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

Requirements for Re-Engineering of the Web

ANSA ISF Group

Abstract

The objective of this document is to capture the requirements for the web re-engineering work discussed at the May ANSA WWW Workshop and Technical Committee meeting.

For the present, the precise details are confidential to the consortium.

APM.1505.01	**Approved**	12th June 1995
	Technical Report	

Distribution:
Supersedes:
Superseded by:

Requirements for Re-Engineering of the Web

Requirements for Re-Engineering of the Web

ANSA ISF Group

APM.1505.01

12th June 1995

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

Contents

1 Requirements for Re-Engineering of the Web

The objective of this document is to capture the requirements for the web re-engineering work discussed at the May ANSA WWW Workshop and Technical Committee meeting.

The objectives are divided into immediate and longer term. Immediate objectives are things which need to be satisfied by the alpha-release of code in the last quarter of 1995. Longer term objectives are things we would like to achieve in 1996 and on. Further medium to longer term planning is required.

The key, and therefore immediate, priorities of the present work are to put in place the IIOP proxies, library layers and modules, and to represent the necessary interfaces in CORBA-IDL. The other elements identified in the requirements capture exercise, of which this white paper is the result, will be incorporated in the medium term and longer-term working plans.

1.1 Why re-engineer the Web?

There are a number of problems with the existing technology.

- The performance of HTTP is a serious problem:
  - TCP connections are opened and closed too frequently;
  - ASCII text needs to be parsed.
- URLs are too low a level:
  - an additional level of indirection (analogous to interface references) would allow the infrastructure to locate the most appropriate instance of a resource.
- Extensibility using CGI is difficult:
  - demand for sophisticated services is hard to satisfy.
- Resource discovery is not directly supported by existing web technology:
  - metadata formats and management methodology are needed;
  - client-side support for query construction is needed.
- The existing infrastructure does not support mobility of either clients or servers.

1.2 Constraints on the solution

This section lists some of the constraints on the solution.

- Existing functionality must be preserved, specifically:
  - immediate rendering of text and images, and, also, “Real Audio™”;
  - concurrent retrieval/rendering of text and inline images;

  - — a stop/abort button to stop the retrieval of a resource.
- Evolution not revolution:
  - — the solution must interwork smoothly with existing WWW;
  - — existing resources must be usable via the new technology;
  - — it must be easy to replace old tools with new (or upgrade).
- Availability and Deployment:
  - — the solution technology must be very portable;
  - — a “good enough to use” example must be available free;
- Progress towards standards:
  - — respect IETF standards development where possible;
  - — use prototypes to guide new standards where necessary.

Dependability, transactions, migration and security also need to be taken into account. We need the explanation of how they will fit, but not code in the first version.

1.3 Immediate Objectives

This section lists objectives which need to be satisfied by a Beta release of code in the last quarter of 1995. An Alpha release is planned for September which will satisfy some of these objectives.

1.3.1 Performance problems

The existing web suffers from a number of serious performance problems. Many of these are avoidable using existing well understood technologies. In some cases, it may be possible to improve web performance by up to an order of magnitude without major modifications to existing servers or clients. The problems are taken in turn below.

1.3.1.1 Connection management

We want to avoid opening a new TCP connection for each HTTP request, as this introduces extra latency. This should come for free by using IIOP.

There is, however, strong evidence that multiplexing everything over one TCP connection is not always the right thing to do. For large data transfers (in excess of 1 MByte) performance degrades. This implies we need to be able to control how much multiplexing goes on. The initial objective is to support the control of multiplexing; controlling the degree of multiplexing is a longer term objective (see especially §1.4.1). It is expected that achievement of this objective will be aided by the results of the DIMMA Engineering work. In addition, ANSAware R/T contains facilities which can be used to resolve some of the engineering problems mentioned here.

1.3.1.2 More efficient encoding of HTTP

Recent work at ANSA [EDWARDS 95] has shown that one of the major causes of latency in HTTP is the time taken to parse ASCII HTTP headers (43 milliseconds minimum for an HTTP server to parse an HTTP request; usually it will be much worse). This implies we need a better encoding of HTTP request headers. One way of doing this would be to map HTTP to CORBA IDL, and then use CORBA marshalling/unmarshalling technology to decode the headers.

The important issue is for the "usual" media types to be taken into account in the IDL and represented more compactly than as strings.

1.3.2 Extensibility

WWW is often used to create a "uniform information space" by interworking with (and providing a gateway to) legacy sources of information. Service providers often have specific requirements which cannot be satisfied by a standard WWW server.

The target of this work is the provision of ODP for use within the WWW. If we have an IDL encoding of HTTP and a stub compiler which generates the appropriate calls to the IIOP protocol stack, then it should be easy to define new methods as an alternative way of extending HTTP server functionality (c.f. CGI).

This is what differentiates the present proposed infrastructure from alternative new protocols and attempts to re-engineer the web. The HTTPng project, for example, is addressing the performance problems encountered in using raw HTTP: as such, it will fit within our proposed framework by using and appropriate IDL description of it to allow it to be treated as an interface to a faster transport protocol with similar invocation methods to HTTP. It also forces people to think of HTTP as being a service type in exactly the same way as SimpleBank is a service type. The strategy described here is clearly seen to be leading the way towards making the WWW into an ODP environment.

1.3.3 Preserving existing functionality

No existing capabilities should be lost as a consequence of implementing the new protocols.

1.3.3.1 Immediate Rendering and Stream objects

Browsers such as Netscape Navigator render text and images as they arrive. This is a very popular feature: it helps the user make up their mind faster about where their interest lies within the document. This is important for clickable images and, in the future, for user-influenced downloading.

This is not something which fits into the RPC model. Various ideas were discussed at the ANSA project meeting; the one which seems to offer the most promise is for one of the result parameters to be a reference to a pseudo-object. The programmer would make down calls to retrieve the data from this pseudo-object as it arrived. The stream data would be the last result to come down the wire. One problem this presents is that we need threads and fragmentation to do rendering of multiple objects concurrently: see §1.3.3.2.

1.3.3.2 Concurrent retrieval and rendering

This shows a need for integration of threads and communications: for example, Netscape Navigator not only renders data (text and images) as it arrives, but also retrieves it concurrently so that it can be rendered concurrently. In the new model being proposed here, this capability may require thread support, but should be provided wherever possible. Taking full advantage of these capabilities will require session control as well. IIOP does not directly support session control, but can be enhanced to do so with relatively little effort.

This has an impact on the underlying protocol. Being able to render multiple objects simultaneously implies being able to send multiple replies concurrently using IIOP. If the 'TCP stream per request' problem is to be avoided, this implies adding fragmentation to IIOP, and will also need to make use of available session control mechanisms (see above). Currently it can only send one response at a time by sending a reply message which takes a completely marshalled data buffer. Thus a client and server have no way of interleaving the data streams.

There is a proposal to enhance IIOP to support transfers of objects whose size is not known in advance. This provides a possible way to interleave response fragments that needs to be investigated.

1.3.3.3 Controllable Kill Semantics

Users are used to having a stop button on their browsers, which aborts the incoming data streams. We cannot take this away from them. So we need to be able to make down calls on stream objects to tell them to destroy themselves and close the connection gracefully (tell the server to stop sending fragments and maybe close the TCP connection).

1.3.4 Evolutionary approach

1.3.4.1 Interoperability

Any new servers and clients need to be able to interoperate with existing web browsers and servers. This means that they will have to have a way of speaking HTTP (perhaps by using proxies as shown in [REES 95]).

1.3.4.2 Legacy of HTML documents and URLs

All existing HTML documents have URLs using the http:// scheme embedded in them. We need to be able to serve these documents using an IIOP based protocol. Specifically we do not want to force people to change to a new scheme. It is important to be able to route an http:// scheme request via IIOP where it is available. It should also be possible to serve the same document hierarchy using either an IIOP based server or an HTTP based server.

Different ways of doing this were explored at a recent ANSA project meeting. One possibility is to use HTTP as the binding protocol. Initially a client will use HTTP to talk to the server, but will include in its accept headers something like “x-protocol IIOP”. If the server can serve documents using IIOP it will include in its reply headers “x-protocol IIOP”. All subsequent communication between the client and server can then take place using IIOP: the client or proxy will remember which servers can talk IIOP.

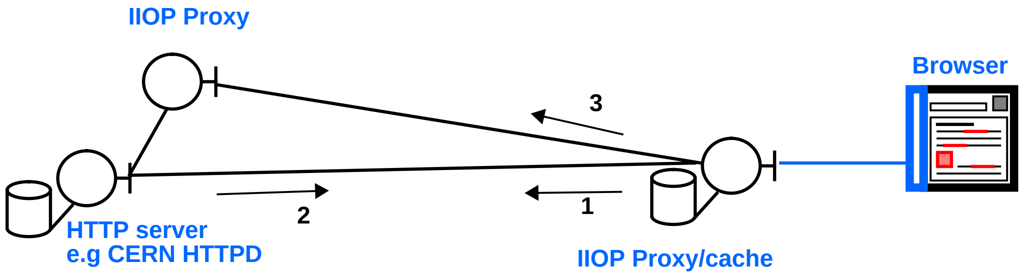
The headers need to include enough for the recipient to be able to invoke the service — this suggests an Interface Reference.

Note: This scheme would work for alternative protocols like HTTPng and DCE.

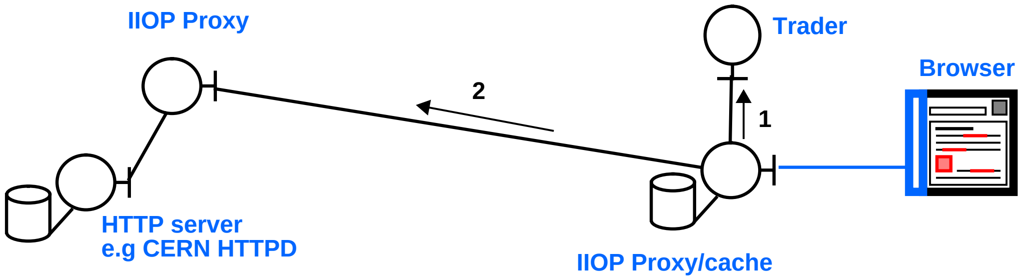
Note: We should look at Dave Kristol's HTTP extension headers as a way of doing this, rather than re-inventing our own headers [KRISTOL 95].

The problem with the extended header approach is that the HTTP server (e.g. CERN HTTPD) needs to be modified to send it. This may not always be acceptable, so other strategies need to be explored, such as trading and routing. Figure 1.1 shows two possible strategies.

Figure 1.1: Two strategies for serving legacy HTML documents



1. IIOP proxy invokes HTTP server using HTTP
2. HTTP server responds with resource and “x-protocol IIOP” header
3. All subsequent resources retrieved using IIOP



1. IIOP proxy queries trader to find IIOP proxy serving resource
2. All resources retrieved using IIOP proxy

1.3.4.3 *Upgrade path for existing browsers and servers*

Proxies need to be provided which can be run on the same machine as existing web browsers and servers (or on the same Ethernet). Proxies will communicate using IIOP, while the existing browsers and servers will use HTTP to communicate with their local proxies. This means that IIOP will be used for the wide area communication (giving benefits of connection management etc.), while HTTP is used for local communication (see [REES 95]).

In parallel, we will build an IIOP class for Java/HotJava. (Java may be an important Web technology which we should become knowledgable about. Netscape have announced plans to incorporate Java into their browser, see http://www.netscape.com/newsref/pr/newsrelease25.html.)

Advantages of the Java route (with respect to the client-side) are:

- It seems like it might be easier to integrate an IIOP protocol module into the Java/HotJava environment (than into libwww);
- If the Java documentation is to believed then HotJava users will be able to download and integrate the IIOP module automatically;
- It cuts away the proxy delay time browser(HTTP)<-->proxy(IIOP) at the client side.

One big advantage of the Java approach is that, once the architecture for incorporating IIOP at the client end is done well, the succeeding new protocols (such as could be required for VRML support, for example) can be automatically and transparently inserted on the client side using the same technology.

1.3.5 Availability and Deployment

1.3.5.1 Portability

Any software we write (proxies, servers and browsers) needs to be portable and easy to install, otherwise people are unlikely to use it.

1.3.5.2 Giving it away

It is planned that the software will be available to the rest of the world by anonymous ftp from the Beta-release stage. Prior to that, testing of interoperation will be accomplished with the aid of other sites belonging to members of the ANSA programme.

1.4 Longer Term Objectives

Script technology, and in particular Java, should provide a good way to explore many of these ideas, so we need to ensure that we can use Java in conjunction with any re-engineered technology.

1.4.1 Connection management

Having agreed that the infrastructure must support multiplexing, we need to provide the applications and tools with capabilities for control of multiplexing. The problem is how to decide when to multiplex and when not to multiplex. One rule of thumb is to multiplex whenever possible, and revert to point-to-point when transfers will exceed a given threshold (current experience suggests that a good value for this threshold lies at about 1Mb). This will work provided the amount of data to be transferred is known in advance.

1.4.2 Incorporate Metadata — URLs are too low a level

The ideal from an information system perspective is to be able to treat the web as a library of knowledge: ask for material by name and have it supplied transparently. If you don't know what it is called, describe your requirements and have your system try to match them and then return the results. This requires a scheme incorporating metadata, along with some form of trading or brokering service that can be added in a scalable fashion. This could be based on metadata stored in the form of URCs (Uniform Resource Characteristics) together with ANSA-type trading technology.

One proposed architecture for the metadata service consists of an encoding of metadata within URCs and URCs wrapped for presentation into HTML documents; the URCs repository is the ANSA MatchMaker, the query engine is a Safe-Tcl module built into the Changeling extensible webserver [MCCLENAGHAN 95], and queries are passed through a bespoke Tcl-to-MatchMaker gateway. See [MADSEN 95a] for details.

1.4.3 Different encoding schemes

It is inevitable that people will want to experiment with different encoding schemes and new MIME types. We must support this.

The trend may be towards active objects rather than dumb documents, and these may require the ability to interact with other services. For example, some kinds of documents/graphical objects may need to be delivered as a series of RPC calls rather than as a stream. The RPC series has an advantage in that it is steerable — as the user moves their mouse pointer (avatar!) towards an object (e.g. along a virtual reality corridor) the browser sends off a series of RPC calls that fetch objects/details as they come into view.

1.4.4 Object Oriented Browsers

The client side will be re-designed and re-implemented, so that it consists of multiple collaborating objects, talking to each other through CORBA IDL defined interfaces using the ORB's favourite protocol to communicate with each other. For example, we may have separate HTTP, FTP, Gopher protocol clients which are ORB objects. The benefit of this is that it allows incremental evolution and promotes backwards compatibility, by making interfaces explicit.

Another important benefit is that browsers can be more lightweight since they can link to the various protocol engines only when they need them.

It is particularly important to take advantage of the fact that existing browsers are already having some of their functionality stripped in newer releases to avoid duplication of other system functions.

It is envisioned that in the medium to long term, the client side will develop support for metadata browsing and metadata querying [MADSEN95]. This is most simply accomplished by writing objects that have interfaces to the requisite script languages (Java, Obliq, Safe-Tcl,...). This will allow dual benefits:

- rich metadata objects can be created and manipulated;
- the infrastructure then exists for the system to support mobile agents.

1.4.5 Object Oriented servers

As with the client side, the server side gains a number of benefits from viewing the information space as a web of distributed objects. One is the ability to service requests robustly: a failure to locate a requested object can be transformed into an interface reference to a migrated instance of the original object, or to a script which can be retrieved and invoked to locate the original object.

1.4.6 Support for Mobile Clients and Servers

The infrastructure need to support mobile clients through asynchronous operation. For example, in MIT’s Rover, the infrastructure is prepared to use several protocols (such as SMTP) as the underlying transport for HTTP. When a page is touched which cannot be accessed immediately, the infrastructure locally invents a page which says “come back here later to see the real thing” [JOSEPH 95].

It is also envisioned that support for server mobility will become an issue, and the design must address this as well. Servers may be mobile, for example, because they are associated with data-gathering aspects of an enterprise. In general, server migration will inevitably occur as a consequence of environmental factors; the infrastructure must allow servers to continue to serve requests. Such a requirement can be met be exporting an interface reference to an ORB. It is worth noticing that the full URN/URC scheme as currently proposed will be able to cope with this situation by allowing name resolution through an intermediary service. These approaches can also be used as a way of dealing with in-service upgrades of servers.

This means that parts of the infrastructure need to support caching or proxies which are “asynchronous capable”.

1.4.7 Integration with DIMMA streams

Dealing with streams which have no QoS requirements is an immediate objective. A longer term objective is to handle QoS requirements such as bandwidth and jitter, integrating the DIMMA work.

1.5 Other issues

1.5.1 Benefits

There are many benefits which are mentioned above. One which has not been mentioned above is that this work will have the affect of migrating the Web onto technology which is the basis of commercial products (CORBA). If we are successful distributed objects in general and CORBA in particular will assimilate the web. The alternative commercial technology is DCE.

1.5.2 Feeding back into GIOP Revision

There is a CORBA task force working on revising the GIOP proposal. We could, if we wished, feed our requirements into this task force. The deadline for submission is some time in August (Andrew Watson has the details).

1.5.3 Dependability and transactions

These need to be supported for commerce.

1.5.4 Security

The need for security to be available is widely recognised as the foundation of future commercial interest in the development of the web.

Although security is not incorporated in the outline design already sketched as a solution to the requirements discussed here, the use of HTTP as the binding protocol will allow the system to drop back to the HTTP level when use of the SSL (Netscape's Secure Socket Layer) is requested, or even to load the S-HTTP (Secure HTTP) module automatically at the client end when the server requests a secure transaction at the HTTP level.

1.6 Acknowledgement

The ANSA Information Services Framework (ISF) Group are: Mike Beasley, Nigel Edwards, Mark Madsen, Ashley McClenaghan and Owen Rees. They would like to acknowledge the contributions of all their colleagues on the ANSA Team and the ANSA Technical Committee.

References

[EDWARDS 95]

"Performance of HTTP and CGI", Nigel Edwards, APM.1500.00.02, APM Ltd, Cambridge UK, June 1995.

[JOSEPH 95]

“Rover: A Toolkit for Mobile Information Access”, A. Joseph, A. deLespinasse, J. Tauber, D. Gifford, M. F. Kaoshoek, M.I.T, March 28, 1995 (to be published).

[KRISTOL 95]

"A Proposed Extension Mechanism for HTTP", D.M. Kristol, Internet Draft draft-kristol-http-extension-00.ps, January 1995.

[MADSEN 95]

"Agents for Knowledge Resource Mapping in the World Wide Web", Mark Madsen, APM.1473.01, APM Ltd, Cambridge UK, May 1995.

[MADSEN 95a]

"Metainformation Management on the World-Wide Web", Mark Madsen, APM.1405, APM Ltd, Cambridge UK, June 1995.

[MCCLENAGHAN 95]

"The Changeling Webserver",Ashley McClenaghan, APM.1453.01, APM Ltd, Cambridge UK, March 1995.

[REES 95]

"Re-engineering the WWW Infrastructure", Owen Rees, APM.1482.01, APM Ltd, Cambridge UK, May 1995.