

# JEE - Java Enterprise Edition

## EJBs - Architectural Overview



# EJB Architectural Overview

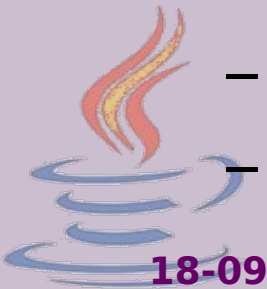
- 2 types of EJBs server-side components
  - *Session Beans*
  - *Message-driven Beans (MDB)*
- *Session Beans* can be accessed using a variety of distributed object protocols
- MDBs process messages asynchronously from systems like JMS, legacy systems and web services



# EJB Architectural Overview

## Session Beans

- Extensions of the client application that **manage processes or tasks**
- Session Beans **manage** particular kinds of **activities**
- Have to do with the **relationship between entities, but do not map entities**
- **Examples** (Singapore Subway):
  - make a ticket reservation
  - check train capacities
  - create ticket
  - calculate routes etc.



# EJB Architectural Overview

## Message-driven Beans

- Similar to Session Beans
- **coordinate tasks involving other session and *entity beans***
- *MDBs* and Session Beans differ in how they are accessed
- **MDBs subscribe to or listen to messages**
- they **respond by processing a message** and managing the actions that other beans take
- Example Bank-Transaction-Manager manages a bank transfer



# EJB Architectural Overview

- The **activity** an EJB represents is **fundamentally transient**
- They do **not represent entities** in a database
- They implement actions on a database and lead to changes though
- **Session- and Message-Driven-Beans** model interactions, but they **do not have persistent state**
- **Extremely flexible** usage



# EJB Architectural Overview

## Session Beans

- To implement a session bean you need to define:
  - *Remote Interface:*
    - **business methods**, accessed **outside of the container**
    - *Plain Java Interface*
    - *tagged with:* **@javax.ejb.Remote**
  - *Local Interface:*
    - **business methods**, accessed **inside the container**, running in the **same JVM**
    - *Plain Java Interface*
    - *tagged with:* **@javax.ejb.Local**



# EJB Architectural Overview

## Session Beans

- To implement a session (continued):
  - Endpoint Interface:
    - **business methods**, accessed **outside of the container via SOAP**
    - based on Java API for **XML-RPC**
    - Plain Java Interface
    - tagged with: **@javax.jws.WebService**
  - *Bean Class*:
    - contains business logic
    - implements at least one of the ext. interfaces



# EJB Architectural Overview

## Session Beans

- To implement a session (continued):
  - *Bean Class (continued)*:
    - tagged with:
      - **@javax.ejb.Stateful** or
      - **@javax.ejb.Stateless**
- **Any combination** of local, remote and endpoint interfaces **can be provided**
- **Clients never interact** with the bean class **directly**
- Instead they use the bean's component interfaces

# EJB Architectural Overview

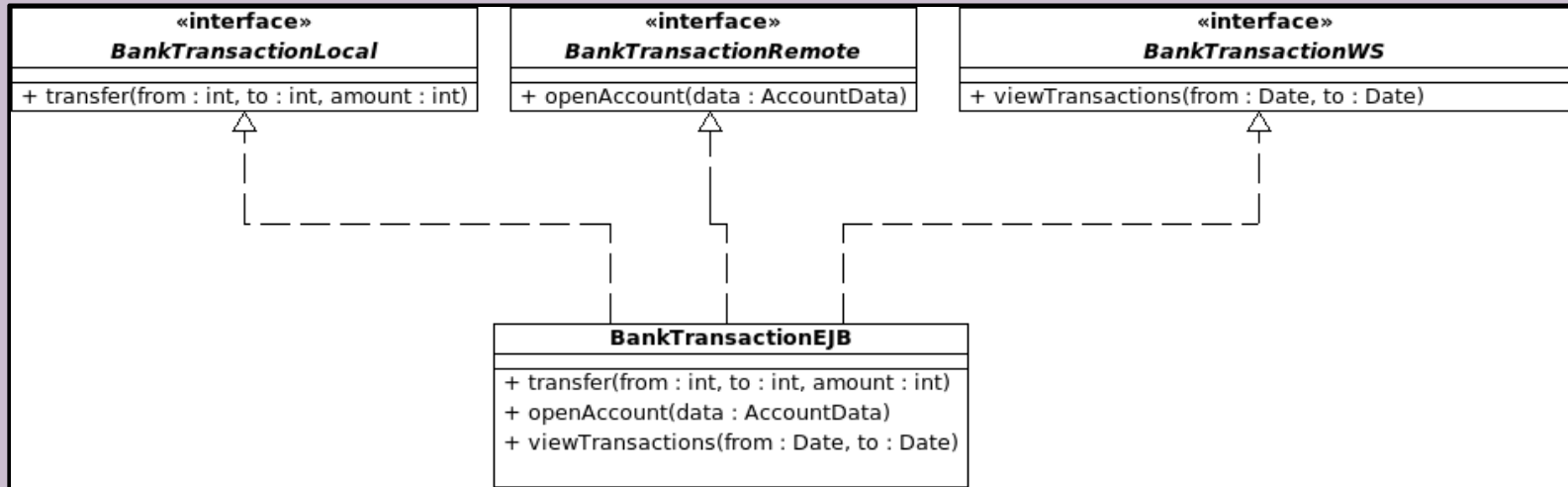
## Session Beans

- Using the component interfaces
  - clients are actually interacting with ***proxies or stubs***
  - generated automatically by the container
  - allows the container to **monitor the interactions** between bean and client and to **apply security** and **engage in transactions**
- Container is responsible for creating new instances of beans

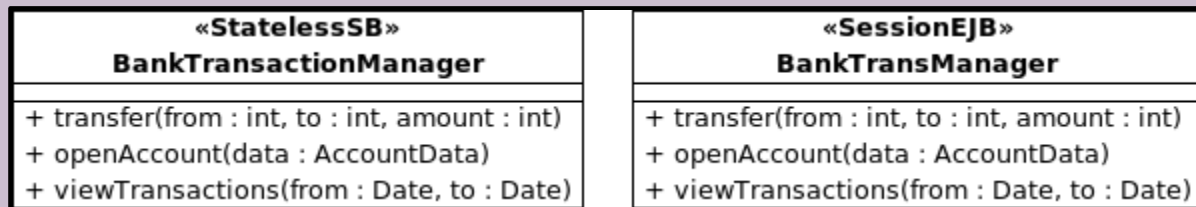


# EJB Architectural Overview

## Session Beans



Alternative:



# EJB Architectural Overview

## Session Beans

- **2 subtypes** of Session Beans exist
  - *Stateful Session Beans*
    - **maintain conversational state**
    - information kept in memory during *active conversation* with a client
    - **instances NOT shared** among clients
  - *Stateless Session Beans*
    - no maintenance of conversational state
    - methods are completely independent from each other
    - **provide better performance**, as **instances are shared** among clients



# EJB Architectural Overview

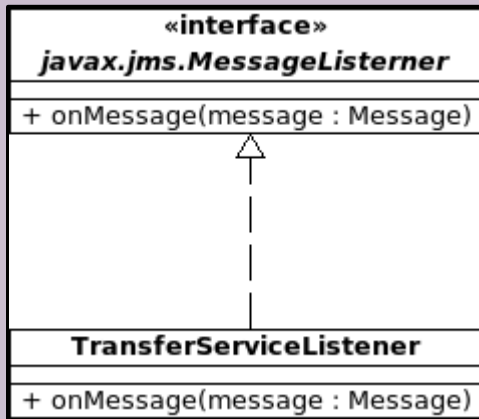
## Message-Driven Beans

- To implement a message-driven bean you need to define:
  - Message Interface:
    - defines **methods** by which **messaging systems**, e.g. JMS, **can deliver messages to the bean**
  - *Bean Class*:
    - **implements** one ore more **message-delivery methods**
    - tagged with: **@javax.ejb.MessageDriven**



# EJB Architectural Overview

## Message-Driven Beans



or



# EJB Architectural Overview

## The EJB Container

- **business logic does not directly work with instances of the bean class**
- it's working through the bean's remote or local interface
- when methods are invoked on the interface, the **object instance used is called *proxy stub***
- **proxy stub implements the remote or local interface**
- sends **method invocation** over the network, or routes the request to an EJB container in the JVM



# EJB Architectural Overview

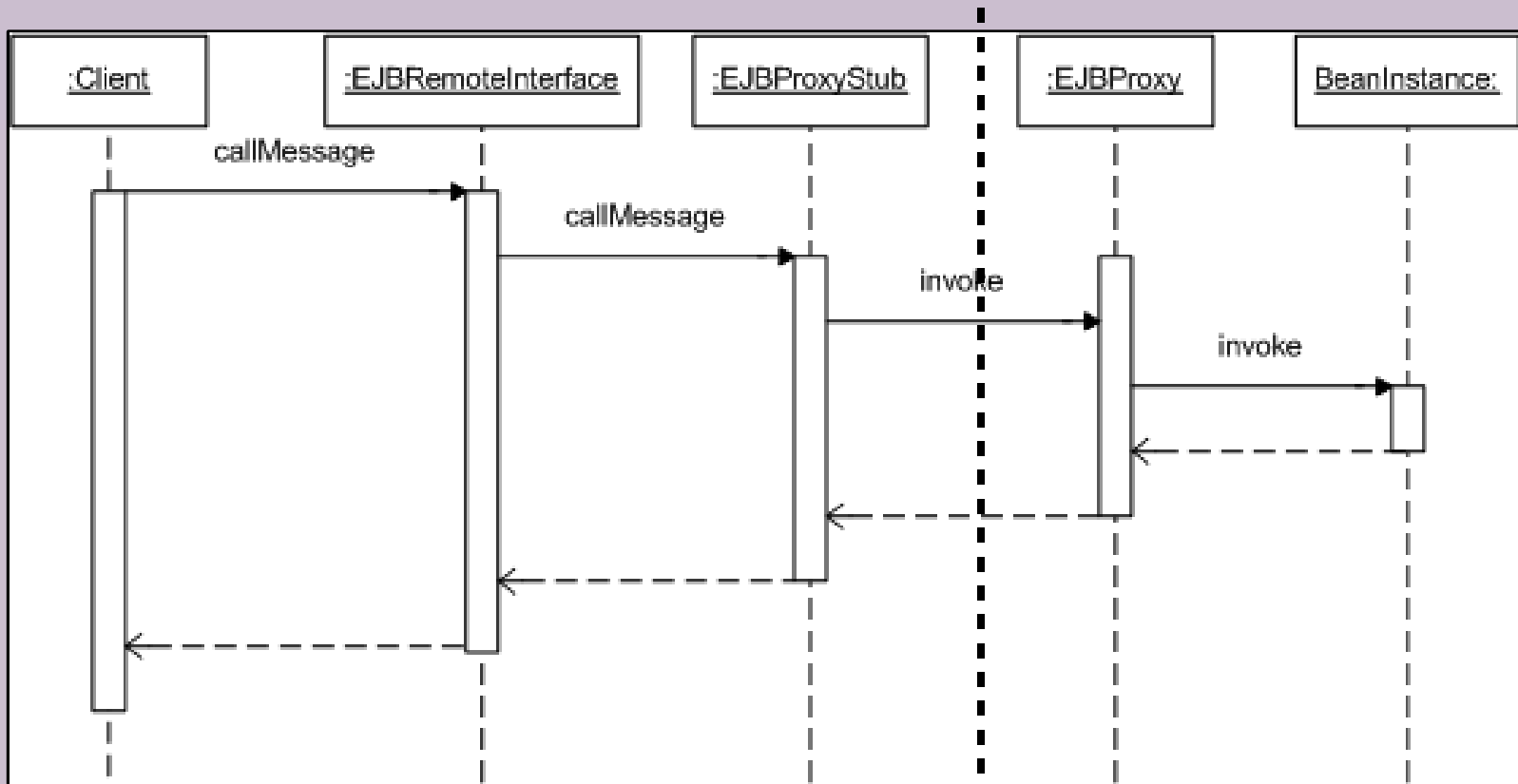
## The EJB Container

- proxy stub can be **generated by a precompiler**
- in case of JBoss, it is **dynamically generated at deployment time**
- EJB container **manages bean class instances, security and transactions**
- manages the **life cycle of the bean instance**
- **routes the request** from the *proxy* to the real bean class instance



# EJB Architectural Overview

## The EJB Container



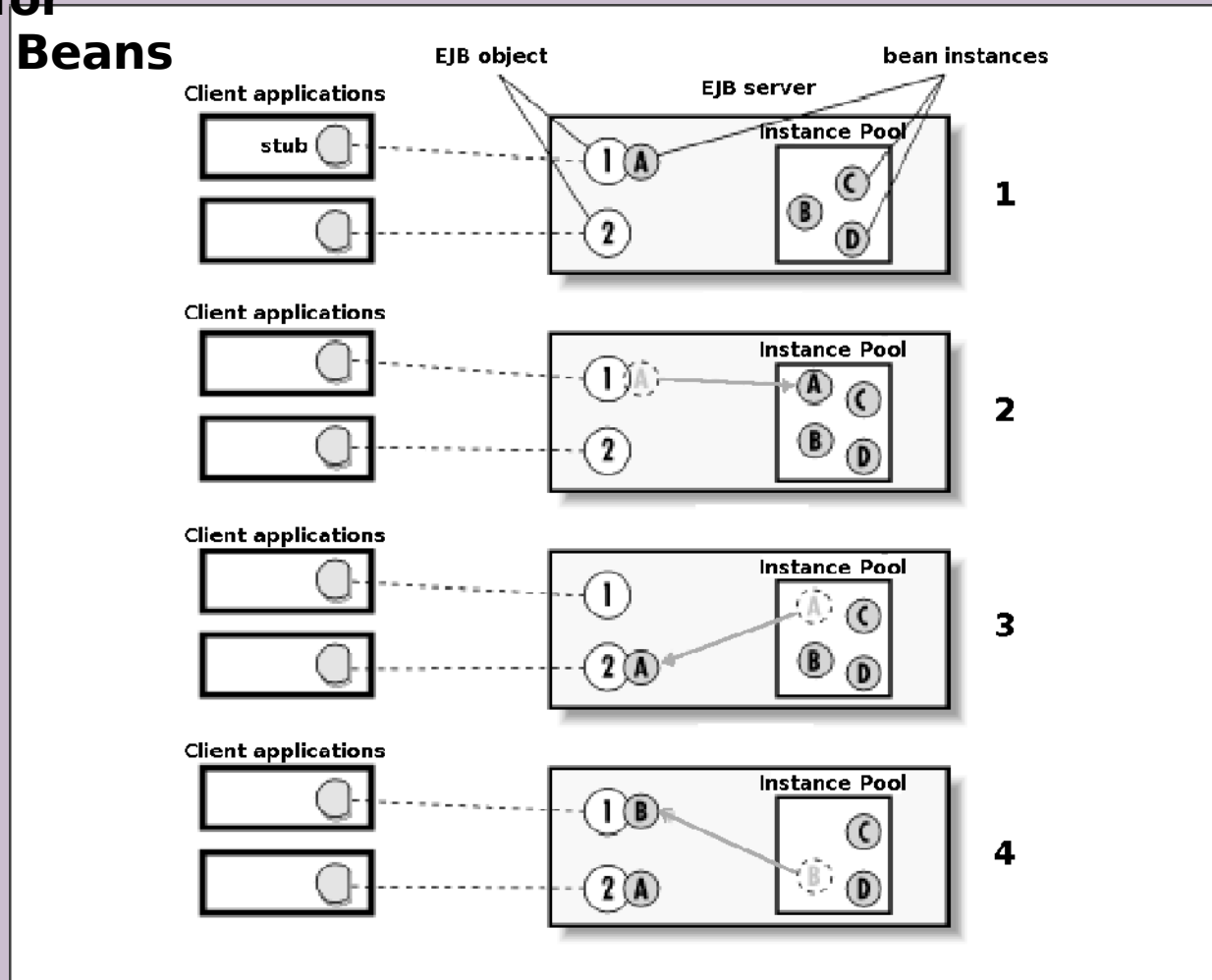
e.g. network



# EJB Architectural Overview

## The EJB Container

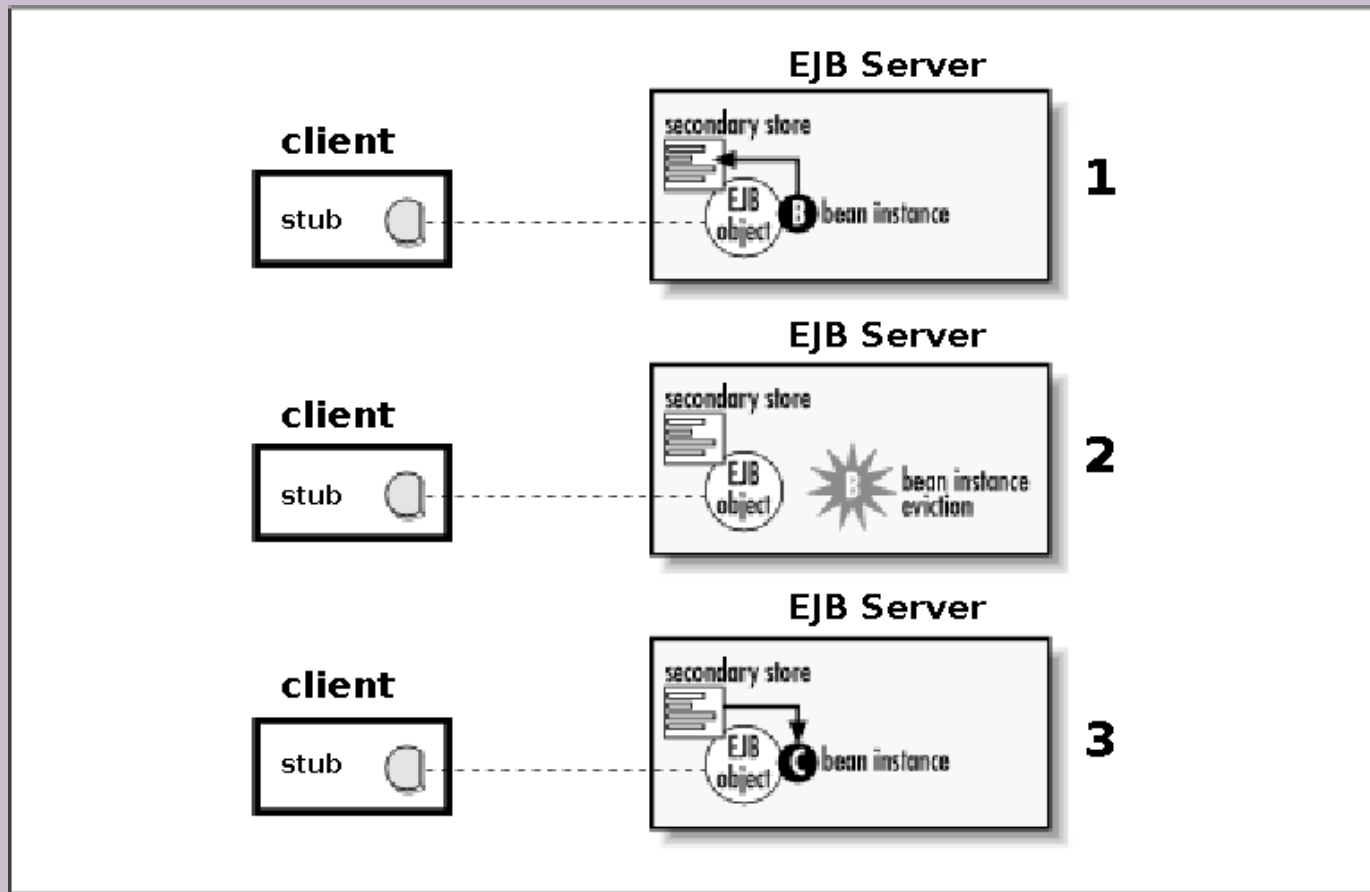
### Instance pooling for Stateless Session Beans



# EJB Architectural Overview

## The EJB Container

### Stateful Session Beans



# NetBeans Demonstration

**Be aware: No Step-by-Step!**



# JEE - Java Enterprise Edition

## EJBs - Architectural Overview - Questions?

