Poseidon House
Castle Park
Cambridge CB3 0RD
United Kingdom

TELEPHONE: **Cambridge (01223) 515010**
INTERNATIONAL: **+44 1223 515010**
FAX: **+44 1223 359779**
E-MAIL: **apm@ansa.co.uk**

ANSA Phase III

Object Wrapping (for WWW) - The Key to Integrated Services? (Extended Abstract)

Nigel Edwards

Abstract

In the world of information services heterogeneity is inevitable: not only because of the huge amount of legacy data which exists on a wide variety of different platforms in different formats; but also because there is unlikely ever to be a format, protocol or platform which is suitable for all applications.

This paper explores how heterogeneity is managed within WWW (the World-Wide Web) and CORBA (Common Object Request Broker Architecture). It shows how the concept of object wrapping is useful for incorporating external information services and providing an integrated view of that information. In doing this work we are developing a programming model for WWW which is very close to that of CORBA.

IDLs (interface definition languages) are useful notations for describing existing (or legacy) information services. Encapsulation and integration of these services is made much simpler if the middleware (or ORB) makes it easy to plug new protocols into it and can support multiple, parallel protocol stacks: then the ORB can use the information service's own protocol. In addition such an ORB could make servers accessible via multiple protocols.

The paper begins by comparing the CORBA and WWW models, and goes on to show how object wrapping can be applied to both.

APM.1463.01 **Approved** 25th April 1995
External Paper

Distribution:
Supersedes:
Superseded by:

1 Object Wrapping (for WWW) - The key to Integrated Services?

In the world of information services heterogeneity is inevitable: not only because of the huge amount of legacy data which exists on a wide variety of different platforms in different formats; but also because there is unlikely ever to be a format, protocol or platform which is suitable for all applications.

This paper explores how heterogeneity is managed within WWW (the World-Wide Web) and CORBA (Common Object Request Broker Architecture) [OMG 93]. It shows how the concept of object wrapping is useful for incorporating external information services and providing an integrated view of that information. In doing this work we are developing a programming model for WWW which is very close to CORBA. We conclude with some recommendations about the kind of facilities middleware needs to incorporate to support easy integration of information services.

We begin by comparing the CORBA and WWW models, and go on to show how object wrapping can be applied to both.

1.1 Comparing CORBA & WWW

Both CORBA and WWW (using HTTP) are distributed object oriented systems: clients invoke methods (or operations) on server objects.

In CORBA the methods a service supports are defined in an Interface Definition Language (IDL). Objects communicate with each other using Object Request Brokers (ORBs) which provide a transport protocol for passing invocation requests and replies between objects. ORBs can support multiple transport protocols and abstract the programmer from many aspects of distribution (e.g. object location).

Many WWW services use the Hypertext Transfer Protocol (HTTP) [BERNERS-LEE 95]. This defines both a transport protocol and a set of standard methods; two of the standard methods are mandatory for all “general purpose” HTTP servers (GET and HEAD). Servers are also free to implement their own “extension” methods.

CORBA defines no standard methods which all application services are required to support. However, it does define an environment for building clients and servers which hides most of the complexity of the underlying platform from the programmer. From the IDL description of a service, client and server stubs are generated which hide the Application Programmer's Interface (API) of the underlying ORB from the application programmer. Hence remote invocation looks very similar to local invocation. This makes it relatively easy to write new clients and servers as well as making it easy to extend existing implementations —something which is important to assist integrating new information services into systems.

In contrast it is harder to write clients and servers which use HTTP, because of the need to drive the protocol stack directly — there is no equivalent to client or server stubs. Despite this a number of very sophisticated WWW clients and servers have been implemented, although I know of no server which implements any extension methods.

1.2 Integrating third party information services

The distributed object, or CORBA, approach to integrating third party services (or legacy) applications is to write an object wrapper which encapsulates that service. The system designer writes a description of the service using IDL. The implementor must then write code which invokes the appropriate set of actions in the legacy application when one of the object wrapper's methods is invoked. This has proved extremely effective in a number of projects (e.g. [DRAHOTA 94]).

Common Gateway Interface (CGI) [McCOOL] is a standard for external gateway programs to interface with HTTP servers. This allows WWW to encapsulate third party information services such as databases. CGI defines the format of the data stream between the WWW server and gateway, and also the environment variables available to the gateway.

Bespoke clients for these encapsulated information services are implemented by HTML (Hypertext Markup Language) "Forms" [CONNOLLY]. HTML Forms can be thought of as primitive agent technology: the client program (in the form of an active document or "fill-out" form) migrates to the WWW browser, asks the user for parameters and sends these back to the HTTP server.

1.3 Object wrapping for CGI: a stub compiler

There are few tools to assist CGI programmers: they must explicitly parse the incoming data stream to extract parameters. This is analogous to CORBA programmers having to write their own unmarshaling routines for servers.

To alleviate this, we have developed a stub compiler for CGI. From a CORBA IDL description of the CGI program it generates a server stub, a client stub and a template HTML form. Currently all CORBA IDL primitives and data types are supported, except the "ANY" data type.

Server stubs make the job of the CGI programmer easier by unmarshaling the incoming data stream.

Client stubs encapsulate libwww (the underlying library used by many WWW applications) in much the same way as stubs encapsulate the CORBA programmer from the API of the underlying ORB. Consequently, it becomes very easy to write bespoke client applications for CGI services: we have written two line C programs which are CGI clients.

Template HTML forms are supported for the simplest CORBA data types (e.g. strings and longs). Template forms generated by the stub compiler are guaranteed to produce a parameter stream consistent with the CGI server stub generated from the same IDL definition. This helps to ease the problem of keeping an HTML form consistent with its corresponding CGI program. Template forms need to be annotated with text to explain the input fields to humans.

The programming model provided by the stub compiler is very close to the CORBA programming model. The main difference is that CGI programs are stateless: they are forked for each invocation by the HTTP server. This means that any state change made by the client has to be stored externally (e.g. on the local file system).

Further details of the stub compiler are given in our programmer's guide [EDWARDS 95].

1.4 Object wrapping for HTTP

Currently we are exploring object wrapping for HTTP. Specifically we are writing a mapping of HTTP to CORBA IDL.

This will allow us to apply our stub compiler technology to build an HTTP server which supports extension methods and WWW clients which can use these extension methods. The aim is to provide a distributed object programming model for WWW similar to that of CORBA, making it easy to encapsulate legacy information services within WWW.

This IDL mapping will also make it very easy to plug an implementation of the HTTP protocol into an ORB, giving CORBA programmers the ability to encapsulate WWW inside CORBA. This kind of encapsulation requires an ORB which supports multiple, parallel protocol stacks. Given such an ORB it will be possible to write CORBA clients which access HTTP servers and also to write CORBA HTTP servers (supporting at least the GET and HEAD methods). Such servers might be accessible via multiple protocols, one of which will be HTTP.

In principle it is possible write an IDL description for other application protocols such as FTP. This is slightly less interesting, because unlike HTTP, most of these protocols are not designed to be extensible.

1.5 Conclusions

Object wrapping is a powerful technique for integrating information services. It also enables extensions to existing functionality. This promotes innovation and competition by allowing competitors to offer different functionality.

By using (CORBA) stub compiler technology we have been able to make object wrapping technology available to WWW CGI applications and, in so doing, have a CGI programming model which is close to that of CORBA.

The same idea can be applied to HTTP itself: if a reasonable IDL description of HTTP can be obtained, a stub compiler for HTTP can be built. This will make it very easy for HTTP server writers to provide extension methods. These extension methods could be used as an alternative to CGI to encapsulate other information services in WWW.

A similar effect would be achieved by plugging an implementation of HTTP into an ORB: this requires obtaining a reasonably tractable mapping of HTTP to CORBA IDL.

IDLs have proved useful notations for describing existing (or legacy) information services. Object wrapping relies on encapsulating these information services inside an object which is either resident in the same process or acting as a gateway. The gateway approach is made much simpler if

the middleware (or ORB) makes it easy to plug new protocols into it and can support multiple, parallel protocol stacks: the ORB can then use the information service's own protocol (e.g. HTTP).

References

[BERNERS-LEE 95]

T. Berners-Lee, R.T. Fielding, H. Frystyk Nielsen, “Hypertext Transfer Protocol - HTTP/1.0”, Internet-Draft (work in progress), March 1995, <URL:ftp://nic.nordu.net/internet-drafts/draft-ietf-http-v10-spec-00.ps>.

[CONNOLLY]

Daniel Connolly, “HyperText Markup Language (HTML)”, <URL:http://www.w3.org/hypertext/WWW/MarkUp/MarkUp.html>.

[DRAHOTA 94]

A. Drahota, D. Hutcheson, “DAIS and its use at Hydro-Electric”, ANSAworks 94, Cambridge, UK, April 1994.

[EDWARDS 95]

N.J. Edwards, “A Stub Compiler for CGI: The Programmer’s Guide”, Technical Report APM.1465, APM Ltd., Cambridge UK, April, 1995.

[McCOOL]

Rob McCool, “The Common Gateway Interface”, <URL:http://hoohoo.ncsa.uiuc.edu/cgi/overview.html>.

[OMG 93]

“The Common Object Request Broker: Architecture and Specification”, Revision 1.2, Draft, December 1993, OMG Document Number 93-12-43. (Available by anonymous ftp from ftp.omg.org, file: pub/docs/93-12-14.ps.z).