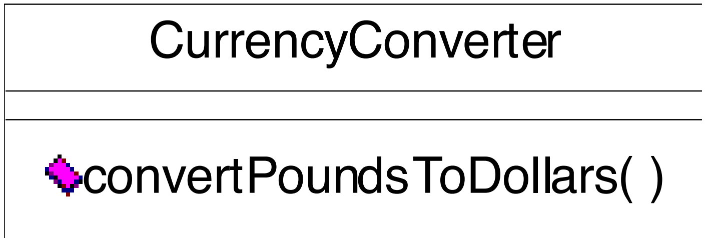
# Enterprise Java Beans

# Distributed System Scenario

- Financial Company (worldwide)
- 10000 Customers, on-line
- New software component:
  - » Currency Converter
  - » Heavily used (1000 hits/second)
- Design
  - » Business logic
  - » Distributed infrastructure

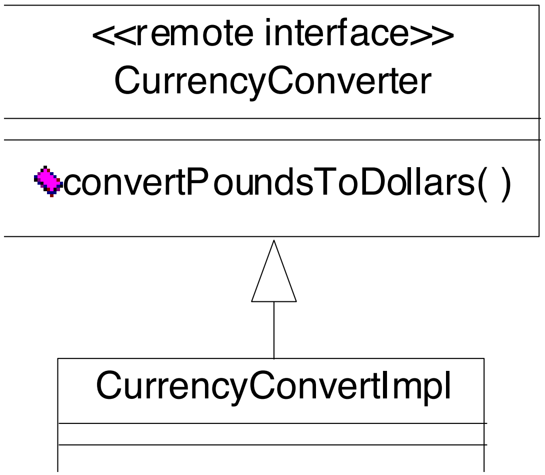
# Business Logic



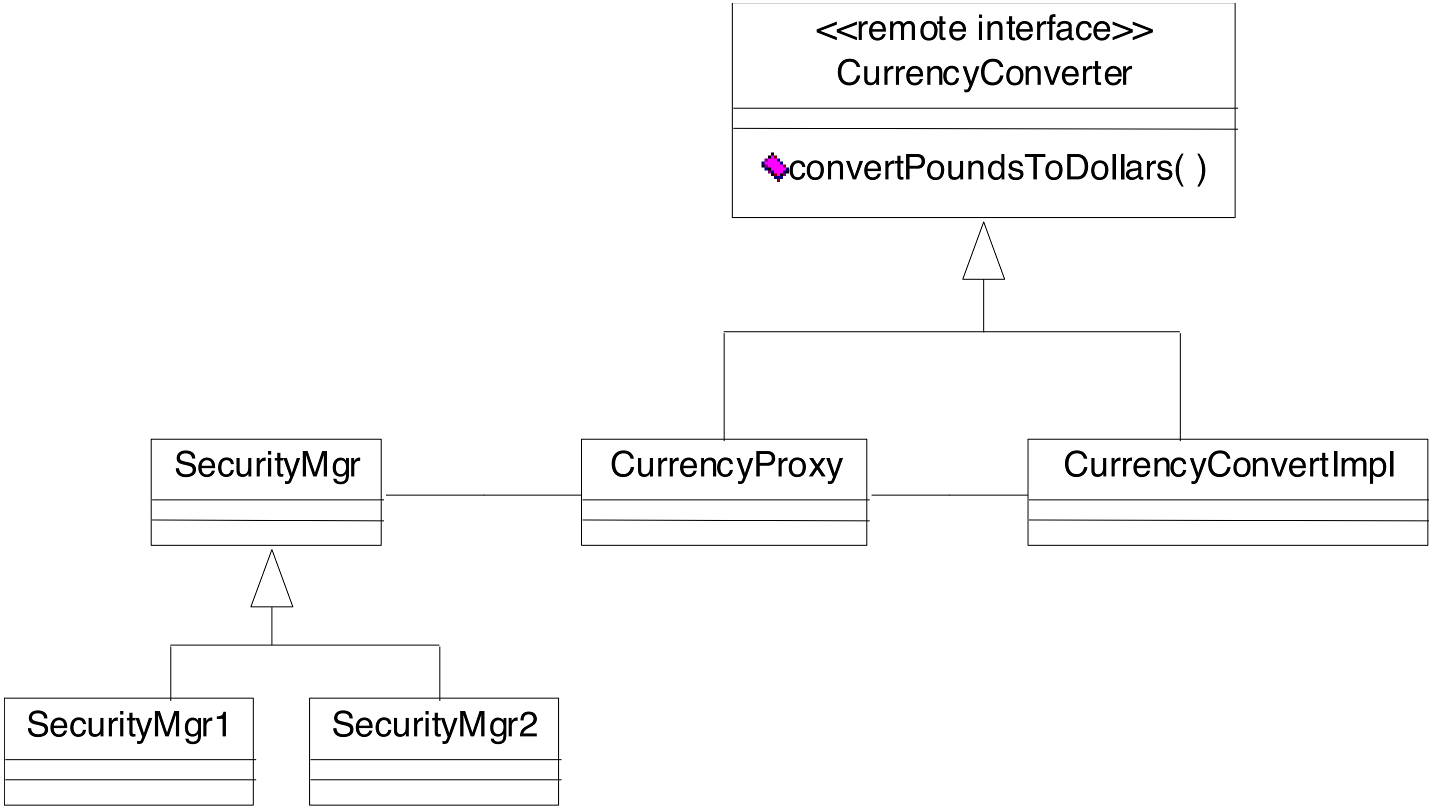
CurrencyConverter

convertPoundsToDollars( )

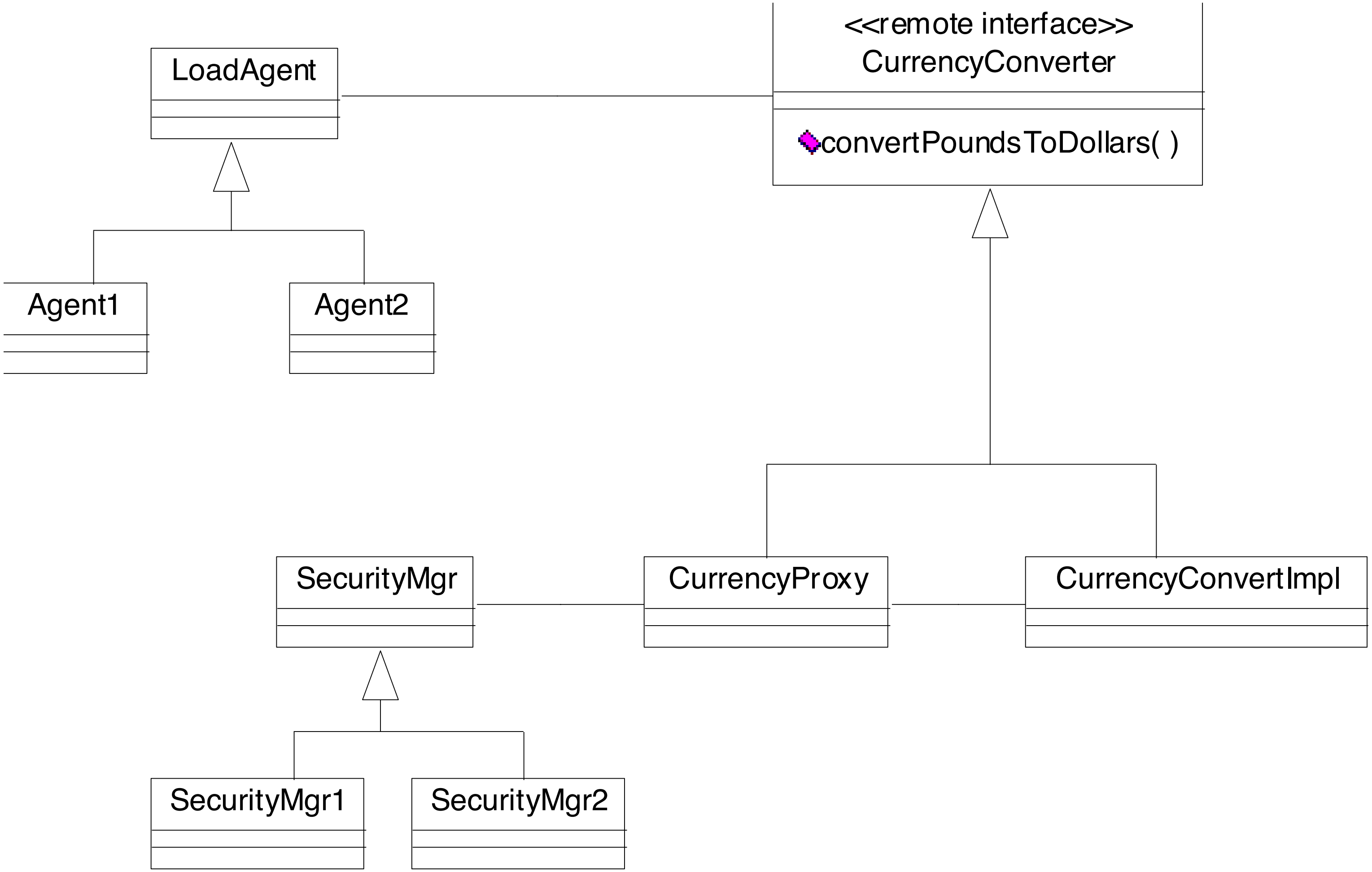
# Distribute the object



# Add Security Manager



# Add Load Balancing Agent



# Development Effort

(1) Business Logic

(2) Distributed Infrastructre

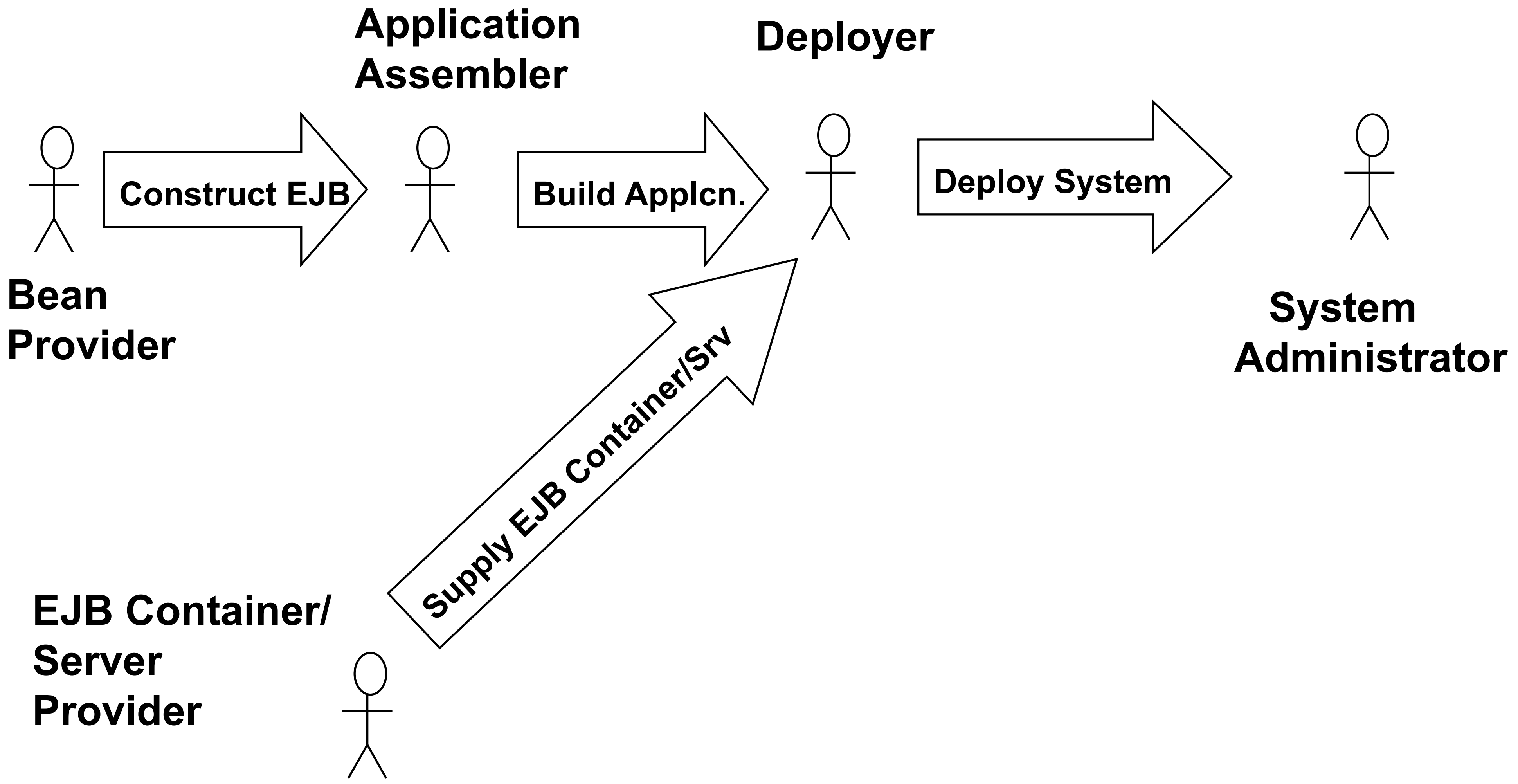
- » Security
- » Load Balancing
- » Transaction Management
- » Object-Relational Mapping

- ● EJB takes care of (2)
- ● Provides API & framework

# J2EE / EJB

- EJB is a server-side component architecture that enables and simplifies the process of building enterprise-class distributed object applications in Java
- You can write scalable, reliable, and secure applications without writing your own complex distributed object frameworks
- EJB is a specification

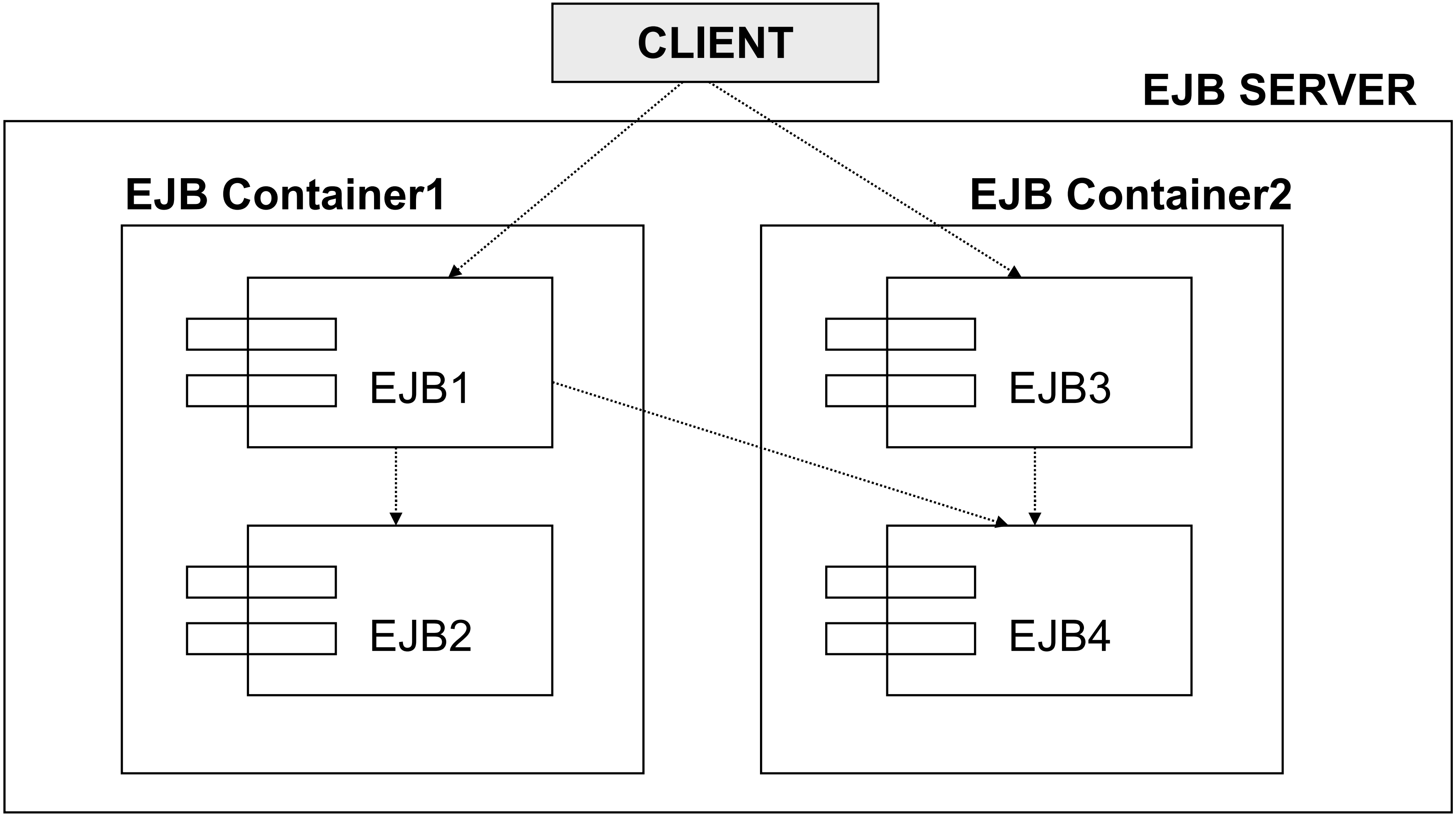
# The EJB Process



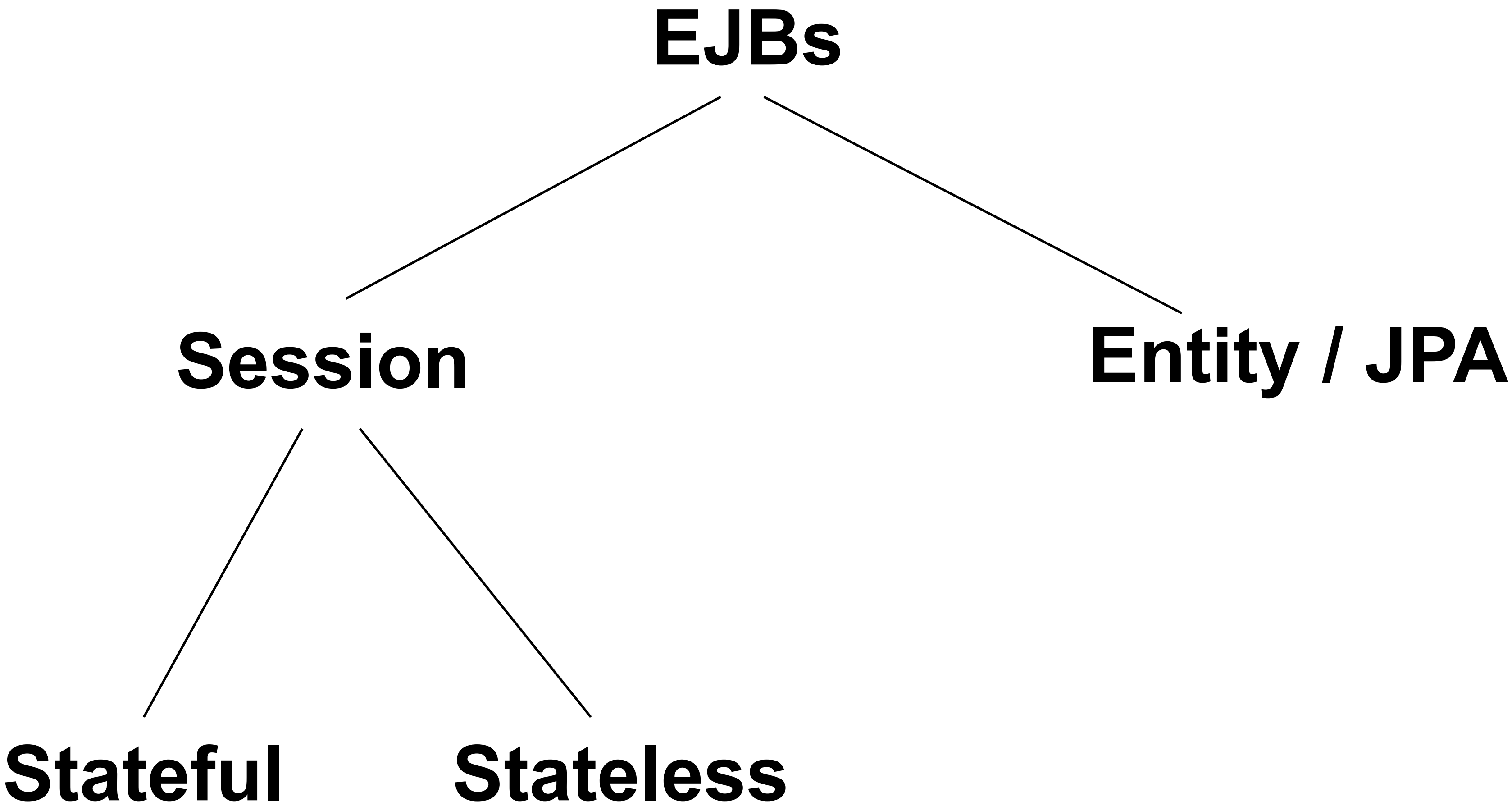
# EJB Server & Container

- Container:
  - » Where the EJBs run
  - » Responsible for managing EJBs
- EJB Server:
  - » Runtime environment for container(s)
  - » Manage low-level system resources

# EJB Server & Container



# EJ Bean Types



EJBs

Session

Entity / JPA

Stateful

Stateless

# Session Beans

- “Business Process Objects”
  - » Price quoting
  - » Order entry
  - » Video compression
  - » Stock Trades…
- Live for as long as the client’s session
- Usable 1 client at a time
  - » They are NOT shared
- EJB Server manages lifetime of beans

# Stateful Session Beans

- Business process that spans many requests
- “State” is kept
  - » Adding products to shopping cart
  - » Banking transactions for client
    - – Pay a bill
    - – Top up phone card
    - – Withdraw money

# Stateless Session Beans

- Single request
- No state kept
- Examples:
  - » Currency converter
  - » Compression utility
  - » Credit card verification

# Entity Beans / JPA

- Represents persistent data
- “Object-oriented in-memory view of data in an underlying data store”
- Long lasting and Shared access
- Subtypes of Entity Beans
  - » Bean Managed Persistent Entity Beans
  - » Container-Manager Persistent Entity Beans

# Bean Managed Persistence

- Must be persisted manually…
- Must look after:
  - » saving
  - » loading
  - » finding
- Use persistence API
  - » JDBC
  - » SQL/J

# Container Managed Persistence

- Automatic persistence
- Container/server looks after:
  - » loading
  - » saving
  - » finding component data
- Must describe what you want persisted
- Deployment tool support
  - » Define simple object/relational mapping

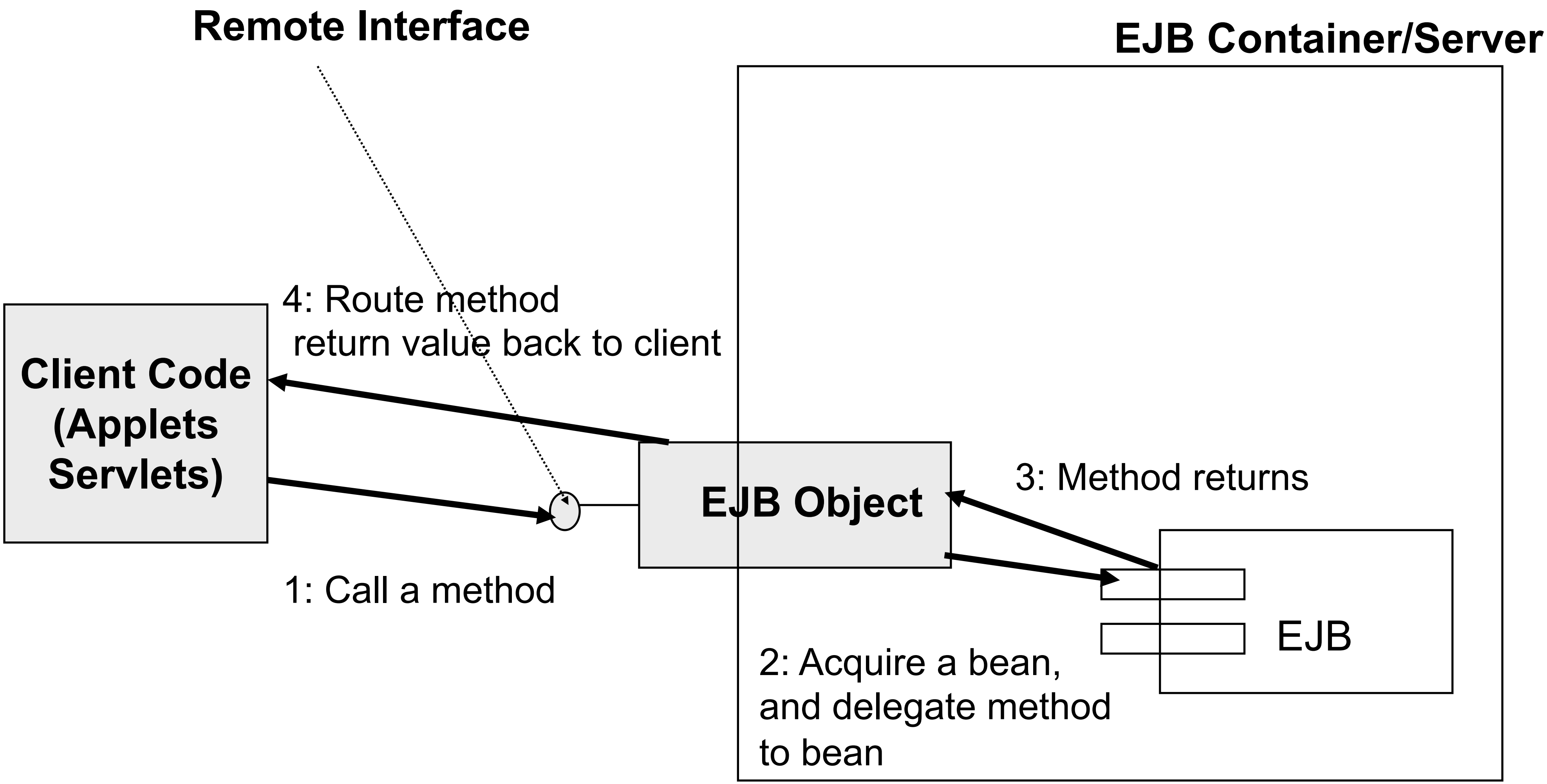
# EJB Object

- Client never invokes the bean instance
- Invocation:
  - » Intercepted by the container
  - » Delegated to the bean instance
  - » A network-aware wrapper object
  - » A layer of indirection between client and bean
  - » A surrogate object
  - » “Glue” between client and bean

# Remote Interface

- EJB Objects must:
  - » Clone every business method that your bean class exposes
  - » Which methods?
  - » Specified in the *remote interface*
- All remote interfaces derive from:
  - » javax.ejb.EJBObject

# EJB Objects


Remote Interface

EJB Container/Server

Client Code (Applets Servlets)

EJB Object

EJB

1: Call a method

2: Acquire a bean, and delegate method to bean

3: Method returns

4: Route method return value back to client

# Session Bean Interface

- Implemented by all session beans
- Lifecycle methods that may be implemented in the bean:
  - » setSessionContext
  - » ejbCreate
  - » ejbRemove
  - » ejbPassivate
  - » ejbActivate

# Finding an object

- Java Naming & Directory Interface
- Steps:
  - » Associate the resource (e.g. bean) with a nickname when deploying
  - » Clients of the bean can use this nickname to look up the resource across a deployment

# Client Code

- Look up reference in JNDI
- Call business methods on EJB object

Example to follow…