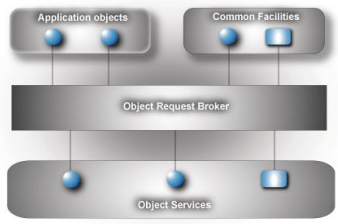
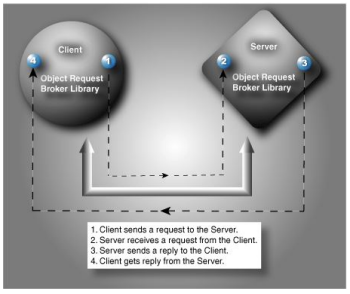
CORBA - Common Object Request Broker Architecture

- Big Picture
  - Strategy for distributing objects over the internet



  - Object Request Broker connects a client to objects owned by a server
  - Makes it feel like the client is talking directly with the server



- IDL - Interface Definition Language
  - Automates the creation of a set of files that define the interface between the client and the server

```
module HelloApp
{
    interface Hello
        {
            string sayHello();
            oneway void shutdown();
        };
};
```

  - If the module we define is XApp, generates a set of files:
    * XPOA.java - POA stands for Portable Object Adapter. This is an abstract class that is the structure of the server
    * X.java the java version of the interface we are defining

        * XHelper.java manages the streams connecting the client to the server
        * XHolder.java Holds the public instance of our X class. For any out or inout methods, the appropriate streams are defined and details are delegated to XHelper.java
        * XOperations.java interface defining the operations in our interface
        * _XStub.java Client stub that implements the X.java interface

- So, what do we build? (look at the code fpr each)
    - Operations - I coded them in _FilesSystemStub.java, but I really should have extended that in the client
    - XClient.java - this is the runner for the client and controls what the client asks for.
    - X.POA - It maps requests to code that handles each request. If you can add more operations to the interface, they get built here
    - XServer.java - this is the runner for server - sets things up and sits there.

- So, how do we run it?
    - Everything is in a package - know how to run them from the command line?
    - On the server
        * orbd - starts up the Object Request Broker that maps incoming requests to the correct server
        * the server
    - On the client
        * the client class
    - the magic parameters
        * ORBInitialHost - the host that orbd should pay attention to (I have no idea why it doesn't default to localhost . . .)
        * ORBInitialPort - the port that the name service of orbd is listening to. You could have multiple servers running with one orbd running and it would connect clients to them all
        * port - the port your server will be using