Unit - II

Enterprise Java Beans

Enterprise Java Beans

- Write once, run-anywhere, middle tier components

Evolution of Technology

- Mainframe/Terminal model
- Transaction processor
  - To handle concurrent client requests
  - Several statements as on logical unit
  - Guaranteeing successful execution or non would be executed

Transaction processor

- Provides API with ‘begin’, ‘commit’, and ‘rollback’.
- Logging mechanism

ACID properties of Transactions

- Atomicity
- Consistency
- Isolation
- Durability

OLTP Vs DSS/EIS/OLAP

- Updates in a Database
- Response time is critical
- Can handle large volume of transactions

- Reviews information
- Involves long-running queries
- Smaller number of requests, longer think time

Evolution…

- Two-Tier architecture
  - Transaction integrity by DBMS
- Three-Tier architecture
  - Transaction integrity by Middle tier components
- **Sockets**
  - Limited distributed computing
- **RPC**
  - A thin layer on top of Sockets
  - Stub-Skeleton
  - Stub uses IDL

CORBA

- An object oriented RPC Mechanism
- Objects written in one language can be called by objects written in a different language
- CORBA clients can access EJB objects

RMI

- Java version of CORBA
- No need to write IDL. RMIC handle automatically
- EJB allows client side RMI calls to EJB objects

EJB's role

- EJB specifies an execution environment
  - EJB is a java class implements Session bean or entity bean
  - Container provides services to EJB
  - Container provides proxy object for each bean
- EJB exists in the middle tier
  - To encapsulate business logic
- Supports transaction processing
- Can Maintain state

- **Enterprise JavaBeans** (EJB) is an architecture for setting up program components, written in the Java programming language, that run in the server parts of a computer network that uses the client/server model.
- Enterprise JavaBeans is built on the JavaBeans technology for distributing program components (which are called Beans,) to clients in a network.

- Enterprise JavaBeans offers enterprises the advantage of being able to control change at the server rather than having to update each individual computer with a client whenever a new program component is changed or added.
- EJB components have the advantage of being reusable in multiple applications.
- Can be deployed across all major operating systems, not just Windows.

EJB's Architecture

- Logically three-tier system
- EJB server & DB reside on the same machine – EJB server includes built in functionality for persistent storage
- EJB server & Client – EJB bean makes a call to another EJB bean
- All three tier might reside on a single machine

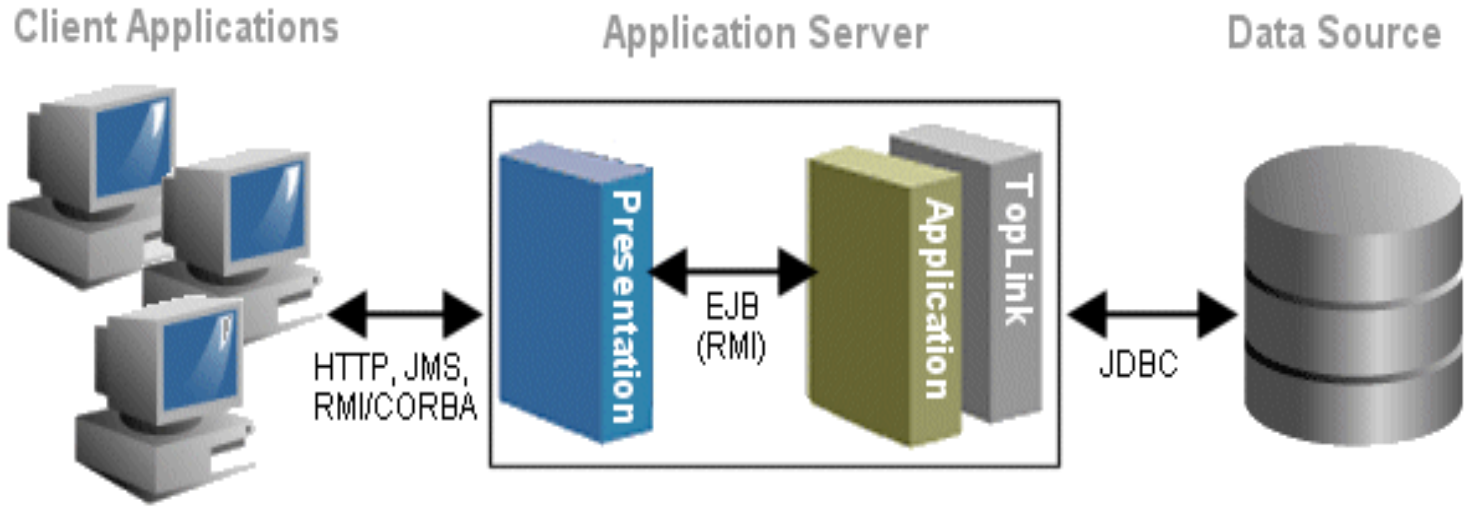
Client - EJB Server - Database

EJB's role in the three layers

- Client calls remote EJB's
- EJB components live in the middle tier,

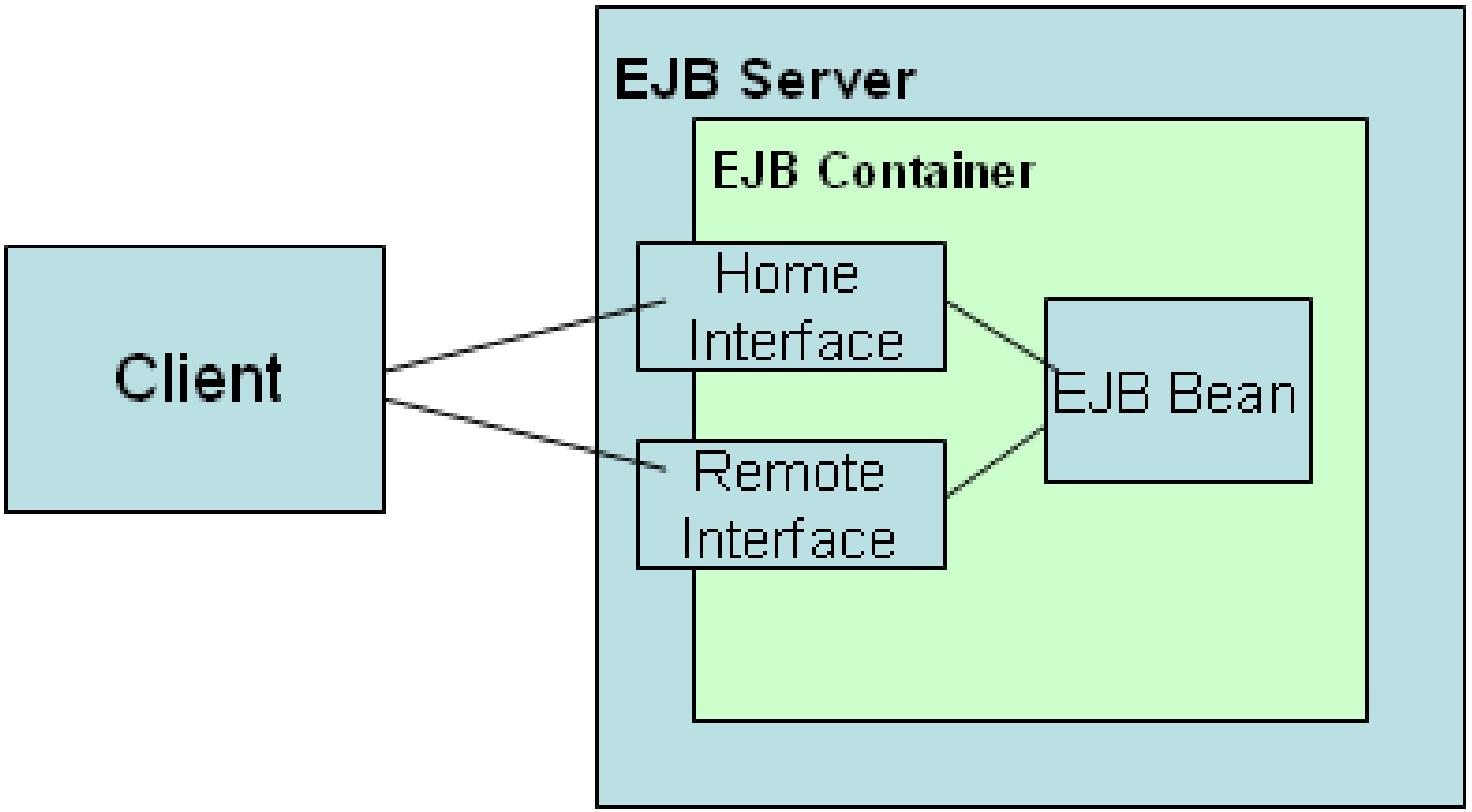
  EJB objects reside in an EJB container which is in side of an EJB server
- DB resides in the third layer

  EJB beans accesses the DB through JDBC



Overview of EJB's Software Architecture

- EJB bean exists within the container
- Client communicates with bean through home interface, remote interface



Overview of EJB's Software Architecture

EJB server

- Provides container with lower level services such as network connectivity
- Layered approach

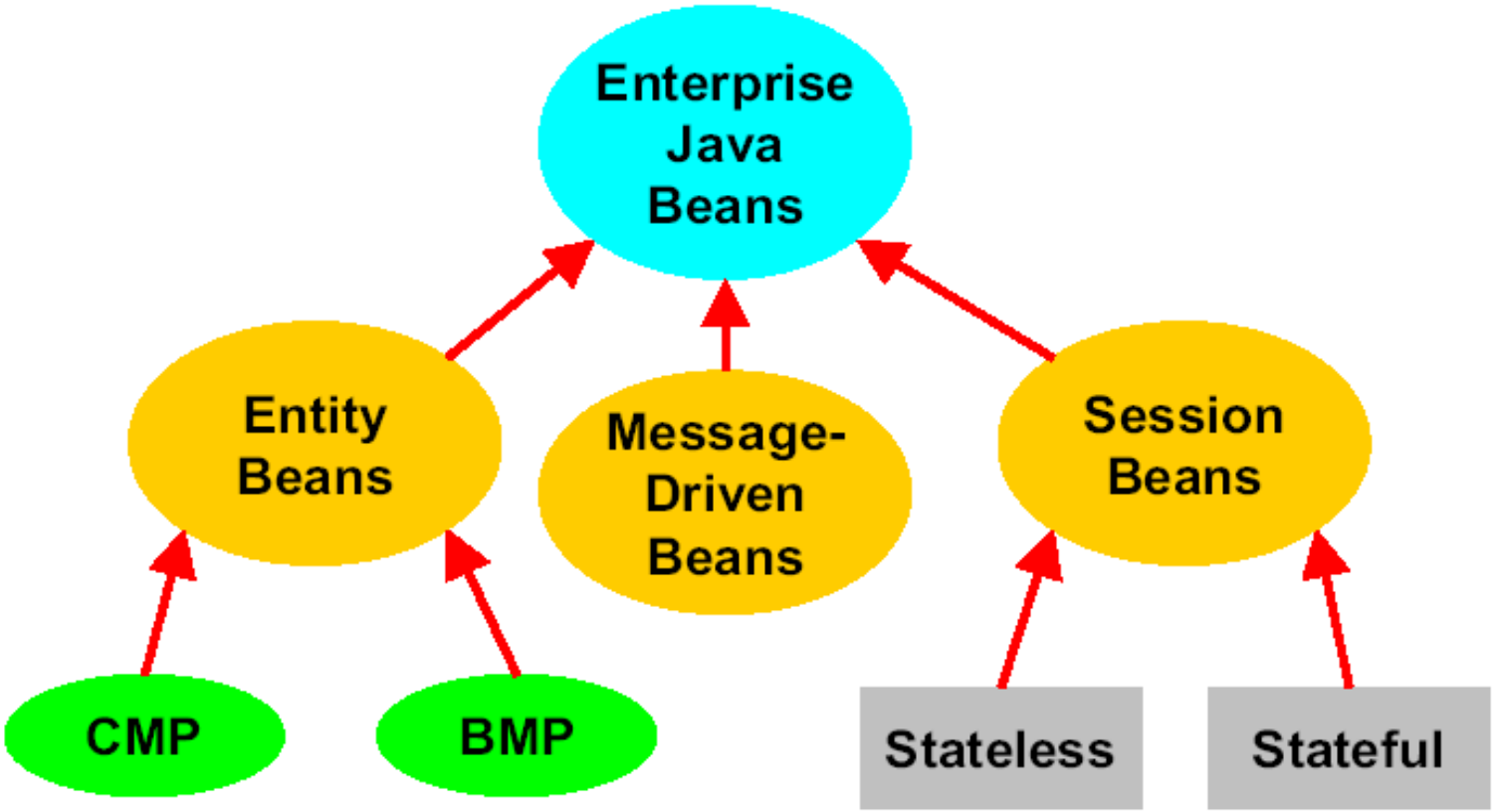
EJB Container

- Interface between EJB and outside world
- Can create pool of beans
- Provides services to Beans
  - Support for transactions, management of multiple instances, persistence, and security

Enterprise Bean

- EJB object is implemented, in addition Home interface and Remote interface implemented

Types of EJB beans



- **Session Bean** – is created by a Client and usually exist only for the duration of a single client/server session.

- **Entity Bean** – represents a business objects in a persistent storage mechanism
  - Ex : Customers, orders & products

- A **stateless session bean** is a distributed object that does not have an associated conversational state, thus allowing concurrent access to the bean.
- The contents of instance variables are not guaranteed to be preserved across method calls.
- **Stateful session beans** are distributed objects having a conversational state. The state could be persisted, but access to the bean is limited to only one client.

Session bean Vs Entity bean

- EB's are persistent, allow shared access, have primary key, and may participate in relationships with other entity beans

When to use Entity bean

- If the bean represents a business entity, not a procedure
- If the bean's state must be persistent
  - Ex: CreditcardEJB - Entity bean
    CreditcardverifierEJB – Session bean