Object Management Group

Framingham Corporate Center
492 Old Connecticut Path
Framingham, MA 01701-4568

Telephone: +1-508-820 4300
Facsimile: +1-508-820 4303

<u>Common Facilities RFP-4</u>

Common Business Objects

and

Business Object Facility

SUBMISSIONS DUE: October 15th 1996

OMG TC Document CF/96-01-04

1. INTRODUCTION

1.1 Purpose

The Object Management Group's central mission is to establish an architecture and set of specifications, based on commercially available object technology, to enable **distributed integrated applications**. Primary goals are the *reusability*, *portability* and *interoperability* of object-based software components in distributed heterogeneous environments. To this end, the OMG adopts interface and protocol specifications that define an Object Management Architecture (OMA) that supports applications based on distributed interoperating objects.

The OMG is issuing this Request For Proposal (RFP) to address the OMA component called **Common Facilities**. As described more fully below and in Section 5, this RFP solicits proposals for the following:

- **Common Business Objects**

  Objects representing those business semantics that can be shown to be common across most businesses.

- **Business Object Facility**

  The infrastructure (application architecture, services, etc...) required to support business objects operating as cooperative application components in a distributed object environment.

For the purposes of this RFP application components are manifestation of business objects within the context of the Business Object Facility, however there may be other application components that do not correspond to business objects.

1.2 Problem statement

1.2.1 Business requirements

Global business competition and a shift from commodity to short life-cycle or custom products has created an environment of continuous business structure change. In order to effectively compete, businesses are constantly revisiting products, corporate collaborations (virtual enterprises), processes, suppliers, and customer concerns. A successful information strategy must enable business flexibility by provisioning business solutions at a rate commensurate with the increasing rate of business structural change.

Rapid solution delivery in response to continuous business process changes requires direct involvement of empowered users to dynamically change their business processes, workflow, rules, policies, presentation, and other aspects of their environment. Enterprise IT organizations have, however, repeatedly failed to achieve productivity, performance, and cycle time gains necessary to adequately support business change. They have failed to provide units of delivery which will meet the dynamic change requirements. This is not surprising, since the software industry as a whole has failed to meet this challenge.

As business complexity increases, there is an non-linear adverse impact on requirements interpretation, productivity, reliability, delivery cycle time, flexibility, and cost. The challenge is to provide rapid implementation of accurately specified business requirements.

Mechanisms for achieving these goals are likely to utilize concepts of componentization, model-based specifications, and end-user solution composition. Such composition may be achieved through "business objects" as self-contained and independently-developed "application components" which can be used in different combinations at different times.

A useful analogy is the computer hardware area. Semiconductor (and downstream hardware) technology has consistently sustained an annual doubling of productivity and performance gains. However, similar significant productivity gains in software have not materialized. The software industry needs to capitalize on the proven concepts of specialized, application-independent, encapsulated, units of functionality - ***components***.

There is a spectrum of such software component technology, from components such as integers, stacks or arrays, to larger scale application components that make sense in the business context. It is this latter part of the spectrum which is relevant to this RFP.

Exploitation of such concepts within the software domain will be enabled by open standards for the interoperability of business objects; the existence of a marketplace (suppliers and consumers) for reusable business objects; and standardization of facilities to hide underlying software technology complexities from IT developers who are engaged in the construction of such business solution oriented application components.

In summary, the business requirement is for interoperable business objects as high-level application components relevant to and usable by application domain developers and, eventually, by end-users, through which they can provision their own business solutions. The OMG, through this RFP, is now addressing the two major pre-requisites to meeting this need: industry and cross-industry models for business objects; and the ability for IT developers to build and deploy them such that they will interoperate in ways which make sense for the users, but which may not have been foreseen by the developers.

1.2.2 Vision

The OMG vision described in the OMA (section 1.1) is not only a vision of freely interoperating, reusable and portable software objects spread across the distributed enterprise. It also a vision of removing the "major hurdles" of development time, maintenance and enhancement capability, and program complexity.

The OMG has further defined this vision in its "Direction for Common Business Object" press release (18th October 1995). This statement also talks about managing complexity, and goes on to mention "packaging the essential characteristics of business concepts".

The keynotes of the OMG Common Business Object (CBO) vision are:

- Interoperability of OMG Business Objects as both design-time and run-time constructs including the possibility of ad-hoc integration

- Simplicity, so that design and implementation, configuration and deployment is viable for the average developer

OMG CBOs from different suppliers should be able to interact effectively for an end user when that interaction may not have been foreseen by any developer. This is the hallmark of ad-hoc integration.

OMG CBOs are models, templates, designs and/or patterns which include the necessary defined semantics for interaction. They map one-to-one onto design/analysis concepts such as Customer, Invoice, Insurance claim, Road map, etc.

OMG CBOs are also run-time software constructs, and map without significant transformation to the design models. They can be implemented relatively simply, without the need for systems programming level software engineering skills. For deployment, the binary or binaries must be packaged so as to minimize dependencies, and present the essential characteristics of the business concepts. The Business Object Facility will hide complexity such that CBOs can be implemented by, for example, the average IT department business solution developer.

Multi-Tiered model for information systems

Current client-server technology serves to separate the concerns of data storage from the application. However, it fails to separate business concerns from the application, user interface and underlying technology. Perhaps more importantly, it also fails to reflect the peer-to-peer and "multi-tier" reality of today's business systems. The result is that the "traditional" client-server model (or, “fat client”), while being very useful in a number of situations, does not scale to encompass the needs of the enterprise and change within the enterprise.

Several multi-tier models have been proposed as better ways to build information systems. Rather than present each model in turn, this section draws out the common factors. These models exhibit two important features:

- They define three general areas rather than two:
    - Presentation: (variously referred to as "Presentation and desktop", "User Interface", "Rendering", etc.)
    - Business: (variously called "Business Objects", "Business Process Objects", "Corba Entity", "Agents", “Workflows” etc.). Business objects represent business semantics independent of storage system or user interface.
    - Persistence: (variously described as "Storage Systems", "DBMS", etc.).
- They explicitly support the reality of separation - both physical (many computing nodes within a single system) and logical - of the above general areas and the consequential need for peer-to-peer interactions as distributed objects.

Many models embrace the concept of Business Objects as things which represent business concepts. These models encapsulate the business relationships, attributes and rules as well as isolating the business logic from both presentation detail and details of the storage system.

This means that the location and implementation of storage systems and of presentation and rendering mechanisms becomes an engineering decision that does not have to effect the business object model. *The business object model becomes the driving force of the information system, not the technology. The business object model is derived directly from the “shape” of the enterprise*.

Finally, models which address implementation of business objects also address:

- Deployment of the executables (units of delivery from developers) in today's distributed peer-to-peer environment. It is the degree of correspondence between logical design models and implementation/deployment models which will meet the objectives of interoperability in the run-time domain.

- Minimal loss of semantic information, between design, code and deployment.

1.2.4 Scope

OMG recognizes that, while standards based solutions may never provide the technology to solve all of the above problems, this RFP is a step in the process of filling the gap between the current technology oriented standards and the business requirements.

1.3 RFP organization

This RFP is organized as follows:

Section 2	*Context* - background information on OMG's Object Management Architecture (OMA).
Section 3	*Process* - background information on the OMG specification adoption process.
Section 4	*RFP Scope, Objectives and Requirements* - requirements and criteria that apply to all specifications proposed to OMG.
Section 5	*Common Facilities RFP-4 Items* - description and requirements for the specific Common Facilities this RFP covers.
Section 6	*Instructions for Responding* - explanation of how to make a submission to this RFP.
Section 7	*Submission Review Process and Schedule* - summary of how RFP submissions will be reviewed and the schedule for review.
Appendix A	*Commercial Availability Requirements* - OMG requirements on the commercial availability of technology on which submissions are based.
Appendix B	*End User Special Interest Group Requirements*
Appendix C	*Conventions and Guidelines*

1.4 References

The following documents are referenced in this RFP:

Object Management Architecture Guide, Revision 3.0.

The Common Object Request Broker: Architecture and Specification, Revision 1.2. OMG TC Document 93-12-43

Common Facilities Architecture, January 1995. OMG TC Document 95-1-2

Object Services Architecture, Revision 1.1, January 1995. OMG TC Document 95-1-47

Object Services Roadmap, October, 1992. OMG TC Document 92-8-5

Common Object Services Specification, Volume 1. OMG TC Document 94-1-1

BOMSIG white paper, OMG TC Document 95-4-1

ICL response to CFTF RFI #2, Financial services and accounting facilities, OMG TC Document 95-8-28

Data Access Corp. response to CFTF RFI #2, Financial services and accounting facilities, OMG TC Document 95-8-27

2. CONTEXT

2.1 Goal of OMG Common Facilities

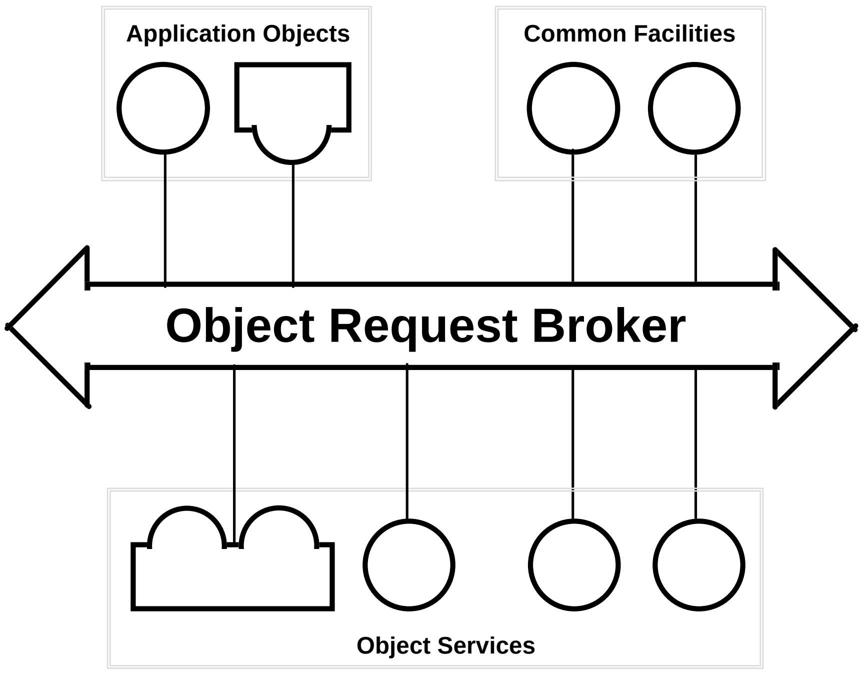
The Object Management Architecture Guide (OMA Guide, Third Edition), published in 1995, defines OMG's technical objectives and terminology and provides the conceptual infrastructure upon which subsequent supporting specifications are to be based. The guide includes a Reference Model which identifies and characterizes the components, interfaces, and protocols that compose the OMA.

Figure 1 shows the four major parts of the OMA Reference Model. The solid boxes represent software with application programming interfaces. The dotted boxes represent categories of objects with object interfaces.

- The *Object Request Broker* (ORB) enables objects to transparently make and receive requests and responses in a distributed environment.
- *Object Services* is a collection of services (interfaces and objects) that support basic functions for using and implementing objects.

• *Common Facilities* is a collection of facilities that provide general purpose capabilities useful in many applications.

- *Application Objects* are objects specific to particular industries or end-user applications.



Please note that the above diagram is undergoing change as a result of the reorganization of OMG.

Through a series of RFIs and RFPs, OMG is populating the OMA Reference Model with detailed specifications for each component and specification. The **OMG Object Model**, published in 1995, defines *common object semantics* for specifying the externally visible characteristics of objects in a standard implementation-independent way. The **Common Object Request Broker Architecture** (CORBA 1.2), adopted in 1993, defines the programming interfaces to the ORB component.

An ORB provides the basic mechanism for transparently making requests to and receiving responses from objects located locally or remotely without the client needing to be aware of the mechanisms used to communicate with, activate, or store the objects nor where the objects are located. As such, it forms the *foundation* for building applications constructed from distributed objects and for interoperability between applications in homogeneous and heterogeneous environments.

Using an ORB, requests for an object's services are made without regard to the location or implementation of the object providing the service, i.e., without regard for the mechanisms used to represent, store, manage, invoke or communicate with the object. Objects made available through an ORB publish their interfaces using the Interface Definition Language (OMG IDL) as defined in Chapter 4 of the CORBA specification. The OMG IDL provides a programming language-independent way of specifying an object's operations and attributes.

Common Facilities comprise facilities that are useful in many application domains and which are made available through OMA-compliant object interfaces. Unlike Object Services, which will be supported on all platforms, Common Facilities are optional. Not all standardized facilities will be available on all platforms, but, if available, they will provide the OMG approved semantics. OMG, through its Common Facilities Task Force, intends to focus initially on facilities that are important in the sense that they are either *fundamental* for developing useful ORB-based applications composed of distributed objects, or that they provide a *universal* (application domain-independent) basis for interoperability.

The cost, performance, quality, and other properties of the specification implementation may or may not be important in particular situations, so long as it has the expected behavior. These specification interfaces define expected behavior for developers implementing this software, as well as for users of the facilities. To support maximum use and reuse of these software components, it is essential to design them to depend on only the behavioral aspects (i.e. interfaces) of other components they use. Facilities should be defined so that they will support a variety of implementations, each potentially making a different set of engineering trade-offs.

It is also important that there be agreement on the overall style and conventions to be used for Common Facilities. Failure to do so may cause problems ranging from inconvenience to non-interoperability. The characteristics of the first specifications to be adopted will set the “style” for later facilities.

Common Facilities should be suitable for a wide variety of object implementations and be designed to operate across multiple object systems. Common Facilities should be capable of supporting a wide range of environments and application domains that should not be application, programming language, hardware or operating-system dependent.

It is not expected that Common Facilities would provide all of the operations which may be useful to an application, rather, they solve specific problems within a well defined problem domain, which should be useful to a broad class of applications.

The broad class of application that is the subject of this RFP is that of business applications. OMG is soliciting responses for the infrastructure required to support business applications as a set of cooperating, distributed components and for the common extensions of those application components common to all or most businesses.

3. PROCESS

OMG adopts specifications for interfaces, based on existing technology, by explicit vote on a technology-by-technology basis. The specifications selected each fill in a portion of the OMA Reference Model. OMG bases its decisions on both business and technical considerations.

The OMG Technical Committee (TC) provides technical guidance to the OMG in making decisions about specifications. The TC is composed of representatives of all interested OMG member companies. It is chaired by the OMG Vice President of Technology, a full-time employee of the OMG. The submissions to this RFP, taken within the specific response period, will be evaluated by the Common Facilities Task Force of the TC with the full TC then voting on a recommendation to the Board for approval. Once a specification (a technology, not source code or product) is adopted by the OMG Board, it will be available to both OMG members and non-members alike.

In addition, the End User Special Interest Group (SIG) will be reviewing all submissions made in response to this RFP. On the basis of the End User Track criteria (found in Appendix B) the End User SIG will make an independent recommendation to the OMG Board before any technology is approved as an OMG specification.

Limitation: Any service submitted, if of sufficiently general nature in the determination of the OMG Architecture Board, are subject to a future service RFP, rather than being accepted as an OMG specification under this RFP.

In responding to this RFP, the OMG feels that potential Submitters should keep the following points in mind:

- Unlike a submission to an OMG Request for Information (RFI), it is expected that an RFP submission will require significant effort; several staff months of effort might be necessary.
- The OMG specification adoption process is an open process. Responses (including specifications) to this and all other RFPs are public and available to members and non-members alike for perusal. No confidential or proprietary information of any kind will be accepted in a submission to this RFP.

Due to the reorganization of OMG which is taking place within the same timeframe as this RFP, changes in process may take place during the submission or evaluation period. Submitters are urged to say abreast of current OMG policy and procedures. Evaluation of RFP submissions may be handled in the DTC (Domain technical Committee) or the PTF (Platform Technical Committee) by an appropriately formed task force.

4. RFP SCOPE, OBJECTIVES, AND REQUIREMENTS

4.1 Introduction

The goal of this RFP is to solicit submissions from the industry for specifications that provide basic Common Facilities in the context of OMG's Object Management Architecture. This section describes the objectives, requirements, and criteria for *all* OMG specifications. Submissions must describe how they meet the general technical and non-technical requirements listed in Sections 4.3 and 4.4 below.

A premise of this RFP is that Common Facilities can be expressed in terms of object interfaces defined using OMG IDL. Such interfaces will either define types from which subtypes can be derived or facilities that can be requested by a client via an ORB.

Note that while Common Facilities will need to provide CORBA-compliant interfaces and be consistent with the OMG Object Model, the facilities themselves need not be constructed using an object-oriented paradigm.

4.2 General Technical Requirements on Submissions

The submissions to this RFP shall satisfy the following general technical requirements:

- **Interfaces shall be object-oriented and shall be expressed in OMG IDL**

- Any part of an OMG specification that is viewable as an object shall be viewed that way and given an OMG IDL interface description.

- Any requirements that a specification places on general objects shall be defined in OMG IDL. For example, a presentation facility might define what a "presentable" object is.

- The specification shall use OMG IDL naming conventions for interfaces, types, operations, attributes, etc.

- The specification shall use OMG IDL conventions for exceptions and exception parameters.

- Semantics that are beyond those expressible in OMG IDL may be expressed in English or any standard notation. Semantics, constraints, and protocols should be clear, unambiguous and precise.

- **Proposed extensions to OMG IDL, CORBA, Object Services, and/or the OMG Object Model shall be identified**

  OMG specifications shall be consistent with OMG IDL, CORBA, CORBAServices, CORBAFacilities, and the OMG Object Model. It is a goal to minimize extensions required to the CORBA and define additional components that use the CORBA wherever possible.

  Moreover, specifications should assume that there exist good implementations of existing CORBA specifications. They must not work-around early CORBA implementations if this will sacrifice uniformity and generality in the long term.

- **Operation sequencing shall be included where applicable**

  Behavior, protocol and sequencing of operations shall be part of each specification. (e.g. transactions)

- **OMG specifications shall not contain implementation descriptions**

  Maximum implementation flexibility should be preserved. However, provision should be made for application developers to control implementation choices (e.g., by pragmas or in some other way that does not affect the functional interface).

- **OMG specifications shall be complete**

  There shall be no “magic” required. For example, object creation does not just happen, some operation must be called to make it occur. Lifecycle Service’s definitions shall be defined as appropriate to provide these operations (See COSS, Volume 1, Lifecycle Service).

- **OMG Specifications should have precise descriptions**

  Specifications should have a precise description of their semantics and side-effects, if any. For example, a specification should distinguish interfaces and behaviors provided by a specification from those it expects other Facilities to supply.

In addition, submissions should demonstrate adherence to the following architectural principles:

- **Independence and modularity of Specifications**

  It should be possible to separately specify and implement each specification.

  (It may also be possible to view an existing software module as implementing a collection of one or more OMG Facilities.)

- **Minimize duplication of functionality (the Bauhaus principle)**

  Functionality should belong to the most appropriate specification.
  Each specification should build on previous specifications where appropriate. For example, if the presentation facility defines object presentation interfaces, an application should be able to use this facility and should not need to reinvent presentation for itself.

- **No hidden interfaces among specifications**

Interfaces and behaviors must be specified sufficiently well so that implementations can be replaced and the parts will still work together. For example, the operation *PutInside(container, member)* works independently of the implementation of members (via a database, ORB, etc.).

- **Consistency among OMG specifications**

  Different OMG specifications must be able to work together.

- **Extensibility of individual specifications**

  Specifications should be extensible through an iterative process. It should be clear how extensions can be defined (e.g., via inheritance, delegation, etc.).

  Specifications should use the multiple inheritance and multiple interface capability of OMG IDL wherever possible. Multiple inheritance makes it easier to define a set of interfaces that can be used by different facilities. A sub-interface can inherit from several different interfaces and use multiple inheritance to specify the set of operations it implements. Clients may choose to use only part of the behavior of any object, and view the object with just the interface they need

- **Extending the collection of OMG specifications**

  It should be possible to define and standardize new specifications without having to redesign a system that uses existing OMG specifications.

- **Configurability**

  It should be possible to configure OMG specifications in different combinations for different purposes. It should be possible to use two or more specifications in arbitrary combinations.

- **Integrity, Reliability, and Safety of OMG specifications.**

  Safeguards should be provided to guard against corrupting the integrity of objects.

- **Performance.**

  OMG specifications should be designed with performance trade-offs in mind.

- **Scalability.**

  OMG specifications should be designed with the goal that there may be many implementations that are optimized for different environments. OMG IDL and a language mapping can make remote facilities look like object invocations. Most OMG specifications have a possible implementation across a spectrum from a shared library to a remote server. The user of an OMG specification should not have to care which variety is being used.

- **Portability.**

  OMG specifications should be designed to accommodate portability of implementations across a wide range of platforms. They should not require use of a particular programming language.

In addition, submissions should *define, explain and/or demonstrate* the following:

- **Consistency with Common Object Services Specifications (COSS)**

  Consistency and compliance with COSS, Volume 1 (and other COSS specifications, if they exist at the time of submission) wherever compliance is relevant and appropriate, including:

  Provision of Lifecycle Object's Factory Service as needed (see COSS, Volume 1) to use the service or create objects managed by the service including the definition of all factory interfaces.

- **Conventions and guidelines**

  Conventions and guidelines defined, used or implied in the submission. See Appendix C for a description of why conventions and guidelines are important and for a list of conventions and guidelines that may be applicable to a submissions.

- **Mandatory versus optional interfaces**

  Mandatory interfaces that those that are required to be supported by compliant implementations as distinguished from optional interfaces that a particular implementation may or may not support. Optional interfaces need not be supported for an implementation to be said to be compliant. Complex optional interface schemes are not encouraged.

- **Constraints on object behavior**

  Constraints on object behavior over and above those defined by the object's interface semantics (a.k.a. "qualities of service") that implementations must take into account.

- **Integration with future OMG specifications**

  How the specification can evolve to integrate with relevant future specifications.

4.3 General Non-technical Requirements on Submissions

Your organization must be an OMG Corporate Member and must provide a statement about your willingness to comply with the OMG's requirements (e.g., your willingness to make the proposed technology commercially available) in your Letter of Intent (see Section 6.1).

Submissions must include a "proof of concept" statement, explaining how the submitted specifications have been demonstrated to be technically viable.

It is important for the Task Force to understand the technical viability of an OMG submission during the evaluation process. The technical viability has a lot to do with the state of development of the technology submitted. This is not the same as commercial availability, which is an OMG Board of Directors consideration. Proof of concept statements can contain any information deemed relevant by the submitter, for example:

> "This specification has completed the design phase and is the process of being prototyped."
>
> "An implementation of this specification has been in beta-test for 4 months."
>
> "A named product (with a specified customer base) is a realization of this specification."

It will also be incumbent upon the finalists to demonstrate to the CFTF's satisfaction the technical viability of their submission.

Any technology adopted by the OMG must be commercially available from a Corporate Member. A statement to the OMG Board of Directors describing how the submission meets this commercial availability requirement is required on adoption of a submitted specification.

5. COMMON FACILITIES RFP-4 ITEMS

This RFP calls for submissions to support two related Common Facilities:

- **Common business objects**

  Objects representing those business semantics that can be shown to be common across most businesses.

- **Business Object Facility**

  The infrastructure (application architecture, services, etc...) required to support business objects operating as cooperative application components in a distributed object environment.

This RFP recognizes the two objectives of Common Business Objects and the Business Object Facility and that...

- responses to these objectives may be interdependent
- it may not be possible to define a precise boundary between the two
- a smooth transition - with as few transformations as possible - from analysis and design to implementation to units of delivery is highly desirable.

Your submission may address either one or both RFP items.

5.1 Business objects and application components

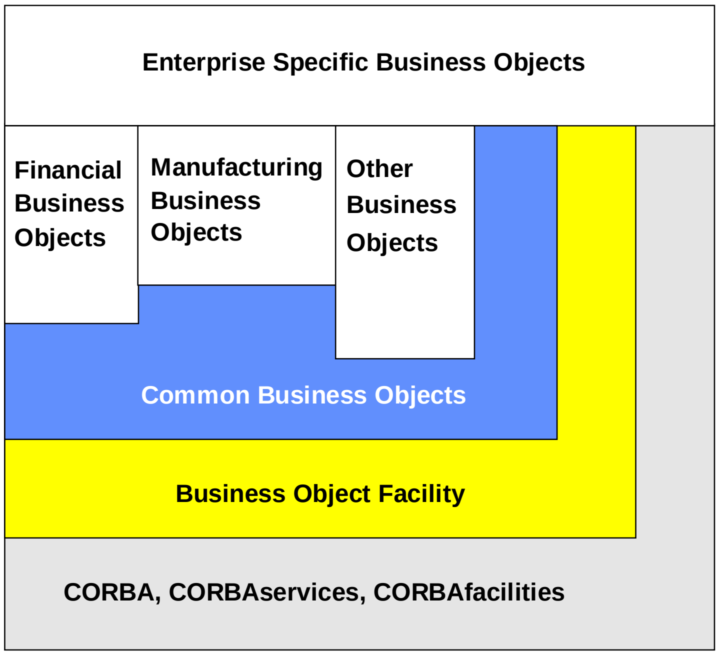
A business object is defined as a representation of a thing active in the business domain, including at least its business name and definition, attributes, behavior, relationships, rules, policies and constraints. A business object may represent, for example, a person, place, event, business process or concept. Typical examples of business objects are: employee, product, invoice and payment.

The business object abstraction, which models a real world thing, is implemented by one or more objects in the information system. Each such object in the information system is a application component of that information system and must be supported by a technology infrastructure. A business object is a specialization of the general concept of an object as it applies to representing business concerns.

The technological infrastructure to support "plug and play" business application components is the **Business Object Facility**. This infrastructure must have the semantics to support business objects (common or those specific to an enterprise).

Therefore, "**Common Business Objects**" represent the business semantics that may be found in any enterprise, such as the concept of a "purchase", and the **Business Object Facility** represents the information system semantics required to support business objects.

As shown in the diagram below business objects can be specialized to meet the needs of specific domains and enterprises.


Enterprise Specific Business Objects
Financial Business Objects
Manufacturing Business Objects
Other Business Objects
Common Business Objects
Business Object Facility
CORBA, CORBAservices, CORBAfacilities

5.2 Common business objects

Common business objects are those objects that represent things and concepts so common to every business that some of the semantics of those things and concepts can be standardized.

For example, the concept of a “trade” must involve at least two legal entities and some exchange of goods or services - this is the essence of every purchase or sale. Specific objects representing these concepts are common business objects. Another example is the common concept of a “document”.

5.2.1 Description and requirements

Common business objects may be expressed abstractly, such as: financial-transaction, legal-entity and resource, or, may be expressed at a higher level such as: sale, customer and machine-tool. It is left up to the submitter to demonstrate the level to which business semantics are common.

Submitters should specify common business objects in terms of the following:

- Names of common business objects
- Attributes of common business objects
- Relationships between common business objects
- Constraints on common business objects and relationships
- Business rules associated with common business objects and relationships
- Interfaces and methods of common business objects specified in OMG IDL

.In addition to the definition of these objects submitters are asked to provide information to explain:

- The ease of developing domain specific business objects from these common business objects
- The interoperability across sets of domain specific objects (verticals)
- The extensibility of common business objects

If the submitter is separating common business object semantics from the supporting infrastructure as suggested in the business object facility then infrastructural support does not need to be specified for each common business object. If however, the submitter considers infrastructural support integrated with the concept of common business objects then all infrastructure related questions listed under the business object facility should be included in the response for common business objects.

5.3 Business Object Facility

Components, by definition do not stand alone, they only provide value within some well defined architecture, specification or framework. By their very nature components must have a way of being part of something larger.

For example, consider the boards that go inside a “PC” computer system: Each board provides some useful function and is a component of the whole system. Before the PC bus and other standard computer bus architectures, boards were all different, each manufacturer had their own and there was no open market in such components. Once the *standards and infrastructure* were in place for this type of component the marketplace flourished. Components require standards for interaction at some level of detail. Just being able to wire boards together is insufficient; pinouts, timing, voltages and protocols all have to exist for these components to work together. Besides the standards, an infrastructure is required to make them work (the PC bus and the BIOS).

Application components need the same kind of infrastructure and standards to make them work and it is to this end that we need the **Business Object Facility.** Consider also that below the level of the PC board component is the integrated circuit (IC) component, these are like the lower level software technology components that are used to implement the higher level business objects.

The **Business Object Facility** provides the interoperable infrastructure and/or framework required to support business objects as application components (common, domain specific or enterprise specific) of a business solution. It provides the interfaces and protocols for the application components to collaborate and the infrastructure to support them as “Plug & Play” application components.

The lower level technology interfaces that now exist in CORBA can be combined for a particular purpose and given more specific behaviors as a “profile”. The Business Object Facility has just such a purpose, it can use the existing CORBA interfaces in well defined ways to provide an interoperability infrastructure, in other words to provide a profile for application components. Other interfaces may be required as well, ones that are specific to the needs of business application components.

Components are defined in terms of their interfaces so that differing implementations of each application component can exist for a particular purpose. In this way application components can be used, integrated, combined, replaced and changed without effecting the overall integrity of the system. Finally, application components should be easy to build, use and assemble into functional applications by the typical applications developer.

Submissions to this item should not specify this facility in terms of implementation, but in terms of interfaces and external behaviors. Differing implementations of the specification should be able to coexist so as to represent different engineering tradeoffs.

5.3.1 Facility Description and Requirements

The description of the **Business Object Facility** is expected to include details of the following interfaces:

- Interface and protocol between application components.
- Interface and protocol between application components and any non-application components. Other components may represent the user interface, agents, world wide web, palmtop computers or any other system that may use application components.
- Interface and protocol between the business object facility and the underlying CORBA infrastructure.

Submitters should also explain:

- Ways in which application components can be used to form applications
- How business objects relate to application components.
- How application components are built, including several examples.
- Constraints put on application components by the facility.
- How business policies and workflow can be combined with application components.
- How dependencies between application components and rules are handled.
- How application components collaborate, including coordinated transactions.
- Application component boundaries; what differentiates application components from any other object?

5.4 Relationship with other Common Facilities, OMG Object Services, CORBA and OMG Object Model

Submitters need to explain how the business object facility, application components and business objects relate to the following:

5.4.1 CORBA

Submitters should explain if and how application components and business objects are specializations and/or aggregations of the basic CORBA object types.

5.4.2 CORBAServices

Business objects and application components and the business object facility are expected to use object services such as:

- Persistent object service
- Transaction service
- Relationship service
- Trading service
- Query service
- Event service

Submitters should identify the consistent set of object services required to support application components and any profile of their use. Particular attention should be given to any specializations of the interfaces of these services.

5.4.3 Other CORBAFacilities

Application components, business objects and the business object facility are expected to exploit most of the Common Facilities such as:

- Presentation facility that could be used to present application components to users.
- Information management that could be used to provide persistent storage for application components
- Systems management that could be used to manage application components
- Task management: that could be used to access application components and to integrate them with other components and/or objects to form applications

Submitters should identify the interfaces provided by their application components and business objects to integrate with these facilities.

5.5 Related References

- *ICL response to CFTF RFI #2, Financial services and accounting facilities,* OMG Document 95-8-28
- *Data Access Corp. response to CFTF RFI #2, Financial services and accounting facilities,* OMG Document 95-8-27
- *BOMSIG business application architecture white paper,* OMG Document 95-4-1
- Business Objects, Oliver Sims McGraw-Hill 0-07-707957-4
- Object Advantage, Ivar Jacobson Addison-Wesley 0-201-42289-1
- The Object-Oriented Enterprise Rob Mattison McGraw Hill
- Business Engineering with object technology, David Taylor Wiley, 1995
- The OMA Guide OMG Wiley

<u>5.6 Requirements submissions must address</u>

Section 5.6 details the requirements that a submission <u>must</u> address. A response is required under each subsection.

Section 5.7 details additional requirements that a submission <u>may choose</u> to address. These requirements are <u>optional but are, nevertheless considered to be important,</u> and responses to them will affect the evaluation of the quality and scope of submissions.

Section 5.8 identifies a number of technical issues that are considered as important to the development and deployment of business objects. While some of these issues are covered by existing facilities or interfaces, the submission should explain how each of these issues is handled by new or existing technology and how the parts of the technology are used and work together to meet the needs expressed in this RFP. Any technical criteria that the submitter does not feel applies should be so identified.

5.6.1 Interoperability

It is a primary goal of this RFP that application components representing business objects be interoperable regardless of the implementation of the framework, implementation of business semantics, engineering methodology, source language, operating system, human language or business domain of the application components.

Submissions must address how interoperability is maintained at a high semantic level between independently developed business objects.

5.6.2 Separation of technology issues

Business objects and application components should be independent of the technology required to implement or use them. Submissions must address how issues such as model, process, persistence and interface are abstracted so as to provide independence from each other.

5.6.3 Extensibility of business objects

Business objects and application components are expected to form the base of distributed object applications but will certainly need to be extended for use in any enterprise. Submissions must address how business objects and application components are extended through inheritance, delegation, configuration or other mechanisms.

5.6.4 Reusability

Submissions must address how business objects and/or application components promote reusability and/or an open market in business objects.

5.6.5 Scalability

Submissions must address the scalability of business objects and/or application components.

5.6.6 Ease of development and deployment

Submissions must address how common business objects and/or application components ease the development and use of business objects and their deployment in information systems from the perspective of the business system developer and user.

5.6.7 Application integration

Submissions must address how business objects and/or application components are integrated to form an application.

5.6.8 Security

Submissions must address how business objects and/or application components support or provide for security of information and enforcement of security policy.

5.7 Requirements submissions may address

5.7.1 How business objects implement the business model

Submissions may address how business objects and application components implement concepts that exist in the business domain. Explain how the business concept of, for example, a sale may be translated to a business object representing that sale.

5.7.2 Legacy applications

Submissions may address how business objects and/or application components may be integrated with or implemented using legacy applications.

5.7.3 Flexibility and longevity

Submissions may address how business objects and/or application components enhance the flexibility, maintainability or longevity of the application implementation.

5.7.4 Generality and desktop integration

Submissions may address how business objects and/or application components enable integration and compatibility with other standards and defacto standards such as Microsoft OLE-II and Windows(r). Submissions should state whether infrastructures other than CORBA may be used to implement application components and business objects.

5.7.5 Proof of commonality

Submissions may address how it has been shown that the concepts and/or facilities proposed are common among business domains and enterprises.

5.7.6 Specification of business objects and metadata

It is recognized that OMG IDL may not be sufficient to represent the semantics of business objects. True interoperability may require exposing information as to what the interface means and requires.

Submissions may address how information about business objects, other than their OMG IDL interface, is represented. This may include: constraints, rules, roles, policies, relationships, states, attributes, visibility, dependencies, protocols, pre and post conditions, error conditions, warning conditions or events.

Such information may have a textual form (Such as **OMG IDL extensions**) and/or an executable form, such as **metadata**.

5.7.7 Multilingual use

Because the users of CBOs would be domain specialists, it would be natural that object names are borrowed from the terms found in the system specifications written in their mother tongue.

Submissions may address how multilingual issues are addressed.

5.8 Technical Criteria

The following technical issues have been identified as being important to the development and deployment of business objects. While some of these issues are covered by existing facilities or interfaces, the submission should explain how each of these issues is handled by new or existing technology and how the parts of the technology are used and work together to meet the needs expressed in this RFP. Any technical criteria that the submitter does not feel applies should be so identified.

5.8.1 Change and Event Notification

Change notification is a service that should be available from any business object without explicit programming by the application developer. An object must be able to request notification of events or changes in any other object or class of object and later cancel the request with standard message forms. The notification message must also be in a standard form. Notification should be provided for any change that occurs in a specified aspect of a target object. This service should be available in a consistent fashion across heterogeneous environments.

5.8.2 Active Views

As models are developed and implemented, their size and complexity will grow. There is a need to be able to implement solutions to particular problems using simplified abstractions of the enterprise model. Views, as intended here, are much like traditional database views where a view is a simplified representation that excludes some attributes, relationships and operations, and also flattens a complex structure. In addition, views can provide aliases and signature conversions for specific methods. Consequently, views may be used to insulate individual applications from evolutionary changes in the enterprise model. An active view is linked to the underlying objects such that attributes and relationships represented in the view are kept current with the states of the underlying objects.

5.8.3 Transparent Persistence

Application programmers should not be concerned about persistence of objects or their identity. If an object is required to be persistent, then the business object should be able to save its state in external storage at appropriate times and be able to recover its state to be consistent with the rest of the system if a system failure should occur. These mechanisms, which may use the CORBA persistence facility, must assure appropriate storage and recoverability across heterogeneous environments.

5.8.4 Search mechanism

A flexible search mechanism should permit locating qualified objects based on logical versus implementation considerations.

5.8.5 Backout

Backout is the ability to restore an earlier state. This facility is necessary for termination of transactions, it may be required to support searches, violation of business rules or exception resolution, and it is useful to enable a user to undo a mistake. It must operate across heterogeneous environments so that related or dependent changes will be backed out in a consistent fashion.

5.8.6 Concurrency and Serialization

When concurrent transactions share objects, changes applied by one transaction process may cause the processing of another transaction to be invalid. Serialization assures that the actions of competing transactions are controlled in such a way that the result is equivalent to the transactions being executed one after another instead of concurrently. Serialization must be achieved across heterogeneous environments to the extent that transactions traverse multiple environments. Standard mechanisms are required for identity of transactions, detection of deadlocks and suspension and resumption of processes.

5.8.7 Nested Transactions

It should be possible to implement business transactions that are invoked by other business transactions and be assured that the serialization, recoverability, backout and commitment of the associated transactions are handled properly. To achieve this, the management of transactions must be consistent and coordinated across heterogeneous environments.

5.8.8 Referential Integrity and Garbage Collection

When objects are referenced across environments, there is a risk that the reference may become invalid if the object is deleted. Likewise, if there many external references to objects in an environment and the references become obsolete, the environment may become overburdened with objects that appear to be in use but are not. A consistent mechanism that minimizes programmer responsibility must be defined to determine objects no longer referenced and remove them from the execution environment. Likewise, a consistent mechanism to handle references to deleted objects may be required.

5.8.9 Encapsulated attributes and relationships

The concept of encapsulation can be used to define a standard protocol for access to attributes and relationships that conceals their implementations (reification). In addition to promoting interface consistency, encapsulation of these aspects can help ensure the integrity of other features such as change notification, backout and serialization.

5.8.10 Constraints, rules and policies

A mechanism to support the specification of constraints, including rules and policies, and satisfy these constraints is needed. Constraints are essential for assuring the integrity of models according to natural as well as business rules. Mechanisms must exist to ensure that the system never commits an invalid state and that dependencies are properly propagated.

5.8.11 Relationship Management

Relationships between objects may be complementary such that each has access to the identity of the other. In some manner, this requires complementary references which must be consistently maintained. The integrity of these relationships should be maintained without explicit programming by the application developer. Operations on relationships should conform to a standard protocol.

5.8.12 External name management

External names, as used here, are identifiers used in the real world. A single business concept may have several names which apply in different contexts. A external name management facility would provide a standard protocol for accessing objects in a distributed environment using names. The facility must model the contexts and classifications used in the real world so that a name entered by a human may be properly interpreted and the appropriate attributes will be used when referring to an object in a report or display.

5.8.13 Exception/Fault Resolution

When exceptions or faults occur in a distributed heterogeneous environment, it is not acceptable for the entire environment to come to a halt. An exception/fault resolution mechanism is needed to support analysis of the state of processes and to determine at what point they might be restarted. Standard mechanisms are required to report, analyze, reconcile states and restart processes when failures occur.

Exceptions should be able to handle messaging to objects that no longer exist in a predictable manner.

5.8.14 Configuration Management

In a large, distributed, heterogeneous environment, it will be necessary to upgrade business objects (and other components) without interrupting the operation of the networked system. This will include changes to the implementations of active objects and relationships which must be coordinated across heterogeneous environments. A business object should be configurable after it has been delivered and deployed and should be able to be subclassed and used by other objects in unanticipated ways without recourse to development. Objects should cooperate with other objects in ways that do not bind them to a specific location or implementation.

5.8.15 Composite Object Bounds

A composite object is a structure that represents a complex application concept with a principal object and associated application component objects. Typically, if the principal object is deleted, the application component objects will no longer be meaningful. Sometimes, however, a composite object may "contain" other composite objects as components. Generally, the components of a composite object will be saved or moved with the principle object and the principle object may have versions which incorporate different component objects. The problem is to define the bounds of composite objects and facilitate versioning, storage and transport of the composite as a unit consistently across heterogeneous environments.

5.8.16 External Resource Representation

External resources, typically input/output devices, must be represented as objects with consistent protocols and straightforward operations. These should be accessible across heterogeneous environments.

5.8.17 User Attributes and Preferences

User attributes and preferences should be represented in a consistent fashion so that an application can respond accordingly. This should include the user's domains of interest for defaults and warnings, and it should include the user's experience/proficiency level as a basis for providing assistance.

5.8.18 Textual Representation

Object structures should be translatable to a textual form for storage, transport and editing. The textual form must represent circularity and terminate at the bounds of complex structures. The textual form should be language and environment independent to enable exchange of object state between diverse environments.

5.8.19 Executable Object Expressions

There is a need to be able to express operations on objects in the form of objects. Applications need to be able to create and sometimes analyze executable expressions and pass them from one computing environment to another. These may be used for rules or constraints, or they may be used in executable blocks for exception handling, iteration, or event processing. The language used to specify or display these expressions should be independent of the particular programming environment.

5.8.20 Loose binding

Interoperability means integrating separately-developed binaries in the run-time environment. This will probably require loose binding of method and/or message data between business objects. An important aspect of such loose binding would be significant levels of version insensitivity, location independence and implementation independence.

5.8.21 Instance Specialization

Instance specialization is the capability to add methods and state to an object instance to incorporate unique capabilities. The extensions may be temporary for solution of a unique problem, or they may be persistent for a continuing requirement. They may be used to support analysis or reasoning about the object, or to evolve the representation of a concept as it is discovered in the real world.

5.8.22 Reflection

Reflection is the ability of a system to analyze and report its state and activities and to alter its state and activities based on this analysis. This requires the ability to examine meta-information, call-paths and executable expressions. The ability should be ad hoc so that active processes can be compiled for performance but switched to interpretation for reflection. Such capabilities are important for such activities as exception resolution, automatic code generation, interactive query, machine learning, performance tuning and adaptation to particular users. Reflection should also support tools for analysis of system performance, debugging, design of tests and failure mode analysis.

5.8.23 External interfaces

External systems, such as user interfaces or desktop programs should have a known and consistent interface to any business object. All constraints, dependencies and rules should be enforced regardless of the source of any change to any object.

6. INSTRUCTIONS FOR RESPONDING

6.1 General

Companies responding to this RFP shall designate a single contact within that company for receipt of all subsequent information regarding this RFP and RFP submissions. The name of this contact will be made available to all OMG members.

As a forewarning to organizations who intend to respond to this Common Facilities RFP, please note that responding to an RFP requires:

- A Letter of Intent signed by an officer of your organization signifying your intent to respond to the RFP and a statement of your organization's willingness to comply with the OMG's requirements (e.g., a statement describing your willingness to meet the commercial availability requirements). For a definition of commercial availability see Appendix A.
- The submission of the specification of technology as described in this RFP.

Documentation submitted in response to this RFP will be distributed to all of the members of the CFTF, and will be available to all OMG members.

The following points should also be noted:

- Independent submissions are solicited for each separate item in the RFP.

A company may respond to any or all items. Each item will however be evaluated independently by the CFTF. Thus submissions that do not present clearly separable proposals for multiple items may be at a disadvantage

Submissions to each individual item must be "complete"

A submission must provide complete specifications for the item and address all the relevant requirements stated in the RFP.

It is anticipated that RFP items may not map cleanly onto technology described in some submissions. Typically:

- A given technology may provide a solution to an RFP item and, in addition, provide *(or its design may necessitate*) an integrated approach to one or more specifications not included in the RFP. Submitters are invited to provide information on these additional specifications and give a rationale as to why they should also be considered for adoption at the same time. However, information on these additional specifications should be clearly distinguished.

Submitters may provide alternative definitions, categorizations, and groupings so long as the rationale for doing so is clearly stated. Equally, Submitters may provide an alternative model or framework for how business objects are provided for within the OMA architecture if there are compelling technological reasons for a different approach.

6.2 Format of RFP Responses

The following outline describes what your submission should consist of and is offered to assist in the development of your submission. Responses should follow the following template and include specific references to supporting documents as necessary. Submitters should realize that submissions that are concise and easy to read will inevitably receive more consideration.

Please include an overview or guide to the material in the submission. Submitters are encouraged to confine submissions to the facilities requested. If this is not practical then Submitters must make it clear what portion of the documentation/specification pertains directly to the RFP and what portion of the materials submitted are not relevant to the RFP.

According to the Policies and Procedures of the OMG Technical Committee, proprietary and confidential material may not be included in any submission to the OMG. Responses become public documents of the OMG. If copyrighted, a waiver of copyright for unlimited duplication by the OMG is required. A limited waiver of copyright to allow OMG members to make up to fifty (50) copies of the document for OMG review purposes only is also required.

You should include, for each specification, a description using the following template.

6.2.1 SPECIFICATION DESCRIPTION

Responses to this RFP should be clear and understandable. As a guideline, we expect the following levels of description to be needed to explain a specification. Submissions without this level of description will be at a disadvantage when being evaluated by the CFTF.

6.2.1.1 Rationale

Describe the specification and how it provides (or does not provide) the functionality identified with the phrase “should address” in the description of the facilities in section 5. This description should include the goals and objectives of the specification. In addition, if there are goals that are outside the scope of this specification, they should be specified.

6.2.1.2 Conceptual Model

For each object defined by this facility, describe that object's functionality, abstract data model and relationships to the other objects in the facility. Specify pre-conditions, post-conditions, and invariant assertions, relationships and any other information required to define the expected behavior of the objects when providing services defined in the OMG IDL Interface Description.

6.2.1.3 Interface Description: OMG IDL

Describe the signature of the interface i.e. the list of operations along with the parameters that are required. Provide a list of exceptions that may be returned by the operations. The OMG Interface Definition Language (OMG IDL) must be used to specify the signatures associated with each object. Include definitions for all factory objects as part of the specification or objects created or managed by the specification.

Describe any extensions that may be needed to OMG IDL in order to support the specification.

6.2.1.4 Interface Description: Behavior

Describe the semantics of the behavior observed as a result of using each specified operation. These descriptions may be provided in informal English. Describe any limitations in the specification's behavior.

6.2.1.5 Glossary

Because many terms do not have generally agreed upon definitions, provide a glossary of technical terms used in your submission.

6.2.2 RESOLUTION OF TECHNICAL AND NON-TECHNICAL ISSUES

Explain how the submission meets the general technical and non-technical requirements listed above in Sections 4.3 and 4.4.

Provide a discussion and rationale in response to the technical issues indicated by the phrase "should address" in the specification descriptions in Section 5.

6.2.3 SPECIFICATION DEPENDENCIES

To provide the specification described in submission, the objects identified may be designed to take advantage of CORBAFacilities or CORBAServices that are available. Identify any other objects that the described objects are composed of or take advantage of and provide an explanation of how and why these other objects are relied upon to provide the described service.

Clearly identify any relationships, actual or perceived, between the proposed technology and the components of the OMA and Common Facilities Architecture. Include relationships to facilities components: Presentation, Information Management, System Management, and Task Management (see the Common Facilities Architecture).

Identify any other technology dependencies such as other software that is presumed to exist to support the specification.

6.2.4 RELATIONSHIP TO CORBA

Describe how the specification uses CORBA features and/or identify any extension of CORBA that are required, including extensions to OMG IDL. For additional guidance on this see Section 2.4 of the Object Services Reference Model.

6.2.5 RELATIONSHIP TO OMG OBJECT MODEL

Describe how the specification conforms to the OMG Object Model and/or what new components or profiles may be needed

6.2.6 STANDARDS CONFORMANCE

In accordance with Section 3.4.17 of the OMA Guide, submissions should identify their use of relevant existing industry standards or should justify why such use is not appropriate.

In particular, submissions should identify how their provisions relate to the architecture for system distribution defined in ISO/IEC 10746, Reference Model of Open Distributed Processing (ODP).

6.2.7 OTHER INFORMATION

Provide any other materials deemed to be relevant to the evaluation process.

6.3 How to Submit

OMG requests that electronic versions of your submission (typically ASCII, FrameMaker, Word, WordPerfect, Acrobat PDF or PostScript format) be sent to the Common Facilities Technology Desk (cftf@omg.org) at OMG by the submission deadline.

In addition, 50 hard-copies of the submission should be sent to the Common Facilities Technology Desk at OMG within three working days after the submission deadline and 50 hard-copies should be made available to attendees of the CFTF meeting immediately following the submission deadline. Your organization should be prepared to handle requests for additional copies of your submission.

Responses to this RFP (and other communication regarding this RFP) should be sent to:

Common Facilities Technology Desk
Object Management Group Inc.
Framingham Corporate Center
492 Old Connecticut Path
Framingham, MA 01701-4568
USA

Responses to this RFP must be received at OMG no later than 5:00 PM EST (22:00 UTC) on the date defined in section 7.2 below. The outside of packages/envelopes containing submissions or any other communication regarding this RFP should be clearly marked "COMMON FACILITIES RFP-4 SUBMISSION".

6.4 Reimbursements

The OMG will not reimburse Submitters for any costs in conjunction with their submissions to this RFP.

7. SUBMISSION REVIEW PROCESS AND SCHEDULE

The OMG Common Facilities Task Force (CFTF) will conduct the initial evaluations of the RFP submissions. It will recommend selected specifications to the Technical Committee (TC) which in turn will make a formal recommendation to the Board of Directors (BOD) to adopt the specifications. The CFTF is composed of interested OMG member companies and the TC is composed of representatives of all OMG member companies.

7.1 Review Process

The CFTF plans to a use a four step process for handling submissions to Common Facilities RFPs:

1. *Letter of Intent*

   A Letter of Intent (LOI) to submit will be 60 or more days before the deadline for submissions.

2. *Submissions*

   Full submissions will be due by a specific deadline.

3. *Revised Submissions*

   There will then be a 120 day preliminary evaluation by the CFTF during which companies will have the opportunity to revise and/or merge their submissions.

4. *Specification Selection*

   Finalists may be requested to make demonstrations to the CFTF. The final specification recommendation by the CFTF will take place 90 days later. The TC will then vote to recommend a specification to the BOD 3 to 5 weeks later.

All the above dates are approximate. For more detailed information on the OMG’s Common Facilities adoption process and plans see Chapter 5 of the *CFRM*.

7.2 CF RFP-4 Schedule

Event/*Activity*	**Date**
Preparation of RFP by CFTF	
CFTF votes to issue RFP	11 January 1996
Review by TC ("Three week rule")	
TC votes to issue RFP	11 January 1996
Preparation of submissions	
LOI to submit to RFP due	15 Aug 1996
Preparation of submissions	
Submissions due	15 Oct 1996
Preliminary evaluations by CFTF and	
preparation of revised submissions	
Revised submissions due (As necessary)	15 Feb 1997
Specification selection by CFTF	
CFTF votes to select specifications	March 1997
Review by TC ("Three week rule")	
TC votes to recommend specifications	March 1997
BOD votes to adopt specifications	May 1997

APPENDIX A: Commercial Availability Requirements

A.1 Commercial Availability

For technology to be accepted and adopted by the OMG board of directors (see OMG policy titled “Policy Document on Adoption of Specifications - 2/12/90”) it must be commercially available within 12 months of adoption. This is required for proof of concept and expedient implementation of actual product and licensing procedures. Commercially available is delineated as:

- Technology that has been publicly announced as a product or embodied within another product.
- Technology that is of production/manufacturing quality, has cleared a process of product shipment authorization, and can be demonstrated at OMG request (including installation, documentation, specification, and support,). Demonstrations must be available following RFP presentation to the OMG Technical Committee.
- Technology that can be referenced by at least two consumers (customers) of the technology.

This must be accompanied by a letter of authorization by an officer of at least one of the companies proposing the technology.

A.2 Terms and Conditions

The OMG Business Committee has produced a document entitled “OMG Policy on Adoption of Specifications.” When reviewing submissions to each RFP, the specific items that the OMG Business Committee will be considering during the selection process are outlined below:

- The optimization of interoperability and portability goals across multiple platforms.
- Commitment by the proposed technology supplier to make the implementation available on commercially reasonable terms, applied in a non discriminatory fashion.
- Submission of a proposed price schedule and Standard License Agreement.
- A preferred, but not required, method for achieving multi-platform interoperability is source code licensing. Please include any provisions as such.
- Assurance that the results in the duplication of the “look and feel” of any aspects of such proponents implementations from specifications will not result in infringement or any obligation to pay royalties.
- Plans for future revisions, enhancements, maintenance and support.

- Agreement to grant the OMG a world-wide copyright including the right to copy and distribute the adopted interface specification at no cost to OMG. Implementations or instantiations of the specifications are owned by the developer.
- Upon OMG's acceptance of the sponsoring company's interfaces, the sponsoring company agrees to provide all documentation in an OMG prescribed format and in OMG endorsed terminology.

APPENDIX B: End User Special Interest Group Requirements

Under the auspices of the END-USER TRACK, the OMG End User SIG will review all requirements in the body of their responses to the RFP, then a minimum response to this appendix should be a separate part of the submission containing a list of references against each numbered requirement indicating where each point is addressed. On the basis of the End-User Track technology evaluation against these requirements, the End-User SIG will make independent recommendations to the OMG Board of Directors on technology adoption. Below is the list of End-User Requirements which will be attached to all RFPs.

In this document, interface specification means IDL syntax and object behavioral and sequencing semantics.

0 Completeness Requirement

Submissions for consideration for adoption as OMG adopted technology shall provide complete and precise semantic definitions of interfaces.

1 Primary Requirements

1.1 Interoperability

The OMA Guide defines Interoperability as the ability to exchange requests using the ORB in conformance with the Object Management Architecture Guide. This means that a client is able to invoke operations on an object implementation, where all of the following apply:

- the client resides on platform A and is constructed with vendor X's OMA-compliant implementation;
- the object implementation resides on platform B and is constructed with vendor Y's OMA-compliant implementation;
- each is developed exclusively from the interface specification.

Universal interoperability with all other OMA-compliant ORBs directly out of the box is a requirement.

1.2 Portability

It must be possible to rehost an OMA-compliant client or object implementation on an alternative OMA-compliant environment, with no changes to the source code.

1.3 Substitutability

It must be possible to replace an implementation of an interface specification with another implementation without requiring changes to any client of that interface.

1.4 LifeCycle

Specifications shall identify the degree to which the LifeCycleObject interface is supported: mandatory, optional or not applicable.

2. Administrative Requirements

Interface specifications shall address the administration of objects, whether they be services, facilities, or ORB features, to include the following:

2.1 Installation/De-installation

Standard interfaces shall be provided for automated remote installation and de-installation of the specified ORB feature, common facility, or object service.

2.2 Upgrade

Standard interfaces shall be provided for automated remote upgrade of the specified ORB feature, common facility, or object service.

2.3 Performance Management

In order to support the management of large, heterogeneous OMA- compliant environments, standard interfaces shall be defined which provide for the querying and tuning of performance-critical resources.

3. Additional Requirements

3.1 Testing and Problem Determination / Resolution

Interface specifications shall address testing and problem determination/resolution in OMA-compliant environments in a uniform way across objects.

The interface specification shall specify a unique exception code for each possible error condition.

There shall be an interface to determine the state of every object.

There should be an interface to report the history of object invocations.

3.2 Version Compatibility

Extension of an interface specification with a new version shall preserve substitutability for all of the original operation.

APPENDIX C: Conventions and Guidelines

An important aspect of defining Common Facilities is reaching agreement on overall style and conventions. Failure to select and follow such conventions can cause problems ranging from inconvenience to lack of interoperability. For example, lack of naming conventions for operations can make programming error-prone, whereas disagreement on the inheritance hierarchy can make some facilities unusable by some components.

Some issues that must be resolved before interfaces can be defined include:

- **How inheritance should be used in the interfaces.**

  The interfaces of Common Facilities should take full advantage of multiple inheritance. Each specification should define a self-contained interface. A particular object implementation might support several different interfaces, and it can use multiple inheritance to specify the complete set of operations it implements. Clients may choose to use only part of the behavior of an object, and can view the object with just the inheritance they need.

- **Conventions for exceptions and exception parameters.**

  The CORBA specification describes a multiple termination model for operations. That is, for each request of an operation, one of a set of terminations will be returned. The interface designer designates one of the terminations as the “normal” termination, in that the outcome of the request is returned via the return parameter and out/inout parameters in the operation invocation. The interface designer may also define a set of operation-specific additional terminations via exception declarations and the syntactic association of these terminations with the operation; each operation may also return any of the set of standard terminations defined in the CORBA specification. Each of the operation-specific terminations may return additional, termination-specific return results.

  Given the availability of this specification in OMG IDL, and that the specification is mapped to programming languages, exceptions/terminations should be used extensively to define alternative outcome from operation requests. This provides enhanced documentation in the OMG IDL specifications themselves, and leads to improved error detection by clients of operations so defined, especially if the client is programmed in a language that has a natural mapping from terminations into language-specific exception handling.

- **Assumptions about storage usage for local and parameter objects.**

CORBA puts no constraints on a specification's private use of dynamically allocated memory. However, a Common Facility must coordinate with the ORB to reclaim parameter memory - memory whose contents are returned to the client. Common Facilities should standardize the method of allocation and deallocation of this memory.

- **OMG IDL naming conventions for interfaces, types, operations, attributes, etc.**

Programmers will find Common Facilities from different providers a lot easier to use if there are conventions for the various components of an interface definition. The following conventions are suggested. They are the same as those used in the CORBA specification:

Interfaces and Types. Tokens representing interface and type names are capitalized. If the name of an interface or type consists of multiple words, each word is capitalized.

```
interface Foo {
        struct FooData {
                long value;
                float data;
        };
};
```

Operations and Attributes. Tokens representing operations, attributes, formal operation parameters, struct and exception member names, and union branches are all lower-case. If the token consists of multiple words, the words are separated by underscores (_).

```
interface Bar {
        void operation1 ();
        float another_operation (in float value);
};
```

- **Access Control Model.**

It will be necessary for Common Facilities to provide for access control to the facilities they offer in a consistent manner. There should be conventions in place for controlling permissions to use Common Facilities, perhaps through the use of Access

Control Lists. The Common Facilities should reuse and specialize the Security Object Service as appropriate to provide access control capabilities.

- **Internationalization and Localization.**

  Common Facilities should present interfaces and permit implementations that support languages and conventions other than American English. This includes European and Asian languages.

  Different implementers of Common Facilities may decide to do this in different ways. For example, there may be different ways of signifying the character set of parameters in an interface specification.

The model of programs including start-up, termination, and dynamic linking.

- **The Availability of other interfaces such as POSIX threads etc.**

  To promote maximum portability and interoperability of separately developed Common Facilities, it may be necessary for OMG to encourage responders to RFPs to standardize their UNIX-based solutions on POSIX interfaces where available.

- **The approach to defining and evolving the proposed Common Facilities while providing stability of interfaces.**

  It is likely, and desirable, that the first facilities adopted by the OMG will not be the final word on the subject. As users gain experience with Common Facilities, it is certain that the facilities will evolve – for example, to add features, improve performance, or be applicable to more application domains.

  In order to minimize disruptions to applications that use the first generation Common Facilities, the best way of evolving the facilities will be to design new interfaces that inherit from the existing, adopted interfaces. Submissions should take this into account when designing their interfaces.