CORBA Programming

B.Ramamurthy

Chapter 3

Programming Steps

- See Fig.3.1
- Step 1: Develop required interfaces for an application by writing the IDL files and compiling them. This generates stubs and skeletons, to support development of client and server program and to facilitate communication between the client and server.
- Step 2: Develop client application using the interfaces specified in the IDL files. If interfaces are known at compile time the client uses SII else use DSI(?). Compile.
- Step 3: Implement the server using Static Skeleton Interface (SSI). Compile.
- Step 4: Run server; register; Run client and test.

SSI and DSI

- Client and Server style is transparent to both sides.
- Details of SII, SSI, DSI are implementation-dependent and do not concern application interfaces.
- Programmers are free to mix and match these styles.
- For example, SII clients can talk to a DSI server and DSI clients can talk to SSI server and so on.

iService Application

- See Fig.3.2

iService Application Description

- A company wants to develop a prototype of an application iService that provides three kinds of Internet services: free, paid, and tryable.
- All transaction fees are quoted in dollars.
- Users must register and are given initial credit of $100. Can transfer more money directly from a bank.
- User has account with username and password.
- iService operates three services: news, retail, and price quoting service.

IDL for iService

- P.83-85
- Complete understanding is needed.

Naming Service

- A naming service is a generic directory service that provides similar functionalities as the White Pages, such as finding an address of a person given a name.
- Given a name of an object, the naming service is responsible for returning the reference to that object.
- Names require management. They can be treated uniformly within its own name space.

Name Management Functions

- **Structuring**: The structure of a name space must reflect the naming policy adopted. (design decision)
- **Name allocation**: Within some context the name should be unique. Ex: IP address is unique in a network. Name management includes the activity of administering and allocating the unique part of a name to a particular entity.
- **Name Registration**: This is a function of the directory service management that includes the activity of making persistent the information which permits mapping between names and entities.
- We will examine "Naming" in CORBA OMG. Other Examples: DNS and X.500 directory service.

Naming (Chapter 7)

- Names may refer to a variety of resources such as computers, services, ports, and individual objects.
- Names cover three different concepts: symbolic names (Ex:yeager@cse), unique identifiers (Ex: UUID, IOR), addresses (Ex:128.205.36.1 for yeager). See Figure 7.1
- Association between a name and an object is called **binding**.
- A naming service is responsible for maintaining a database of bindings between textual names and attributes of objects. Operation **resolve()** looks up attributes in a database for a given name.
- Other operations: create new bindings, delete bindings, list names. For large applications the database is distributed and replicated.

Name Mapping Algorithm

- Performing the name mapping algorithm to determine a value that associated with a name is known as resolving.
- This controlled by another parameter called the context.
- For a given naming system, there may be many contexts. Implementation of context: table of bindings
- A single name may map to different values in different contexts.
- Thus the three main operations are:
  - Resolve (name, context)
  - Bind (name, value, context)
  - Unbind (name, value, context)

Common Algorithms

- Simple table lookup (most common)
- Path names
- Search (for complex systems)

OMG Naming Service

- Service allows one or more local names associated with an object reference.
- A server that holds the object reference can register it with the naming service.
- Client applications can use the naming service to obtain the object reference by using the logical name assigned to that object.

Naming Structure

- Syntax-independent, in-memory hierarchical, similar to Unix, DOS, and X.500
- There is no absolute name. Name binding is always defined relative to a context.
- To resolve a name is to determine the object associated with the name in a given context.
- To bind a name is to create a name binding in the given context.
- Context itself is an object.
- Contexts form a graph called naming graph. See Fig.7.3

Name IDL

- In general, the OMG idl for a name is follows:

```
Typedef String Istring;
Struct NameComponent {
   Istring id;
   Istring kind;
};
Typedef sequence <NameComponent> Name;
```


NameContext IDL

- CORBA Naming Service is provides operations for navigating and updating the naming graphs.
- This is defined in NamingContext idl.
- We will discuss each operation in this idl in details.
- See the enclosed pages.224-230

Lessons Learned

- To describe a CORBA service, analyze the needs, define its functions, and structure.
- Define the IDL for the components.
- Define exceptions possible.
- Implement the IDL specification.