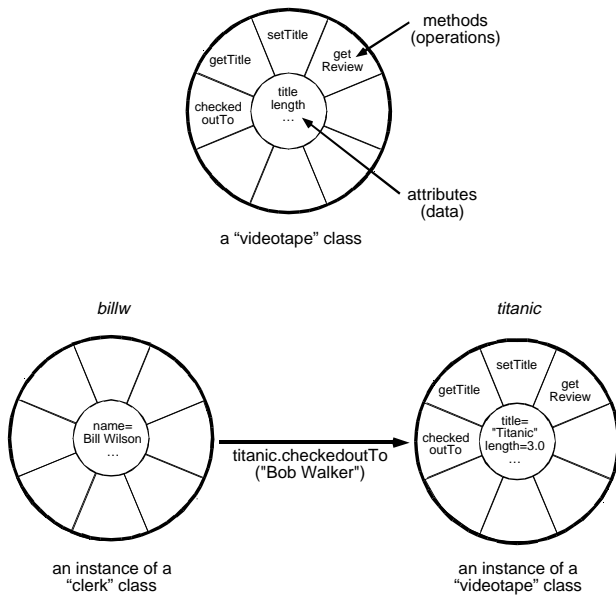


## Objects



### ■ Abstract data types (1970s) → objects

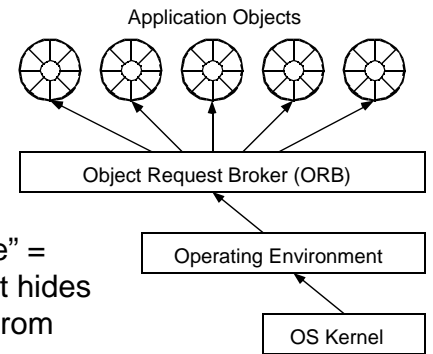
- Encapsulation of data (attributes) along with procedures that manipulate that data (operations, methods, member functions)

1

Spring 1999, Lecture 09

## Distributed Objects & The ORB

- The ORB acts as the platform for distributing objects

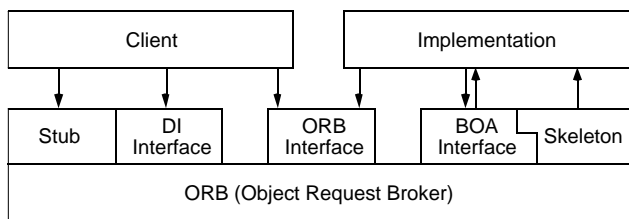


- "Middleware" = software that hides OS details from applications
- Using middleware for client-server communication allows software developers to write software that is portable between platforms (support for hw/sw heterogeneity)
  - Intraprocess, interprocess, intermachine
- System vendor supplies the operating environment (platform) on which the middleware is based
  - May offer multiple platforms to support different customer needs

2

Spring 1999, Lecture 09

## Common Object Request Broker Architecture (CORBA)



### ■ Interface Definition Language (IDL) compiler produces client stub and implementation skeleton

- On client side:
  - Code to locate skeleton, marshal parameters, and transport the message
  - DII is an alternative to IDL (details later)
- On implementation (server) side:
  - Similar: unmarshaling, etc.
  - Programmer must fill in method calls
  - Skeleton represents object's formal type, basic object adapter (BOA) (details later) represents style of implementation
- Can also use ORB directly if necessary

3

Spring 1999, Lecture 09

## RPCs vs. CORBA (History, Politics)

- Three versions of RPC in common use:
  - Courier (Xerox PARC) → Xerox Network Services (1979), still used inside Novell Network
  - SUN's RPC (mid 1980s), used in their Network File System (NFS), provided on most Unix systems
  - Distributed Computing Environment (DCE), supported by HP, DEC, IBM, later by Open Software Foundation (OSF)
- Result: 3 incompatible systems
- CORBA can be built on top of RPC or any other package
  - Others propose to base CORBA on TCP/IP, which is vendor-neutral

4

Spring 1999, Lecture 09

## RPCs vs. CORBA (Technical Details)

- RPC does not understand objects; CORBA does
- RPC supports remote function calls, CORBA supports intraprocess, interprocess, and intermachine method invocation
- RPC supports server activation, but not object activation (CORBA does both, thus has finer granularity)
- RPC is not as transparent as CORBA
  - Incompatible versions, etc.
  - RPC needs to know location of object it calls, and needs to connect and disconnect from a known service; ORB client doesn't need to know any of this

5

Spring 1999, Lecture 09

## Static vs. Dynamic Invocation

- Static approach:
  - All objects known at compile time
  - Use IDL to describe objects; IDL compiler produces source code to be compiled into client and implementation
  - May be able to improve implementation through compiler optimizations
- Dynamic approach:
  - No IDL — add a Dynamic Invocation Interface (DII) to the ORB so that requests can be made to objects that were unknown at implementation time
    - Create a generic Request object, then add arguments, then invoke object
  - New objects do not require recompilation
  - Takes 80% less code than static approach, according to some studies

6

Spring 1999, Lecture 09

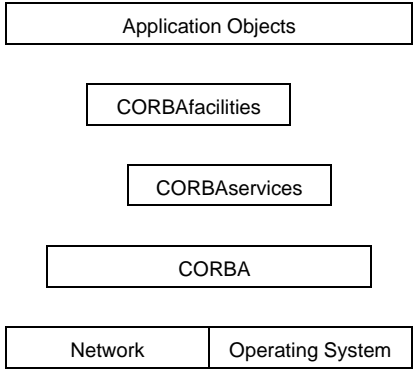
## Object Adapters

- Object Adapter suggests a style of implementation for an object
  - Should centralize functionality (style) that is common across a range of implementations
- CORBA supplies a general-purpose object adapter called the Basic Object Adapter (BOA)
  - Centralizes style of object activation
  - Supports object creation, destruction, activation, and deactivation
- Other object adapters possible:
  - Library Object Adapter — dynamic loading and linking of a library
  - Load-Balancing Adapter — select the appropriate implementation, or aid in migration of processes, as load shifts

7

Spring 1999, Lecture 09

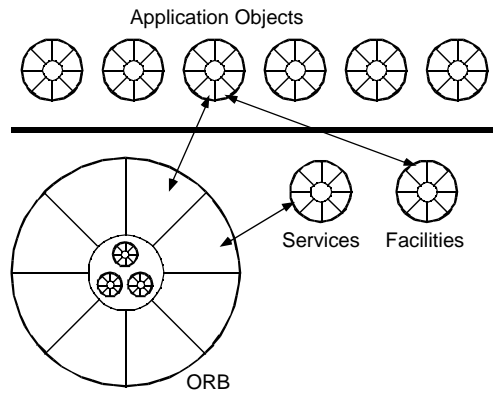
## OMG's Object Model (Layered View)

- CORBA consists of much more than "just" the ORB!
 
- Object Management Group (OMG) formed in 1989
  - Around 800 member companies
  - Sets unofficial "standards", driven by technology and market forces
- CORBA (the ORB) supplies objects
  - Application objects can use CORBA directly, but they can also use higher-level CORBAservices and CORBAfacilities

8

Spring 1999, Lecture 09

## OMG's Object Model (Object View)



- Five components of the OMG's model:
  - The ORB (CORBA) — communication platform upon which interesting objects are constructed
    - Provides naming, request dispatch, parameter encoding, synchronization, exception handling, and security
  - Services, Facilities, Application objects

9

Spring 1999, Lecture 09

## CORBA services

- CORBA services interact closely with the ORB, and provide a “higher” level of service than the ORB
  - These are the fundamental objects that everyone uses
  - Class management — create, delete, modify, copy, distributed, move
  - Instance management — create, delete, modify, copy, invoke
  - Storage for objects
  - Integrity for objects (locks, transactions)
  - Security — access control
  - Versions

10

Spring 1999, Lecture 09

## CORBA facilities

- CORBA facilities are optional, and interact closely with applications by providing application frameworks and promoting common interfaces
  - Not many facilities exist yet
  - Catalog and browser for classes and objects
  - User interface components
  - Printing and spooling
  - Error facilities, help facilities
  - Email facilities
  - Computer-based training and distance learning
  - Information repositories
  - Agents and intelligent macros
  - Querying facilities

11

Spring 1999, Lecture 09

## OLE, COM, and DCOM

- OLE, COM, and DCOM
  - Object Linking and Embedding (OLE)
    - Placing references to one object (“embedding it”) inside compound object
  - Common Object Model (COM)
    - Also specifies access and binary layout of objects, allowing multiple language access
  - Distributed COM (DCOM)
    - Distributed version of OLE and COM
- OLE, COM, DCOM vs. CORBA
  - OLE and COM are for non-distributed environment; CORBA supports distributed heterogeneous systems
    - COM, etc. are primarily Microsoft
  - Different object models
    - CORBA supports multiple inheritance, but COM supports only single inheritance
    - COM class may be composition of a set of COM interfaces (an aggregate)

12

Spring 1999, Lecture 09