Objects Mapping in Object-Oriented Service Bases

5G−8

Qianshan He and Hidehiko Tanaka
Department of Electrical Engineering, University of Tokyo

1 Introduction

We are implementing a distributed system which provides extensible as well as integrated environments for users and application programs to access and combine distributed resources conveniently. The system is based on the concept of object-oriented service bases. Distributed resources are abstracted as objects. Three layered views about objects are divided: (1). The external view is the view shown to users or application programs. Objects in the external view are transparent and easy to be used. (2). In the conceptual view, attributes of a resource is abstracted as an object. The difference of media types, heterogeneity and location distribution of resources are absorbed in this layer. Object management in this layer is based on a directory [1]. (3). An object in the internal view is the entity of a resource. The internal view about an object concerns its physical realization in a computer. Objects in the conceptual view are reflected to their respective objects in the internal view.

A service base can be seen as a software package built on an existing operating system to provide mapping from the objects in the external view to the objects in the internal to realize transparency and to absorb heterogeneity and different media types. The overview of our system can be referred to [1] and [2]. This paper mainly discusses the objects mapping mechanism, and the query methods provided to users.

2 Objects Mapping

The relationship of objects in the three layered views has following characteristics: first, objects inside one node are referenced by object pointers. Objects distributed in networks are referenced by remote objects mapping. We do not use remote object pointers to reference remote objects. Second, logically, a global inherited relationship of distributed objects can be seen in the external view of a node.

Our system is being implemented by object-oriented language C++. Using this kind of language to implement remote object pointers is difficult and complicated, in that we must extend the language to a distributed one. Instead, we use the remote objects mapping mechanism to realize remote object references. In the follows, we use a simple example (in Figure 2) to explain the mapping mechanism.

We suppose there is a file registered in the service bases, its object name in the external view of node(N) is called *file_name*. An user in node(N) wants to display this file. We suppose he does not know where the file is actually located and what media type the file is expressed by. Therefore, he is not certain to use what kind of editor to display this file. The logically inherited objects tree in the external view of node(N) (this tree is shown to users in a command shell window) indicates that this file can be imposed various operations: it can be displaied by using the *edit* operation or be printed by using the *print* operation (an operation which is inherited from the super-object) etc.. After *edit(file_name)* is requested to the service base in node(N), the file (which actually is an image file) is displaied in an image editor selected by the service base in node(N).

Inside the service base in node(N), the objects registered in the external view are first searched. If there is a object with the same name as the *file_name*, this object is selected. This object has a pointer to an object in the conceptual view. By using this pointer, the object in the external view of node(N) is mapped to the object in the conceptual view in node(N). The object in the conceptual view (attribute information of the resource) indicates that the file is located in node(N+1) and the media type of the file is image media, and that there is an image editor (object name in the external view is called *edit*) in node(N) which can display this file (in the conceptual view, an object not only contain the attributes of a resource, but also contain the information which indicate what operations can be applied to this resource.). In the conceptual view, the requested operation is checked if it can be applied to the data. In this example, the requested operation *edit* can be applied to the data *file_name* Then, the operation part of the object in the conceptual view is mapped to an object in the internal view in node(N) (the entity of the resource), and a request (object name) is sent to node(N+1) to map an object in the external view in node(N+1). In node(N+1), same processes of objects mapping as in node(N) are done and finally the file is copied

Objects Mapping in Object-Oriented Service Bases
Qianshan He and Hidehiko Tanaka
Department of Electrical Engineering
University of Tokyo

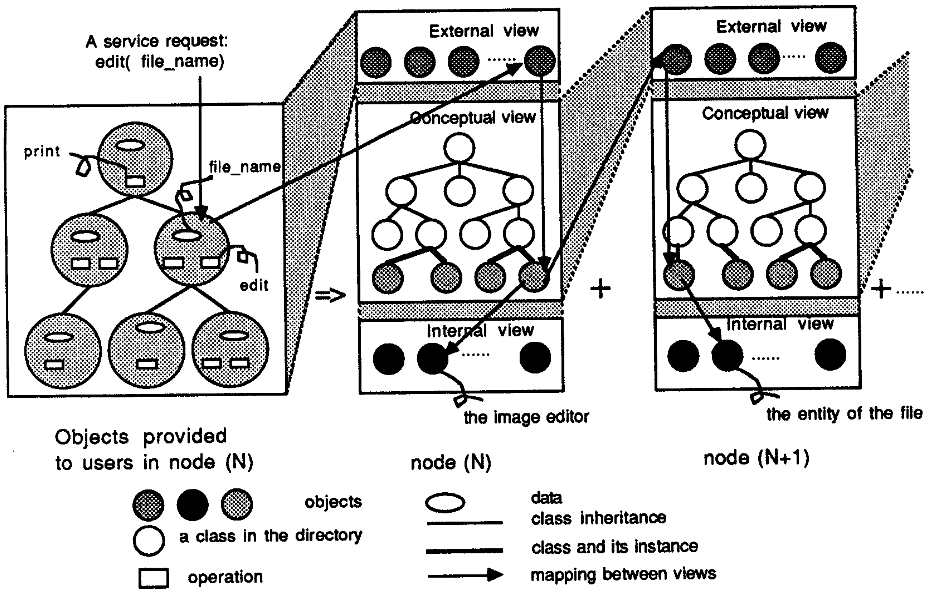

Figure 1: The Relationship of Objects in the Three Layered Views

to node(N) to be displaied by the editor. In this example, we can see that the remote object mapping is realized by containing the attributes of remote node name and remote object name in the objects in the conceptual view, and by searching the object in the external view in the remote node.

From this example we can also see that logically the objects in the external view of node(N) have a global inherited relationship. The global object image is realized by the so-called forward mapping: if objects is not local objects, the objects in the conceptual view of node(N) are mapped to the objects in the external view of node(N+1), and the objects in the conceptual view of node(N+1) are forward mapped to the objects in the external view of node(N+2).

3 Query Methods

To search or use a service registered in the service bases. we provide two kinds of query methods. The first method is called "query by object name". The example given above is using this method. The second method is called "query by object attributes". In some cases, users may want to use or search objects with certain characteristics. Figure 2 shown some attributes and their attribute values of objects in the conceptual view. Users can query the service bases by selecting a set of attributes and attribute values to get needed objects. For example, by setting

$\{\ Date >= "1989.11.1", Format = LATEX\ \}$

the data which are in LATEX file format and are created after Nov. 1, 1989 will be selected.

```
{ ExtName = "doc1",  HostName = ENTERPRISE,  Media=MULTIMEDIA,
  Format  = ODA,  Status = LAYOUT_CONTENT, Date = "1989.11.1",
  Author  = {"Q.HE","Other"}, Title ="A Test ODA document(1)",
  KeyWord = {"distributed", "ODA"},        IntName = "doc1.oda",
  Resouce = DATA
}
{ ExtName = "doc2",        HostName = DISCOVERY, Media = IMAGE,
  Format  = RASTER,       Date = "1989.10.5",   Author= "Q.HE",
  Title   = "A Test Image document", RemoteExtName ="doc2.ra",
  KeyWord = {"network","distributed", "raster"},Resouce = DATA
}
{ ExtName = "doc3",        HostName = DISCOVERY,  Media = TEXT,
  Format  = LATEXT,        Date = "1989.11.5",  Author= "Q.HE",
  Title   = "A Test Text document", RemoteExtName ="doc3.tex",
  KeyWord = {"network","distributed", "text"},  Resouce = DATA
}
{ ExtName = "editor1", HostName =ENTERPRISE, InputType = TEXT,
  IntName = "emacs",             AccessPass = "/user/local/bin",
  Resouce = FUNCTION
}
{ ExtName = "editor2", HostName=ENTERPRISE, InputType =BITMAP,
  IntName = "bitmap",            AccessPass  =  "/user/bin/X11",
  Resouce = FUNCTION
}
```

Figure 2: The Attributes of Objects

4 Future Work

As a future work, we are considering to construct a distributed object database system based on the framework of object-oriented service bases.

References

[1] Q. He and H. Tanaka, "Implementation of A Multimedia Resource Mangement System Based on Distributed Objects", 39th National Conference of IPSJ, 7H-4, Oct. 1989.

[2] Q. He and H. Tanaka, "A Distributed Resource Management System Based on An Object-Oriented Approach", Proc. of 4th International Joint Workshop on Computer Communication, pp. 387-396, Tokyo, July 1989.