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

ANSA plans: 1996 - 1997

Rob van der Linden

Abstract

The business problem addressed is...

The technical problem created by that business problem is ...

The solution being offered is....

APM.1708.00.01 **Draft** 20th August 1996

Project Management (confidential to ANSA consortium for 2 years)

Distribution:
Supersedes:
Superseded by:

ANSA plans: 1996 - 1997

ANSA plans: 1996 - 1997

Rob van der Linden

APM.1708.00.01

20th August 1996

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 APM 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.

APM 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 Overview

The following pages present the techncial plans for the ANSA Workprogramme for the period January 1996 to February 1997.

The ANSA workprogramme has been divided into a series of projects, each with well defined deliverables, which are of real practical use to our sponsors.

ANSA projects take place in two distinct areas:

1. CORBA advanced features: recognising that different applications require different features from their distributed systems infrastructures, activities in this area aim to extend CORBA functionality with real time and multimedia capabilities in a modular framework.
2. The Internet meets CORBA: recognising that the web now occupies an important place in distributed computing, activities in this area aim to prototype applications which cross the divide between CORBA and the Internet (web).

CORBA advanced features are of interest to telecommunications equipment producers and system integrators and providers of services with real time or mixed media requirements (e.g. command and control). Due to the complexity of distributed systems infrastructure and relatively low rate of market change, projects in this area are strongly related and planned on a longer timescale.

The Internet and CORBA activities are aimed at end users and service providers. They reflect the rapid rate of change of technology and applications in the marketplace and aim to offer benefits in the near term. The projects are also more self contained.

1.1 The projects

Amber: multimedia streams in DIMMA

Amethyst: DPE resource management

Jet: CORBA API on DIMMA

Jetstream: explicit binding and streams in CORBA

Ruby: concurrent multimedia stream processing

Jade: service access and provision on the Internet through hollowed out computing and CORBA

Quartz: service distribution through electronic mail

Titanium: the structured web server

Need a section on planning details: resources etc. Need not be exposed on web pages.

2 CORBA advanced features

Building distributed applications using the ANSA principles, perhaps using a CORBA compliant product, is currently limited to operational style interactions; real time and multimedia capabilities are not available. Writing multimedia capable distributed applications is possible with tools such as Vic, but these tools are difficult to use and require programers no or limited control over system resouces from their application programs. This leads to applications which are not portable or hard to maintain.

Business requires all kinds of applications, including those which require a variety of multimedia, real time and network control capabilities. Application diversity is inevitable: multimedia conferencing applications need to run side by side with billing applications and ATM network management applicatons for instance. Diverse applications require different resources and guarantees from the distributed processing environment. One DPE cannot cater for the whole spectrum of requirements.

2.1 The adaptor and stream concepts

The concepts of adaptor and stream are introduced to alleviate the above problems.

Adaptors allow

- distributed system suppliers to produce plug and play DPEs
- system integrators to configure DPEs to application type
- application programmers to choose and configure DPE functions (software and hardware) which are appropriate for their application.

Streams allow

- service providers to specify interfaces to isochronous multimedia data flows
- system integrators to supply multimedia service components
- application programmers control over the stream through standard operational interfaces
- end users a way to specify what quality of service they are willing to pay for

An adaptor encapsulates a specific protocol. It is resourced explicitly to deliver predictable behaviour. It is resourced staticly by a DPE supplier or system integrator or dynamically by the application programmer perhaps as late as the binding stage. It also provides the application programmer with that part of the API which is specific for their application.

Adaptors for streams offer stream interfaces. Typing these interfaces allows type safety as well as an opportunity to specify stream componentware for multimedia applications.

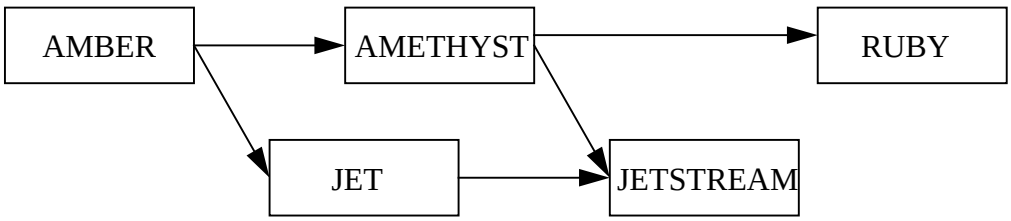
The adaptor approach thus leads to extremely modular distributed processing infrastructures, with built in extensiblity, whilst allowing conformance to standards such as OMGs CORBA.

2.2 The projects

The approach taken builds on the earlier work on the Distributed Interactive MultiMedia Infrastructure (DIMMA) which was released in December 1995. The DIMMA DPE acts as the framework for developments in current and future projects.

Figure 2.1 illustrates the projects in this area and how they are related to one another.

Figure 2.1: The projects and their dependencies



The AMBER project proved that the adaptor and stream concepts are indeed workable. It produced an adaptor for the M-bone stream, including a buffer class for RTP, protocol modules for IP multicast, a binder and ORB interfaces based on Sun's InterORB engine. The adaptor is placed "under" the DIMMA DPE. Stubs and InterORB interfacing was handcrafted due to a lack of automated tools to do this job. AMBER demonstrates a live video multicast within the DIMMA DPE.

Adherance to standards is important and the JET project was started to provide a CORBA API over the previously released DIMMA API. JET also provides a CORBA IDL compiler. Special attention is paid to ensure that all resource management is built in such a way that future multimedia support, explicit binding, streams and QoS support are assured. For this reason it was not possible to simply adapt an existing CORBA implementation. JET provides you with the ability to port simple CORBA applications onto the DIMMA DPE.

The AMETHYST project is to roll out the adaptor and stream concepts. This includes implementation of improved resource separation and management in the DPE. It will allow you to produce your own adaptors using the results from JETSTREAM. QoS management within the adaptors will also be supported as will explcit binding at the application level.

JETSTREAM aims to extend the tools provided in JET so that access to the added functionality provided in AMETHYST can be exploited. JETSTREAM allows programmers to write sophisticated applications which use specific functions offered through adaptors. Adaptor templates can be generated automatically. Extensions of CORBA IDL and API will be properly implemented and experience gained brought back to OMG standards activities.

JET, AMETHYST and JETSTREAM thus roll out the adaptor concept and provide the tools required to write advanced distributed applications.

2.3 Future work

Beyond the above projects we see the need to pilot real time reactive programming techniques in support of synchronous programming. The RUBY project would create an exploratory pilot implementation.

3 The Internet meets CORBA

Provison of services in the Internet is still not straightforward. Despite the advantages on offer to the service provider, standards such as CORBA are not adopted because they require the service user to install a distributed systems infrastructure. The present user-led market requires different solutions.

ANSA projects in this area aim to bring traditional distributed systems infrastructures and services closer towards the Web and its attractive user interface without requiring significant investments in new technology at least from the end user.

ANSAweb, which was delivered in December 1995, provided a set of components from which gateways between HTTP and IIOP could be built. A locator and a stub compiler completed the package. It provides the components needed to access CORBA services from the Web.

Two other ways to distribute and provide CORBA services to Web users are in work:

1. The JADE project delivers “hollowed out computing” loading IIOP enabled JAVA applets into JAVA enabled web browsers, thus allowing end users to buy service access, instead of service access software; that is the *service* instead of a *means* to use a service.
2. The QUARTZ project employs standard e-mail to distribute access to services and implements wide area workflow support.

The size and complexity of web servers is becoming a risk in many organisations. The TITANIUM project proposes to structure the web server using standard CORBA technology.

4 Amber

-- this should be replaced by the current release page which needs to be restructured so that general descriptive information is on page 1, then links to installation information etc.

The Amber project investigated adaptive, scalable, multi-cast based multi-media techniques in the context of the DIMMA multi-media DPE. Amber integrated the DIMMA nucleus with the MBone (the experimental Internet Multi-cast Backbone) and the VIC video tool and investigated streams (typed directional flows of multi-media frames) support for MBone based multi-media.

A release of Amber VIC has been made available to ANSA phase 3 sponsors. The future phases of the Amber project will continue to use the MBone as a multi-media demonstrator. The Amber release note provides further detail on the deliverable.

The Amber work on multi-media streams has provided input to the OMG's Telcom SIG and to ReTINA, as well as directly furthering the design and implementation of DIMMA.

5 Jet: CORBA API on the DIMMA platform

5.1 Aims of JET

JET aims to provide a subset of CORBA API supported on the ODP/DIMMA platform. JET provides:

1. A subset of a CORBA compliant platform which will enable experimentation with the necessary extensions to CORBA, such as real- time, streams, signals, explicit binding and QoS.
2. Inter-operability with CORBA platforms: it will be possible to run management applications to control real-time applications running on DIMMA, for example, from remote CORBA platforms.
3. Conclusions from the work on the extensions listed above can be used as input to OMG to show how they can be added to the CORBA API.

5.2 JET implementation approach

It is necessary to determine for each CORBA facility in the subset whether to implement it in the:

- ODP API
- Stubs
- CORBA personality module (CPM).

In deciding where to support a specific CORBA feature, the approach taken is that the ODP API should be kept as “clean” as possible i.e. provide a convenient minimal set for distributed application designers to implement distributed applications. Functionality not included in the ODP API or in the stubs will have to be supported in JET.

5.3 CORBA C++ support in JET

5.3.1 Supported invocation structure

At-most-once if an infrastructure exception is raised exactly-once if invocation returns successfully

- IN, OUT and INOUT parameters.
- First return parameter is the function’s result; it’s type is the functions type
- Exceptions (requires C++ compiler with full support for exceptions)

5.3.2 Supported IDL Types

- Basic types, constants, enums
- Structures, strings, unions, sequences, arrays
- Abstract ANY: object references (e.g. for trading).

5.3.3 Supported usage of references:

Smart pointers T_var including:

- Shallow copy for T_var
- Deep copy for T_var.

5.3.4 Supported argument passing conventions:

- For IN arguments use: const T_var&
- For OUT use: T_var&
- For INOUT arguments use: T_var&.

5.3.5 Features of CORBA not supported in JET:

The omitted features of CORBA are: Context parameter, Attributes, Oneway invocations, Concrete ANY, Dumb pointers T_ptr.

The omitted features:

- could lead to memory management problems which can be alleviated or overcome by the use of features easily developed in a language such as in C++ (e.g. restricting references to smart pointers)
- can be provided by different means.

6 Amethyst

[Amber Phase 2]

The next phase in the DIMMA project will add more extensive support for resource management to the DPE, allowing controlled allocation and multiplexing of all resources in the nucleus. These include: threads, message buffers, communication channels as well as low-level and system-level abstractions such as memory and network endpoints.

Resource separation, into managed pools, is required to provide predictable performance under unpredicatable loads and to avoid application cross-talk. For example, this is required for in supporting the concurrent processing of multiple multi-media flows in real-time.

Resource pools can provide proper resource separation and a basis for low-evel QoS management. They can also provide structure which can be used as a basis for concurrency control, providing cleaner resource management and locking policies than are currently employed.

Pools will also make the nucleus more modular and provide a simpler environment to develop in. Pools will form the basis of resource scheduling and may be parameterized with different resource scheduling policies. This will provide the implementation basis for both highly scalable, (late binding, aggressively multiplexed) DPE services (such as are required in the large scale Internet) and for QoS controlled dedicated connections for high throughput and real-time responsiveness.

Support for the DIMMA explicit binding architecture, based on plug-able object and stream adapters, which allow typed application interfaces to be bound to QoS controlled communication endpoints, will be implemented. Adapter implementations will make heavy use of resource pools. This activity links in with the [Jet Streams] project and leads the work into [Amber Phase 3].

Deliverables:

1) DIMMA nucleus re-structured for resource management using the concept of resource pools.

2) Enhanced nucleus development environment: support for developing stream and object adapters, support for low - level QoS management.

Estimated resources and timescales: 2.5 X 6 mm

7 Jetstream

Jet is providing a CORBA based API for DIMMA, based on a clean subset of the CORBA 2.0 C++ mapping. Jetstream extends the work to handle explicit binding and streams. Streams allow applications to interface to typed directional data flows. These are often high bandwidth and time structured, such as continuous audio and video multi-media flows are. Explicit binding allows applications to choose and configure network protocols, connecting typed interfaces to created network endpoints and to dedicate resources to established bindings (via resource pools), thereby achieving QoS control. Protocol bindings are abstracted from an application via stream and object (operational) adapters. Adapters export IDL based control interfaces to an application for connection management.

Both the Jet IDL definition and the stub generator will be extended with support for streams and explicit binding. The extensions will support typed stream interfaces, allowing applications to interface to high-bandwidth streamed communications. The design also facilitates diverse interaction models which may be partly supported via the automatic generation of adapter templates from the extended IDL and binder specifications.

The Jet API will be extended with support for application level threading, which will allow applications to perform concurrent processing of multi-media flows, as well as providing support for implementing resource controlled, multi-threaded CORBA servers.

[Application level marshalling support will be designed to support application specific marshalling schemes, to be used for multi-media encapsulations or protocol specific type representations]

7.1 Deliverables

Jet stub compiler extended with stream support.

Adapter template generation.

API support for application concurrency.

Estimated resources and timescales: 2 X 6 mm

8 Opal

8.1 Overview

Within the framework of the DCAN project, the Opal sub-project will develop a simulator which will provide a graphical display of an ATM network, consisting of a number of connected “switch”, “controller” and “external device” entities, and be able to model the interactions between these entities required to implement a given control architectural model.

The intention is to provide a framework that we may then use to investigate the behaviour of our candidate control architectures.

In parallel with simulator construction we will, in colabaration with our DCAN partners, define a control architecture specifically oriented to the management and control of “dumb” switches and devices in the context of private ATM internets.

The ATM network simulator will be used to annimate the chosen control architecture. This will provide feedback on the practical operation of the model, allowing us to then iteratively refine the architecture.

The simulator will also provide a framework, into which we can progressively integrate real ATM devices that have been programmed to support the control architecture.

8.2 Implementation

We have chosen to implement the ATM network simulator in Java, as we believe that its object oriented nature, integrated multi-threading, and good graphics support make it an attractive option for such a grapical simulation environment.

An additional advantage of a Java implementation, is that its inherent multi-platform support should make it easy for sponsors to download and use the resultant simulator without having to port it to their chosen platform.

Where appropriate the control architecture interfaces will be expressed in IDL to a achieve a comprehensive protocol description, and a clean separation between interface and implementation.

8.3 Deliverables

1. Graphical ATM network simulator.

2. A description of a control architecture for an ATM internet of dumb switches and devices, controlled by a largely separate distributed system

A phased approach to the deliverables is recommended, with the goal of producing and demonstrating an initial network simulator, implementing a very simple control architecture as soon as possible. In the light of the experience gained during this exercise, both the model and architecture may then be iteratively refined.

8.4 Timescales

Initial simple, simulator: [to be discussed]

Initial control architecture:

Refined control architecture:

Refined, extended simulator :

9 Ruby

This project phase will apply the results of the previous Amber deliverable (resource pools and QoS controlled adapters) to demonstrate concurrent multi-media stream processing based on the initial MBone demonstrator.

[Examine building a useful MBone tool such as a video encoding converter (gateway) or rate adapter or reflector.]

Application level QoS will be investigated. Application level QoS for multi-media will be mapped onto low level MBone QoS, possibly incorporating network resource reservation (for example, based on the RSVP model). ATM QoS will also be investigated in the DCAN context, to help tie the demonstrator into the DCAN connection management architecture.

[The Jet API support for concurrency and the Jet / Amber support for generating and constructing adapters will be evolved to provide support for interfacing to synchronous programming environments (such as that which the Esterel language provides). This together with the resource and QoS managed DPE, will provide a basis for constructing real-time reactive objects.]

[Protocols: RT communications, multi-cast protocols]

Deliverables:

Enhanced Amber multi-media demonstrator. Application level QoS management.

Estimated resources and timescales: 2.5 X 6 mm

10 Jade

Overview, timescales and deliverables on page 1

References to pages with the technical proposal to follow.

Text below to be structured along these lines.

10.1 Abstract

The Jade project is about Service Access and Provision on the Internet. Jade aims to fuse corporate CORBA infrastructures into the ubiquitous Web by developing software which will allow Java (Web objects) to access CORBA objects. The core component of this software is a CORBA IIOP engine, written entirely in Java.

10.2 Introduction

“Customers won’t buy software, they’ll buy services.”

Internet growth is continuing at a staggering pace. Already it has become an important medium for businesses. Service provides in the banking, travel, retailing, insurance, TV and publishing sectors are clambering to go on-line as they realize the marketing and service provision potential of the Internet.

The Web is fastest growing part of the Internet, and Service providers have begun to create multimedia Web sites to tap into the growing number of on- line customers.

Until recently customer access to on-line services has been limited. Customers have had to interact through static pages and simple fill-in form interfaces because of the limited capabilities of the technologies used to build the Web. But things are changing fast. Recent technologies -- such as Java, Netscape plug-ins, VRML and HTML3 -- have opened up many new opportunities for doing business on the Web.

The aim of the Jade project is:

- to realize the potential of the Web and develop new opportunities for Service Access and Provision on the Internet.

Towards this aim, Jade will:

- harness new Web technologies (focusing on Java), and * integrate the existing corporate solutions (focusing on CORBA).

10.3 Motivating Observations

Jade’s solutions are very closely linked to developments in Internet technologies. With this in mind, below we list a number of our observations on Internet related technologies which help explain Jade motivations and design decisions.

- Connectivity: The Internet is the largest information network that ever existed and will, during the next decade, be a major focus for service provision. The Web is the killer application in the Internet.
- Web Operating Systems: Web browsers, such a Netscape, are set to become the new Net based operating systems.
- The Web Programming Language: Java has gained a widespread following as the language for programming Web based applications. Many software companies have licensed it from Sun. (JavaScript and Visual Basic Script are possible rivals.)
- Integration and Access Technologies: Efforts are underway to provide interworking through Java between technologies such as: OLE, SQL, CORBA and the Web.

10.4 The Core Idea

Jade's core idea for Service Access and provision on the Internet is to allow service providers (like banks, high street retailers, travel agencies, libraries, on-line music and video stores, traffic information systems, etc.) to offer their on-line services to their customers through Web based, downloadable user interfaces.

Initially Jade will focus on providing Web access to CORBA based services.

In Jade, a downloadable user interface (to an on-line service) will be implemented as a Java applet. To enable such an applet to talk to it's CORBA based server, Jade will develop a CORBA IIOP package for Java. This package will be written entirely in the Java language itself and so will be downloadable and portable across all Java supporting platforms.

We envisage the following scenario:

1. The service provider constructs a CORBA based service. 2. The service provider programs a customized user interface to the service using Java and Jade's CORBA IIOP package. 3. The service provider uploads the user interface to the Web. 4. The customer sees the service advertised on the Web, and downloads the self-installing user interface from the Web. 5. The customer accesses the service through the customized user interface.

10.5 Benefits

10.5.1 Benefits to Web Based Consumers

- "APM's Jade empowers Web users by giving them the capability to access corporate information, commerce and entertainment systems."

The Java component in a Jade based interface will free customers from all software management issues --- Jade interfaces will automatically download and self-install at the click of a button. The Web component in a Jade interface will wrap CORBA based services in an easy-to-use, multimedia interface.

10.5.2 Benefits to Service Providers

- "APM's Jade gives you the power to add Web based consumers to your customer base."

Jade will enable service providers to expose their services to a large and growing community of Web users, no matter which computing platforms they are using. Service providers can fix bugs, improve and upgrade their customer interface software without

having to disrupt customers. The flexibility of the Java component in Jade interfaces also opens up many possibilities for new customer-service interaction models.

10.5.3 Benefits to APM:

- "The company that pioneered the integration of CORBA into the Web."

Jade will bring APM publicity and recognition as a company doing practical research using the latest Internet and Object technologies.

10.6 Deliverables

1. The Jade CORBA 2.0 IIOP Engine Description: A downloadable software package written entirely in Java which will facilitate Java applets to interwork with CORBA 2.0 services. Delivery date: March 28, 1996.
2. The Jade Web Space Description: A set of Web pages on APM's external Web server which introduce, describe and demo the Jade concept, and which provide access to the Jade software. Delivery date: March 28, 1996.
3. Jade Documentation Description: Documents which describe the Jade business case, the architectural design of the Jade CORBA IIOP Engine, and how to program Java applets using Jade software. Delivery date: March 28, 1996.
4. Jade Proof-Of-Concept Projects Description: Involvement in industrial projects in order to validate and prove the usefulness of Jade. Delivery date: March 1996 onwards.
5. Jade Publicity and Dissemination Description: Publicize the Jade work on the Web, through mailing lists, and by giving outside presentations. Delivery date: Februrary 1996 onwards.

11 Quartz

11.1 Motivation

Since enterprises rely more and more on computer-provided information, so they regard their IT applications as services provided to users, rather than as applications installed and run by those users. As the world marketplace develops, these application service users are more likely to be located far away from the central site and to be equipped with different types of machine. Installing applications and data is a costly process, especially if they are not to be used frequently or if complete consistency has to be achieved across all sites, for example at the time of making the annual operating budget. Often external users, such as customers of the company or corporate partners, are involved and there is an increasing need for temporary use of an application or for a punctual connection to a service. Therefore, a technology is required for making software and information widely and easily available to a user, wrapped together as a service with an easy understand, familiar interface.

11.2 Aim and Concept

Tools such as electronic mail and the World Wide Web (WWW) are familiar to almost everyone. The Web's simple and user-friendly interface makes users (even non-technical ones) able to access, provide, and exchange information easily and widely. The OBJECTmail concept is to make CORBA objects widely and easily accessible from desktops and consumers by wrapping CORBA objects with a Web interface and using email and Web infrastructure for distribution. This will make them simple to use and remove the need to learn a particular interface for each. The aim of the Quartz project is to show how the concept of OBJECTmail can be used to solve the above problem and demonstrate its feasibility and benefits.

11.3 Approach and Benefits

To provide interoperability between WWW and CORBA -- a fundamental issue to realise the OBJECTmail concept -- Quartz will take an approach based on Java over IIOP. A client interface for a CORBA object is defined as a Java applet that can be embedded right into HTML pages, therefore accessible using the WWW. However, to enable such an applet to talk to its CORBA object, a CORBA IIOP engine implemented in Java is required. Such an engine will be developed in an on-going project called Jade.

Based on such an approach, OBJECTmail can be used to provide a simple method of sending CORBA services to a user, and be used to develop a CORBA-based application oriented to Web users. The resulting system will have the following advantages:

- Accessible from different sites and different platforms
- Easy connection and installation
- Simple but powerful interface
- Easy access to related information

- Rapid application development
- Re-usable and extensible

11.4 Deliverables

The result of the Quartz project will include some example demonstrations, together with an object library containing some key components, a user builder package for reducing the difficulty of developing similar applications, and documentation describing the system principles and design decisions.

The target timescale for the project is from January to May, 1996.

12 Titanium: A structured Web Server

-- these pages need information from the WWW5 paper refered to later.

Consider replacing the text with WWW5 paper itself, as follows:

Introduction with information about effort and deliverables and time scales.

references to thearchitecture - the technical overview/approach.

The text below is then to be saved and used for the presentation.

12.1 Present Situation

Web services require more interaction than in the past when they only delivered prefabricated documents. The existing technology for building documents and services on the fly is to use the CGI (Common Gateway Interface) to invoke a stored program at the server when a URL is requested. This leads to a monolithic server architecture as well as causing poor performance, due to the need to fork a new process to execute the CGI program.

12.2 Risks of Present Situation

CGI is hard to program accurately, and CGI programs are even harder to maintain or move from server to server. As the need for more interaction between server and client increases, this will become more of a burden on server mainteners. Performance will also be affected by the poor scaling property of CGI: servers cannot simply hand off CGI execution to other servers. Finally, the present mechanism has no provision for extensibility and interoperability without programming from scratch.

12.3 Issue

The main issue is how to build extensible web servers that are easy to maintain and interoperate with the wide range of services that will be required in the future.

12.4 Options

- We could improve the development methods for CGI programs, and merge some of the toolkits that have been developed to help interoperate with other backend systems. This option involves a lot of effort in joining incoherent technologies.
- We could implement the structured web server architecture described in the paper submitted to the WWW5 conference.

12.5 Recommendation

Implement the structured web server architecture by building the necessary technology components. This has the benefits of leveraging a lot of existing ANSAweb technology, straightforward interoperation with all CORBA or DCE services and will produce far better server performance.

The architecture in simple form is:

service1 |---| |---| | |--|<------| |--|<-\ |---| |---| \ Launcher \ Web Server service2 \|------| |---| | | | | |--|<--------------------| |---| |---| | | | |------| Locator / |---| / | |--|<-----------------/ |---|

The Locator returns service references to the web server in response to the web server's requests for services. When an instance of the service is not available, the Locator returns a reference to the appropriate factory for the web server to pas to the Launcher, which then creates a service instance and passes its reference back to the server. Multiple service instances can be created for load balancing purposes.

12.6 Implications

Most of the above componentry and almost all the necessary technology exists and is available. The Launcher will need to be written from scratch, most simply as a CGI program for either CERN or NCSA httpd. The Locator will be adapted from the existing ANSAweb locator by rebuilding it with service-level granularity in place of site-level.

The work is expected to take 6 person months (+- 25%), with a number of deliverables, roughly in the order of:

1. Design presentation and document
2. Launcher program and code
3. Locator program and code
4. Demonstrator application
5. Performance testing
6. Technology transfer presentation
7. Final project document and summary.
8. Paper submission for WWW6 conference based on deliverable vii.

12.7 Yields

The expected yields of this work are that it will produce:

1. Web server interperability mechanisms that can be used to leverage existing serices built around object technologies such as CORBA.
2. Simple extensibility/upgradeability for the server developer and maintainer.
3. Better performance characteristics through using object mechanisms for service instantiation and load balancing.
4. Compatibility with existing standard web server systems: because of the way the structured web server architecture is designed, the basic object support mechanisms can be implemented in CGI and further object services added on to a single CGI middle tier.
5. A testbed for the Java technology developed within Jade, and for other projects involving the need to deliver live services or mobile code in some form.

References

[LINDEN 93]

van der Linden R. J., *An Overview of ANSA*; **AR.000.00**, APM Ltd., Cambridge U.K., May 1993.