IIOP-domain Infrastructure for Inter-ORB Half-bridge-based Cooperation

Małgorzata Steinder, Andrzej Uszok, Krzysztof Zieliński
e-mails: {gosia, uszok, kz}@ics.agh.edu.pl
Institute of Computer Science
University of Mining & Metallurgy
Al. Mickiewicza 30, Cracow, Poland

Abstract

This paper concerns IIOP-domain infrastructure that represents a collection of selected Object Services and technique of their usage by mediated request-level Inter-ORB Bridges built around IIOP-ORB backbone. We introduce an Inter-ORB half-bridge concept and analyse its activity that is a base for identification of the Object Services required in IIOP-domain. Then, we envisage two techniques of IIOP-domain extension with the necessary Object Services. As a case study, the implementation of Interface Repository Access Server is described. [1]

Keywords: computational models, CORBA, interoperability, UNO, bridges, ORB-backbone, Object Services

1 Introduction

Interoperability of various CORBA compliant environments is still a challenging area of research. Universal Networked Objects specification [3] provides a conceptual framework within which ORB components relevant to interoperability are positioned and in which interoperability compliance points are identified. This paper concentrates mainly on ORB services, and the correspondence between them, as a key issue for interoperability achievement. ORB Services are used in ORBs and related to particular transparencies, e.g. message encoding or reference resolution, request transportation, etc.

ORB Services differ from Object Services in that they are positioned below the application level and are invoked transparently to the application code. However, many ORB Services required for implementation of Inter-ORB Bridges include components which correspond to conventional Object Services.

This paper concentrates on IIOP-domain infrastructure that represents a collection of selected Object Services and technique of their usage by mediated request-level Inter-ORB Bridges built around IIOP-ORB backbone. The proposed infrastructure results from a practical work [6] related the Inter-ORB Bridge implementation for Orbix [7] and DOME [8] which proved that IIOP-domain must be extended with several Objects Services to make Inter-ORB Bridges' cooperation possible.

A layout of the paper is as follows. First, in Section 2 Inter-ORB half-bridge concept is introduced. Next, a functional analysis of Inter-ORB Bridge activity is presented in Section 3. This section describes the generic Inter-ORB Bridge activity [6] that is a base for identification of the Object Services required in IIOP-domain. These services, that build infrastructure for IIOP-domain, are described in Section 4. Section 5 elaborates techniques of IIOP-domain extension with the necessary Object Services. Two alternative approaches are considered and compared: Object Service may be accessed from the outside of the IIOP-domain or implemented inside it. With regard to the first approach, a construction of the service-specific bridge, called Access Server, which in fact constitutes a proxy of the given Object Service located in other CORBA system, is proposed. As a case study, the implementation of Interface Repository Access Server (IRAS) is described in Section 6. The paper is ended with conclusions.

2 Inter-ORB half-bridge Concept

Interoperability can be viewed as extending transparencies to span multiple systems. Conceptually, a

[1] This work was sponsored by the Polish Research Council under grant no. 8T11C01210.

mapping or *bridging mechanism* resides at the boundary between domains, transforming requests expressed in terms of one domain's model into the model of the destination domain.

Construction of a bridge depends on where the bridge components are located: inside or outside ORB. The second option leads to application style code outside the ORB and is termed as request-level bridging. Request-level bridges which mediate through a common protocol between distinct execution environments involve components, one in each ORB, known as "half-bridges". As such a common protocol the IIOP is proposed according to the UNO specification [3]. Mediated request-level half-bridges can be built by anyone who has access to an ORB, without needing information about the internal construction of that ORB. It is a main reason why this model has been chosen for further consideration in the reported research.

In the paper [6] a framework for constructing generic request-level half-bridges has been proposed (Fig. 1). It introduces the notion of *InterORB_Proxy* which transparently represents the server from other domain in the client domain. A general principle of the proposed architecture is as follows:

1. The original request is passed to a InterORB_Proxy object in the client ORB,
2. The InterORB_Proxy object translates the request contents to a form that will be understood by the server ORB,
3. The InterORB_Proxy invokes the required operation on the apparent server object,
4. Any operation result is passed back to the client via a complementary route.

Translating request from a format of the client ORB to the format of the server ORB, InterORB_Proxy exploits functionality of a Interface Repository and Dynamic Skeleton Interface (DSI) of the client ORB and Dynamic Invocation Interface (DII) of the server ORB.

In mediated half-bridges one of the ORBs connected by InterORB_Proxy is always the IIOP-domain, which determinate minimum functionality of this environment. It is as follow:

- A mechanism for dynamic server activation and destruction should be provided,
- Object Adaptor functionality for object creation and destruction, as well as for object reference comparison should be provided,

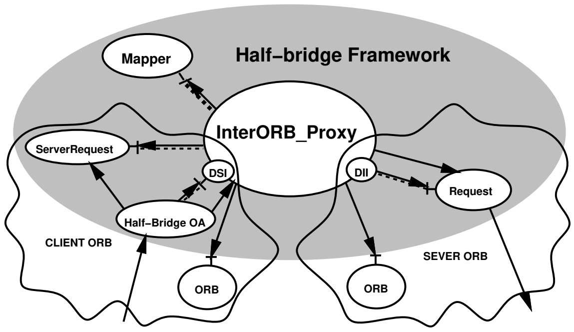

Figure 1: Half-Bridge Framework Architecture

- DSI and DII interface should be implemented,
- Interface Repository should be available,
- A mechanism for localization of services implementation should be given.

This ORB functionality of IIOP-domain should be extended with inter-ORB half-bridges management layer, that should provide inter-bridge protocol and bootstrapping mechanisms. To identify these requirements more precisely the further analysis of the inter-ORB half-bridge activity is necessary.

3 Infrastructure for half-bridge cooperation

There are two kinds of half-bridges: one on a client's side *half-bridge client (HBC)* and one on a server's side *half-bridge server (HBS)* (Fig. 2). *Half-bridge client* translates requests from the client ORB format into the IIOP protocol and *half-bridge server* receives these IIOP messages and converts them into server ORB requests. It is assumed that to initiate cooperation with a remote ORB the initiator - *half-bridge client* must possess a valid IOR (Interoperable Object Reference) [3] of an IIOP-domain object which enables him to access the server in the remote ORB. In the proposed architecture, it is a reference of InterORB_Proxy of a *half-bridge server*. It contains host name and port number of the half-bridge on the server side. Besides, it includes a reference of the object in server domain. A separete problem is creation of half-bridge process which supports such an IOR. Obtaining server's reference, creating an IOR for a half-bridge server as well as launching it may be performed in static or dynamic fashion.

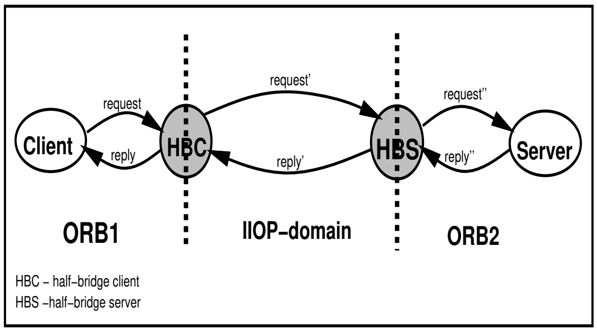

Figure 2: Halfbridge client and half-bridge server cooperation

- Staticaly – server's reference and corresponding to it IOR may be read from a configuration file. Information necessary to do that is kept in a persistent database managed by the system administrator. A half-bridge supporting the IOR can be activated by hand and supplied with all necessary informations in a command line.
- Dynamicaly – a reference of the server object as well as its location is obtained from dedicated service. Similarly, an IOR this reference is translated to and a half-bridge it represents may be created by another IIOP-domain Object Service.

The external service supplying information about servers' implementation is called, in our infrastructure, Service Repository, and corresponds to Trader Service. The service responsible for dynamic creation of half-bridges and IORs – Bridge Factory – constitutes a specialization of CORBA Activation Service and cooperates closely with Implementation Repository Service.

In order to dynamically interpret request messages half-bridges must possess access to Interface Repository.

To verify the proposed infrastructure and locate Ovject Services' usage, in the next point, analysis of half-bridge activity is presented.

3.1 Client-server Connection Establishment and Request Forwarding

InFigure 3 it has been presented how client-server connection is established and used. The steps necessary for remote request invocation are:

1. Client invokes an operation on a service which is not available in its domain, but it directs its call to the *half-bridge client* in the same domain.

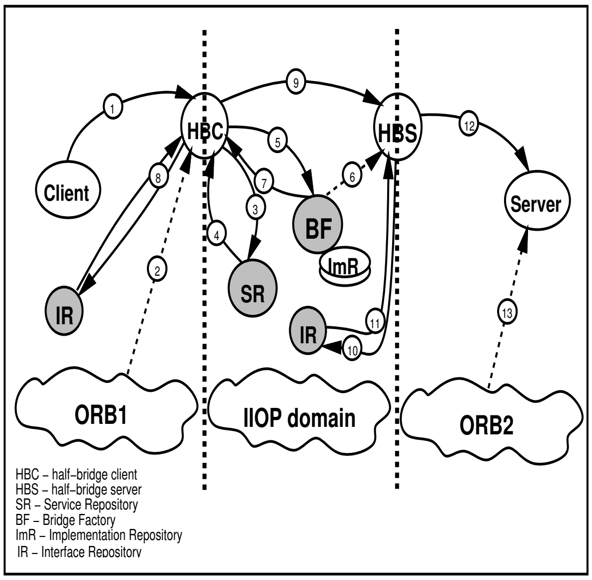

Figure 3: Establishment of client-server connection in a multi-ORB environment.

2. If the *half-bridge client* is not active and there is an Activation Service available in this ORB (e.g. Orbix) the module is bootstrapped. In ORBs which do not possess the Activation Service, *half-bridge client* must be launched by hand (e.g. DOME).
3. In the activated *half-bridge client* a mapping from a requested service name to an object reference of its representative in the IIOP domain must be set. Therefore, the *half-bridge client* contacts the Service Repository specifying a name of the sought service.
4. The Service Repository searches through its database and returns a name of a domain in which the service is implemented, a name of the host on which it is run, and a stringified reference of the server object.
5. Having been supplied with object reference, domain and host names the *half-bridge client* contacts the Bridge Factory in order to receive an IIOP reference of the representative of this object.
6. The Bridge Factory checks up if a *half-bridge server* representing the required service is not already active. If it is, the Bridge Factory returns its reference. If not, the Bridge Factory

activates the relevant half-bridge. To obtain the name of the executable, it uses the Implementation Repository Service.

7. *Half-bridge client* creates a mapping for received IIOP reference in a form of a new InterORB_Proxy object, which possesses reference of the client ORB.

8. *Half-bridge client* uses information from the Interface Repository in the DSI activity to decode the request.

9. *Half-bridge client* translates the request into IIOP and invokes it, using DII of the IIOP-domain, on the *half-bridge server*.

10. To decode the operation invocation, the *half-bridge server* has to possess a definition of the requested operation, therefore it contacts Interface Repository Service in the IIOP-domain.

11. Using DSI of the IIOP-domain, the *half-bridge server* decodes the operation invocation, translates it to a server ORB format, using this ORB DII, and forwards to the server.

12. That causes the Activation Service of this ORB to activate the server. Then, the server performs the operation. Its result is returned the same way as the invocation was made.

3.2 Reference Translation

A client which invokes object's operation may place object references as its arguments which denote other objects in client or server domain. Such references will not be understandable outside. Therefore they have to be mapped from their proprietary form to an Interoperable Object Reference (IOR) for IIOP. Similarly, on the server side the IORs have to be translated into the server ORB proprietary form. These intermediate references have to represent objects belonging to appropriate domains. In case of Inter-ORB cooperation these are, in fact, half-bridges. That means that object reference mapping entails new half-bridge creation. There are two solutions for this problem: eager and lazy reference mapping [6].

Eager reference mapping to IIOP domain In this approach when a half-bridge recognizes an object reference in a request parameter list (step 1 in Fig. 4) it uses the Bridge Factory to perform the translation (2). If necessary the Bridge Factory creates a new half-bridge (3) and returns it reference (4). This reference the partner half-bridge (5). The recipient creates a half-bridge for server environment, using the Activation Service of its domain (6). A reference of the InterORB_Proxy inside the newly created half-bridge is sent to the server in the Request message (7).

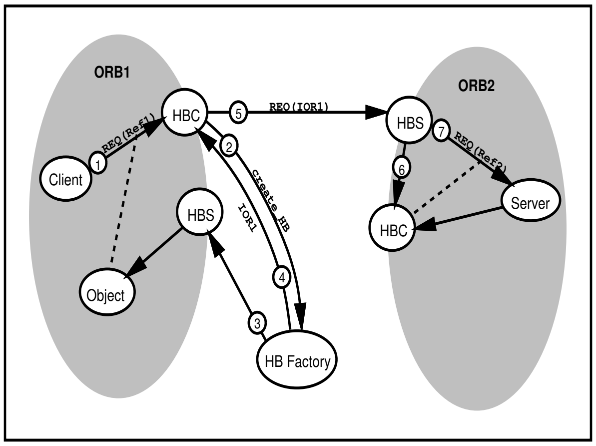

Figure 4: Egear reference mapping

Lazy reference mapping to IIOP domain In this approach, when a half-bridge recognizes an object reference in the parameter list (step 1 in Fig. 5) it does not contact the Bridge Factory so no new half-bridge is created at this moment. Instead, it extends *object_key* field of Bridge Factory IOR with the reference which is to be translated. Such a reference is forwarded to the half-bridge on a server side (2)

Mapping object references from IIOP domain A request arriving from IIOP domain to the half-bridge may contain IOR references which have to be mapped to the proprietary form of the other ORB, meaning that new half-bridge has to be created (step 6 in Fig. 4, step 3 in Fig. 5). In lazy mapping this action is postponed until the first operation on the half-bridge is called (5). The mapping introduced by the new half-bridge replaces IOR in the request (step 7 in Fig. 4, step 4 in Fig. 5). Regardless, when the new half-bridge has been created it has to establish a connection with the half-bridge represented by the IOR, which it was created for. If the encapsulated IOR denotes the eager reference such a connection has already been created. If it is a lazy reference the Bridge Factory has to be connected (6) to create an IIOP translation for the reference in its *object_key* (7).

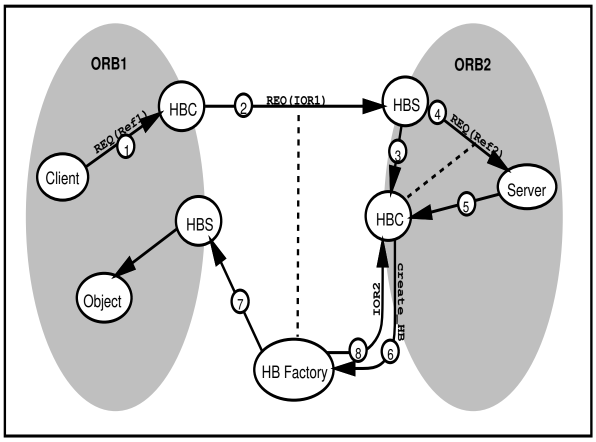

Figure 5: Lazy reference mapping

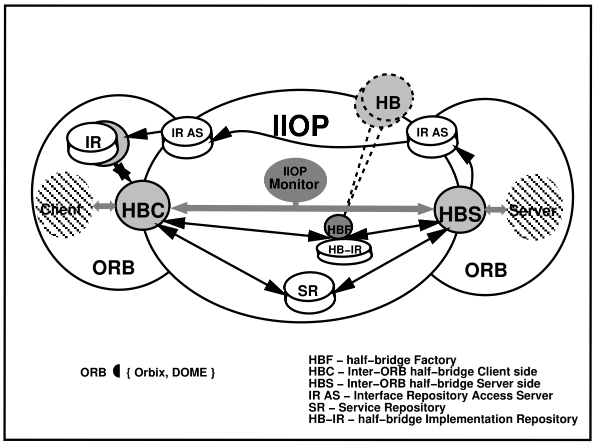

Figure 6: IIOP domain infrastructure for inter-ORB cooperation

4 IIOP-domain Object Services

According to the presented in the previous section analysis the IIOP-domain infrastructure should consists the following Object Services that are specializations of generic CORBA services adapted to half-bridge cooperation needs (Fig. 6):

- Service Repository (related to CORBA Trader Service),
- Bridge Factory (related to CORBA Object Installation and Activation Service),
- Half-bridge Implementation Repository (related to CORBA Implementation Repository Service),
- Interface Repository (related to CORBA Interface Repository Service).

These services will be considered in more details in the following subsections. Furthermore, two other services related half-bridges cooperation are proposed: Half-bridge Management and Monitoring Service and Interface Repository Consistency Service.

4.1 Service Repository

Information about available services does not cross domain boundary. To enable using these services from the outside of the domain, this information must be made available. As the IIOP domain constitutes a backbone accessible from all other domains, a database – Service Repository has had to be constructed in it. It contains information about available implementations for a given service. This information consists of a domain name in which the service is implemented and an object key of a service provider, as well as a host name on which server application is run. The Service Repository enables registration of new services, withdrawing them and browsing through its contents. At this moment it has been assumed that space of services names is global and names are unique in it. Functionality of the *Service Repository* belongs to CORBA Trader Service functionality. Since there is no standard for this service in CORBA, yet, the ISO ODP Trading Function [5] has been used as an inspiration for Service Repository interface definition. Service Repository interface offers three operations that correspond to the following functions defined by *Trading Service Interface*:

- *importOffer()* – information specified as arguments contains: service name and service offer property of interest – domain and host name. On return a list of requested service offers, which include object key of service provider and values for properties of interest specified as argument, is given.
- *exportOffer()* – arguments contain: exported service name and values of offer properties – domain and host name. On return a unique identifier of the offer in the Service Repository is specified.
- *withdrawOffer()* – the function deletes the offer identified by a unique identifier from the Service Repository.

- *discribeOffer()* – the function returns all information about the specified offer.

4.2 Bridge Factory

In the IIOP domain a CORBA Object Installation and Activation Service – a mechanism of dynamic server activation does not exits. It is, however, necessary if the infrastructure of half-bridge cooperation is to be created in dynamic fashion. An implementation of such a service for the IIOP domain is Bridge Factory. It is a kind of deamon, residing on every host of the IIOP-domain. It has the following interface:

```
interface Bridge_Factory{
  sequence<octet> create_HB(
                string domain, sequence<octet> key);
};
```

where `domain` denotes a name of a ORB to which a half-bridge will pass on requests from the IIOP domain, and `key` contains a reference of an object in the specified domain which will be represented by the created half-bridge. The operation returns a mapping for the object key to IIOP IOR representation. To obtain such a mapping it creates a new half-bridge for the given ORB (denoted by *domain*), which represent the object (denoted by *key*) in the IIOP domain and assigns him a port number. Using this mapping a half-bridge on the client side will be able to get through to the created half-bridge on the server side of the IIOP domain.

In order to create a half-bridge the Bridge Factory must possess an information about a name of the file, which contains executable code of it, location of this file in the file system and list of parameters. This information is kept in the Half-bridge Implementation Repository. To avoid retrieving this data from this database too often, which is a time consuming operation, once read the data is cached by the Bridge Factory.

The Bridge Factory is able to create half-bridges in too modes: unshared and shared. With the former a half-bridge may represent only one remote object. The latter case applies for multi-threaded half-bridges able to communicate with many objects at a time. At this moment only unshared half-bridges are being developed, therefore a Bridge Factory creates a new half-bridge for each new object key it receives.

4.3 Half-bridge Implementation Repository

Half-bridge Implementation Repository stores a mapping from names of domains connected by a half-bridge (one of which is always IIOP) to information about:

- name of the executable for this half-bridge,
- its location in the file system (a path name),
- parameter list,
- activation mode (shared or unshared).

Half-bridge Implementation Repository is constructed as a directory containing files in which description of every half-bridge is kept. The directory is accessible only from computers which share the file system. This solution is acceptable at this moment, however, when the IIOP infrastructure extends to span computers with separate file systems a special server will have to exist that will make this information available via IIOP protocol. Half-bridge Implementation Repository as a database requires tools for its management: addition of a new entry, deleting records, reading its contents. For example Orbix system offers such tools in the form of separate programs. Thanks to keeping the structure of the Half-bridge Implementation Repository compatible with Implementation Repository in this system these tools may be also applied to its management.

4.4 Interface Repository specification for IIOP domain

Interface Repository for IIOP domain functionality is a subset of that provided by Interface Repository specified in CORBA 1.2 [1]. The Interface Repository interface has been reduced to only these operation which are necessary for half-bridge activity:

```
interface Contained {
        struct Description {
                Identifier name;
                any        value;
        };
        Description describe();
   };

   interface Container {
        sequence<Contained> lookup_name ( ... )
   };

   interface Repository : Container {
   };

   interface InterfaceDef : Container, Contained {
        struct FullInterfaceDescription { ... };
        FullInterfaceDescription describe_interface();
   };

   interface AttributeDef : Contained {
   };
```

4.5 Half-bridge Dedicated Services

Detailed investigation and implementation of the half-bridges lead to a conclusion that some other services should be also supported by the IIOP-domain infrastructure e.g. Half-bridge Management and Monitoring Service, and Interface Repository Consistency Service. First of these services should be organized in a very similar way as already common SNMP standard. MIBs for half-bridges should be defined and used for their monitoring and management. Analogies between half-bridges and network devices are very evident that make this task conceptually simple. Interface Repository Consistency Service is much more complicated and addresses services specification and naming in various ORBs. This problem solution needs some agreements according to data types representation and global naming convention.

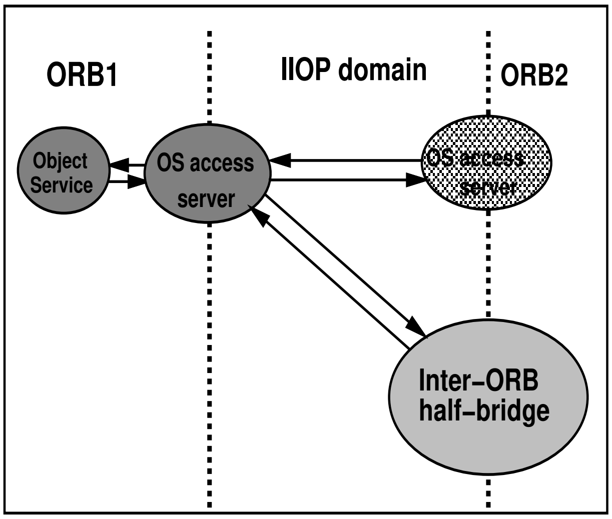

Figure 7: Access to Object Service in the IIOP domain.

5 IIOP-domain extension Techniques

In previous section a set of Objects Services necessary for half-bridge dynamic activity has been presented. They must exist in all domains which the half-bridge cooperates with. Although some of them are defined in the CORBA standard, their definition are not complete. In result, for example, different ORBs are now equipped with different Interface Repository implementations or even deprived of this mechanism. One may apply two strategies to equip the given ORB with necessary Object Services:

- Implementing the Object Service as a completely new service in a given domain from scratch.
- Using Object Service available in another ORB and accessing it via a service specific half-bridge.

The first strategy requires a lot of programming effort. It is in fact rebuilding of the existing ORB. The second approach frees from a lot of work, and is particularly attractive then Object Services provided by various ORBs are inconsistent. In the situation of lack of coordination protocols, it may resolve difficulties in multi-ORB cooperation.

Taking the second strategy one must create a service specific half-bridge – *OS Access Server* from the IIOP domain to a domain which possesses an Object Service. It is also a method for equipping other ORBs with Object Services they are deprived of via a specialized half-bridges from these domains to IIOP domain. This approach is illustrated in Fig. 7.

The proposed technique based on service specific half-bridges is general and possesses all advantages of request-level bridges. Hence, it may be used when approach based on forthcoming multiprotocol servers [3] would not be applicable.

A good example of application of this strategy is IR Access Server. It makes Interface Repository functionality existing in another domain available in the IIOP-domain. It has to support a limited number of interfaces, i.e. those which are defined in the CORBA standard for Interface Repository. In result such a half-bridge is interface specific, so it does not require usage of DSI and DII mechanisms. There is no need of dynamic request recognition, which results in better performance. That justifies application of the second approach, because overheads related to IR Access Server usage should not significantly increase the global overhead of a half-bridge.

6 IR Access Server - Case Study

The described IIOP-domain infrastructure has been implemented over Inter-ORB Engine, experimental software provided by SunSoft. Unfortunately, in Inter-ORB Engine environment Interface Repository (IR) has not been implemented. The goal of constructing IR Access server is possibility of using Interface Repository of Orbix. In this environment, IR is an encapsulated entity with clearly specified interfaces. Such encapsulation allows to use Orbix Interface Repository in IIOP domain. Main idea of cooperation of Orbix and IIOP domains is shown in Fig. 8. IR Access Server works in both environments – its specific parts are responsible for correct data encapsulation and passing it between two domains.

It has been decided to make some assumptions about this module construction. Short list of those assumptions is shown below.

- IR Access Server is a client in Orbix domain.
- IR Access Server is a server in IIOP domain.
- Since Interface Repository uses other objects to supply a client with information, proxy for this objects must be created in IIOP and employed to represent the real IR object to the client in IIOP domain.

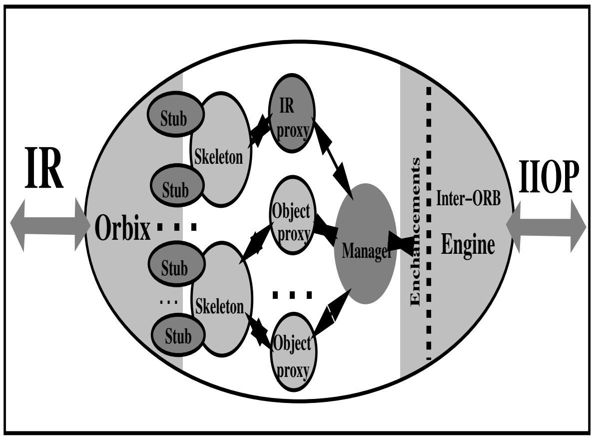

Figure 8: IR Access Server internal structure

The internal architecture of the IR Access Server, which fulfills these assumptions consists of the following elements:

1. Orbix Client Stubs for every interface supported by Interface Repository,
2. IIOP Server Skeletons, for every interface supported by Interface Repository, which are responsible for requests recognition and handling,
3. Inter-ORB Engine layer,
4. Enhancements of Inter-ORB Engine providing management of more than one object, such as creation and destruction of object and two level dispatching of requests — first choosing a target object, second executing an appropriate function,
5. Interface Repository Proxy Object in IIOP domains, which represents access to Orbix Interface Repository in IIOP,
6. IR proxies for other IR objects returned as a result of operations performed on Interface Repository and created dynamically.
7. Main Manager which manages all above components, translates requests from IIOP into Orbix format and vice versa and communicates with all requesting objects.

Since standard Inter-ORB Engine allows to manage only single object it had to be extended to support multiple objects. The Orbix Client Stubs have been generated using Orbix IDL compiler. The IIOP Server Skeletons have been written by hand in face of lack of IIOP IDL compiler. For TypeCode translation a dedicated interpreter has been implemented.

7 Conclusion and Future Work

In this paper it has been presented the IIOP-domain infrastructure that represents a collection of selected Object Services and the technique of their usage by mediated request-level Inter-ORB Bridges built around IIOP-ORB backbone. It has been described how inter-ORB bridges communicate and cooperate using ORB-backbone as an intermediary. For this cooperation availability of certain object services or at least their specializations for cooperation purposes in ORB-backbone is necessary. These services are: the Life Cycle Service, Implementation Repository, Interface Repository and Naming or Trading Service. As a technique of extending the ORB-backbone with these services the Access Server has been proposed and its usage for accessing Interface Repository has been described. The particular emphasis has been put to the problem of mapping object references passed as operation parameters. Two object reference mapping strategies: eager and lazy have been described.

The research described in this paper has led to the conclussion that for a bridge-based ORB interoperability not only bridges have to be designed and implemented but also some, described in this paper, services have to be standardized and made available in an ORB acting as a communication backbone.

References

[1] *CORBA 1.2 Revision Draft*, OMG Report 93-12-43, Object Management Group (OMG) Inc, 1993.

[2] *ORB Interoperability. Joint SunSoft / Iona Submission to the ORB 2.0 Task Force Initialization & Interoperability Request for Proposals* (1994) OMG Inc., TC Document 94-3-1

[3] *Universal Networked Objects*, OMG Inc., TC Document 94-9-32, 1994.

[4] *Interface Repository*, OMG Inc., TC Document 94-11-7, 1994.

[5] *ISO/IEC JTC1/SC2/WG7 N807. Working Document on Topic 9.1 - ODP Trader*, 1993.

[6] Steinder, M., Uszok A., Zieliński K.: *A Framework for Inter-ORB Request Level Bridge Construction*, Proceedings of IFIP/IEEE International Conference on Distributed Platforms ICDCP'96, Drezno, Niemcy, Chapman & Hall, 1996.

[7] *Orbix Programmers Guide*, IONA Technologies Ltd., 1995.

[8] *The DOME User Guide*, Object-Oriented Technologies Ltd., 1994.

[9] Yang Z., Vogel A., *Achieving Interoperability between CORBA and DCE Applications Using Bridges*, (1996) International Conference on Distributed Platforms,