

# Overview of J2EE Architecture and Technologies

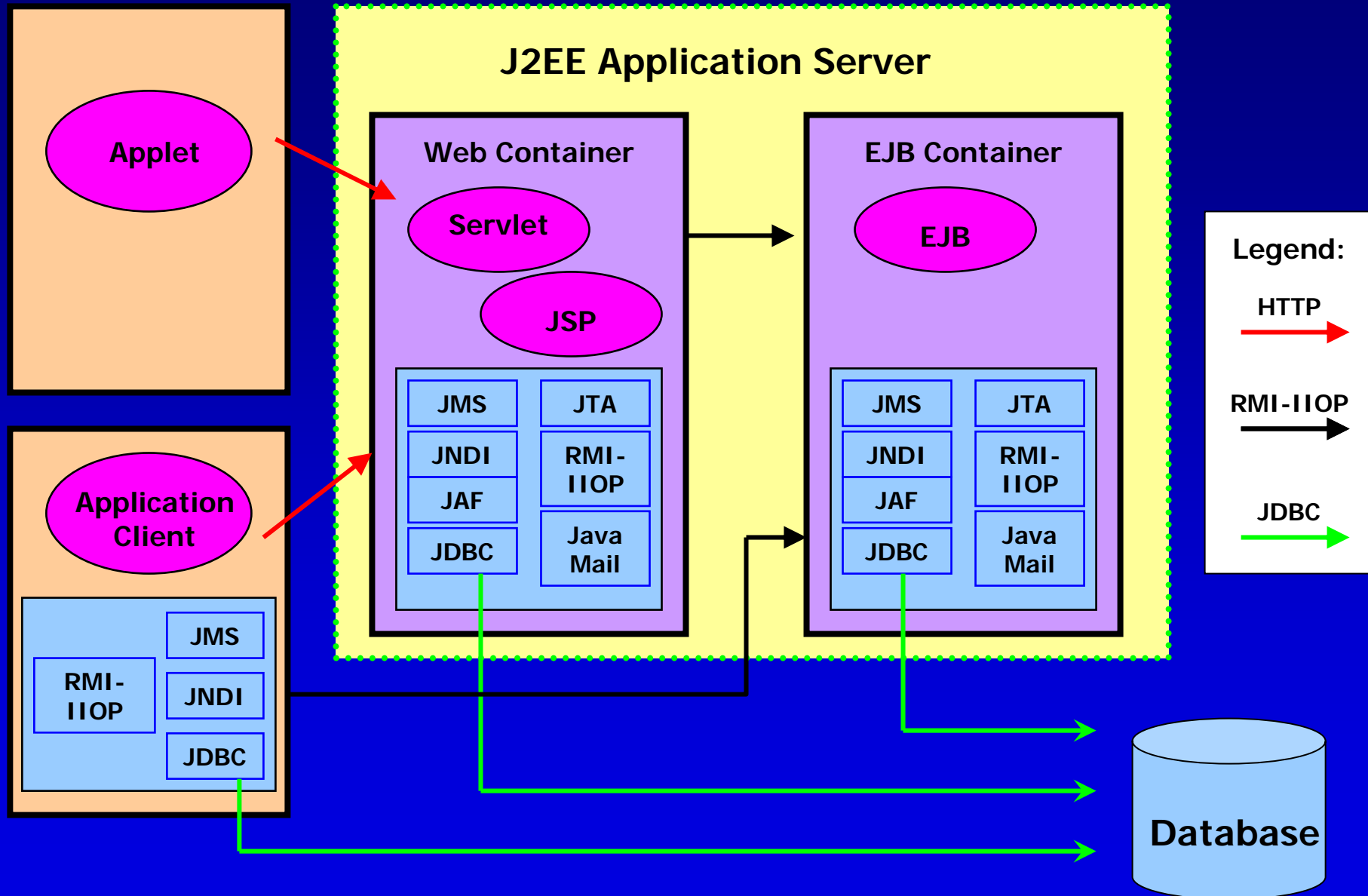
Objective:

The gain a good understanding of J2EE Architecture and technologies



3

# J2EE Architecture



# J2EE Architecture

- J2EE Server
  - Runtime portion of a J2EE product
  - Provides EJB and Web containers
- Types of containers
  - EJB container – manages the execution of the enterprise beans(EJB) for J2EE application
  - Web container – manages the execution of servlet and JSP page for J2EE application
  - Application client container – manages the execution of application client components. Application client and their container run on the client
  - Applet container – manages the execution of applets

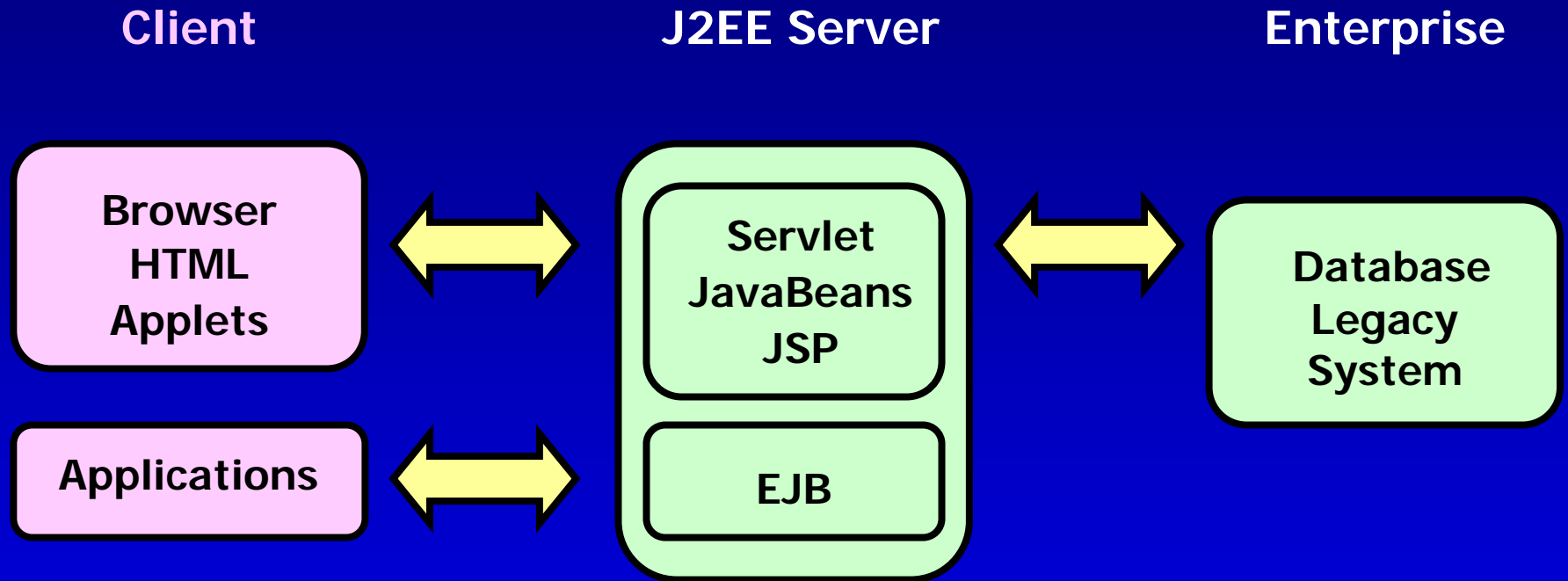
# Communication Technologies

- The J2EE specification requires support for the following types of communication technologies
  - Internet Protocols
    - TCP/IP
    - HTTP 1.0
    - SSL 3.0
  - Remote Method Invocation Protocol
  - Object Management Group Protocol
  - Java IDL
    - Invoke operations on CORBA objects
  - RMI-IIOP
    - Implementation of the RMI API over IIOP allows application providers to write remote interface in Java programming language

# Messaging Technologies

- Provide a way to asynchronously send and receive messages
- The APIs are
  - Java Message Service (JMS)
    - Provides interface for handling asynchronous requests, reports, or events consumed by the enterprise application
  - JavaMail API
    - Provides interface for sending and receiving messages intended for users
  - Java Activation Framework (JAF)
    - Provides mechanism to map filenames extension to MIME types
    - Used by JavaMail API to handle data included in the email messages

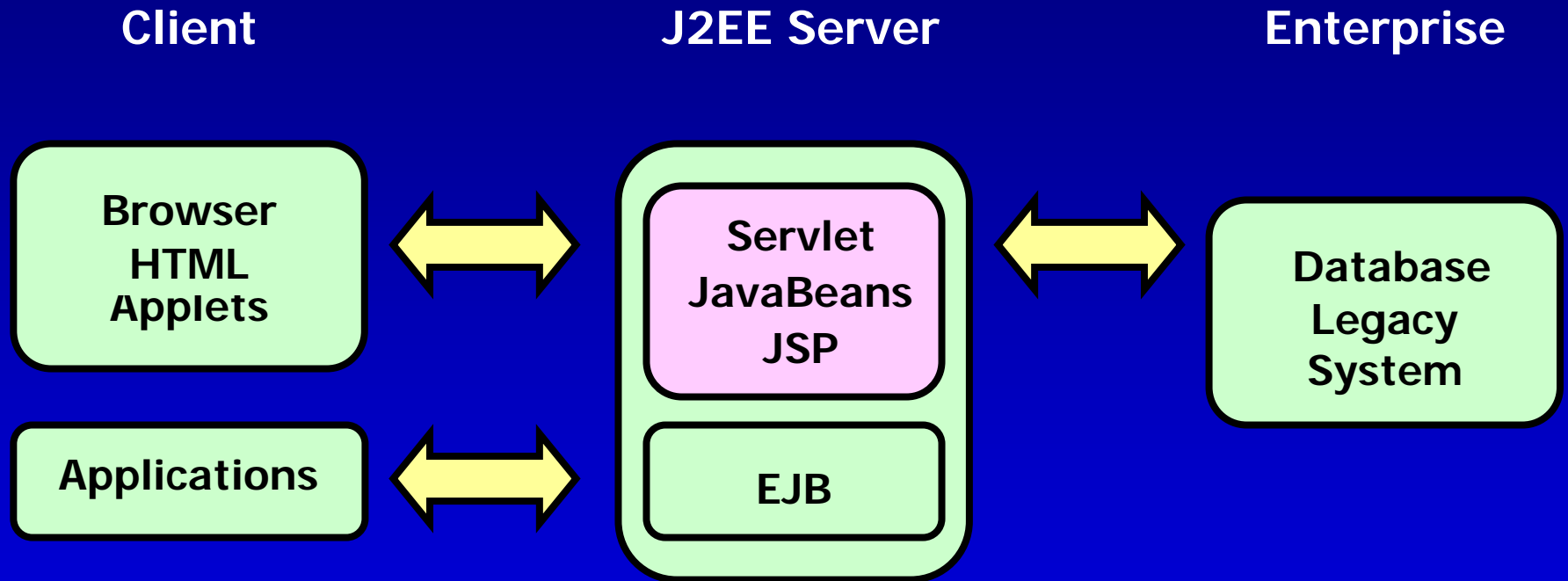
# Client Tier



# Client Tier

- Client tier support for a variety of client type
  - Web browser
    - HTML
    - JavaServer Pages (JSP)
  - Stand-alone Java application
  - Non-Java clients

# Web Tier



# Web Tier

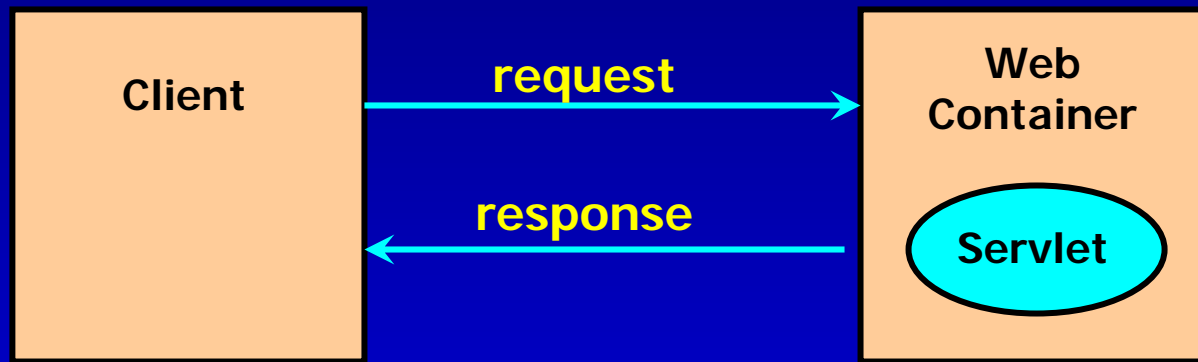
- Performs the following functions in a J2EE application
  - Manages the user interaction and the application business logic
  - Generate content dynamically based on the user's request
  - Controls the screen flows
  - Supports multiple and future client types
- Operational Characteristics
  - Accept the incoming HTTP request
  - Return the resulting HTML page

# Web Tier

- Technologies
  - Servlets and JSP Technologies
  - Custom Tags
  - JavaBeans Components
- Frameworks
  - Model-View-Controller
  - Struts

# HTTP Servlets

- Java classes that are invoked by the Web server to generate dynamic Web pages in response to Web browser requests



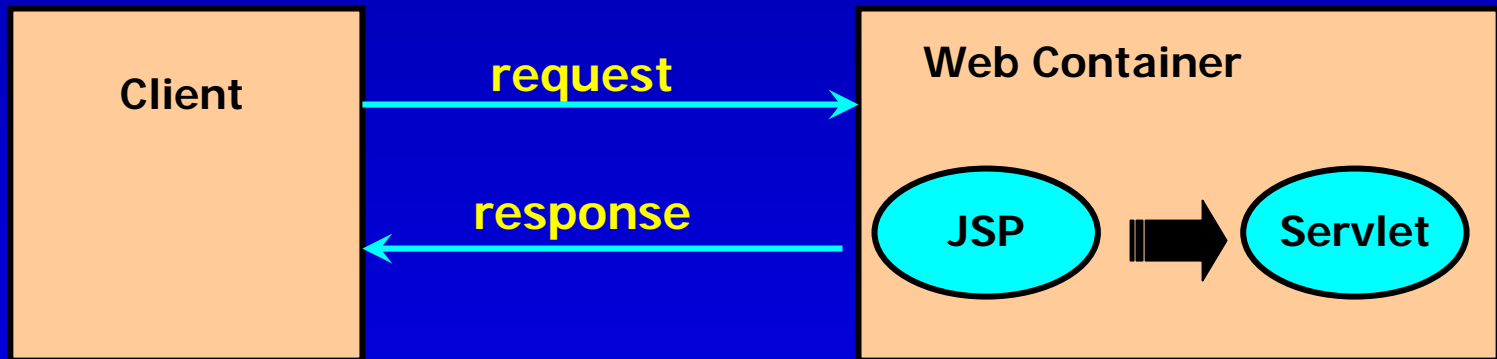
- Main function
  - Accepts request from the client
  - Process the request
  - Return a response to the client (optional)

# HTTP Servlet

- Design Guidelines
  - Use as information service
  - Use an controller
  - Generate binary content
  - Avoid writing servlets that prints static text
  - Use of RequestDispatcher methods
    - forward() – to delegate processing of an entire request to another component
    - Include() – to build response results containing from multiple web resources

# JavaServer Pages

- An HTML file with
  - Static HTML
  - Dynamic HTML
    - Can embedded Java code for dynamic content
- The web container will convert the JSP to a Servlet
- Simplified way of creating a Servlet

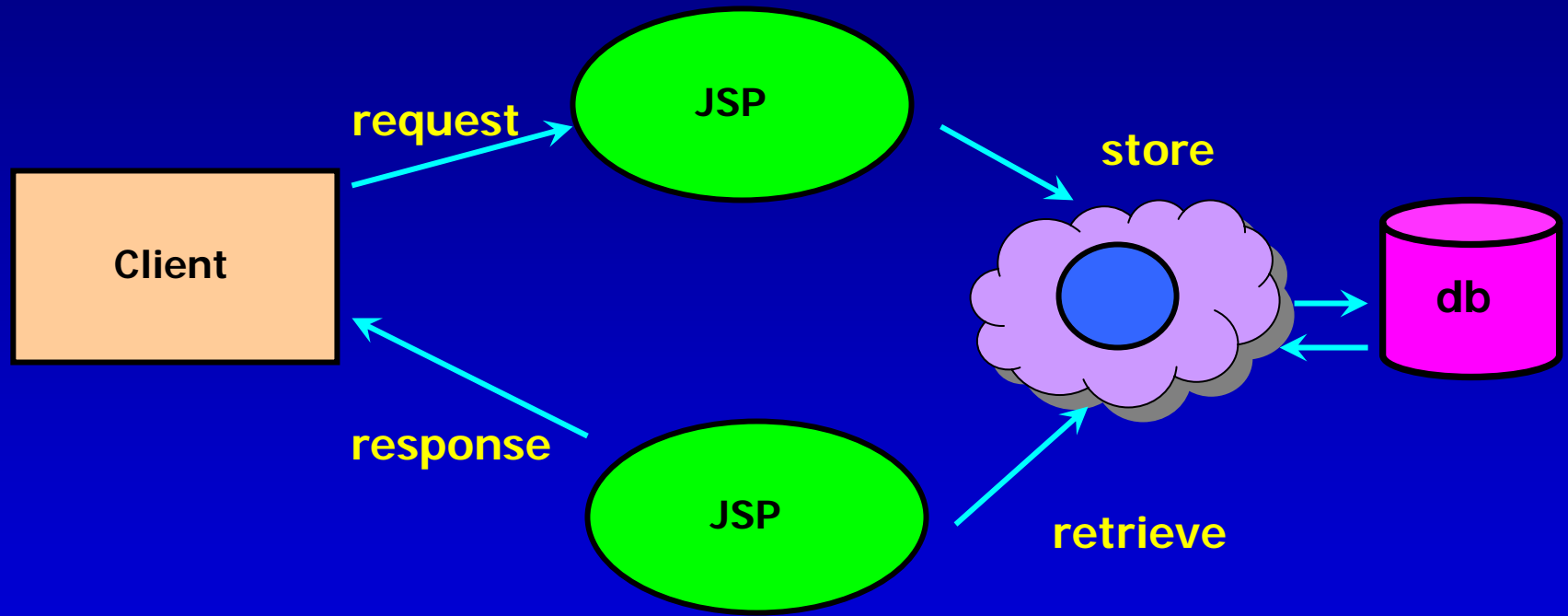


# JavaServer Pages

- Design Guidelines
  - Use for data presentation
  - Use to generate XML document
  - Use to generate unstructured textual content
  - Use of JSP include directive and tags
    - Include directive
      - Translation time
      - Used to modularize web pages
      - To reuse content
      - Keep web page size manageable
    - JSP include tag
      - Occurs each time the JSP page is served
      - Used to insert dynamic content at runtime
      - Flexible but required runtime processing

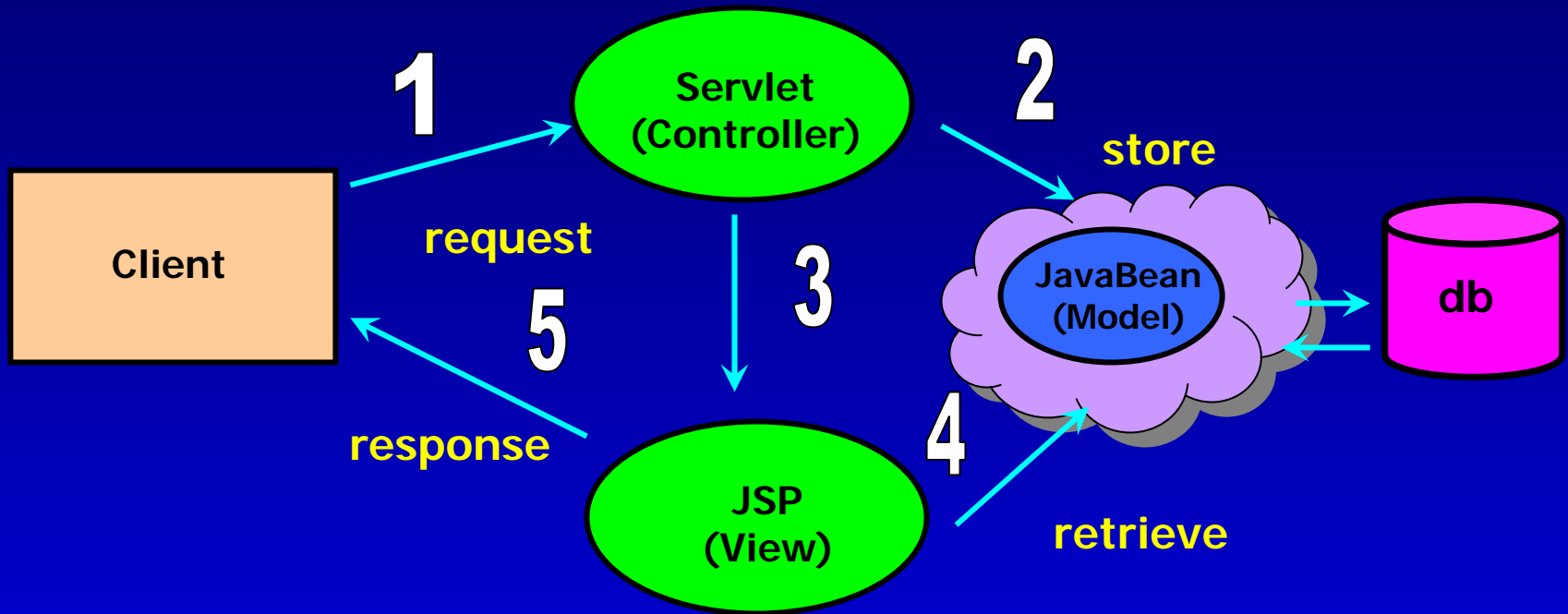
# Model 1 Architecture

- Known as JSP-centric



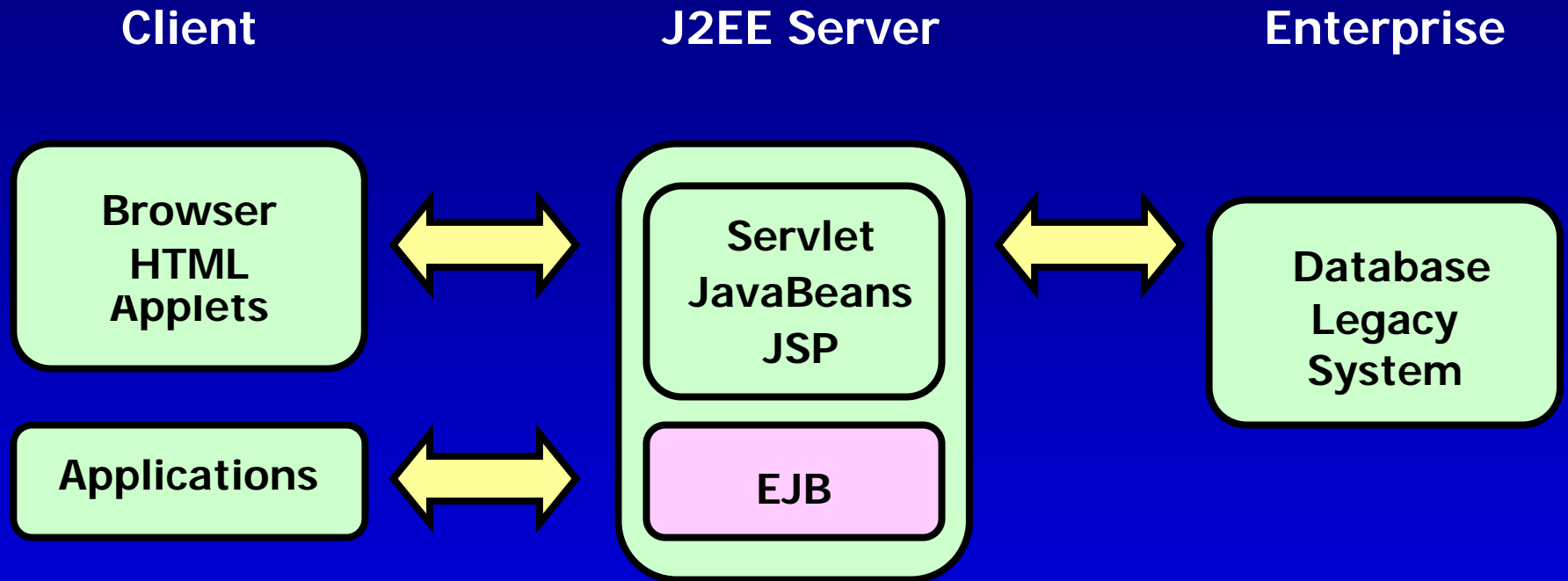
# Model 2 Architecture

- Known as Model-View-Controller



- Model - information the application manipulates
- View – implement the visual display of the model
- Controller – receives all input from the user, decides what they mean, and what to do

# EJB Tier



# EJB Tier

- Provides the EJB tier as the standard server-side distributed components
  - Session Beans
    - State is valid to the client who creates them
  - Entity Beans
    - Object oriented view of the entities that are persistent
  - Message Driven Beans
    - Process asynchronous messages delivered via the Java Message Service (JMS) API

# EJB Tier

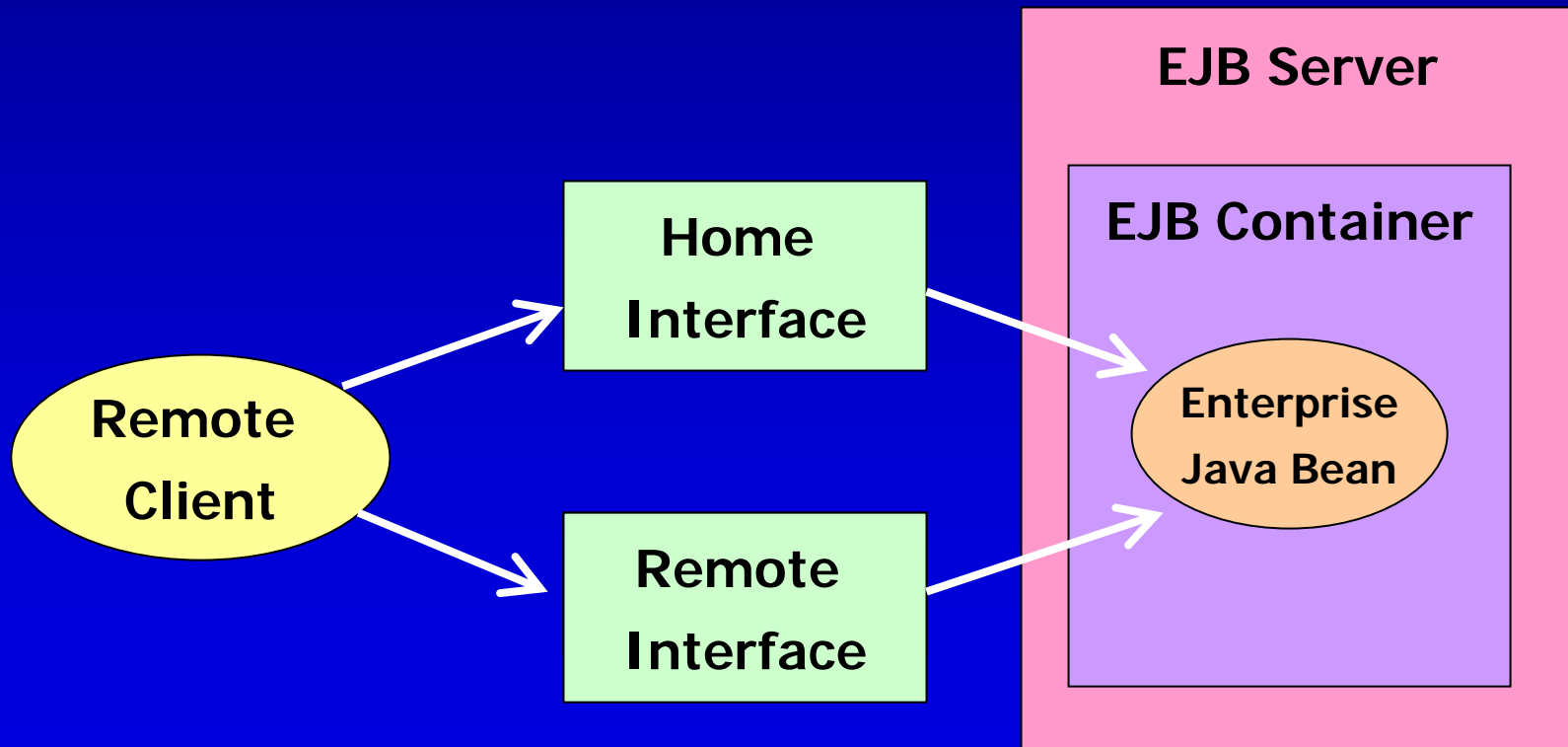
- Operational Characteristics
  - Wraps the individual components so that the container intercepts all requests to them and all responses from them
  - This wrapping mechanism enables the container to perform its work like
    - Checking security credentials
    - Transaction handling
    - Resource management

# Enterprise JavaBeans

- Sun's definition of EJB
  - A component architecture for development and deployment of component-based distributed business applications
- Types of EJB
  - Session Beans
    - Stateless Session Bean
    - Stateful Session Bean
  - Entity Bean
    - Container Managed Persistence (CMP)
    - Bean Managed Persistence (BMP)
  - Message Driven Beans

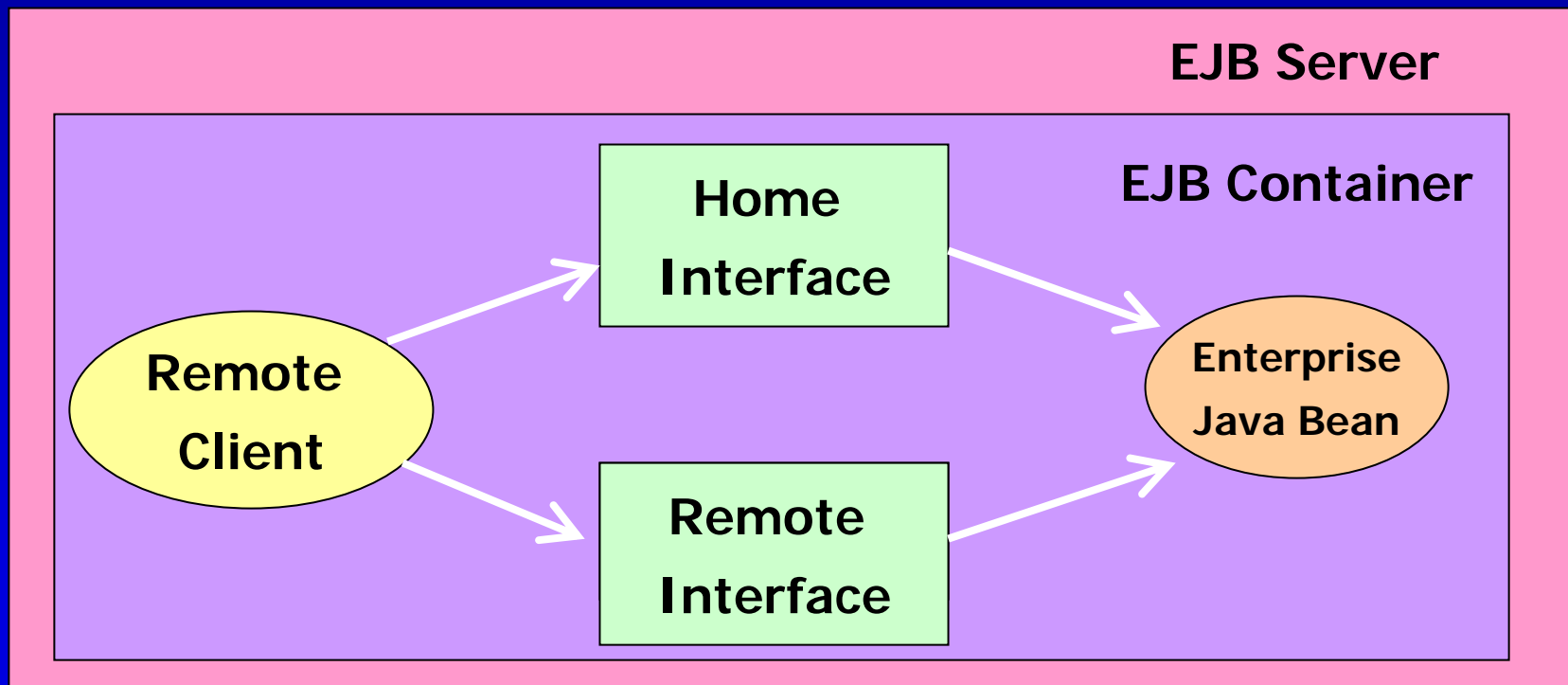
# Remote vs Local Client View

- Remote Client View
  - Designed to be used in distributed environment
  - Client reside in the different JVM
  - Network overhead for each method call



# Remote vs Local Client View

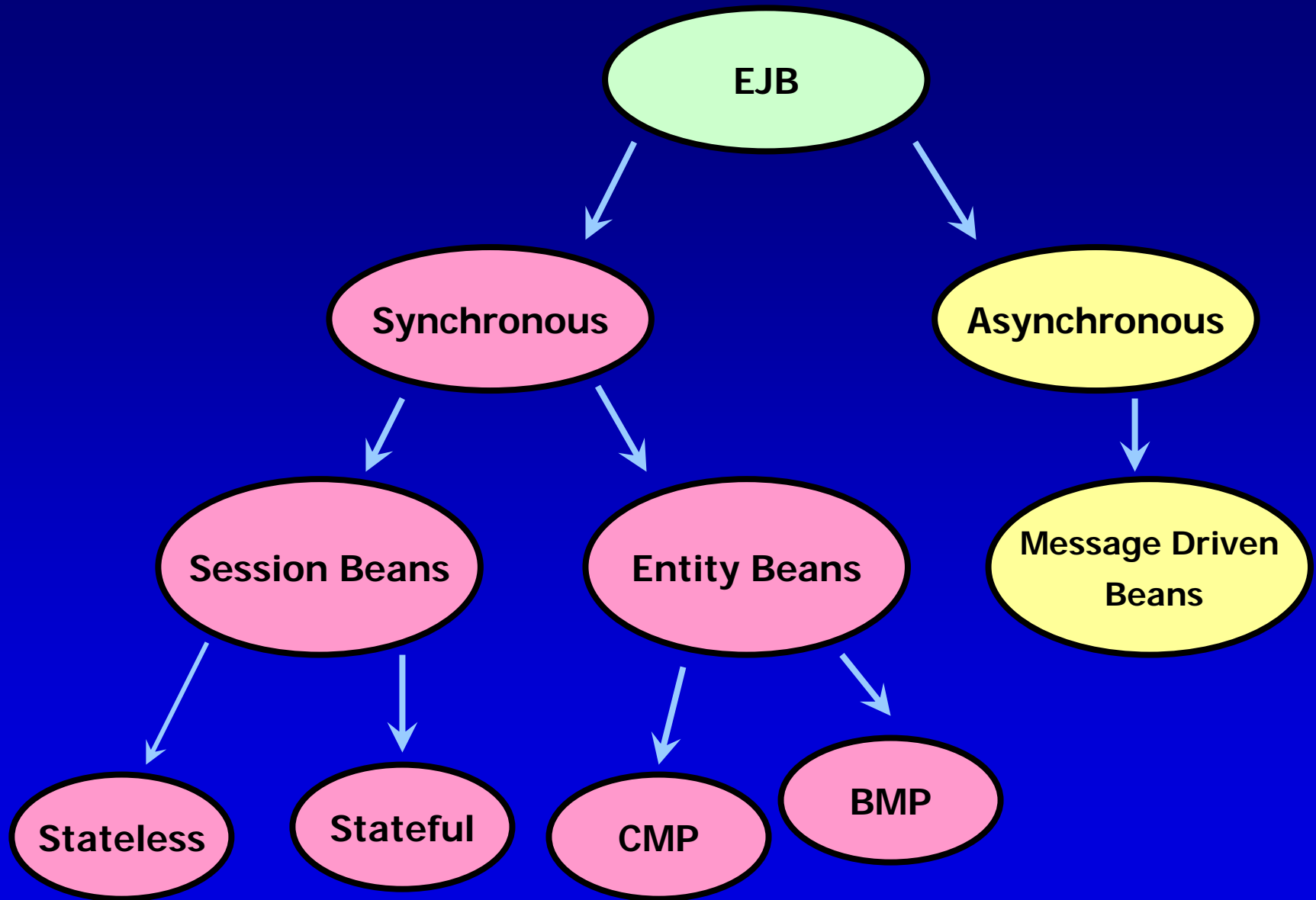
- Local Client View
  - Enterprise bean and the client must reside on the same JVM
  - Invoke local method call directly
  - Avoids the remote invocation overhead



# Remote vs Local Client View

- Using Remote and Local Client View
  - Remote Client View
    - Requires location independence
    - Loose coupling between the a bean and its client is desired
    - Needs to use the pass-by-value in parameter passing between the bean and the client
  - Local Client View
    - Required to be deployed in the same container
    - Needs to use the pass-by-reference in parameter passing between the bean and the client

# Types of Enterprise JavaBeans



# Enterprise JavaBeans

- EJB Session Beans
  - Represents a single client
  - Models business process
  - Does not recover from a server failure
  - May have only one client
  - There are two flavors:
    - Stateless Session Bean
    - Stateful Session Bean

# Enterprise JavaBeans

- Design Guidelines Using Session Beans
  - Stateful Session Bean
    - Needs to maintain client-specific state
    - Short-lived and represents non-persistent object
  - Stateless Session Bean
    - Provides server-side behavior
    - Provides some generic service does not need to maintain any client-specific state
    - Provides the procedural view of data
    - Provides high performance and scales better

# Enterprise JavaBeans

- EJB Entity Beans
  - Model business data
  - Represent a business entity object
  - Persistent, available for subsequent use even after the application that created it terminates
  - May be shared by multiple clients
  - Can survive a system crash
  - There are two flavors:
    - Container-Managed Persistence
    - Bean-Managed Persistence

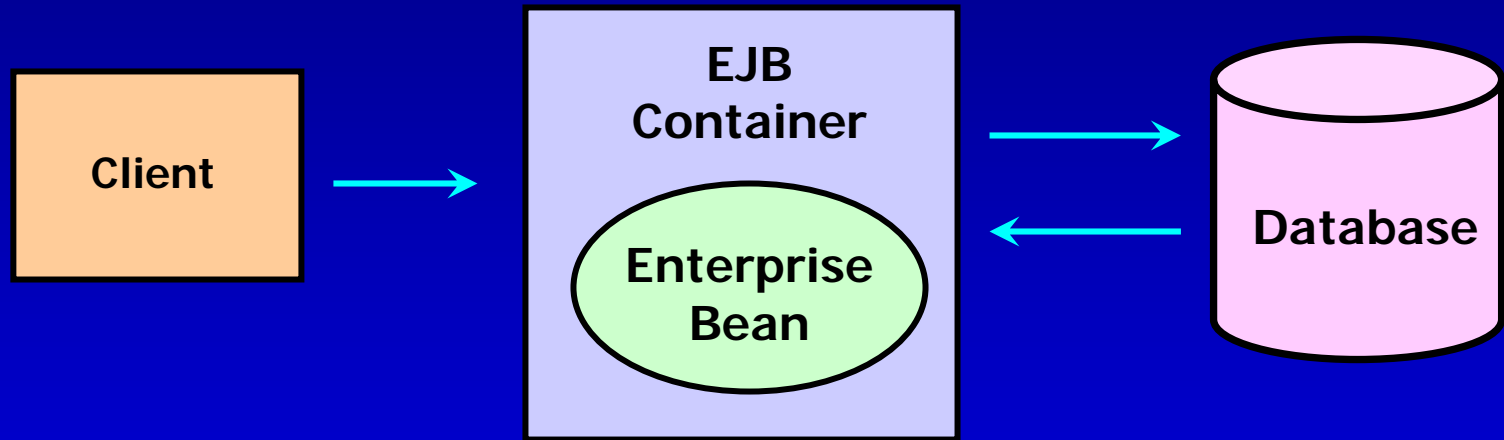
# Enterprise JavaBeans

- Design Guidelines Using Entity Beans
  - Business object needs to be stored in persistent storage
  - Unique identity is important in the application
  - Multiple clients needs to share the state and behavior of the business object
  - Needs to be located by the container based on its state

# Enterprise JavaBeans

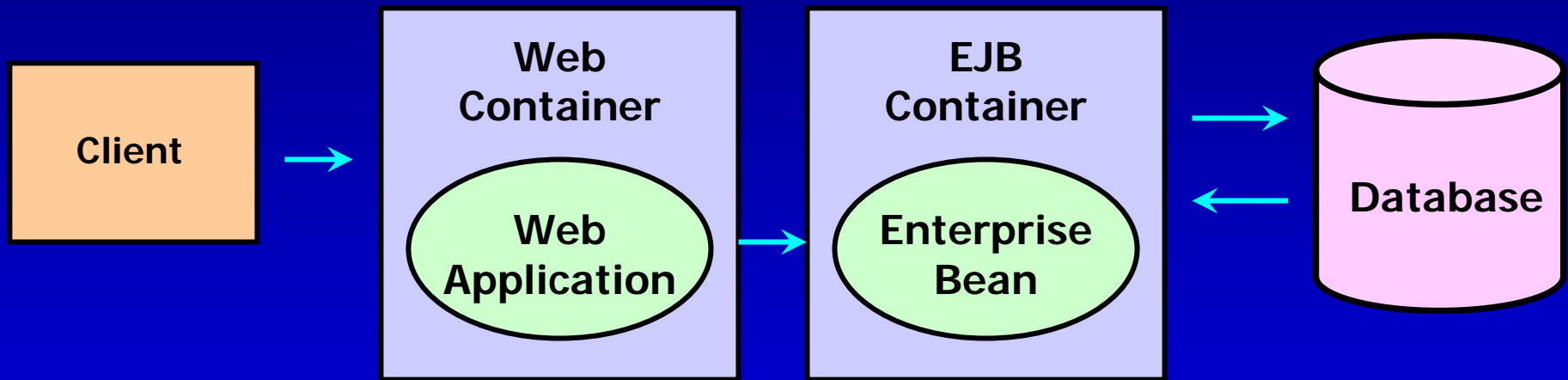
- Message Driven Beans
  - New feature of the EJB 2.0 specification
  - Used for processing of asynchronous JMS messages within the J2EE based application
  - Invoked by the container when a message is received
  - Properties
    - They have no identity to the client
    - Completely managed by the EJB container
    - Stateless, do not maintain any state of behalf of the client
  - Must implement the following:
    - `jms.MessageListener`
    - `javax.ejb.MessageDrivenBean`

# EJB Application Architecture

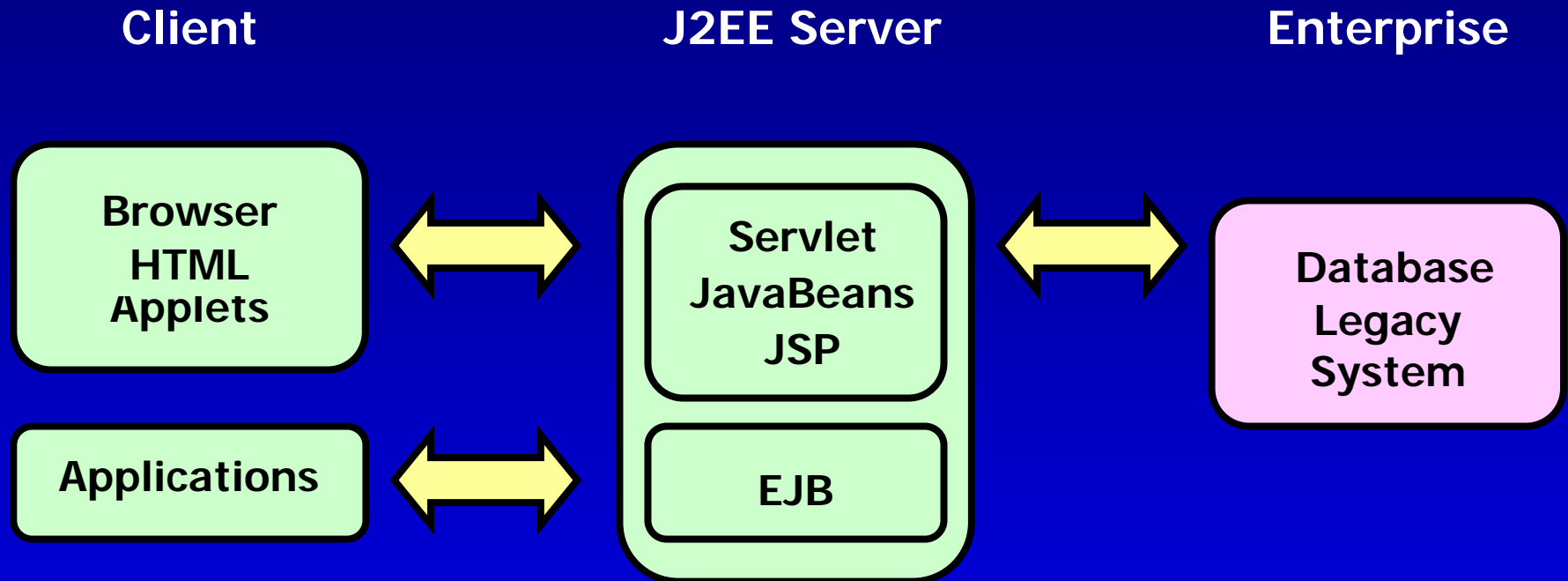


# EJB Application Architecture

- When the client is a Servlet or JSP



# EIS Tier



# Problems in EIS Tier

- EIS stands for Enterprise Information System
- No standard infrastructure for communicating heterogeneous system
- Includes
  - Application integration
  - Data integration
    - Contains more than one database system. Database can be relational, object based, file based, etc.
  - Legacy integration
    - Integrating new enterprise application with applications and EISs that have been in operation for some time

# J2EE Integration Technologies

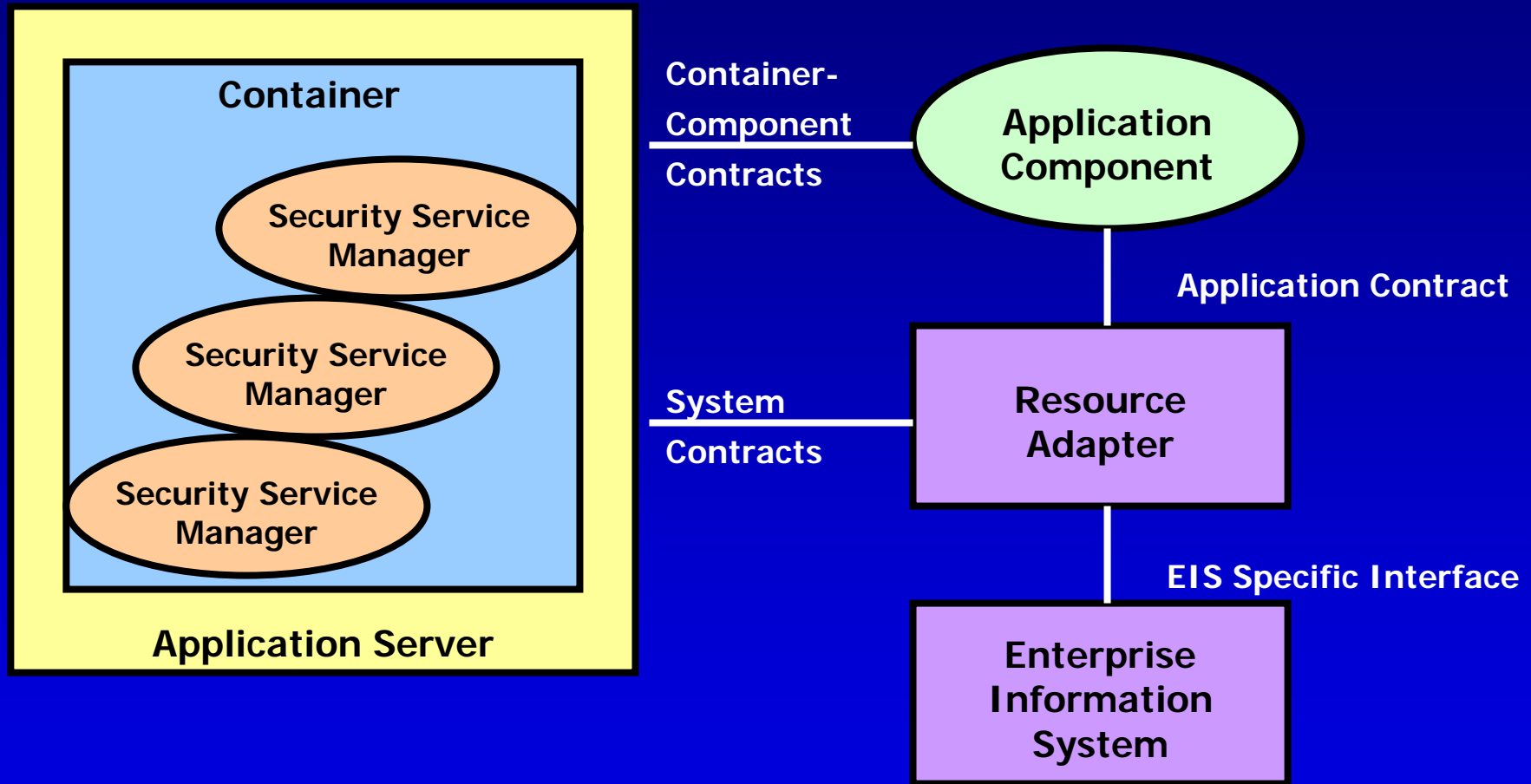
- To address the EIS integration problems, J2EE provides the following technologies
  - J2EE Connector Architecture
  - Java Message Service (JMS)
  - JDBC API

# Java Connector Architecture

- Adds simplified EIS integration to the J2EE platform
- Enables the EIS vendor to provide a standard resource adapter for its enterprise information system
- Pluggable to any J2EE compliant application server
- Defines two types of contracts
  - System level
    - Exists between a J2EE application server and a resource adapter
  - Application level
    - Exists between an application component and a resource adapter

# Java Connector Architecture

Security Service Manager

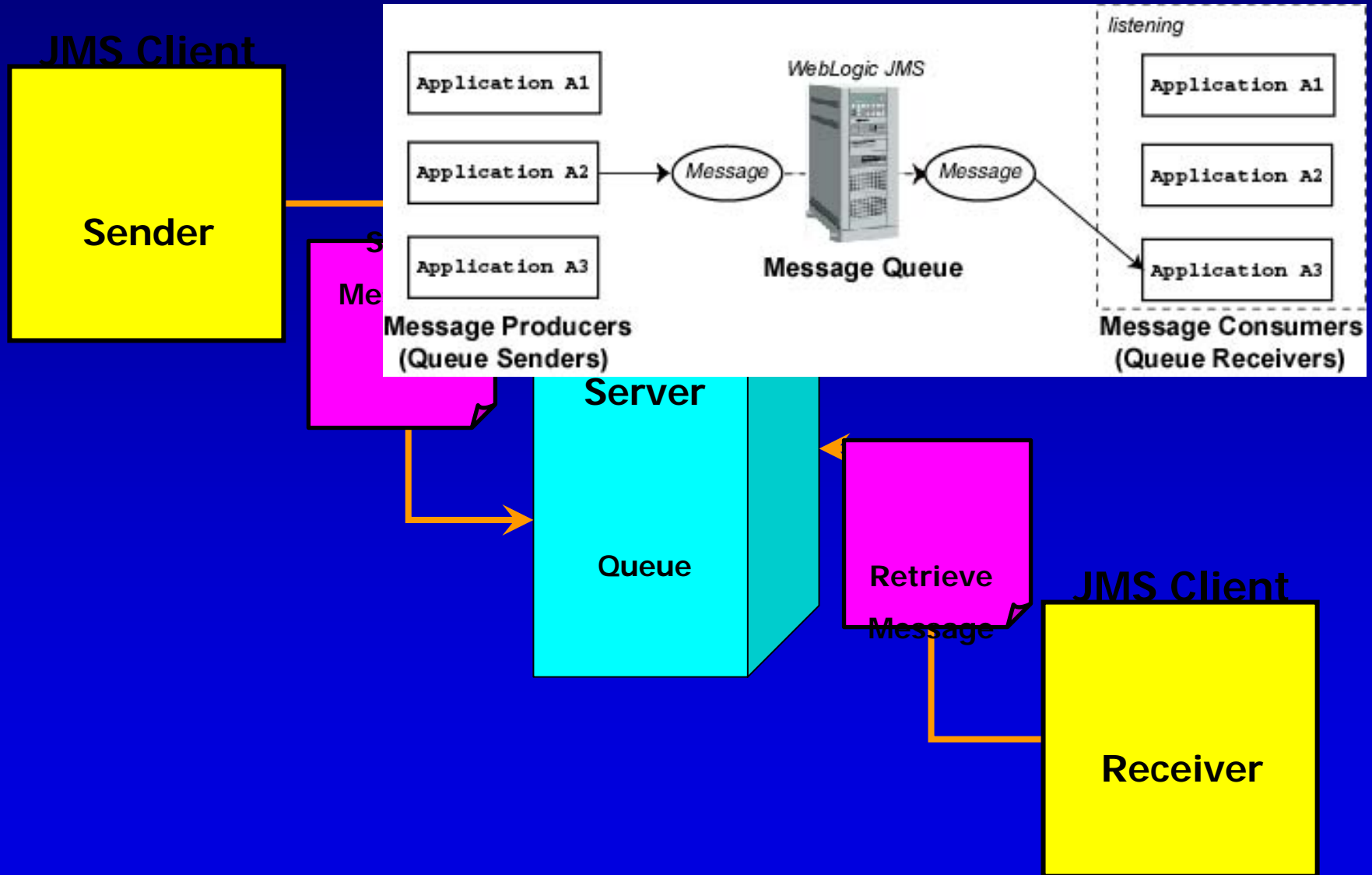


# Java Messaging Service

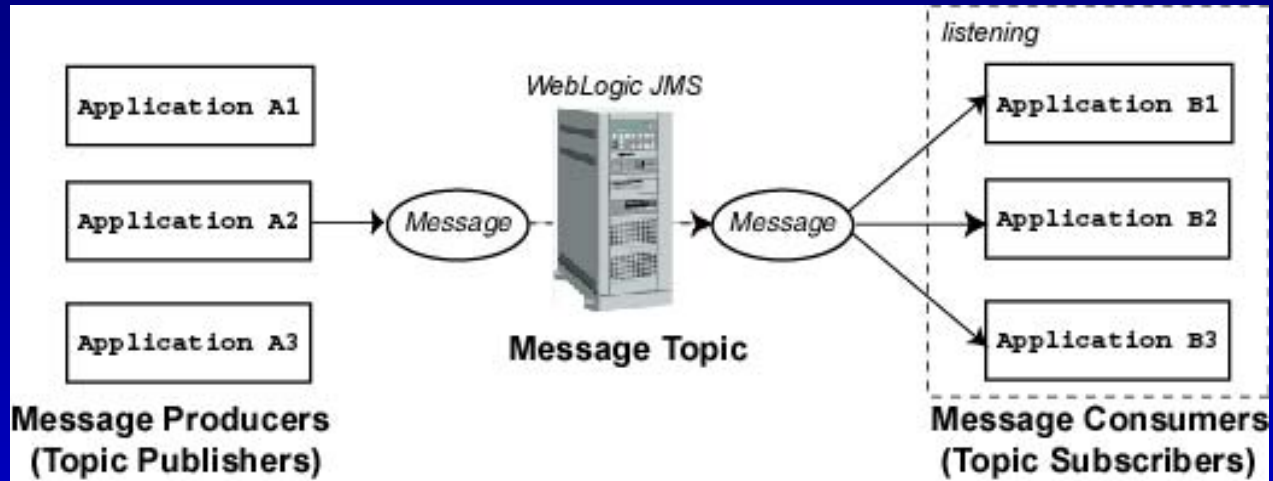
- A messaging standard that allows J2EE application components to create, send, and read messages
- Types of Messaging
  - Topic (Pub/Sub)
    - Topic Publishers
    - Topic Subscriber
  - Queue (Point-to-Point)
    - Queue Senders
    - Queue Receivers

# Java Messaging Service

- Point-to-Point (P2P)

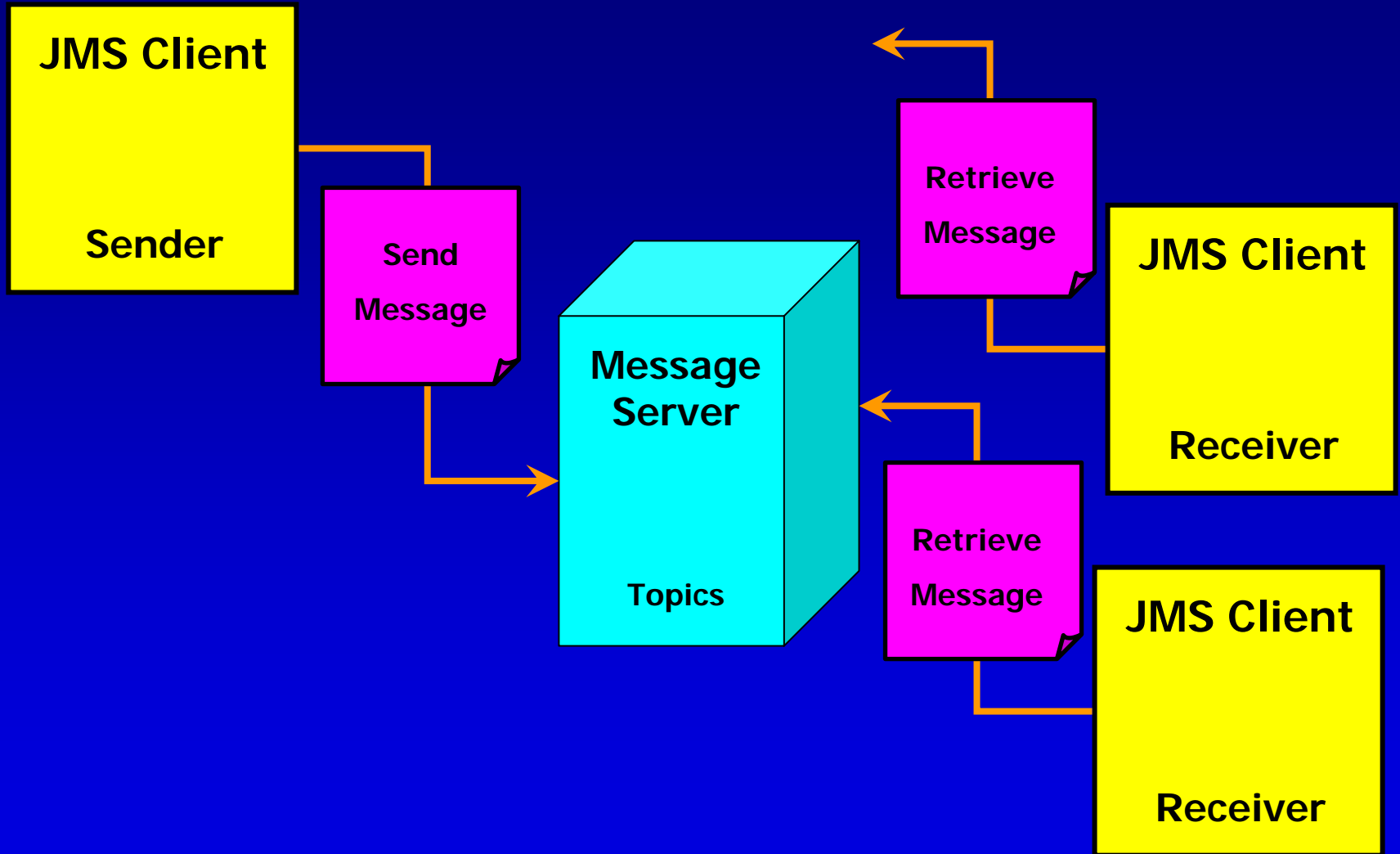


# Java Messaging Service



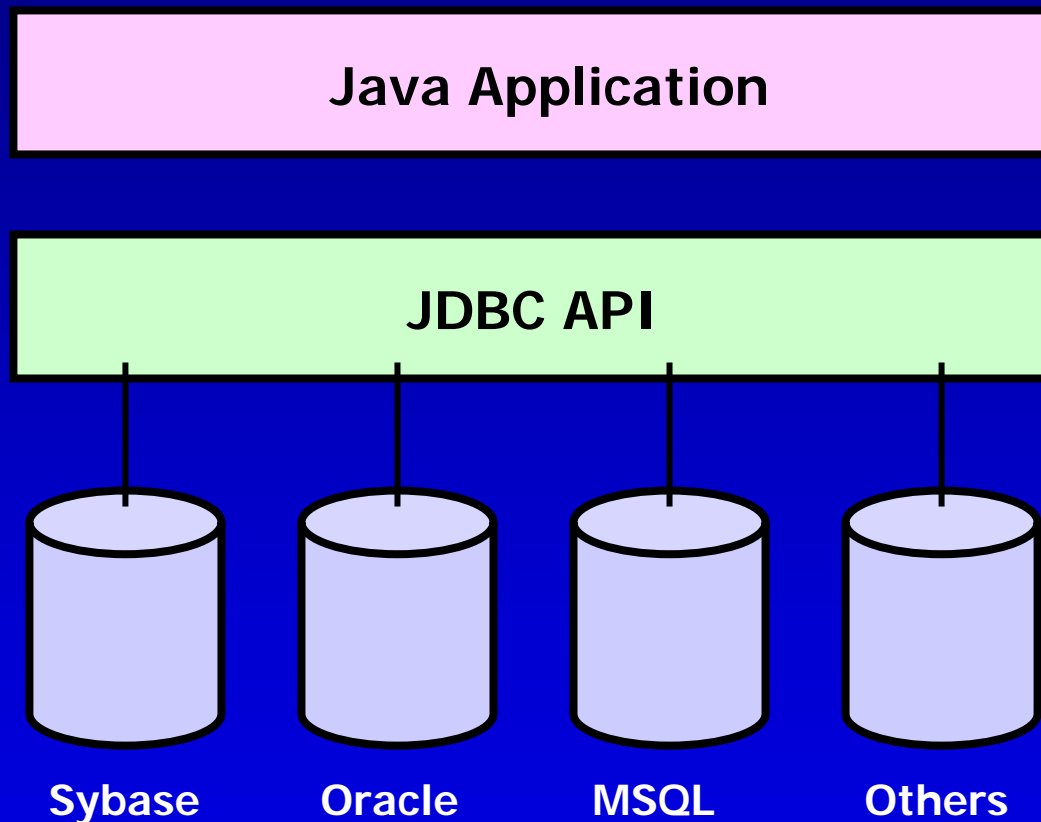
# Java Messaging Service

- Publisher Subscriber



# Java Database Connectivity

- Known as JDBC
- Pure Java API used to execute SQL statements



# Java Database Connectivity

- A standard way to connect to a database from a J2EE application through the JDBC driver
  - JDBC Specification
    - JDBC 1.0 - use direct connection
    - JDBC 2.0 - use of connection pool
    - JDBC 3.0 – access flat files
    - JDBC - Rowset

# Java Database Connectivity

- Advantages of JDBC
  - Leverage existing enterprise data
    - Business are not locked in any propriety
    - Can use different database management system
  - Simplified enterprise development
    - Hides the complexity of data access tasks
    - JDBC API is simple to learn and easy to deploy
  - Zero configuration for network computers
    - No configuration required in the client side
    - Centralized software maintenance

*This slide intentionally left blank*