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

Distributed Objects and The World Wide Web

Nigel Edwards,
Owen Rees

Abstract

The paper explores how ANSA and CORBA technology can be applied to solve some of the management and infrastructure problems which are arising in the World Wide Web (WWW). These problems are becoming more and more severe as WWW grows in popularity and attempts are made to exploit it for commercial purposes.

Current technology has great difficulty coping with resources which can move around the network or resources which are updated: it is very easy to leave dangling links. Further it is desirable to have multiple instances of the same resource in different formats: for availability; for load balancing; to allow clients to select their preferred format (e.g. to take account of reduced bandwidth, distance etc.). Currently technology requires that each instance of a resource is referenced and managed separately: a client is linked to a particular instance of a resource at a particular location.

A concept called a Resource Reference (RR) is proposed. RRs are similar to object or interface references which are used in ANSA and CORBA [ANSA 93]. RRs can refer to many instances of a resource, allowing a client to select the most appropriate instance. In addition a set of management services are proposed based on those developed within ANSA and CORBA. At the time of writing it is not clear to us whether or not a resource reference is an instance of a concept currently being debated in the WWW community: Universal Resource Characteristic.

APM.1283.00.08 **Draft** 24th August 1994

Request for Comments (confidential to ANSA consortium for 2 years)

Distribution:
Supersedes:
Superseded by:

Distributed Objects and The World Wide Web

Distributed Objects and The World Wide Web

Nigel Edwards, Owen Rees

APM.1283.00.08

24th August 1994

The material in this Report has been developed as part of the ANSA Architecture for Open Distributed Systems. ANSA is a collaborative initiative, managed by Architecture Projects Management Limited on behalf of the companies sponsoring the ANSA Workprogramme.

The ANSA initiative is open to all companies and organisations. Further information on the ANSA Workprogramme, the material in this report, and on other reports can be obtained from the address below.

The authors acknowledge the help and assistance of their colleagues, in sponsoring companies and the ANSA team in Cambridge in the preparation of this report.

Architecture Projects Management Limited

Poseidon House
Castle Park
CAMBRIDGE
CB3 0RD
United Kingdom

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

1 Distributed Objects and the World Wide Web

1.1 Overview

The paper explores how ANSA and CORBA technology can be applied to solve some of the management and infrastructure problems which are arising in the World Wide Web (WWW). These problems are becoming more and more severe as WWW grows in popularity and attempts are made to exploit it for commercial purposes.

One of the least developed aspects of the WWW is management of resources in the network. Current technology has great difficulty coping with resources which can move around the network or resources which are updated: it is very easy to leave dangling links. Further it is desirable to have multiple instances of the same resource in different formats: for availability; for load balancing; to allow clients to select their preferred format (e.g. to take account of reduced bandwidth, distance etc.). Currently technology requires that each instance of a resource is referenced and managed separately: a client is linked to a particular instance of a resource at a particular location.

The proposed solution is to modify current WWW software to handle Resource References (RRs), as well as URLs (Universal Resource Locators) and URNs (Universal Resource Names). RRs are similar to object or interface references which are used in ANSA and CORBA [ANSA 93]. RRs can refer to many instances of a resource, allowing a client to select the most appropriate instance. In addition a set of management services are proposed based on those developed within ANSA and CORBA.

At present the WWW community is nearing consensus on the definition of a URL [BERNERS-LEE 94]. However there is still much debate about URNs and also URCs (Universal Resource Characteristics), as subscribers to the mailing list <uri@bunyip.com> will be aware. It may be that RRs are a form of URC, but at the time of writing we are not sure.

1.2 References to resources

In the CORBA world instead of handing out the absolute location of an object (like URLs) we use interface (or object) references. An interface reference is a data structure which contains a lot of useful information, including a list of endpoints (+ protocols) where the resource can be found and a list of management (e.g. relocation) services associated with the resource. The same idea could be used in WWW; to avoid confusion with CORBA object or interfaces references we will call the WWW data structures Resource References (RRs).

The list of endpoints could include multiple alternative instances of the resource, allowing the client to select the most appropriate one and use its favourite protocol to talk to it. Protocols could include http, ftp and CORBA-RPCs.

If a client is unable to find the resource on any of the endpoints it can contact a relocation service for the resource to find out the new location (see §1.3.2 *Relocators*).

The underlying issue here is really one of early versus late binding. At present in WWW if somebody hands you a URL it points to a particular instance of the resource at a particular location (ignoring the possibility of redirection). This is early binding. In contrast, a reference to a resource means that selection of the particular instance and location need not take place until the reference is invoked (hyperlink followed). This is late binding.

We expect the use of RRs to be transparent to users of the Web: they will still see URLs (or eventually URNs) in documents. Servers will hand clients (browsers) URL to RR mappings as part of the document headers. Browsers will store this information; when a URL is selected a browser will lookup and use the corresponding RR to talk to the resource.

Clients which cannot cope with this will just ignore the extra header information and use the URL in the way they do now.

Protecting users from the internals of RRs has ramifications for authoring tools (see §1.3.3 *Authoring and Resource Management*).

The current status of this work is that we think we understand where to slice apart both NCSA Mosaic and CERN httpd to add this functionality and are just starting to make the modifications. We have not yet decided the precise structure of an RR, but the starting point will be in the structure of the interface reference described in [ANSA 93].

1.3 Management and Infrastructure Services

Currently management in the Web (installation and maintenance of services etc.) is ad hoc. In the ANSA project we have built a number of management services running on various CORBA like platforms which we believe would be useful in the Web.

1.3.1 Traders

A trader provides a matchmaker service allowing clients to locate servers at runtime which meet stated constraints on interface type and certain other types of properties. The trading service can be distributed: several traders may collaborate to satisfy a request — it is not the case that all information has to be put in one well known place.

Our current prototype runs on Orbix and uses HP's Allbase to store information about services [THOMAS 94][1]. It tries to provide the information needed to allow matching to take place at earlier phases in the application lifecycle and to allow matching on complex constraints involving properties such as charging and billing policies, security and privacy policies, operation semantics etc. CORBA browsers could drive this directly (although we have not built a CORBA browser — at present this is problematic see §1.4.1 *Constraints on CORBA/WWW Interworking*).

At present the prototype trader is only available within Orbix. We believe a non-CORBA browsers could drive it by using some kind of scripting/forms

1. Earlier versions run on other platforms e.g. ANSAware [ANSA 92] and INA.

technology (see §1.4 *CORBA-WWW Interworking*). (Using Tcl-dp and scripting/ forms we have been experimenting using Mosaic to invoke distributed objects).

1.3.2 Relocators

A relocator is a service which will refresh stale RRs. An RR is stale if a client cannot find the resource at any of the end points listed in the RR. In this case the client can hand the old, stale RR to the relocator; the latter will attempt to return a fresh RR, so that the client can access the resource successfully.

We have built relocation services which client and objects can use to allow resources to relocate and migrate around the network [ANSA 92]. We would like to port this or build something similar for WWW. We also envisage building a migration tool which would migrate resources around the Web, updating relocators so that the WWW resources could always be found.

1.3.3 Authoring and Resource Management

The next step would be track resources as they move to maintain the integrity of links. If a resource instance moves, the links to it will be stale; the first time a link is followed the relocation service would have to be invoked to find the new location of the resource instance. It would be better to allow brokering services such as traders to track resources and refresh RRs which had become stale because the resource had migrated; this would reduce the likelihood of clients getting stale references or links from these services. Some ideas which are applicable here are explored in [EDWARDS 94], although they would need adapting to the specific requirements of WWW.

Currently there is no notion of “installing a document into the network”. Authoring tools will be needed to install resources. This could mean generating multiple instances of the resource and making the management infrastructure aware of these instances. Hence authoring tools start to look more like object (or document) factories which invoke management interfaces on servers to install resources.

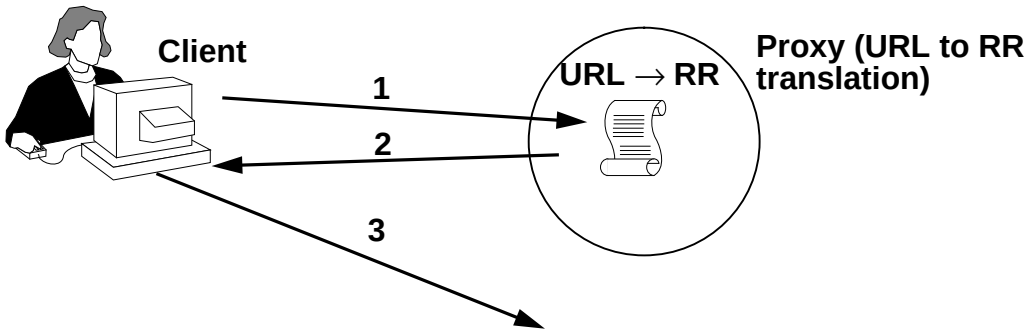
Authors will only use URN's and URLs in their documents. The authoring tool would map these to RRs when it installs the document. This means generating a RR for the new document (perhaps registering it with the trader). The tools would also have to perform URL/URN to RR translation for all links (or anchors) in the document so that a server returning the document can include this information in the headers. (Recall that documents will only contain URLs and URNs, the headers returned by the server provide the information needed to map these to the corresponding RRs.)

Authoring tools will need to maintaining the consistency of multiple instances of resources. For example, if a new version of a resource is published all instances of the old version need to be updated. In addition links need to point to the new instances, rather than old instances of the resource.

We have implemented support for replicated distributed objects [OSKIEWICZ 92]. This involved adapting the interface reference structure to allow clients to treat the members of a group as a single entity. The same ideas could be used to manage groups of Web resources as a single entity. Although there would be an overhead associated with such many to one interactions, it may well be a price worth paying for relatively infrequent management operations which update parts of the Web. Ordinary users would not use management operations and so would not incur this overhead.

We also intend to build a proxy server to allow current (legacy) WWW applications to take advantage of the indirection offered by RRs. Servers would have the option of handing out URLs to a proxy which would maintain a URL to RR mapping. When a browser tried to retrieve a document corresponding to such a URL, the proxy would lookup and examine the appropriate RR. Then the proxy would use the WWW redirection mechanism to direct the browser to the most convenient instance of the resource (see figure 1.1).

Figure 1.1: Proxy URL-RR-URL Conversion



1. Client attempts to retrieve resource from proxy using URL
2. Proxy finds the corresponding RR and selects an appropriate URL, returning it as a redirection to client
3. Client retrieves resource using new URL

1.4 CORBA-WWW Interworking

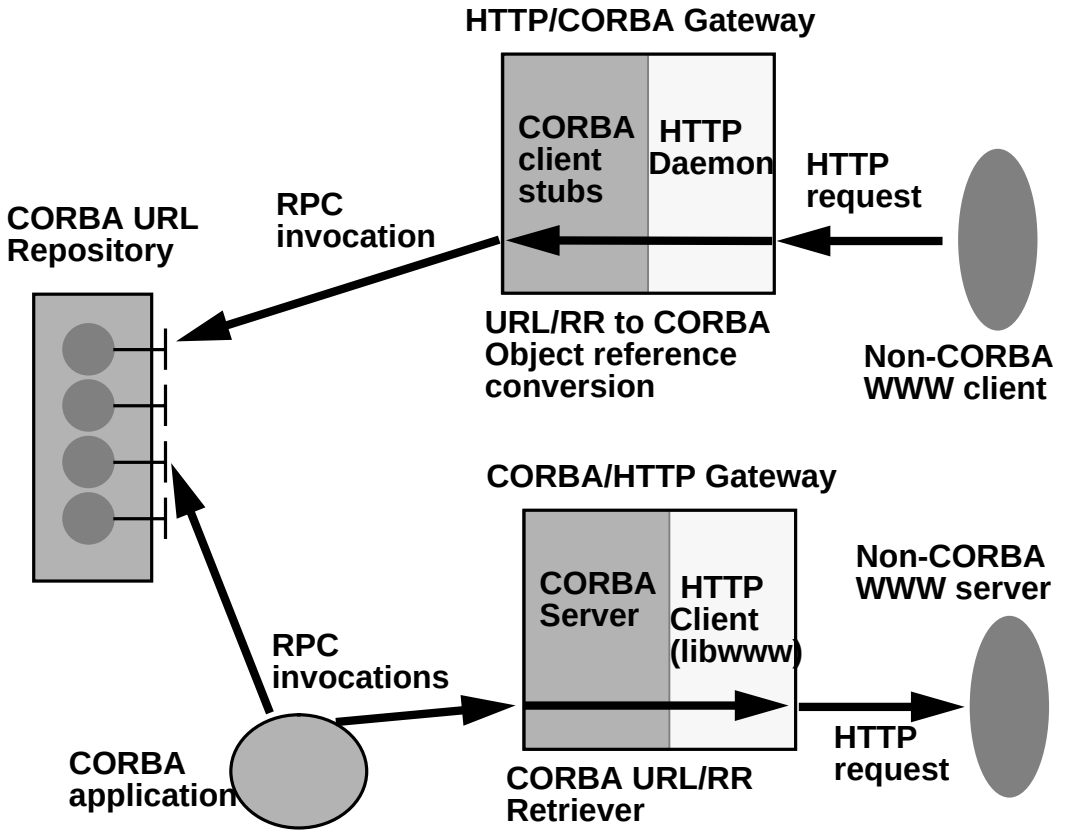
This is about achieving some degree of interworking between CORBA and WWW applications. It is not about protocol tunnelling (e.g. using CORBA-RPC as a transport for WWW). Others, such as OSF [OSF], are looking at protocol tunnelling issues (e.g. using DCE-RPC as a transport for HTTP). The reason for wanted to achieve some degree of interworking is to allow non-CORBA clients and servers to take advantages of CORBA management services (e.g. trading and relocation) and possibly other CORBA services. CORBA clients and servers will benefit by being able to access information available on WWW servers.

A document could be published in the CORBA world by creating a new CORBA object or by passing the text to a CORBA URL repository. (There are various pros and cons to the two options beyond the scope of this note.)

A non-CORBA browser or client would access a CORBA object by parsing a URL which would route it via an HTTP/CORBA gateway (see figure 1.2). This would mean building an HTTP server which maintains a list of (RR or) URL to CORBA object reference translations. Once it had started the server would listen on its socket (select()). When an HTTP request arrives it would do the address translation and invoke the appropriate CORBA object reference, forwarding the results back to the browser. Co-existence of CORBA with HTTP should not be a problem, since the gateway would only ever be an HTTP server and an CORBA client, so the CORBA part of the server should never be trying to listen for an invocation to arrive. The CORBA objects would support an CORBA-IDL version of HTTP.

Allowing CORBA applications to talk to WWW services involves building a CORBA/HTTP proxy (see figure 1.2). The CORBA side would have an HTTP IDL. When it received an invocation it would use libwww (the WWW library available from CERN [CERN 94]) to talk to the Web. For example, invocation of "Get(URL)" would cause the proxy to invoke "HTLoadAbsolute(URL)" in libwww. Some work needs to be done to figure out how to make WWW URLs (or hopefully RRs) look like object references to the CORBA client. (One possible approach would be to use a preprocessor — we have used this in the past successfully to hide replication from programmers.)

Figure 1.2: CORBA/WWW Interworking



An alternative to using a CORBA/HTTP gateway would be to port a set of CORBA HTTP stubs (generated from a CORBA-IDL description of HTTP). The port would be to make the stubs drive HTTP directly (e.g. by interfacing them with libwww). This would mean that CORBA clients would talk directly to Web objects without going through a gateway. It may also be possible using this approach to give the application programmer the illusion that Web objects are CORBA objects (i.e. transparency). An obvious next step would be to take advantage of all the other protocols supported in libwww and do similar things for them (e.g. ftp).

1.4.1 Constraints on CORBA/WWW Interworking

The current CORBA specification says nothing about threads [OMG 91]. Not having a lightweight threads package does pose some constraints on the level of interoperability which might be achieved.

Consider how the HTTP/CORBA gateway in figure 1.2 should be managed (e.g. updating URL/RR to Object Reference mapping). Suppose it is managed by a CORBA interface, so it is both a CORBA as well as an HTTP server. Unfortunately, it is not clear from the CORBA specification [OMG 91] how the two communications stacks should interact: more than likely it would end up waiting either for a CORBA request or an HTTP request, but not both. Getting it to wait for both involves designing an ORB (Object Request Broker) which allows a server to use more than one protocol (for example one thread could handle each protocol). The easier, but less aesthetic approach would be to do the management via the HTTP socket.

Building a CORBA browser is problematic, because this requires a thread safe ORB to interface with X (we would like to slice in an ORB as another protocol into Mosaic). From our own experience of ANSAware (a non-CORBA compliant ORB) we know that it is possible (but non-trivial) to interface an ORB with X.

1.5 Naming

We have been looking at naming issues in the Web and studying the URL/ URN/URC debate (see <uri@bunyip.com> mailing list).

We have been studying the naming of resources on the WWW side and the CORBA side and to what extent we can work around the differences (translation etc.) [REES 94]. We are doing some experiments to try to understand what is and what is not feasible. For example, to what extent can you go beyond treating CORBA as a smart file system.

1.6 Prototyping environment

We have been making use of Tcl/Tk (+ distributed object oriented extensions), and Orbix. In addition we have an in house (CORBA-like) platform called ANSAware.

1.7 Acknowledgement

The authors are grateful to Andrew Herbert and Paul Vickers for their helpful comments on this paper.

References[1]

[ANSA 93]

"ORB Interoperability", TR.43, APM Ltd., Cambridge U.K., May 1993 (also available as OMG Document number: OMG 93-5-9).

[ANSA 92]

"ANSAware 4.0 System Programmer's Manual", APM Ltd., Cambridge U.K., March 1992.

[BERNERS-LEE 94]

Uniform Resource Locators, Berners-Lee, T., Masinter, L., McCahill, M., Internet Draft, August 13 1994 <URL:ftp://cnri.reston.va.us/internet-drafts/draft-ietf-uri-url-06.txt>.

[CERN 94]

<URL:http://info.cern.ch/hypertext/WWW/Library/Status.html>

[EDWARDS 94]

"MED: A CORBA-based Management Engine for Dependability", Edwards, N., Oskiewicz, E., APM.1203, APM Ltd., Cambridge, U.K., 1994. (Abstract also available as: <URL:http://www.ansa.co.uk/phase3-doc-root/sponsors/APM.1203.00.03.html>

[OMG 91]

"The Common Object Request Broker: Architecture and Specification", OMG Document Number 91.8.1, August 1991.

[OSF]

<URL:http://riwww.osf.org:8001/ri/QuadCharts/DCEQuad.html>

[OSKIEWICZ 92]

"A Model for Interface Groups", Oskiewicz, E., Edwards, N., APM.1002.01, APM Ltd., Cambridge U.K., 1992. (Abstract also available as: <URL:http://www.ansa.co.uk/phase3-doc-root/sponsors/APM.1002.01.html>)

[REES 94]

"The Name of the Resource", Rees, O., APM.1282, APM Ltd., Cambridge U.K., 1994. (Abstract also available as: <URL:http://www.ansa.co.uk/phase3-doc-root/sponsors/APM.1282.00.01.html>)

[THOMAS 94]

"Data Management for an Enhanced Trader", Thomas, G., Beasley, M., Hoffner, Y., APM.1162.01, APM Ltd. Cambridge U.K., 1994. (Abstract also

1. ANSA/APM document abstracts are available on the World Wide Web. In case of difficulty access the index: <URL:http://www.ansa.co.uk/phase3-doc-root/sponsors/index.html>. It may be that the document has been superseded (e.g APM.1203.00.03.html replaced by APM.1203.00.04.html)— this will only show in the index.

available as: <URL:http://www.ansa.co.uk/phase3-doc-root/sponsors/APM.1162.01.html>)