# Concurrent & Distributed Systems

## Distributed Systems 5

### distributed communications: CORBA

# distributed communication

- files
- sockets
- message passing
- remote procedure calls
- remote method invocation
  - java RMI
  - *enterprise computing*
  - CORBA

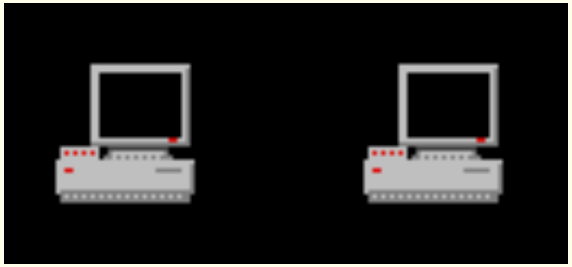
Enterprise computing

# systems today 1

- most real software systems face a number of problems:
  - must run on a network of machines
  - machines are likely to use different architectures
  - machines may run different operating systems
  - components may be written in different languages
  - must incorporate legacy components which are too large and complex to re-write

- for example, BoS and Halifax merger, RBS and NatWest

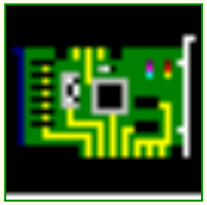
- or, BAe and GEC Avionics

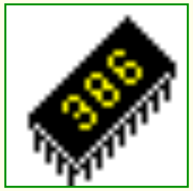


- or, engineering & finance departments

# systems today 2

- **Enterprise Computing**
  - **large**
  - **intra & inter-company**
  - **re-use**
  - **sections may be written by different companies**
    - **subcontracting**
    - **open market**
  - **as in hardware components - component technology**
    - **strictly not just an object**



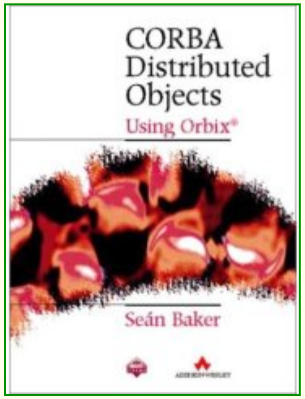


# distributed communication

- files
- sockets
- message passing
- remote procedure calls
- remote method invocation
  - java RMI
  - enterprise computing
  - *CORBA*

# CORBA: background material

- **Reading: CORBA Distributed Objects using Orbix Sean Baker, Addison Wesley.**

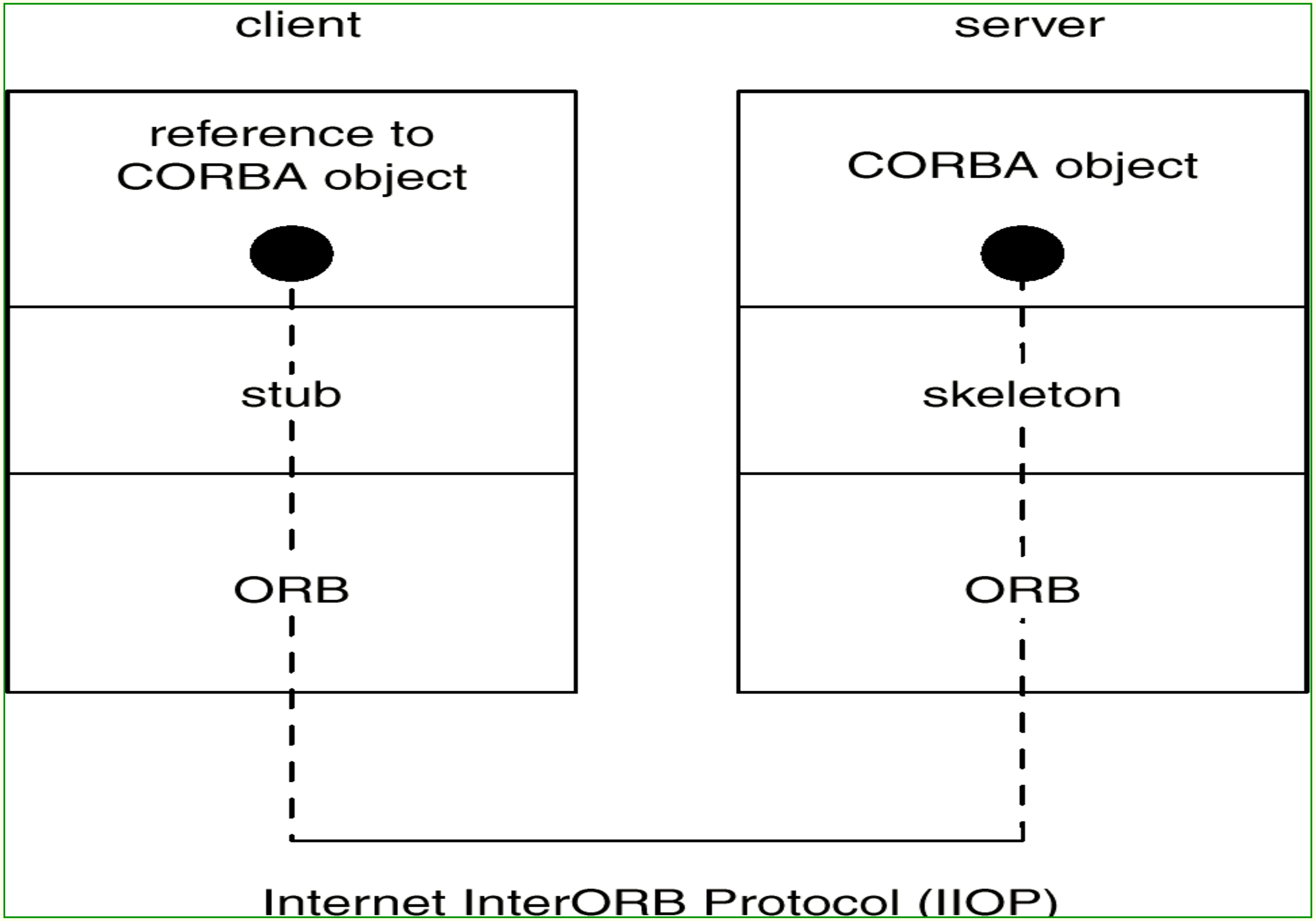


- **Common Object Request Broker Architecture**
  - **Object Management Group OMG:**
  - **http://www.omg.org**
  - **Adopted as a standard October 1991**
  - **First implementation July 1993**

# RMI & CORBA

- **RMI is Java-to-Java Technology**
- **CORBA is Middleware that allows *heterogeneous* Client and Server applications to communicate**
- **Interface Definition Language (IDL) is a generic way to describe an interface to a Service a Remote Object provides**
- **Object Request Broker (ORB) allows Client and Server to communicate through IDL.**
- **Internet InterORB Protocol (IIOP) is a Protocol Specifying how the ORBs can communicate.**

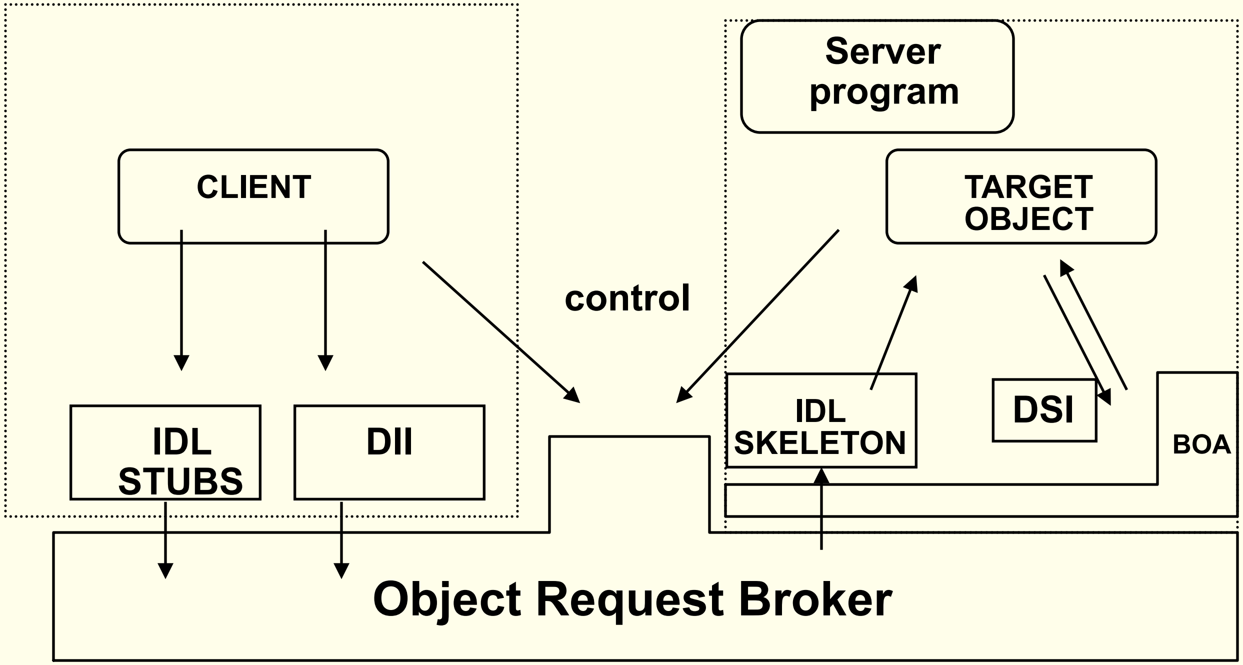
# Corba model


client
server
reference to CORBA object
CORBA object
stub
skeleton
ORB
ORB
Internet InterORB Protocol (IIOP)

# aims of CORBA

- **to make it easier to implement new applications which must be distributed**
- **encourages the writing of open applications that can be used of components of larger architectures : idea of *component architectures*.**
- **CORBA addresses three important difficulties in large scale enterprise computing:**
  - **permits each project to make independent choices about languages, machines, operating systems etc.**
  - **introduces the ability to hide difficulties from programmers: OO extension to the ideas of RPC.**
  - **use of object-oriented design at the enterprise level**

# The CORBA architecture

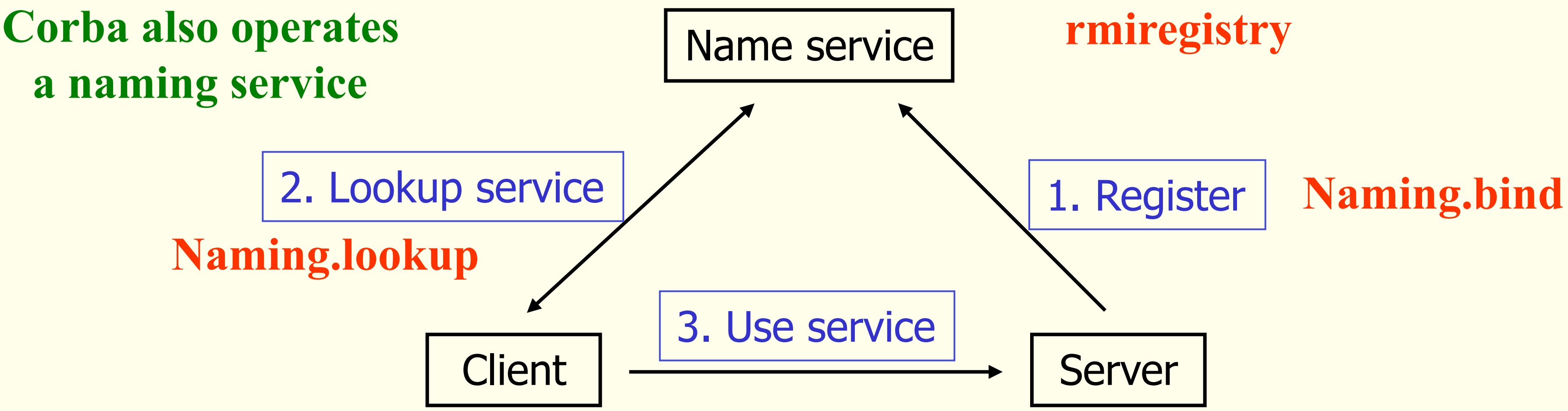


# ORBs

- **each ORB must support the following:**
  - **the IDL language and hence mapping of IDL to a programming language such as C++ or Java**
  - **runtime support for passing an object request (method call) between a caller & a target object**
  - **registration facilities**
    - **acts in a similar manner to java's rmiregistry**
    - **supports the equivalent to Naming.lookup & (re)bind**
    - **actually this form of name service is common.**

# locating named entities

- a ***name service*** **is a common aspect for the distributed system.**
- **when a service is started at any site, it sends its name and location and any other information required to use it to the name service.**
- **when a client wishes to use a service, the information needed is looked up in the name service.**



- **the location of the name service is assumed to be well-known**

# name services

- many distributed systems use a name service.
- examples include the *Domain Name Service* (DNS) used by the internet.
- name services effectively shift the problem of locating services.
  - we can use the name server to locate any service we require.
  - but how do we locate the name server?
- there are two main ways:
  - hardwire the address of the service into programs.
  - issue a broadcast message on the LAN to locate the name server. The name server responds with details of its current location.

# the internet Domain Name Service

- **the Domain Name Service (DNS) is a distributed database that may be queried to obtain attributes associated with a name.**
  - **This is how the *gethostname* method call works.**

```
InetAddress server_inet_add = InetAddress.getLocalHost();
String server_host_name = server_inet_add.getHostName();
System.out.println ("Server name is " +server_host_name );
```

- **queries might include:**
  - **Computer name --> Location (IP address)**
  - **User e-mail address --> Mail server location (IP address)**

# Dynamic Invocation Interface

- ***back to corba …..***

- **static:**
  - **the facilities of the ORB**
  - **must know the *invoked* class**

- **dynamic:**
  - **DII (Dynamic Invocation Interface)**
  - **unaware of invoked class at compile time**
  - **Dynamic Skeleton Interface (DSI) is on the server side**