E32

JDBC Access to IMS DB from DB2, CICS, and Websphere

Christopher Holtz

IMS
technical conference

Las Vegas, NV **September 15 – September 18, 2003**

RETURN TO INDEX

Overview

- IMS Java
  - What Is IMS Java
  - Why Use IMS Java
  - IMS Java Class Library Architecture
- JDBC and J2EE
- Dealership Sample Application
  - Front-end/Back-end split
- Environments
  - Non-Managed
    - IMS
    - CICS
    - DB2
  - Managed
    - WebSphere

What is IMS Java?

- A new feature in IMS v7

- A set of classes that...
  - Offers Java support to access IMS Databases from various environments (IMS, CICS, DB2, WebSphere)
  - Enables SQL access through the JDBC interface

- Java Virtual Machine (JVM) support in dependent regions
  - JDK 1.3 support
  - JDBC 2.1 support
  - Just-In-Time (JIT) compilation
  - Resetable JVM

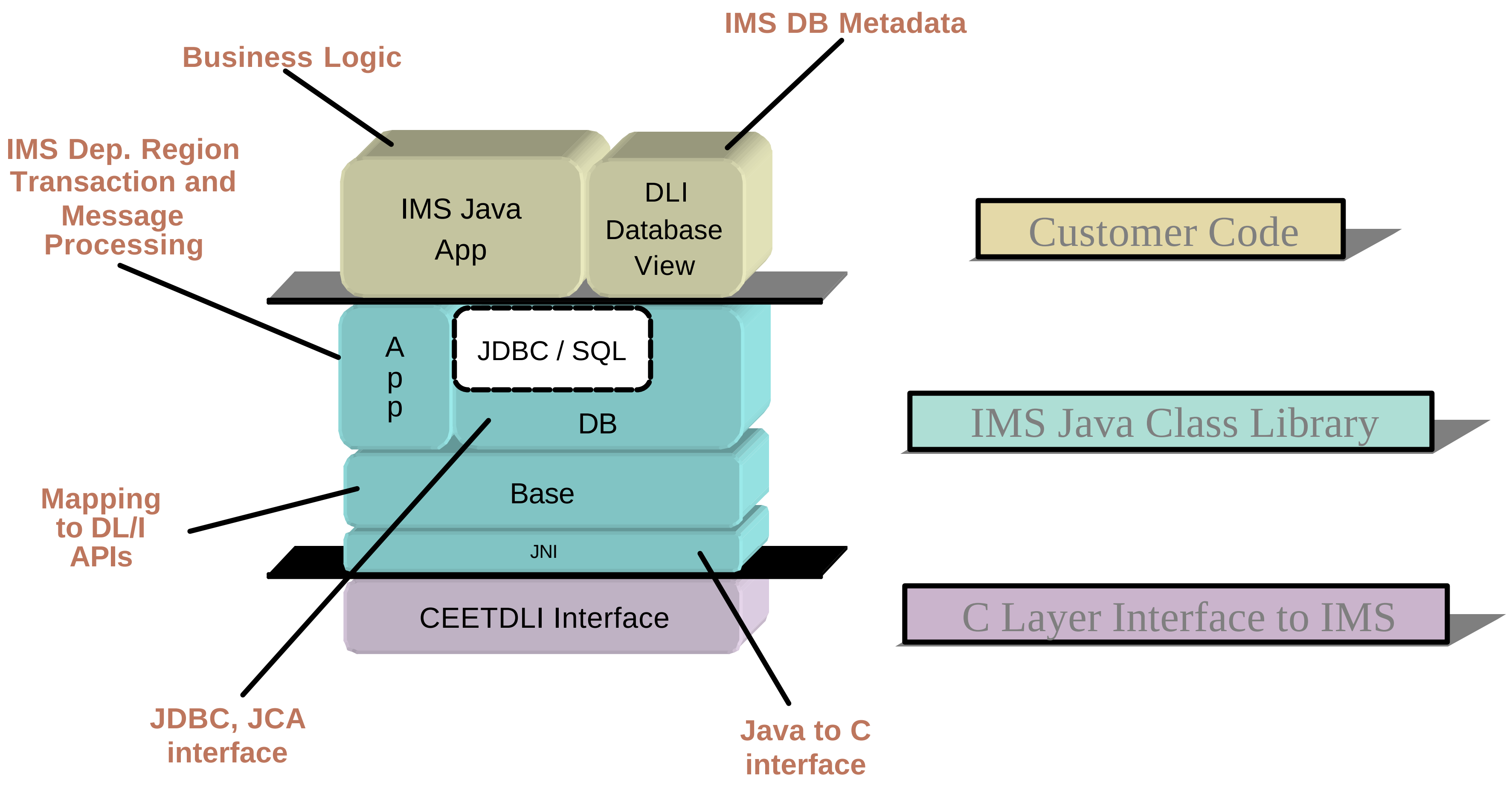
Why IMS Java?

IMS

- Rapid Application Development
  - Reduce the Total Cost of Ownership (TCO) for IT and Data Management needs and Total Time to Value (TTV)
- Extend the life and scope of IMS applications
  - Minimum amount of impact on core applications and effort for developers, system programmers, and DBAs
- Leverage existing marketplace, industry-sanctioned standards - they are the slowest changing and most persistent
  - JDBC and J2EE are standards – help to minimize specific back end knowledge of IMS
- Leverage new and abundant skills in the marketplace and mitigate the loss of 390 skills for customers
- Integrate with other products
  - ☆ Our response is IMS Java, Web Services, WebSphere support, CICS support, DB2 SP support

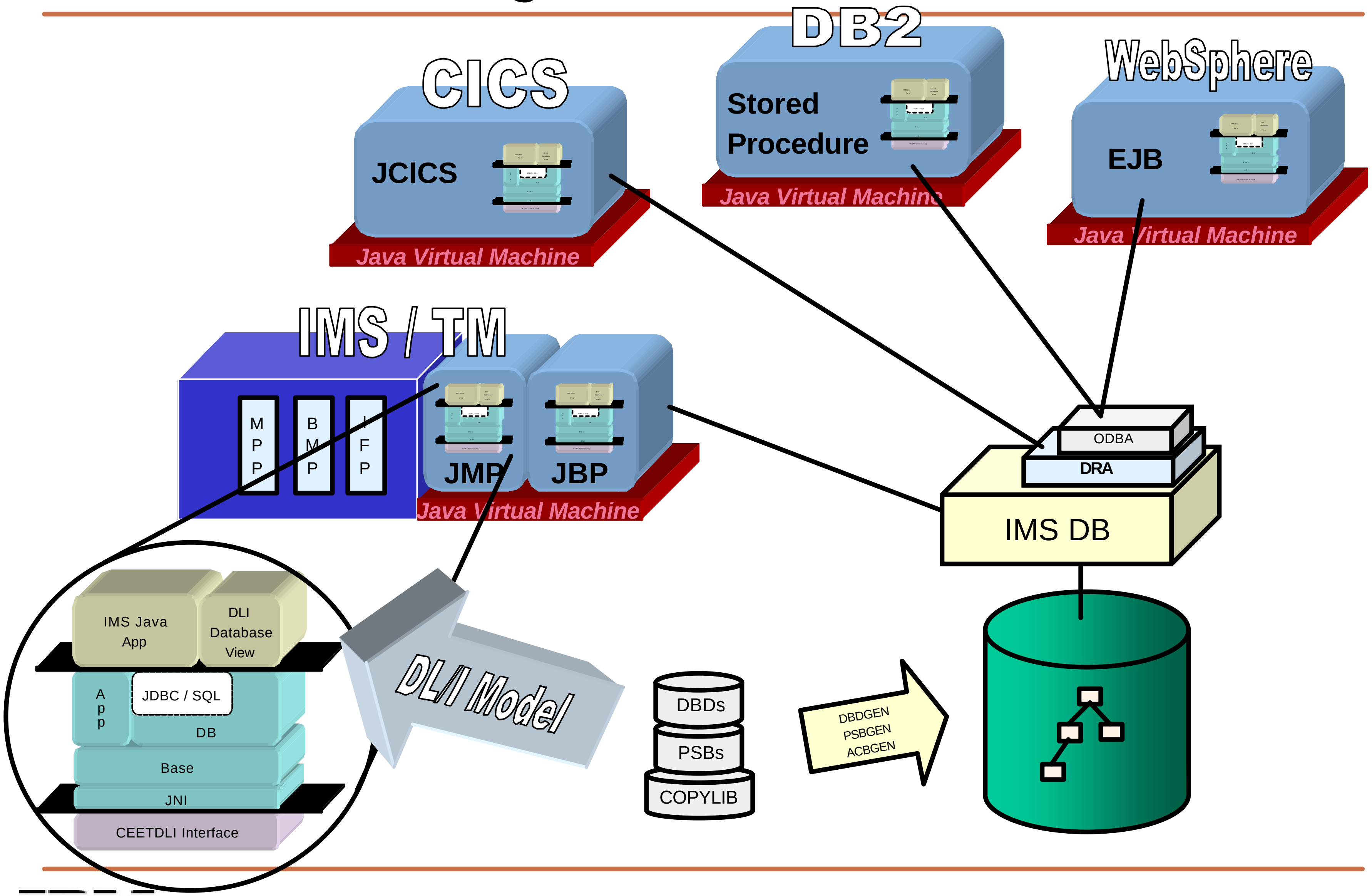
Java Class Library

IMS



IMS Java – The Big Picture

IMS



Overview

- IMS Java
  - What Is IMS Java
  - Why Use IMS Java
  - IMS Java Class Library Architecture
- JDBC and J2EE
- Dealership Sample Application
  - Front-end/Back-end split
- Environments
  - Non-Managed
    - IMS
    - CICS
    - DB2
  - Managed
    - WebSphere

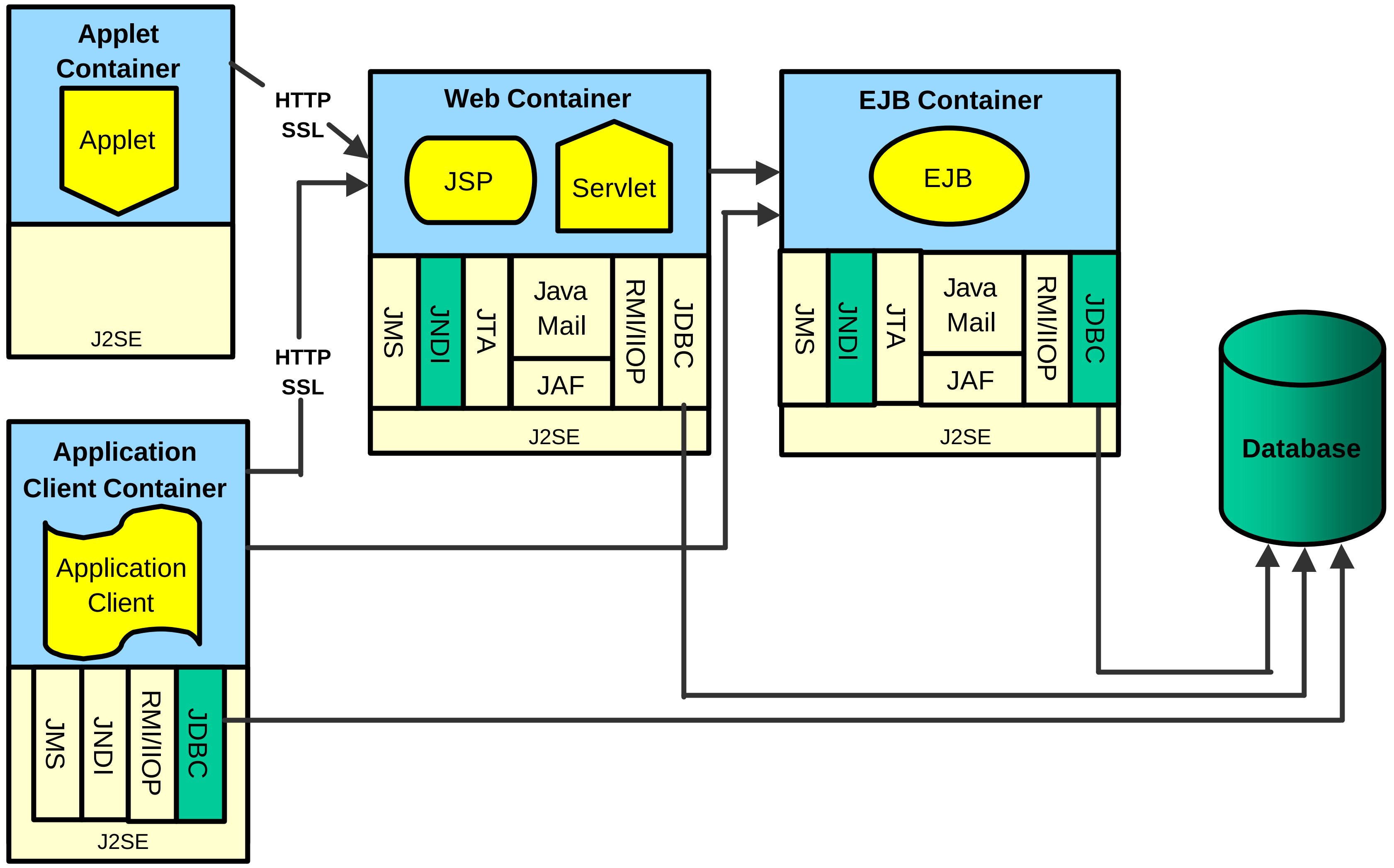
JDBC and J2EE Standards Evolution

IMS

- **Standard way to Query Database (relational)**
  - **Structured Query Language (SQL)**
- **Communicating Query to Database**
  - **Open Database Connectivity (ODBC) – C based**
- **Standard API to Query Database**
  - **“Java Database Connectivity” (JDBC) – Platform/DB Independent**
- **Standard API to Establish Connection**
  - **J2EE Connection Architecture (JCA, J2C)**
- **Standard API to Build Enterprise Applications**
  - **Java 2 Enterprise Edition (J2EE)**

J2EE Architecture

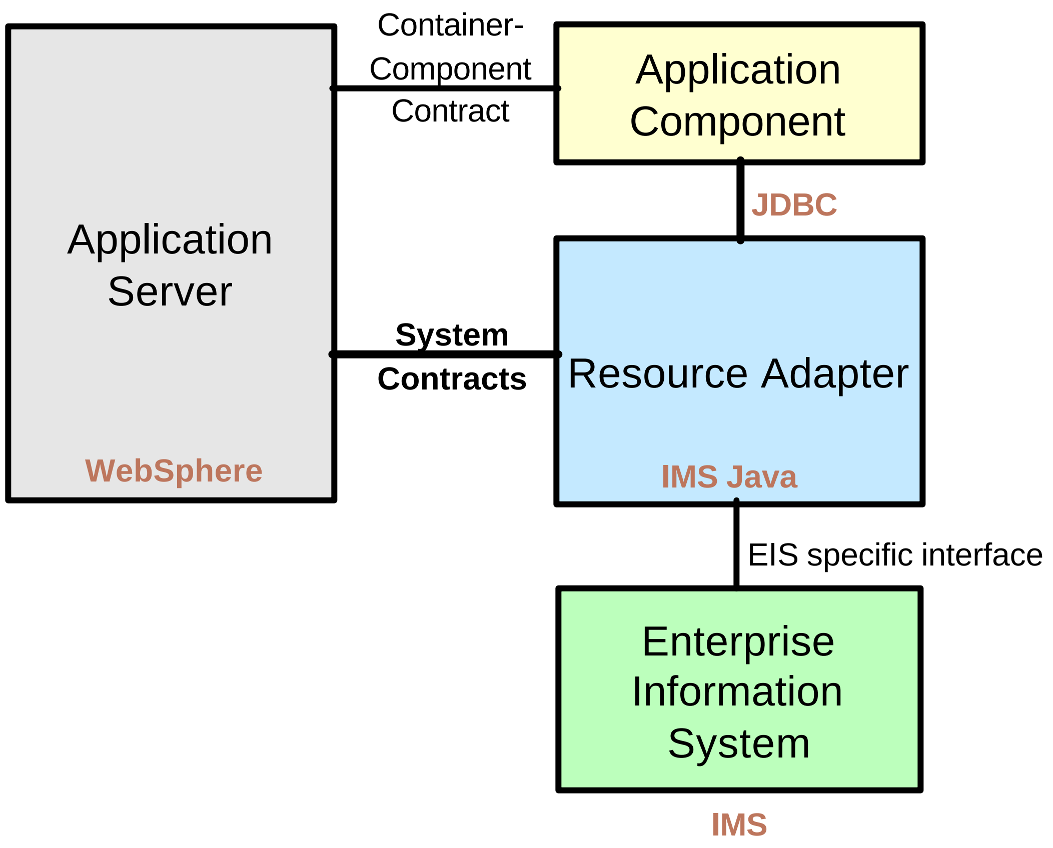
IMS



J2EE Connection Architecture

- Proposed Java standard architecture for deploying a resource adapter in a J2EE compliant application server
- Defines contracts between...
  - resource adapter and the application component
  - resource adapter and the application server

J2EE Connection Architecture

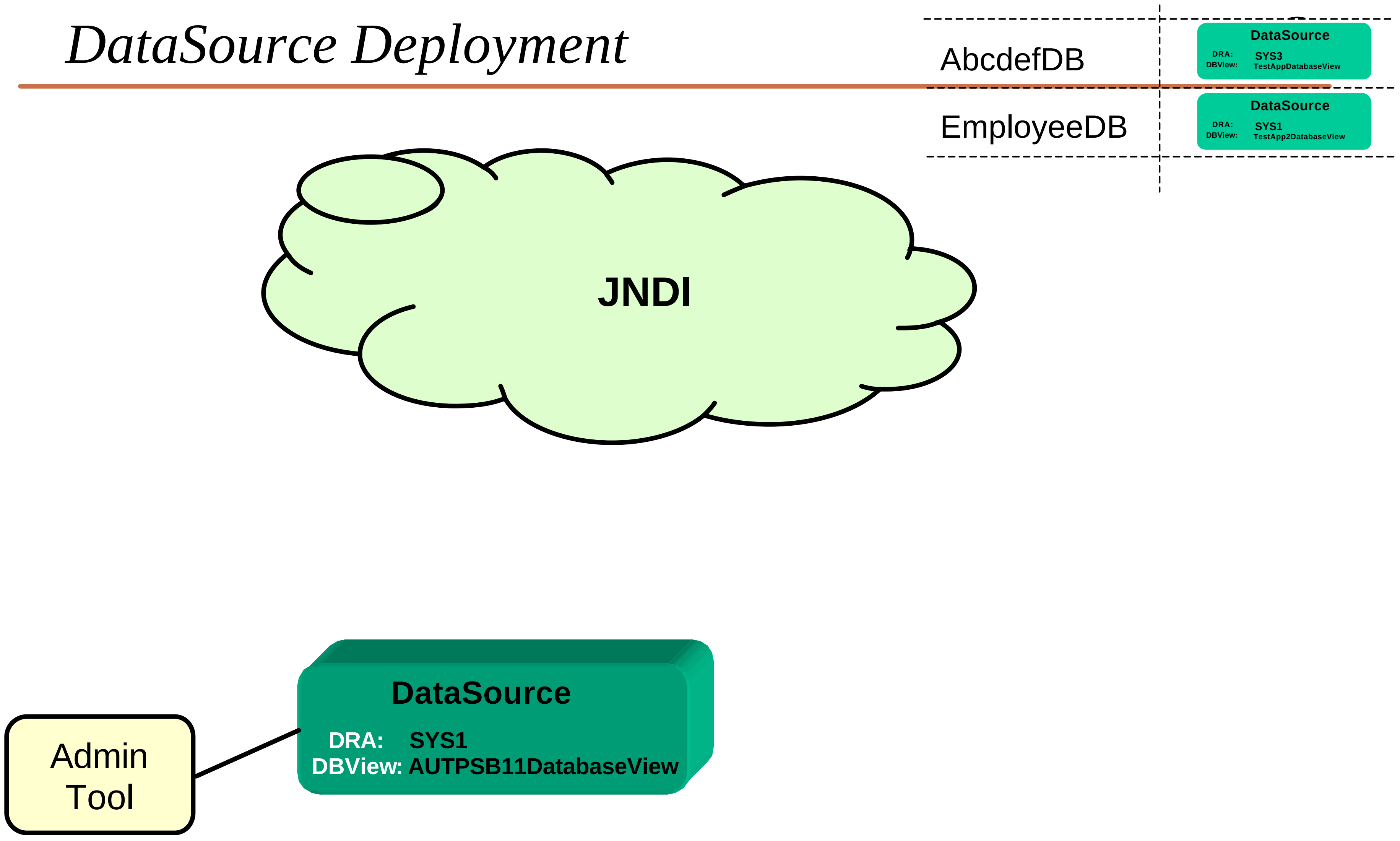


DataSource

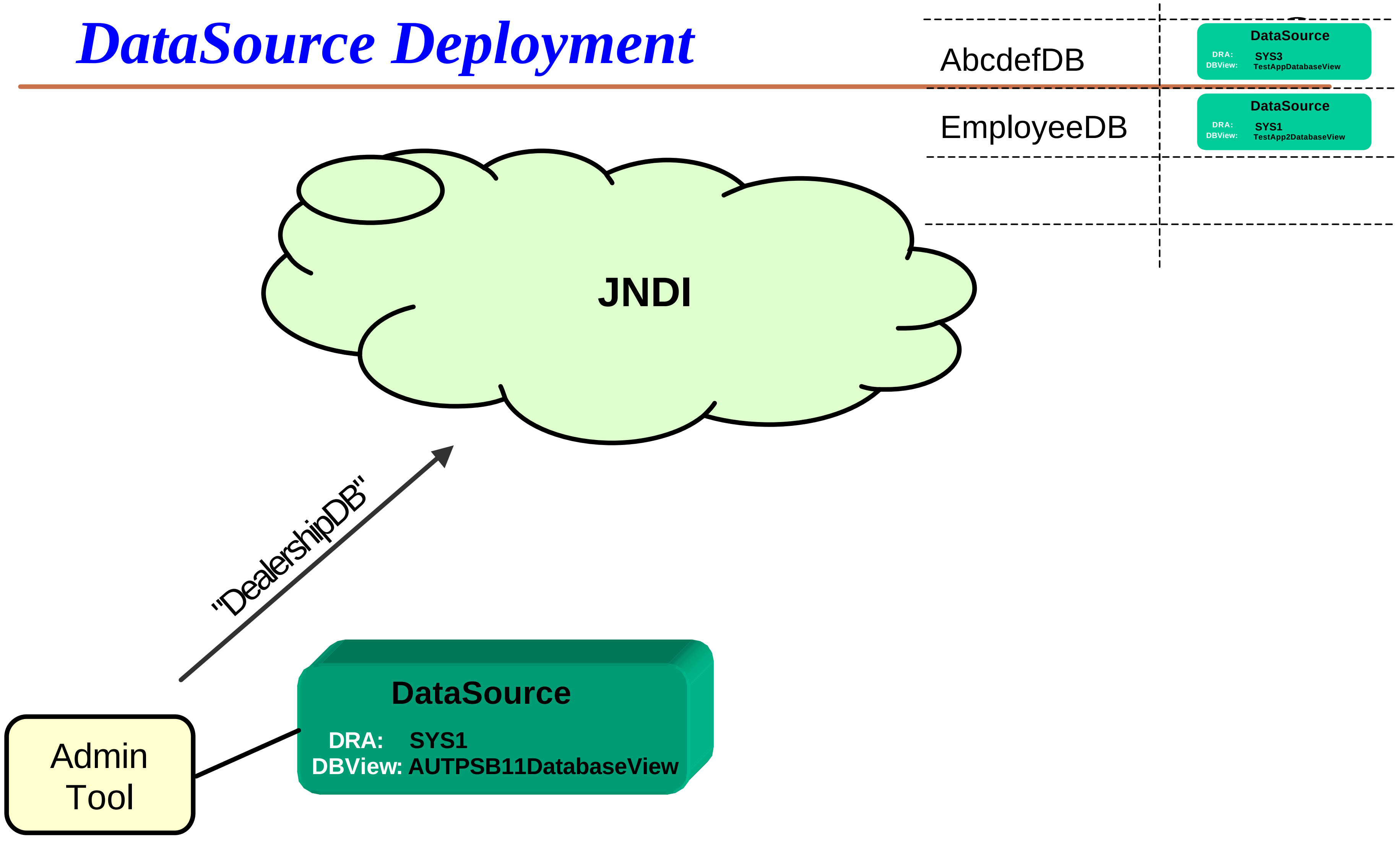
IMS

- Factory for connections to a physical data source
- Replacement to the DriverManager facility
  - Required when running in a managed environment (WebSphere)
- Typically registered with a naming service based on the Java™ Naming and Directory (JNDI) API.
  - Names are associated with objects and objects are found based on their names.
- DataSource objects have properties that can be modified when necessary
  - Code accessing the data source does not need to be changed
  - (Properties include: DRA Name, DLIDatabaseView)

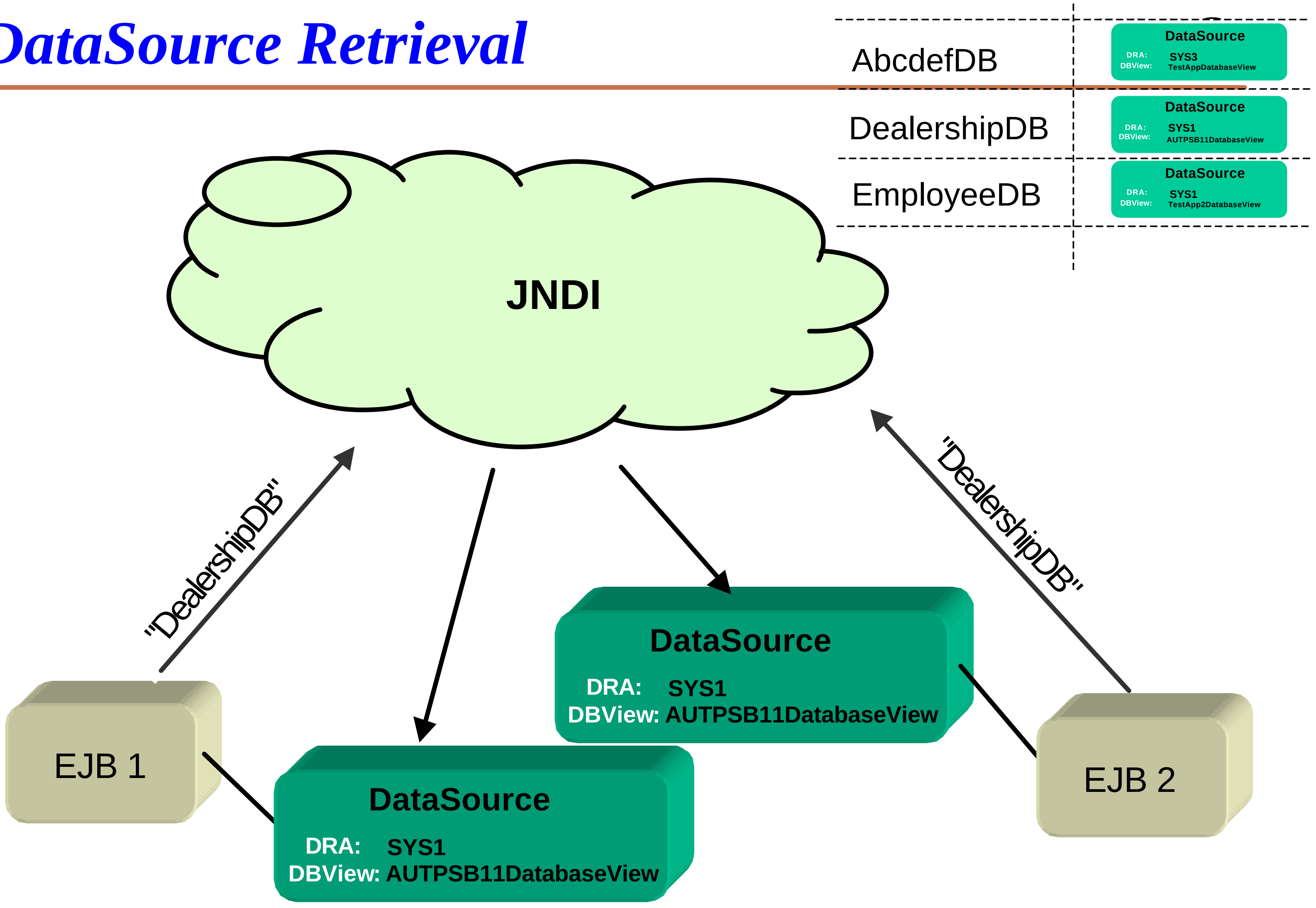
DataSource Deployment



DataSource Deployment



DataSource Retrieval

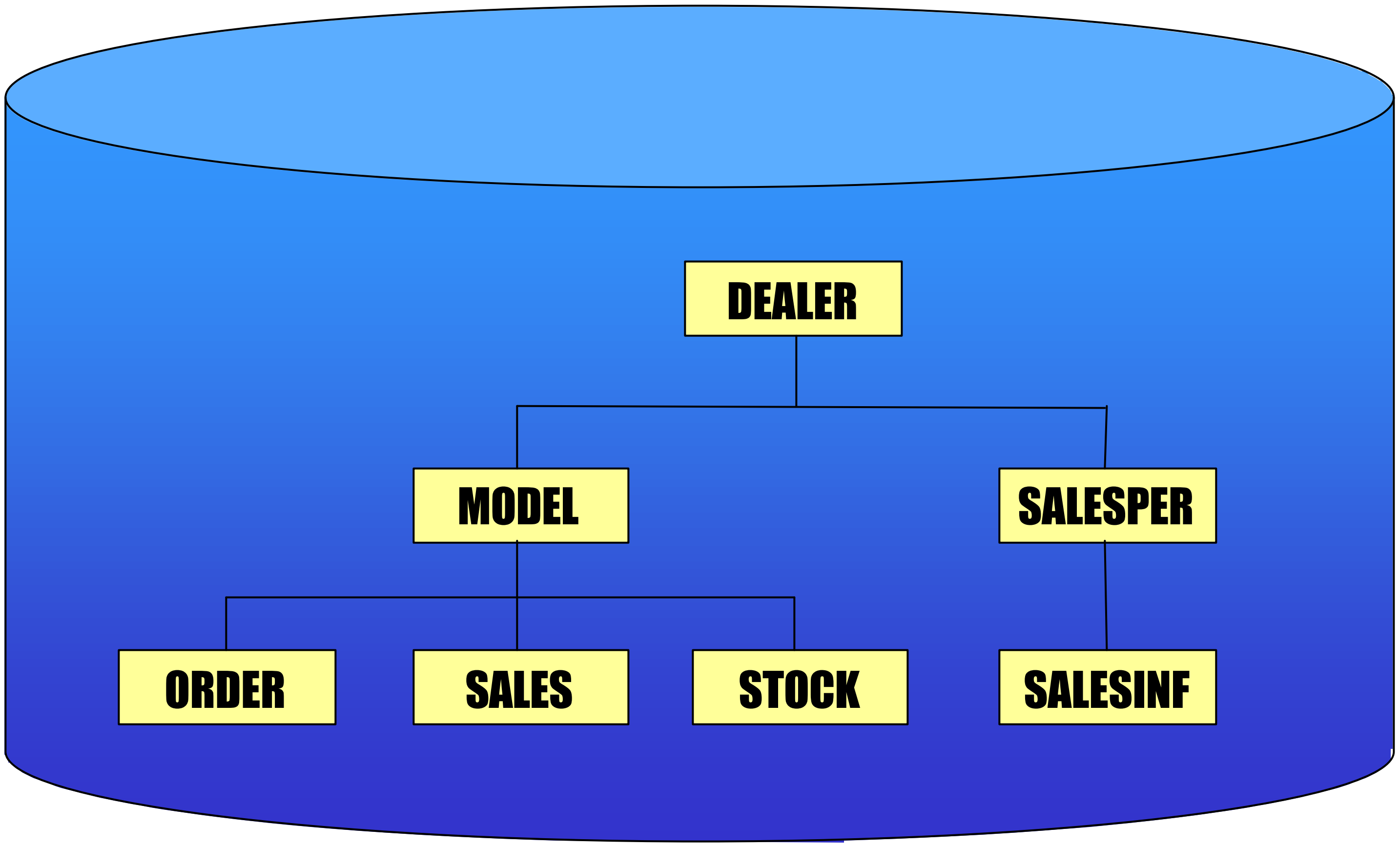


Overview

- IMS Java
  - What Is IMS Java
  - Why Use IMS Java
  - IMS Java Class Library Architecture
- JDBC and J2EE
- Dealership Sample Application
  - Front-end/Back-end split
- Environments
  - Non-Managed
    - IMS
    - CICS
    - DB2
  - Managed
    - WebSphere

Dealership Sample Database

IMS



Dealership Sample Application

- **Performs specific queries to the database**
  - **List all models**
  - **List details of a particular model**

- **Split into a front-end and a back-end**

- **Front-End (Environment Specific)**
  - **Process message queue (IMS)**
  - **Invoke stored procedure (DB2)**
  - **JCICS application (CICS)**
  - **Enterprise Java Bean (WebSphere)**

- **Back-End (Environment Independent)**
  - **Performs all query processing**
  - **Sends data back to the caller (front-end)**

Dealership Sample (Back-End)

IMS

```
public class AutoDealership {

   /** The database connection is created by each front-end and given
       to the single back-end **/
   public AutoDealership(Connection connection) {
     this.connection = connection;
   }

   public Vector listModels() throws SQLException {

       SQL processing logic here...

   }

   public Vector findCar(FindCarInput input) throws
           SQLException {

       SQL processing logic here...
   }
 }
```

Dealership Sample (Back-End)

IMS

```
public Vector listModels() throws SQLException {

  // Create the SQL statement - no inputs needed
  String query = "SELECT * FROM Order.ModelSegment";

  // Execute the query
  Statement statement = connection.createStatement();
  ResultSet results = statement.executeQuery(query);

  process results...

}
```

Dealership Sample (Back-End)

IMS

```
public Vector listModels() throws SQLException {

  create statement and execute query...

  Vector models = new Vector();
  ListModelOutput output = null;
  while (results.next()) {
    output = new ListModelOutput();
    output.setModelType(results.getString("ModelType"));
    output.setMake(results.getString("Make"));
    output.setModel(results.getString("Model"));
    output.setYear(results.getString("Year"));

    models.addElement(output);
  }

  return models;

}
```

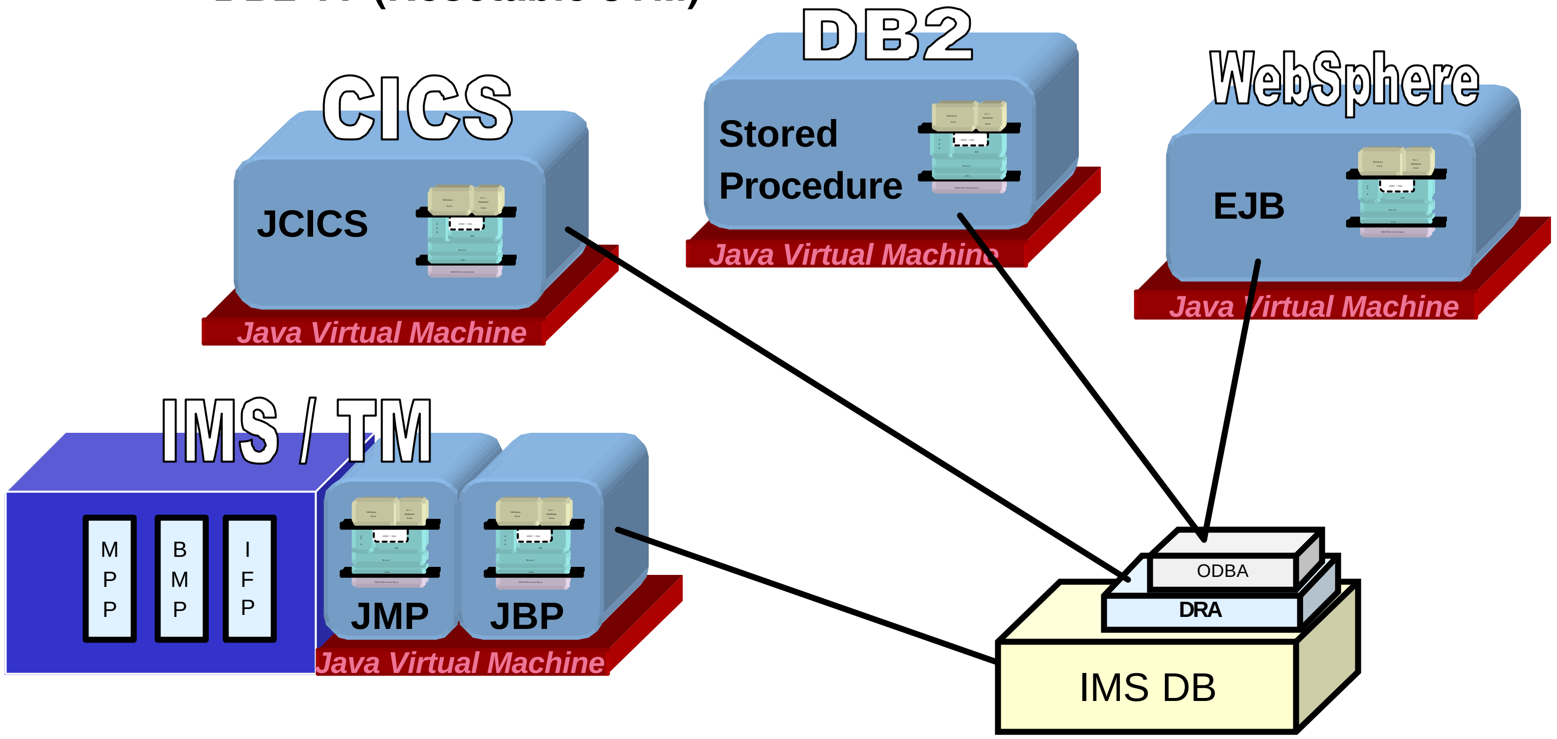
Overview

- IMS Java
  - What Is IMS Java
  - Why Use IMS Java
  - IMS Java Class Library Architecture
- JDBC and J2EE
- Dealership Sample Application
  - Front-end/Back-end split
- Environments
  - Non-Managed
    - IMS
    - CICS
    - DB2
  - Managed
    - WebSphere

Running Environments

IMS

- **Non-Managed Environments**
  - **IMS TM 7.1 (Resetable JVM)**
  - **IMS TM 8.1 (Resetable JVM)**
  - **CICS 2.1 (Resetable JVM)**
  - **DB2 V7 (Resetable JVM)**
- **Managed Environments**
  - **WebSphere 4.01 (JVM)**



Non-Managed Connection Establishment IMS

- **Non-Managed Environment (IMS, DB2, CICS)**
  - **An application can construct a DataSource prior to its use**
    - **Alternatively, a DataSource can be bound to the namespace**
  - **Application acquires a Connection from the DataSource**

```
IMSJdbcManagedConnectionFactory mcf = new IMSJdbcManagedConnectionFactory();

mcf.setDRAName("SYS1");
mcf.setDatabaseViewName("samples.dealership.AUTPSB11DatabaseView");

DataSource dataSource = (DataSource)mcf.createConnectionFactory();

Connection connection = dataSource.getConnection();
```

IMS TM Java Access to IMS Data

- Run in JVM
  - IMS 7.1 or later
  - JDK 1.3
  - JDBC 2.1
- Can only connect to one PSB
- Synchpoints done by calls to application package

IMS TM Java Access to IMS Data

```
// Import the IMS Java packages
import com.ibm.ims.db.*;
import com.ibm.ims.application.*;

public class IMSAuto {

    /* Entry point of the application. */
    public static void main(String args[]) {
        // setup message queue access.
        IMSMessageQueue messageQueue = new IMSMessageQueue();
        InputMessage inputMessage = new InputMessage();

        while (messageQueue.getUniqueMessage(inputMessage)) {
            create connection…
            call back-end…
            reply to message queue…
            commit…
        }
    }
}
```

IMS TM Java Access to IMS Data

IMS

```
/* Entry point of the application. */
public static void main(String args[]) {

  access message queue and get message...

    // Get connection
    IMSJdbcManagedConnectionFactory mcf = new IMSJdbcManagedConnectionFactory();
    mcf.setDatabaseViewName("samples.dealership.AUTPSB11DatabaseView");
    DataSource dataSource = (DataSource)mcf.createConnectionFactory();
    Connection connection = dataSource.getConnection();

    // Pass connection to back-end
    AutoDealership autoDealership = new AutoDealership(connection);

    // Call listModels method and get results in 'output' Object
    Vector output = autoDealership.listModels();

    reply to message queue...
    commit...
}
```

IMS TM Java Access to IMS Data

IMS

```
/* Entry point of the application. */
public static void main(String args[]) {

  access message queue and get message...
     create connection...
     call back-end...

    // Format and insert output message
     OutputMessage outputMessage = new OutputMessage(output);
     messageQueue.insertMessage(outputMessage);

     // Close connection and commit database before we get next Message.
     connection.close();
     IMSTransaction.getTransaction().commit();
  }
}
```

CICS Java Access to IMS Data

IMS

- Run in JVM
  - CICS TS 2.1
  - JCICS API
    - Java version of the CICS API
  - JDK 1.3
  - JDBC 2.1

- Can only connect to (allocate) one PSB at a time
  - Only one Connection active at a time in an application

- Synchpoint done at deallocate PSB

CICS Java Access to IMS Data

IMS

```
// Import the JCICS package
import com.ibm.cics.server.*;

public class CICSAuto {

  private static Task task = Task.getTask();

  /* Invoked when the CICS transaction corresponding to this class
     is executed. */
  public static void main(CommAreaHolder cah) {
      CICSAuto application = new CICSAuto();
      application.listModels();
  }

  /* The listModels method provides a collection of all automobile
     models in the database */
  public void listModels() {

    method logic here...

  }
}
```

CICS Java Access to IMS Data

```
/* The listModels method provides a collection of all automobile
   models in the database */
public void listModels() {

  // Get connection
  IMSJdbcManagedConnectionFactory mcf = new IMSJdbcManagedConnectionFactory();
   mcf.setDatabaseViewName("samples.dealership.AUTPSB11DatabaseView");
   DataSource dataSource = (DataSource)mcf.createConnectionFactory();
   Connection connection = dataSource.getConnection();

   // Pass connection to back-end
   AutoDealership autoDealership = new AutoDealership(connection);

   // Call listModels method and get results in 'output' Object
   Vector output = autoDealership.listModels();

   display results...

}
```

DB2 Stored Procedure Support

IMS

- Run in JVM
  - DB2 Version 7
  - APAR PQ46673 (resetable JVM)
- Stored Procedures that access IMS Databases
  - User-written structured query language (SQL) programs that are stored at the DB2 server and can be invoked by a client application
- DB2 handles synchpoint (not stored procedure)
- DRA table and name required

DB2 Store Procedure Support

IMS

```
public class DB2Auto() {

 public static void listModels(String[] make, String[] model) {

    // Get connection
    IMSJdbcManagedConnectionFactory mcf = new IMSJdbcManagedConnectionFactory();
    mcf.setDRAName("SYS1");
    mcf.setDatabaseViewName("samples.dealership.AUTPSB11DatabaseView");
    DataSource dataSource = (DataSource)mcf.createConnectionFactory();
    Connection connection = dataSource.getConnection();

    // Pass connection to back-end
    AutoDealership autoDealership = new AutoDealership(connection);

    Vector output = autoDealership.listModels();

    // Return data to client
    make[0] = ((ListModelOutput)output.elementAt(0)).getMake();
    model[0] = ((ListModelOutput)output.elementAt(0)).getModel();
  }
}
```

Managed Connection Establishment

IMS

- **Managed Environment (WebSphere)**
  - **DataSource deployed in JNDI namespace using:**
    - **WebSphere Application Server for z/OS and OS/390 Administration tool (4.01)**
    - **Web UI tool (5.0)**
  - **Application (EJB) makes a request for the DataSource and acquires a Connection from it**

```
Context ctx = new InitialContext();

DataSource dataSource = (DataSource)ctx.lookup("DealershipDB");
Connection con = dataSource.getConnection();
```

WebSphere 4.01, 5.0 Support

- Runs in JVM
  - JDK 1.3
  - JDBC 2.1
- Applications run as Enterprise Java Beans (EJBs)
- Access IMS databases through ODBA/DRA
- J2EE Connection Architecture
  - Managed Environment
    - Connections are managed by application server

Enterprise Archive (EJB)

```
public Vector listModels() throws javax.ejb.EJBException {

  InitialContext initialContext = new InitialContext();
  // perform JNDI lookup to obtain the DataSource and get the Connection
  DataSource dataSource = (DataSource)initialContext.lookup("DealershipDB");
  Connection connection = dataSource.getConnection();

  AutoDealership dealer = new AutoDealership(connection);
  Vector output = dealer.listModels();

  return output;
}
```

Overview of Application Development Steps

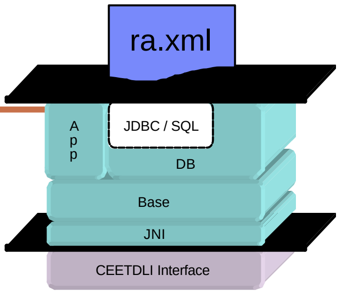
IMS

- **Obtain Resource Adapter Archive (.rar)**
  - **Deploy Resource Adapter Archive (install into server)**

- **Deploy DataSource (represents connection to a database)**
  - **Deploy Resource Adapter Instance**

- **Build and Deploy Enterprise Archive (.ear)**
  - **Two main development components**
    - **Servlet**
      - **Accesses EJB and invokes methods on the EJB**
    - **EJB**
      - **Accesses deployed DataSource (resource adapter instance)**
      - **Uses DataSource to connect to database and performs business logic**

Deploy IMS Java Resource Adapter



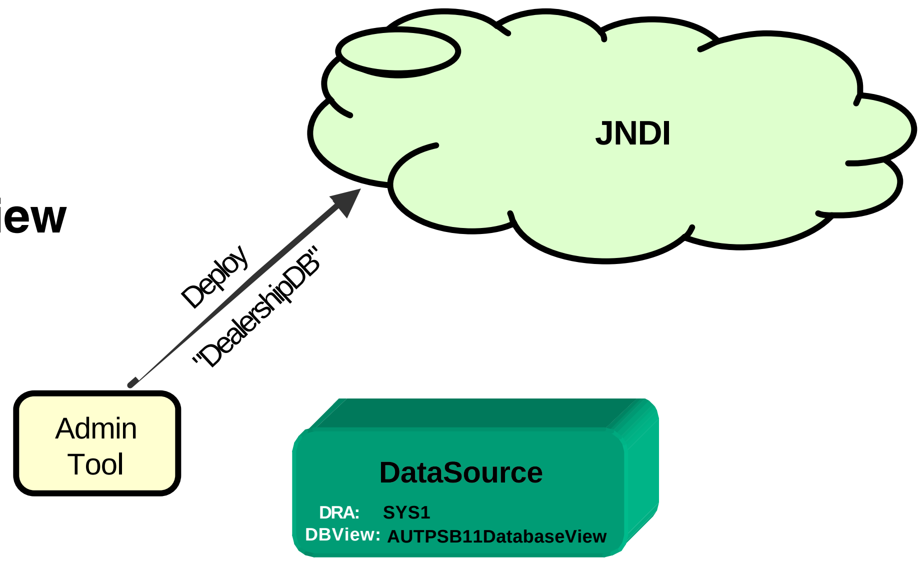
- **We provide a J2EE Resource Adapter Archive (RAR) file containing:**
  - **Deployment descriptor (ra.xml)**
  - **IMS Custom Service (IMSJdbcCustomService.xml)**
    - **Handles initialization and termination of IMS**
  - **Latest information associated with installing the IMS JDBC resource adapter (howto.html)**

- **WebSphere will provide tooling to deploy RAR file into server**
  - **Add IMS Custom Service**
  - **Set CLASSPATH and LIBPATH**
  - **Understand how to deploy an instance of an IMS Java Resource Adapter (via ra.xml)**

DataSource Deployment

IMS

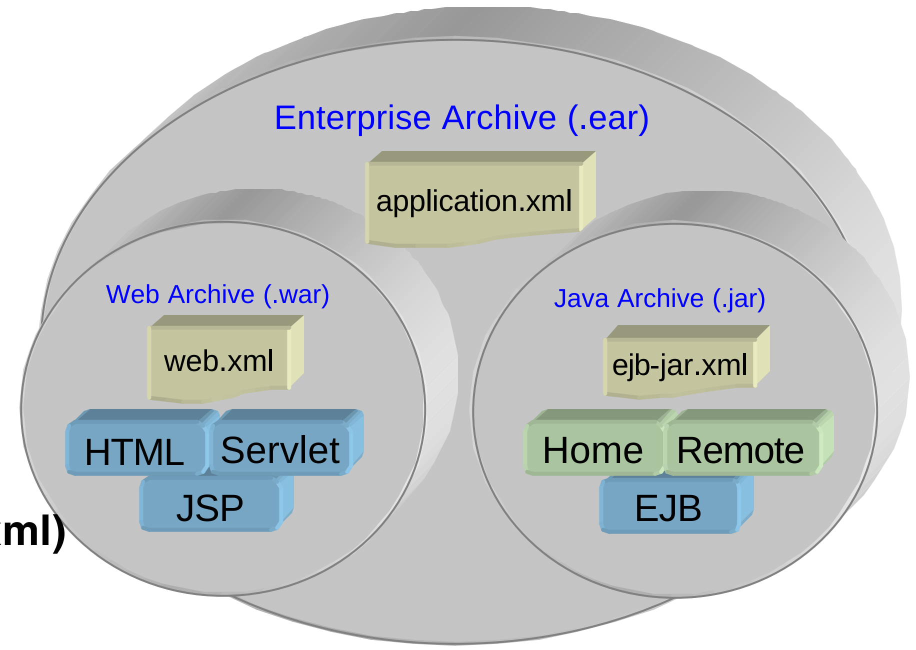
- Use WebSphere Application Server for z/OS and OS/390 Administration tool
  - Shipped with WebSphere on OS/390
- Create J2EE Resource Instance
  - Specify IMSJdbcDataSource as the J2EE Resource Type
- Configure
  - DRA Startup Table
  - Generated DLIDatabaseView
- Deploy



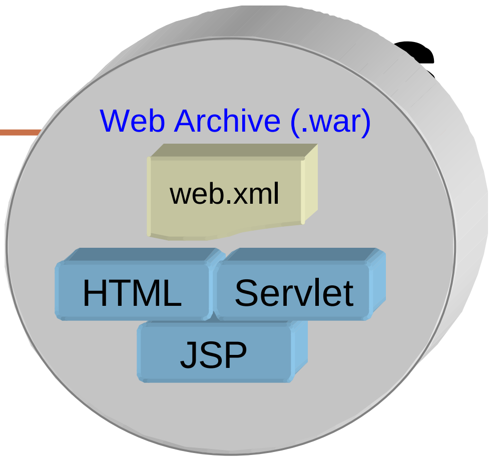
Build and Deploy Enterprise Archive (ear)

IMS

- J2EE Enterprise Application Archive
  - Complete J2EE application
- Must contain
  - One or more J2EE modules (Java Archive)
  - Deployment descriptor (application.xml)
    - Represents a top level view of a J2EE application's contents
- May contain
  - One or more Web modules (Web Archive)
  - Libraries referenced by J2EE modules

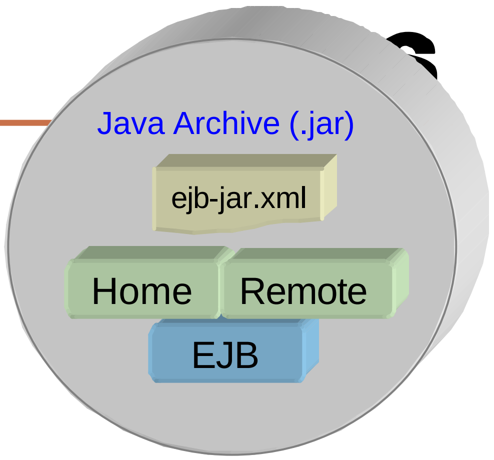


Enterprise Archive (Servlet)



- **Looks up the EJB home interface in JNDI**
- **Using the home interface, creates the EJB remote interface**
- **Invokes methods on the remote interface**
  - **Remote interface uses IIOP to communiate to EJB**
    - **Pass-through interface**
- **Passes results of method to a Java Server Page (JSP) for displaying on a web browser**

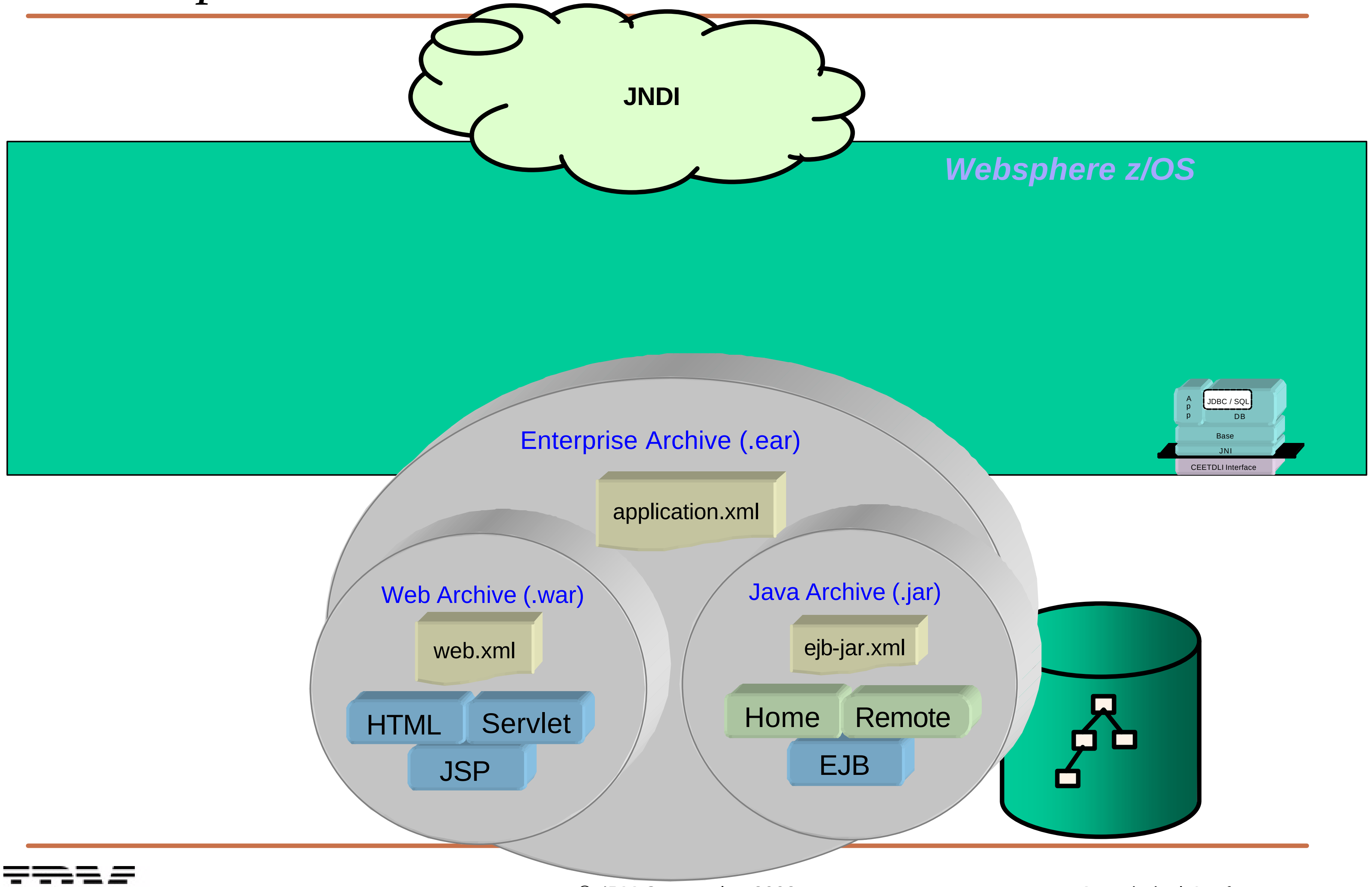
Enterprise Archive (EJB)



- **Receives requests from the servlet**
- **Obtains a connection**
  - **Looks up the deployed DataSource instance and requests a connection from it**
- **Accesses the database(s) and performs business logic**
- **Sends the results back to the servlet to display**

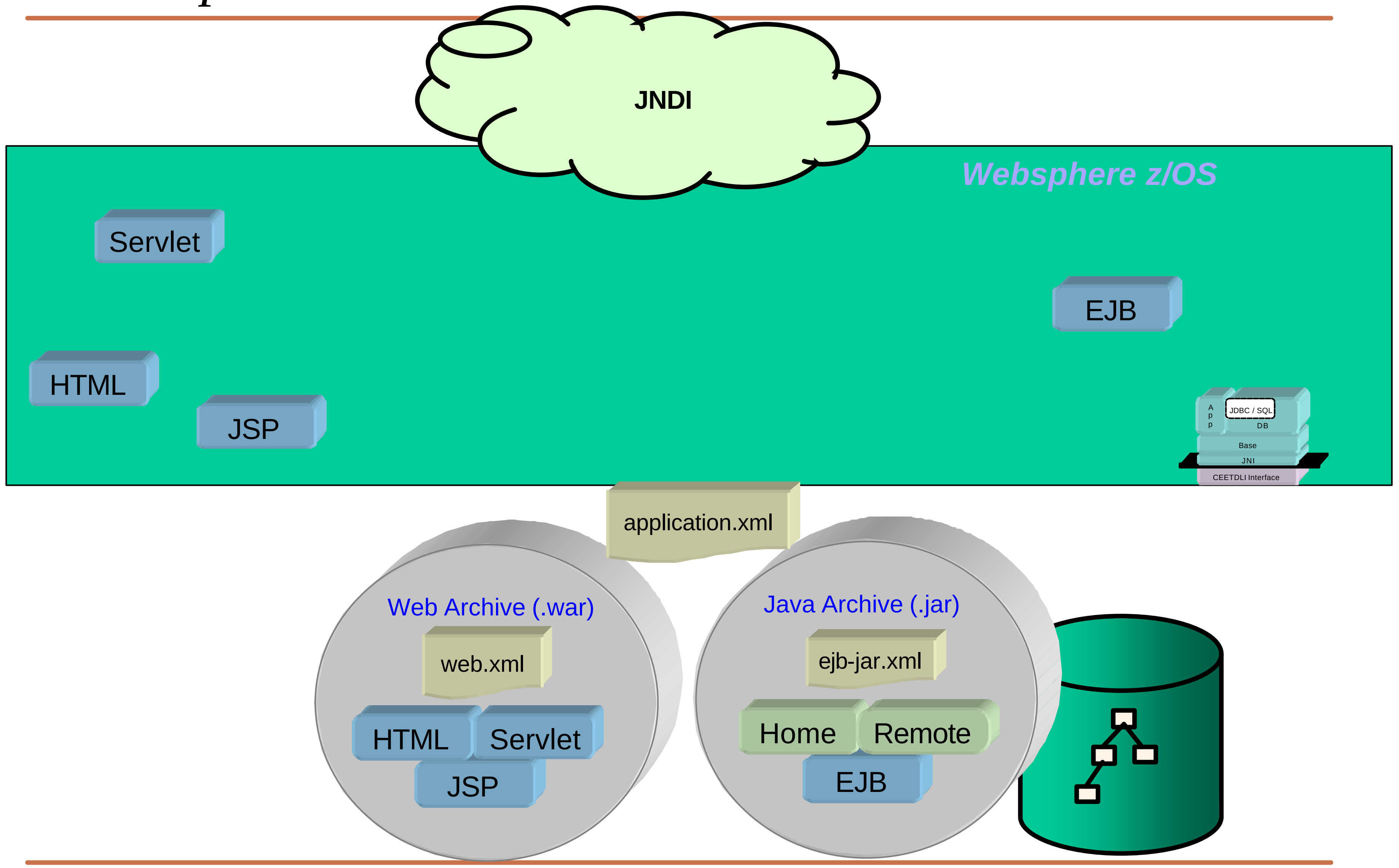
WebSphere Runtime

IMS



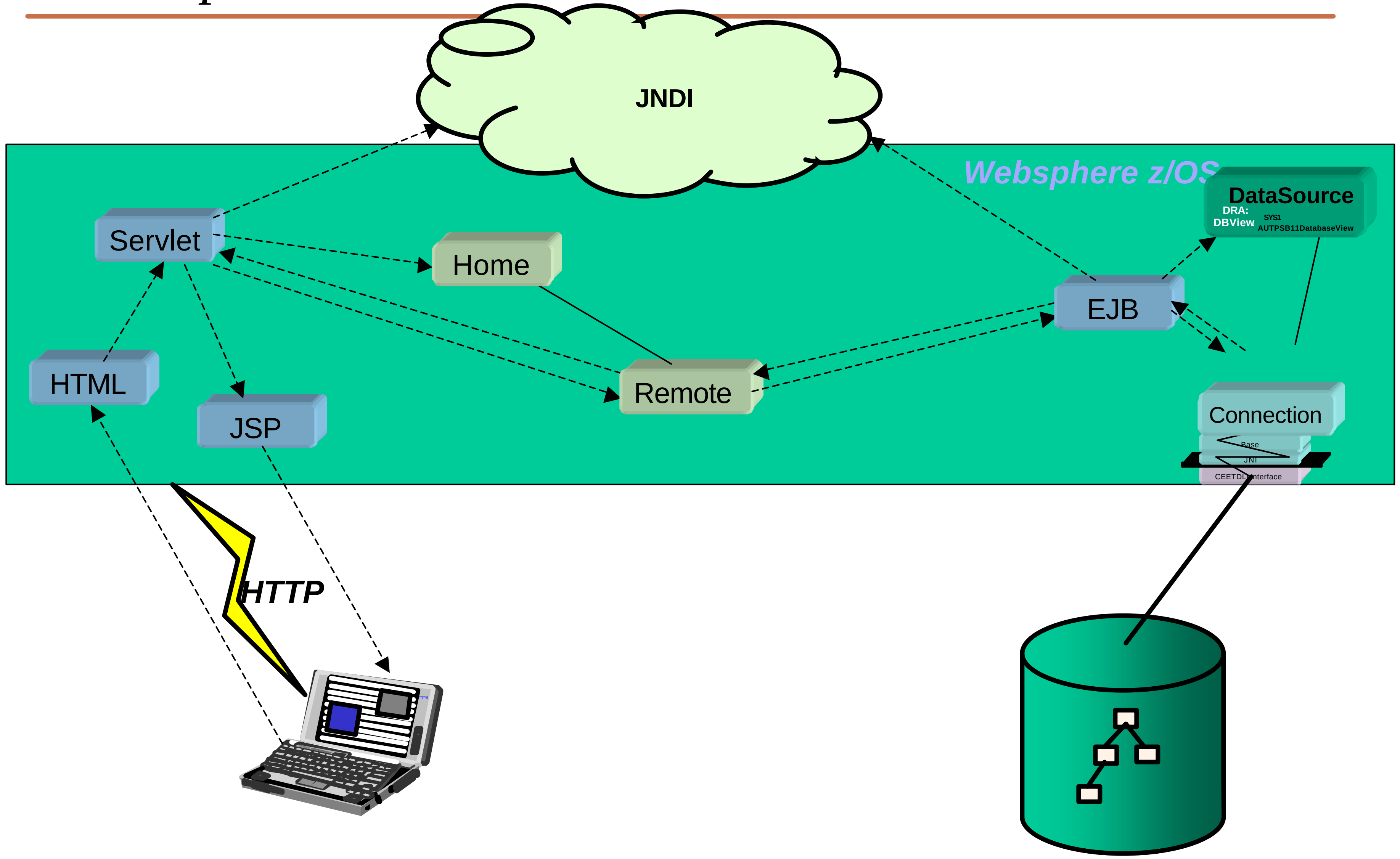
WebSphere Runtime

IMS



WebSphere Runtime

IMS



Remote Database Services

IMS

- **Available IMS Version 9**

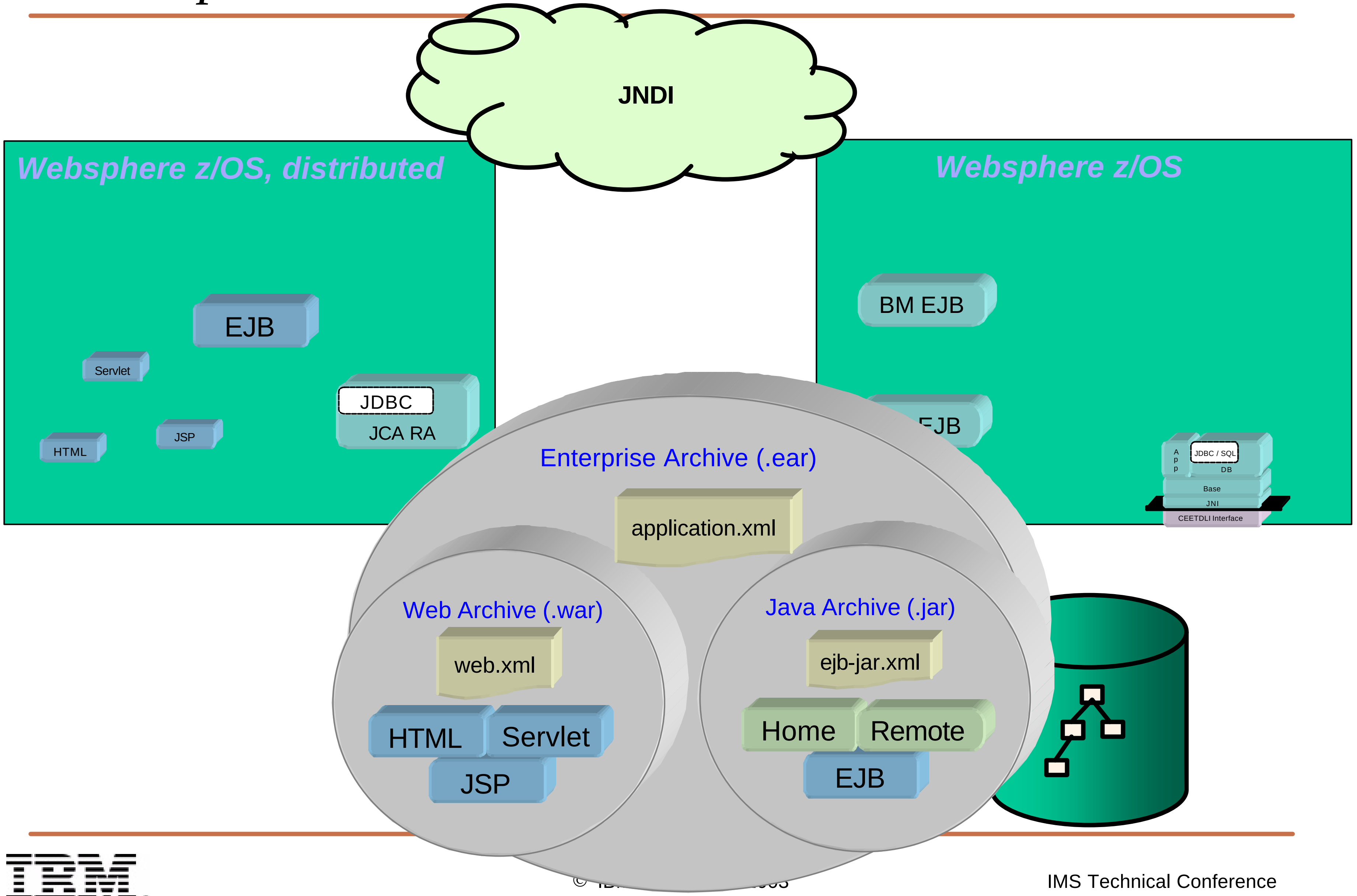
- **Ability to access IMS DL/I data from a distributed J2EE server**

- **Complete client application deployed on distributed server**
  - **Distributed functionality is transparent to client application**
  - **Websphere Application Server 5.0 for z/OS still required (server-side)**

- **All client-server communication is handled by new IMS Java components**
  - **IMS JDBC distributed Resource Adapter (client-side RAR)**
  - **Container Managed EJB (server-side EAR)**
  - **Bean Managed EJB (server-side EAR)**

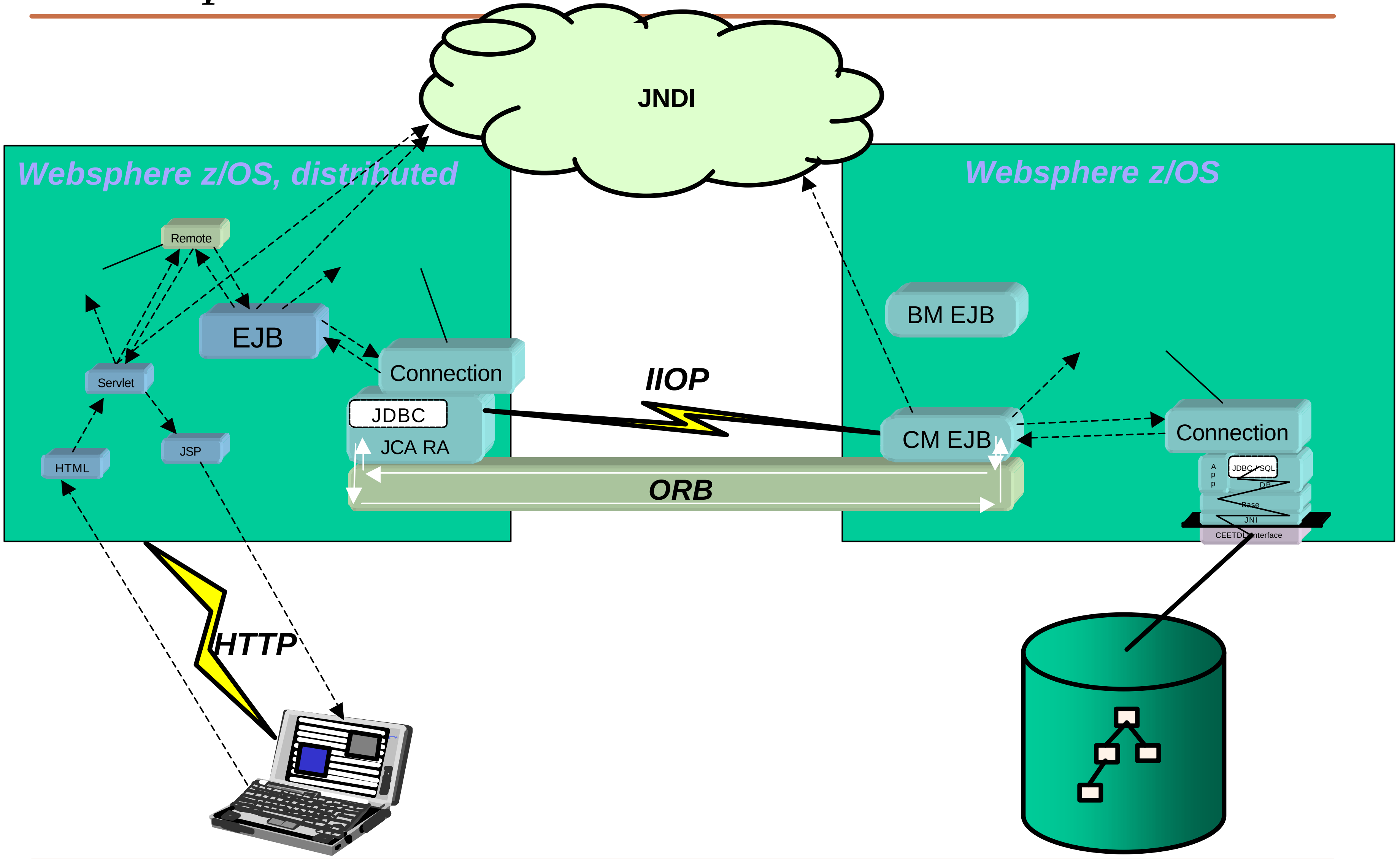
WebSphere Runtime

IMS



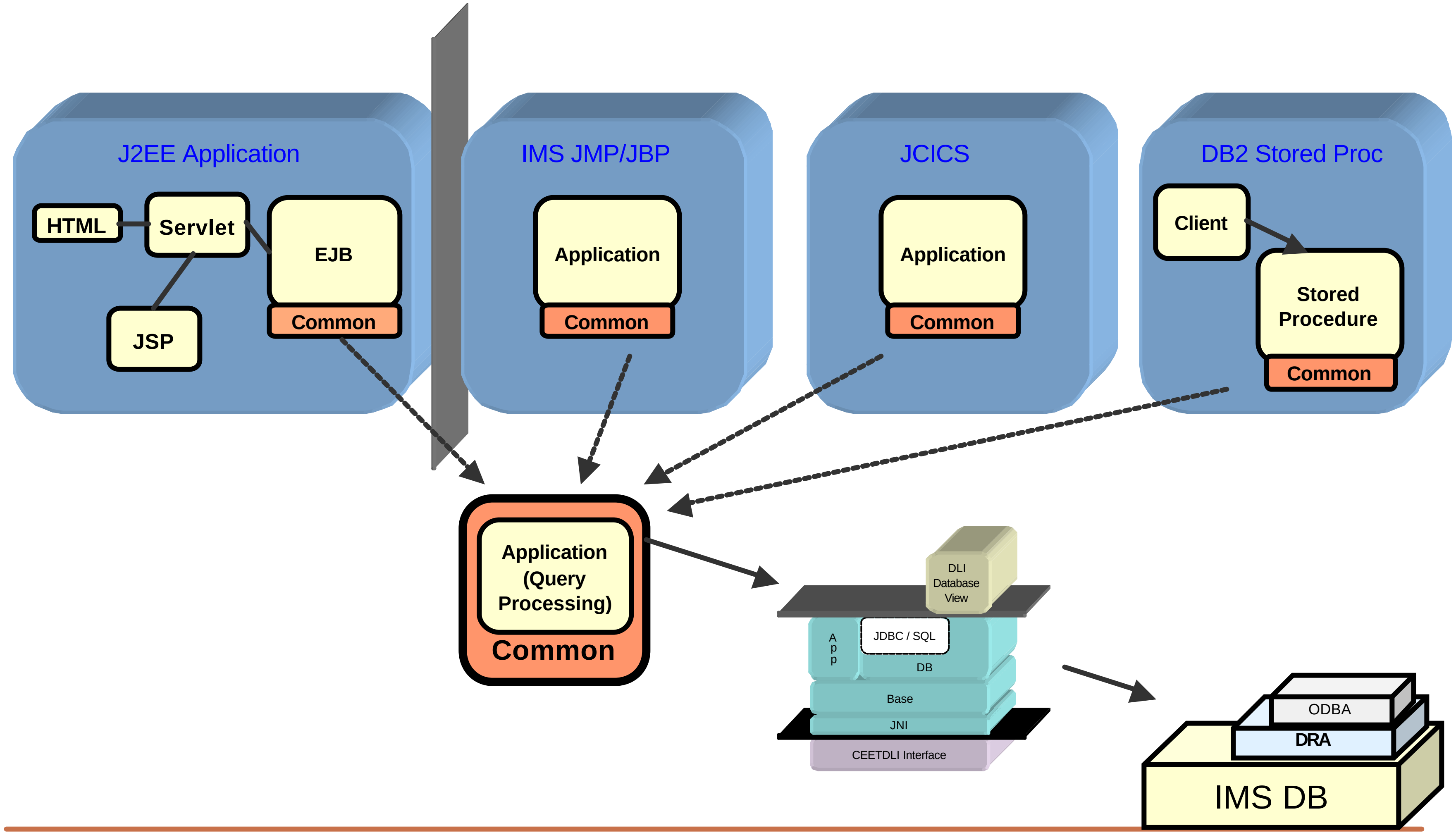
WebSphere Runtime

IMS



Conclusion

IMS



J2EE Application
HTML
Servlet
EJB
JSP
Common
IMS JMP/JBP
Application
Common
JCICS
Application
Common
DB2 Stored Proc
Client
Stored Procedure
Common
Application (Query Processing)
Common
DLI Database View
JDBC / SQL
App
DB
Base
JNI
CEETDLI Interface
ODBA
DRA
IMS DB