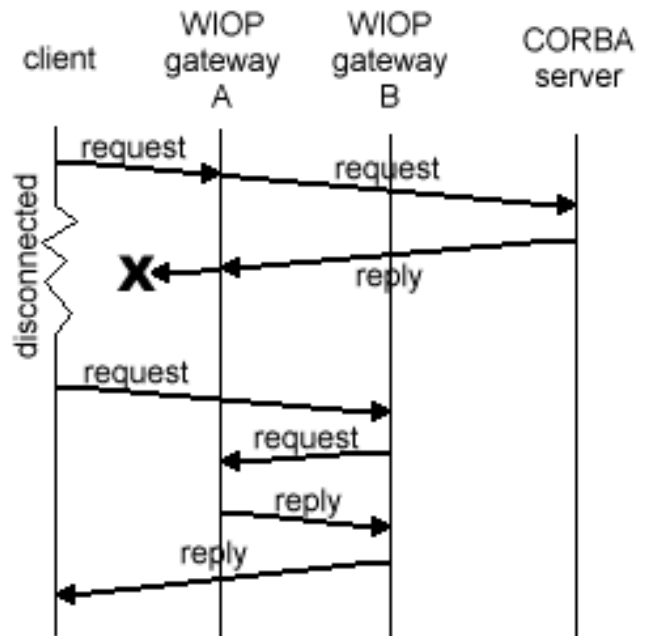


Figure 4 – WIOP Handover

While the original request could have easily been re-sent to the CORBA server, the above method has the advantage of eliminating repetitive work at the server end. In other words, the server only generates the response once, and the gateways simply store this information for ultimate delivery to the proper client. Additionally, no complex tunnelling or state information needs to be exchanged between the WIOP gateways during this handover procedure.

As can be seen, this WAP-based WIOP gateway fulfills the key requirements for an efficient wireless CORBA service. The issues of efficient operation

over low-bandwidth wireless networks, transparency from the perspective of CORBA servers and clients, and the proper handling of disruptions and handover have all been addressed.

3. DISCUSSION/CONCLUSION

Since CORBA is the most popular and widespread middleware for distributed applications, it's important to make it accessible over the wireless environment. Due to the rapid advances occurring in the field of wireless networks with respect to bandwidth and service quality, some people argue that, in the long run, it isn't necessary to develop special gateways or protocols to allow TCP/IP based applications to operate over wireless links. However, with all the advances occurring in the wireless world, similar advances are also occurring in the wired environment. As a result, the wireless environment will continue to be limited in its bandwidth when compared to the wired world, and thus special gateways or protocols to translate between the two environments are still needed. Therefore, the WAP-CORBA architecture outlined in this paper presents an interesting and efficient approach to adapting distributed CORBA applications seamlessly into the wireless world.

The architecture's strength lies in the fact that it uses WAP, which is considered the standard protocol that all wireless devices should support. Additionally, it addresses the key issues that CORBA developers would like to see with respect to deploying applications onto wireless devices, namely seamless integration of existing/legacy code, the ability to handle sudden disconnections and high bit error rates, and efficient use of the limited wireless bandwidth.

However, the architecture isn't without its weaknesses. Although CORBA provides location transparency for server objects via a naming service, the WIOP still needs to be aware of the server's actual bearer-specific address in order to properly transport requests and replies [RUGG99]. This requires special consideration at the WIOP level, since the various bearers that WAP supports can each have their own unique addressing schemes. Additionally, the current architecture does not make any caching, compression, or quality-of-service (QoS) considerations. Most importantly, the proposed architecture isn't a standard, which means that service providers would have to specifically support the WIOP

gateways on each of their access nodes. As other implementations of wireless CORBA services become popular, they too would have to be specially configured at the access nodes. Various other researchers have already proposed different solutions to the problem, each one with their own strengths and weaknesses. [RUGG00] notes that, while the OMG has recognized the importance of deploying CORBA on wireless networks, they still have not developed a concrete architecture. Clearly, a wireless CORBA environment without well-defined standards defeats the whole purpose of using CORBA in the first place.

Nevertheless, the proposed WIOP gateway using WAP as the underlying protocol is an efficient way to bring CORBA to the current mobile environment, since it addresses the key drawbacks associated with using the standard IIOIP over today's wireless links.

4. ACKNOWLEDGMENTS

I'd like to thank Prof. Michel Barbeau for his ideas and suggestions regarding this research paper, as well as for teaching a great course on wireless networks and protocols.

5. REFERENCES

- [OMG95] Object Management Group. *CORBA Architecture and Specification*. July 1995.
- [RUGG99] Ruggaber, R., Schiller, J., Seitz, J. "Using WAP as the Enabling Technology for CORBA in Mobile and Wireless Environments". Proceedings of the 7th IEEE Workshop on Future Trends of Distributed Computing Systems, 1999. Page(s): 69-74.
- [RUGG00] Ruggaber, R., Seitz, J. "Using CORBA Applications in Nomadic Environments". Proceedings of the 3rd IEEE Workshop on Mobile Computing Systems and Applications, 2000. Page(s): 161 -170.
- [WAP98] WAP Forum. *Wireless Application Protocol Architecture Specification*. April 1998.