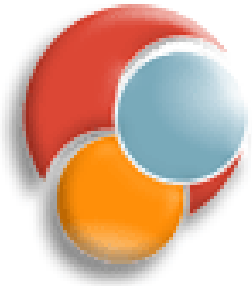


Java y Herramientas de Desarrollo

Sesión 1: Introducción a Java EE

Índice

- Organización del curso
- Java
- Arquitectura de aplicaciones
- Tecnologías Java EE
- Road Map

Profesorado

- Otto Colomina Pardo – otto@dccia.ua.es
- Domingo Gallardo López – domingo@dccia.ua.es
- Miguel A. Lozano Ortega – malozano@dccia.ua.es

Objetivos del curso

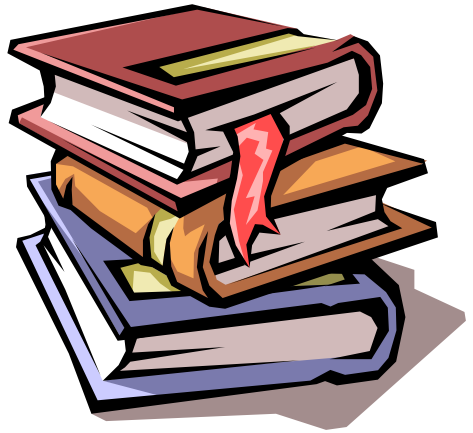
- Ofrecer una formación básica y sólida en las principales tecnologías Java EE
- Proporcionar un “roadmap” para acometer con garantías de éxito el aprendizaje de Java EE
- Proporcionar guías, ejemplos y modelos de desarrollo de aplicaciones Java EE

Horario del curso

- 120 horas
  - 12 horas a la semana
  - 10 semanas
- Clases teórico-prácticas: sesiones de 2 h.
- Estudio del sistema SIGEM: sesiones de 4h

Materiales

- Apuntes de cada módulo y guías de ejercicios
- Trasparencias
- DVDs con material
- Web con el contenido completo del curso
- Espacio CVS



Web del curso

- `http://www.jtech.ua.es/ayto2008` > Material docente
- Apuntes (pdf)
- Ejercicios (pdf + código Java)
- Soluciones (código Java)
- Trasparencias (ppt y pdf)
- Acceso restringido

Módulos

- Java y Herramientas de Desarrollo
- Servidores de aplicaciones
- Componentes web
- Persistencia
- Struts
- Spring
- Servicios Web
- Proyecto de integración SIGEM

Java y Herramientas de Desarrollo

- 10 sesiones (20 horas)
- Una introducción a la programación en Java, y a las herramientas de desarrollo que se utilizarán a lo largo del curso en los demás módulos
- Veremos:
  - Introducción a Java 1.5. Manejo de excepciones, hilos, colecciones, comunicación en red y E/S
  - Herramientas: Eclipse, Ant, JUnit, Log4J, etc.
  - Acceso a bases de datos con JDBC

Servidores Web

- 6 sesiones (12 horas)
- Una introducción a las aplicaciones web, protocolos empleados, estructura de dichas aplicaciones, configuración y puesta en funcionamiento en un servidor web como Tomcat
- Veremos:
  - Elementos del protocolo HTTP
  - Uso del servidor web Tomcat
  - Estructura y configuración de aplicaciones web Java EE

Componentes Web

- 10 sesiones (20 horas)
- Programación de aplicaciones web Java EE mediante servlets y páginas JSP, viendo las correspondencias entre unos y otras.
- Veremos:
  - Introducción a la programación con servlets. Peticiones y respuestas. Sesiones. Comunicación de servlets. Filtros y wrappers
  - Páginas JSP. JavaBeans y lenguaje de expresiones. Librerías de etiquetas
  - AJAX y clientes ricos

Persistencia

- 8 sesiones (16 horas)
- JPA e Hibernate
  - Hibernate es una herramienta ORM (*Object Relational Mapping*) de libre distribución que puede utilizarse sin necesidad de un contenedor
- Veremos:
  - Introducción a JPA
  - Introducción a Hibernate
  - Mapeado de clases persistentes
  - Transacciones, *entity manager* y consultas

Struts

- 4 sesiones (8 horas)
- *Framework* para hacer la parte web de la aplicación (la llamada *capa de presentación*)
- Veremos:
  - MVC, es decir, separación entre presentación, control y trabajo interno (modelo)

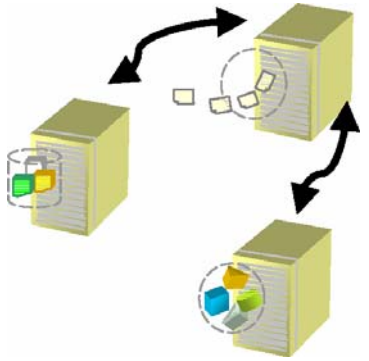
Struts

Spring

- 6 sesiones (12 horas)
- Un *framework* alternativo a la "forma tradicional" de hacer aplicaciones J2EE que incorpora algunas ideas interesantes.
- Veremos:
  - Componentes en Spring: la alternativa a EJBs (versión *clásica*)
  - Inyección de dependencias
  - Programación orientada a aspectos

Servicios Web

- 4 sesiones (8 horas)
- Los servicios Web son servicios accesibles a través de Internet mediante protocolos Web estándar
- Veremos:
  - Invocación de servicios Web
  - Implementación de servicios Web



Proyecto de integración SIGEM

- 12 sesiones (24 horas)
- Estudio de diferentes módulos de SIGEM
  - Utilizaremos la máquina virtual (VM) proporcionada
  - Comenzaremos viendo como poner en marcha la aplicación SIGEM preinstalada en la VM
  - Más adelante estudiaremos y modificaremos el código fuente de la aplicación
- Organización
  - Las sesiones se reparten a lo largo del curso
  - Cada módulo contiene su sesión de integración

La plataforma Java

Carácterísticas más importantes

- Multi-plataforma
- Basadada en estándares
- Soportada por la industria
- Madura
- Fiable
- Gratuita

Arquitectura de aplicaciones enterprise

- Aplicaciones enterprise = aplicaciones multi-usuario
- Elementos que constituyen una aplicación enterprise
  - Lógica de presentación
  - Lógica de negocio
  - Lógica de datos
  - Servicios del sistema (seguridad, logging, transaccionalidad)
- Una arquitectura de una aplicación define cómo organizar estos elementos en un sistema informático

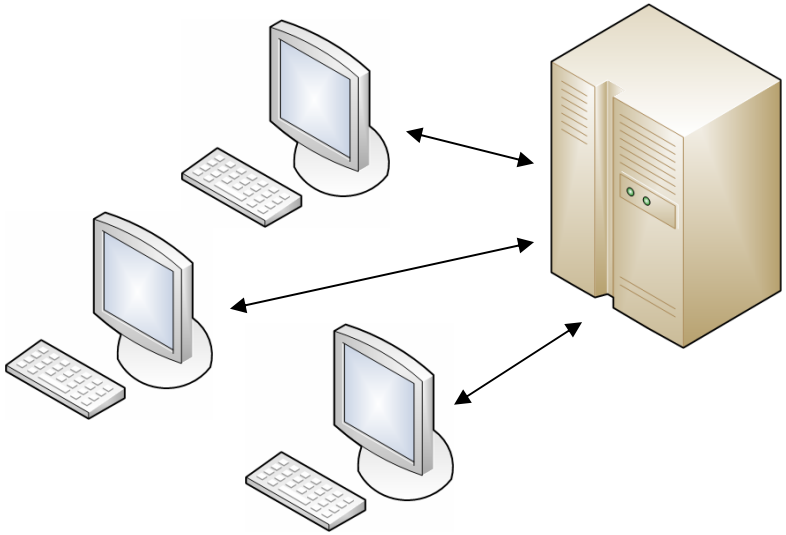
Arquitecturas para aplicaciones cambiantes

- El cambio es una constante en la informática (¡y en la empresa!)
- Un criterio fundamental para comparar arquitecturas es su capacidad para asumir cambios
  - Aumento número de usuarios
  - Incorporación de nuevas funcionalidades
  - Ampliación a nuevos dispostivos (móviles, PDAs, Web, ...)
  - Conexión a nuevas aplicaciones y bases de datos

Evolución de las arquitecturas de aplicaciones

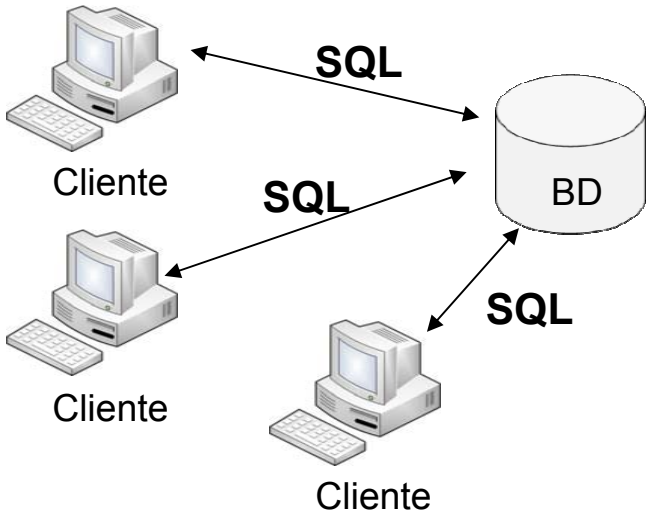
- Una capa
- Cliente-servidor
- Tres capas
- Multi-capa

Capa única



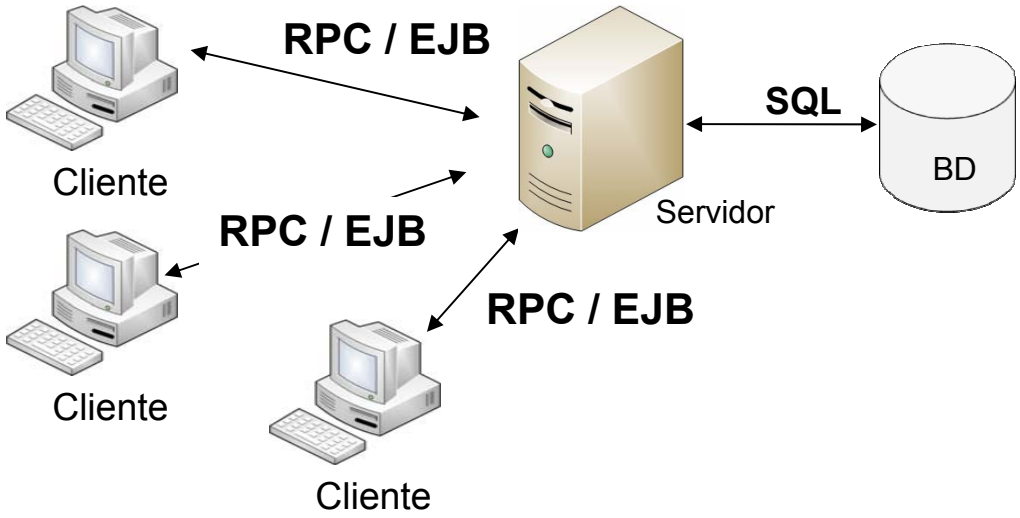
- Terminales tontas se conectan al mainframe
- Modelo centralizado y aplicación monolítica que se ejecuta en el servidor
- Ejemplos
  - Aplicaciones de gestión años 80
  - UNIX y terminales con X11

Arquitectura cliente-servidor clásica



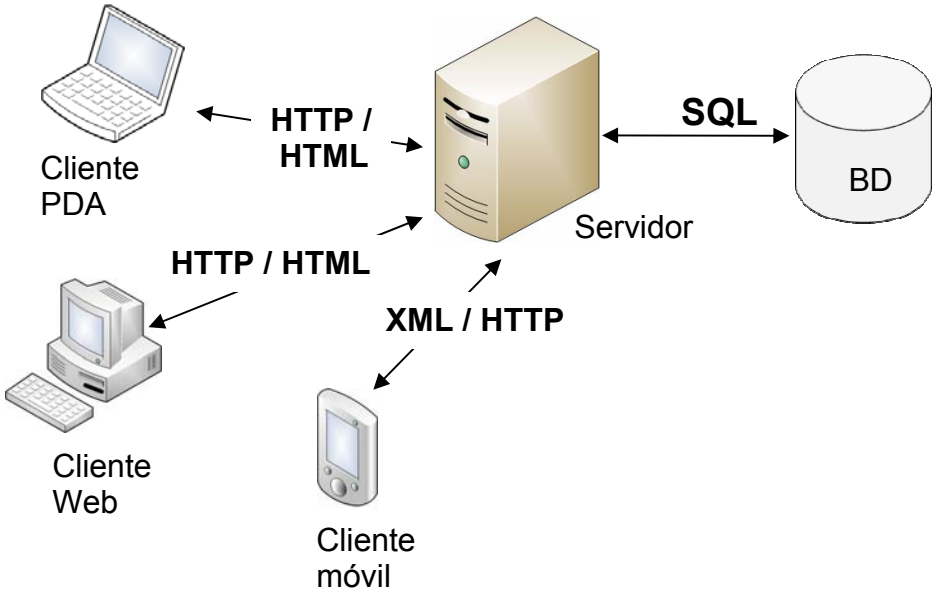
- Los clientes ejecutan programas que se comunican con el servidor de BD utilizando algún protocolo de acceso a datos como SQL
- Aplicación monolítica que se ejecuta en el cliente (cliente grueso)

Arquitectura cliente-servidor moderna



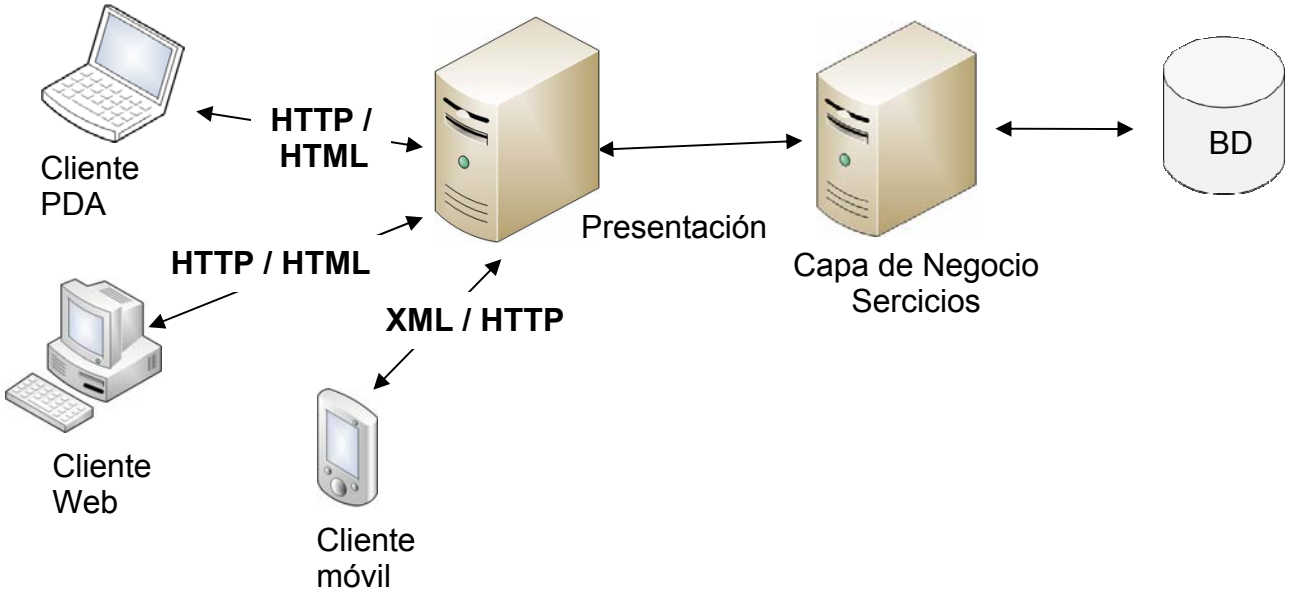
- Similar a la clásica, pero el protocolo de conexión (y transmisión de datos) con el servidor es más avanzado que SQL: una llamada a un procedimiento remoto o a un EJB (objeto remoto)
- Aplicación monolítica que se ejecuta en el cliente (cliente grueso)

Arquitectura de tres capas



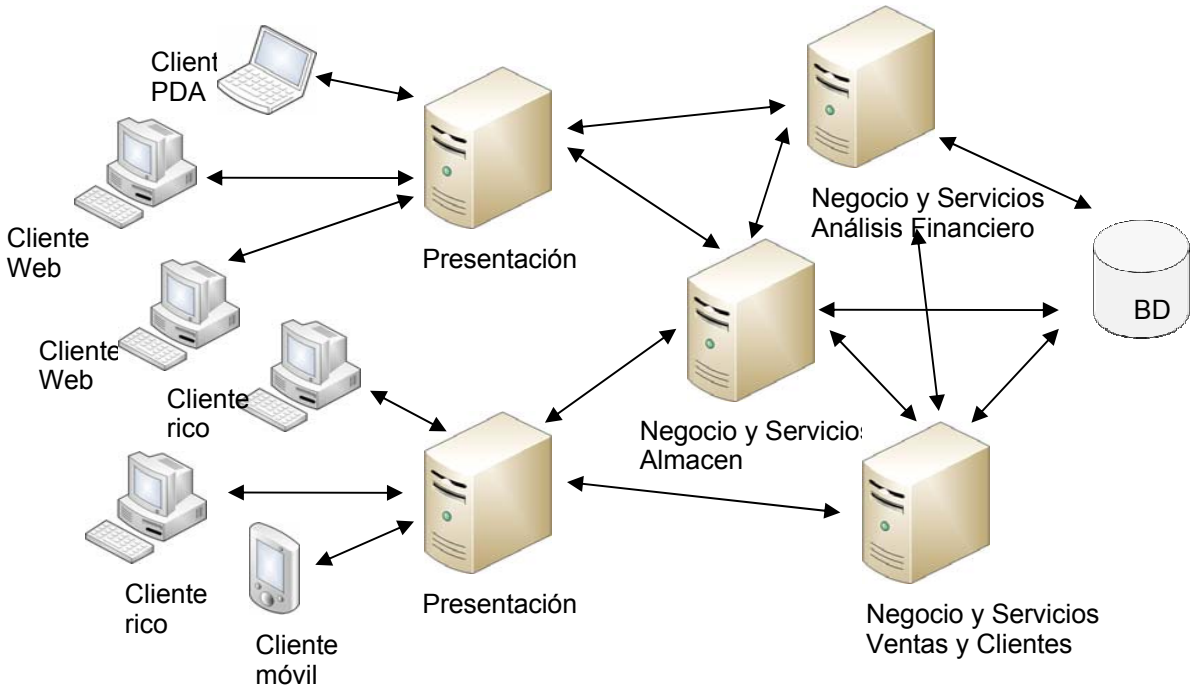
- Los clientes únicamente realizan el “renderizado” de la “vista” proporcionada por el servidor (cliente fino)
- La capa de negocio, de datos y la generación de la “vista” residen en el servidor intermedio
- Ejemplo: aplicación web

Arquitectura multi-capa (1)



- Separación entre la generación de la vista (servidor de presentación) y la capa de negocios y de servicios
- Protocolos de comunicación entre el servidor de presentación y el servidor de negocio: EJB, HTTP, ...

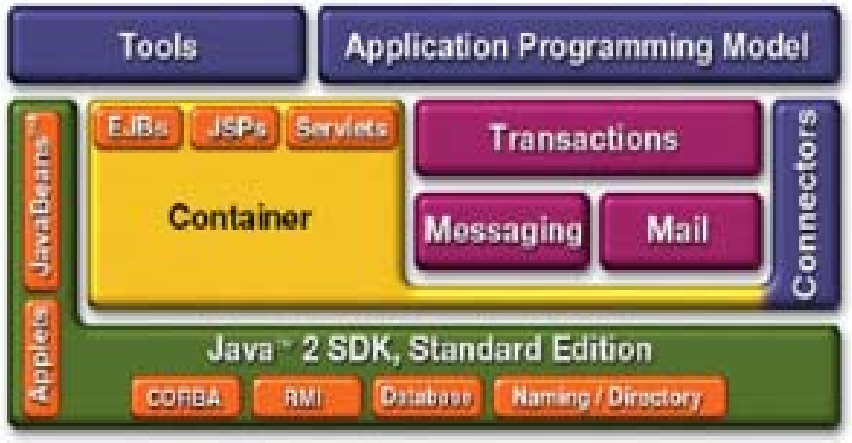
Arquitectura multi-capa (2)



- Múltiples servidores de presentación
- Múltiples servidores de aplicación
- Ejemplo: banca, universidad, concesionarios automóviles, ...

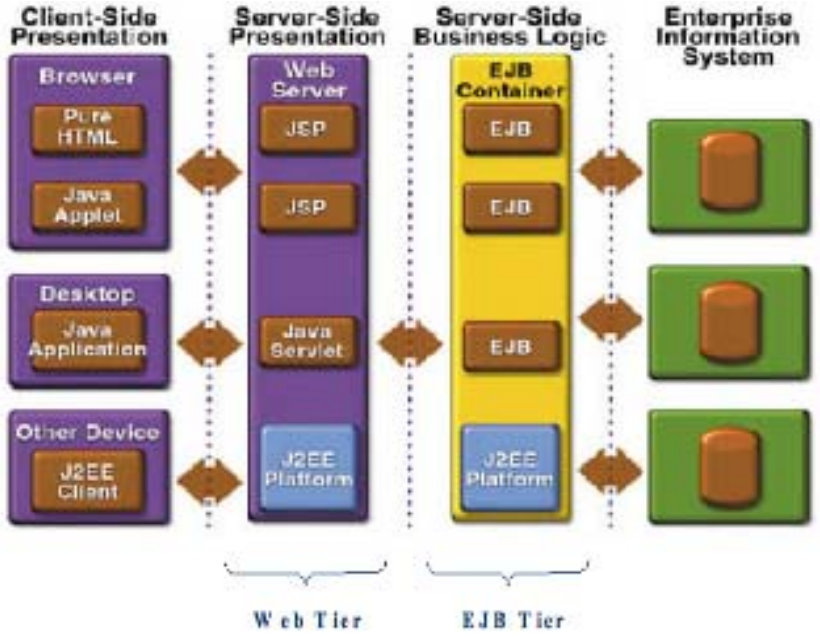
La plataforma Java EE

- Conjunto de especificaciones (APIs) Java para el desarrollo de aplicaciones enterprise
- Las aplicaciones enterprise se ejecutan dentro de un servidor (servidor Web -Apache Tomcat- o servidor de aplicaciones -BEA, GlassFish, Jboss, ...-)
- Las APIs son estándar: las aplicaciones pueden desarrollarse con distintos entornos (Eclipse o NetBeans) y pueden desplegarse en distintos servidores Web o de aplicaciones



Capas de aplicación

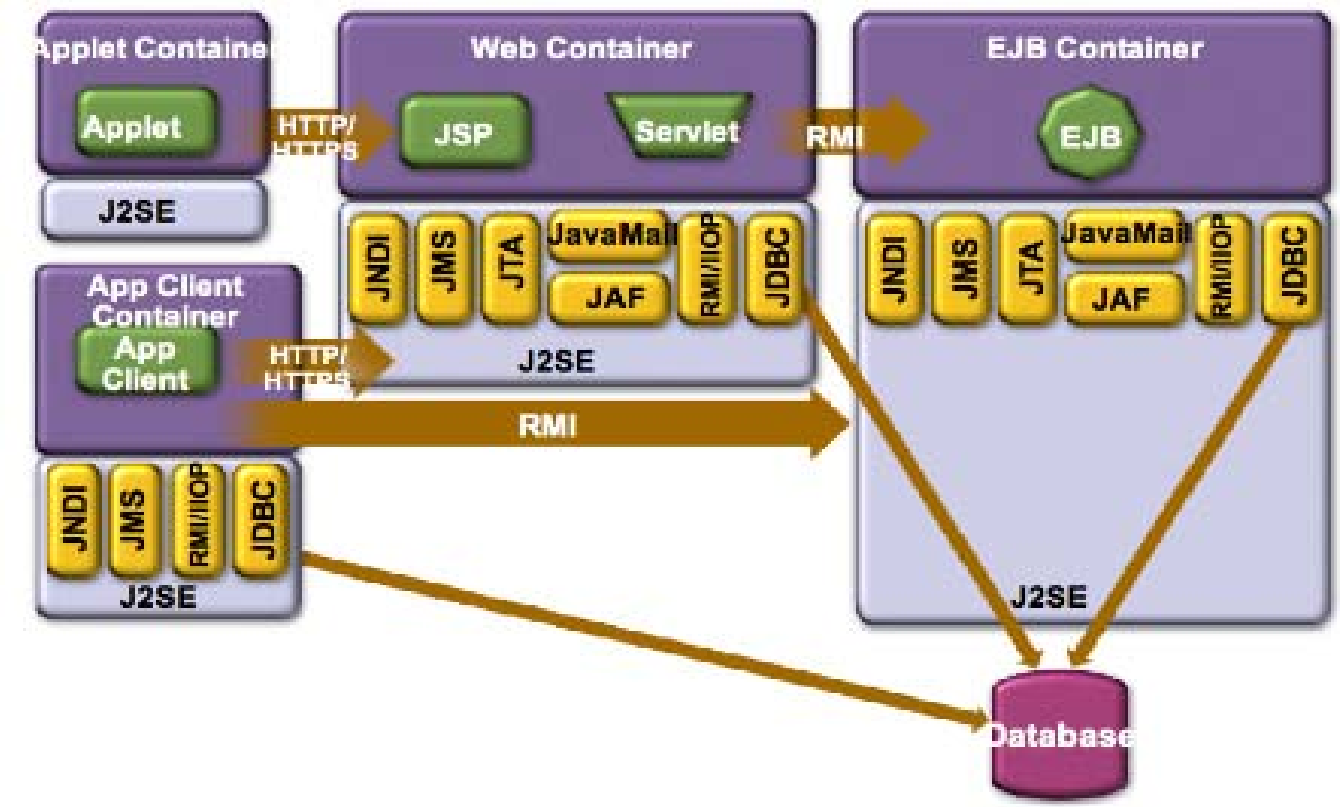
- La plataforma Java EE es completa: tiene APIs para gestionar las distintas capas de una aplicación enterprise
- Capa de presentación en el cliente: Applets Java, Java SE, Java ME (junto con estándares Web como HTML, JavaScript, ...)
- Generación de presentación en el servidor: Servlets, JSP y JSF
- Capa de negocio: EJB
- Capa de datos y EIS: JDBC, Java Persistence API, conectores



Contenedores en Java EE

- Los contenedores son programas Java que están ejecutándose en los servidores y que gestionan y dan soporte a objetos Java Enterprise
- Los más importantes
  - Contenedor Web: presente en los servidores Web; gestiona servlets
  - Contenedor EJB: presente en los servidores de aplicaciones; gestiona componentes EJB
- Proporcionan múltiples servicios Java EE como acceso a bases de datos, transacciones y otros servicios no estándar como clustering o concurrencia

Contenedores en Java EE



APIs y Tecnologías Java EE 5

Aplicaciones Web

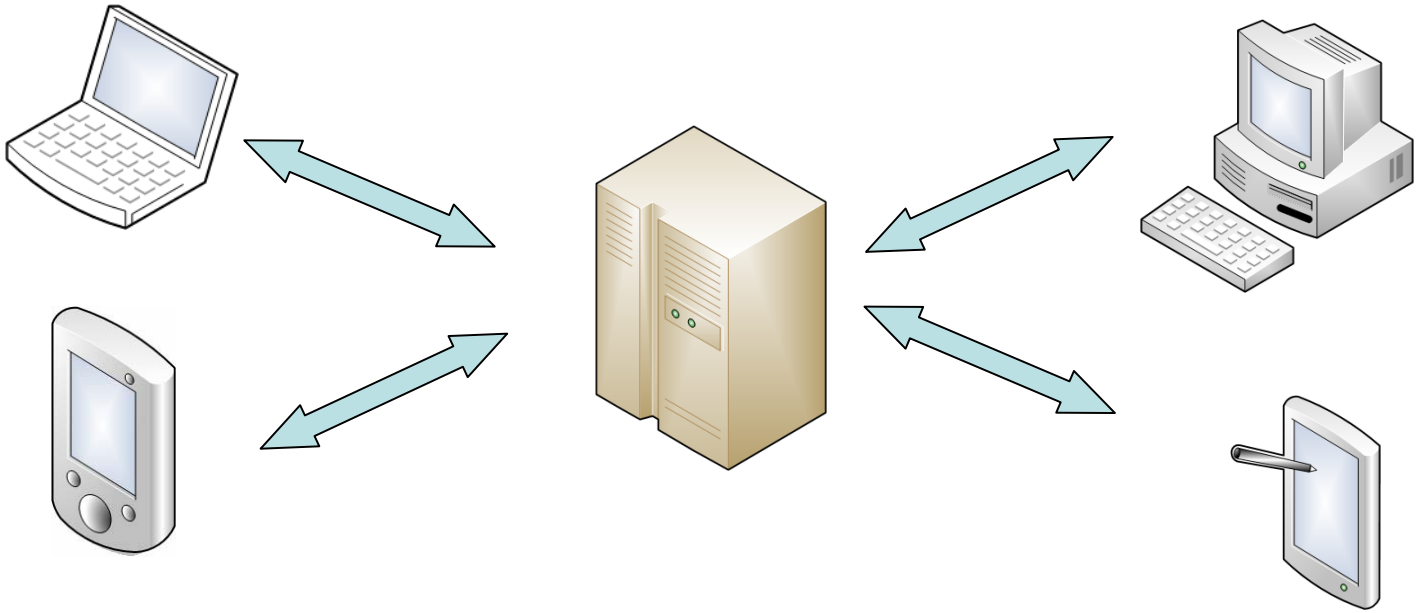
- Java Servlet 2.5
- JavaServer Pages (JSP) 2.1
- JSP Standard Tag Library
- JavaServer Faces (JSF) 1.2

Aplicaciones Enterprise

- Enterprise JavaBeans (EJB) 3.0
- Java Message Service API (JMS)
- JavaMail
- Java Persistence API (JPA)
- Java Transaction API (JTA)
- Java API para Servicios Web XML (JAX-WS)
- Java API para RPC XML (JAX-RPC)

Servidores web

- Un servidor web centraliza todas las peticiones de varios usuarios a una web

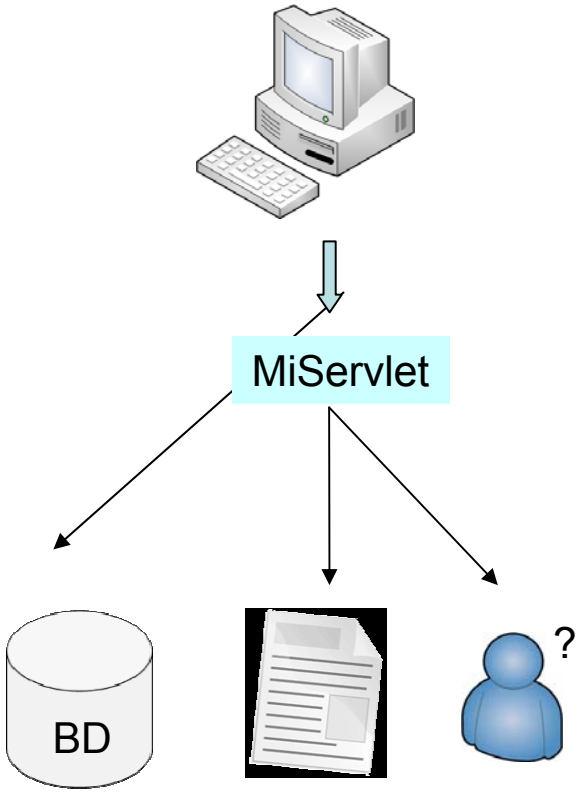


Servidores web

- Bases del protocolo HTTP
- Utilizaremos software libre (Tomcat)
- Gran parte de las configuraciones son aplicables a otros muchos servidores web y de aplicaciones (Resin, WebLogic, JBoss…)

Servlets

- Los servlets son programas Java instalados en un servidor web que:
  - Generan contenido web (HTML)
  - Cargan páginas web
  - Controlan el acceso no autorizado, gestionan las conexiones con bases de datos…
  - etc

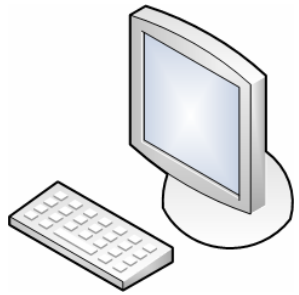


Java Server Pages (JSP)

- Permiten separar la presentación de la lógica de negocio
  - Los JSP son adecuados para generar la presentación en HTML o XML

```
<%@page import="java.util.Date" %>
<html>
<head> <title> Ejemplo de JSP </title> </head>
<body>
Hoy es <%= (new Date()).toString() %>
</body>
</html>
```

  - La lógica de negocio se llevará a otros componentes: JavaBeans, *taglibs*, EJBs, …
- Se construye sobre la tecnología de Servlets
- Ampliable mediante librerías de *tags*
- Lenguaje de expresiones propio



Frameworks

- Un *framework* es un diseño reusable que podemos usar en nuestra aplicación
  - Generalmente consiste en la práctica en una serie de *clases abstractas*, librerías adicionales, etc.
- Framework vs. Librería
  - Una librería es un API "listo para usar". Un *framework* es un "armazón" sobre el que construir nuestro código (de ahí las clases abstractas)
  - Podemos ver un *framework* como **la implementación de una filosofía de diseño**

Frameworks en JavaEE

- Resuelven problemas **comunes**, permitiendo al desarrollador concentrarse en lo **particular** de la aplicación
- Ejemplos paradigmáticos:
  - Spring

    Cubre todas las capas de la arquitectura

    Implementa una serie de “buenos principios”, que se pueden resumir en las ideas de “contenedores ligeros” y “código no invasivo”
  - Struts

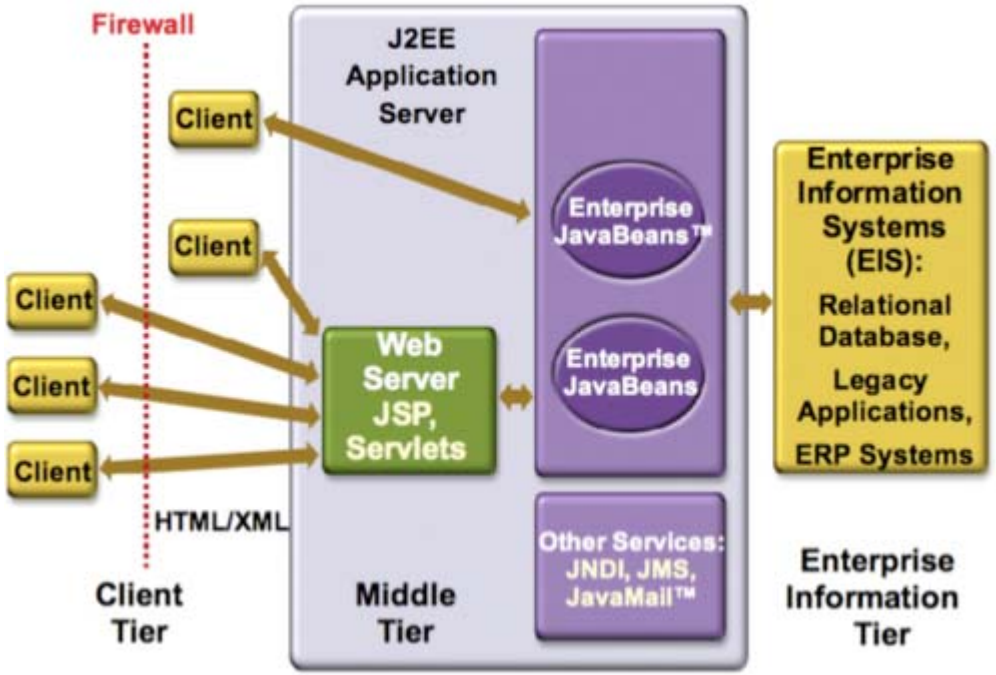
    Cubre únicamente la capa de presentación

    Implementa el patrón de diseño Modelo/Vista/Controlador

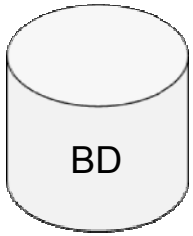
Servidores de aplicaciones

- Proporciona servicios que soportan la ejecución de aplicaciones J2EE:
  - Contenedor de servlets, contenedor de EJBs, clustering (balanceo de carga, recuperación ante fallos), etc.



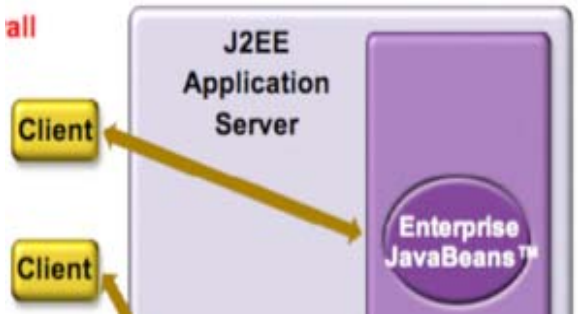
Java Persistence API

- Clases y objetos persistentes
- Los objetos persistentes se mapean con la base de datos:
  - Las clases se mapean con tablas
  - Las objetos se mapean con filas de las tablas
- Las operaciones habituales de una BD (creación, actualización y búsqueda) se realizan mediante la creación y actualización de objetos de clases persistentes.



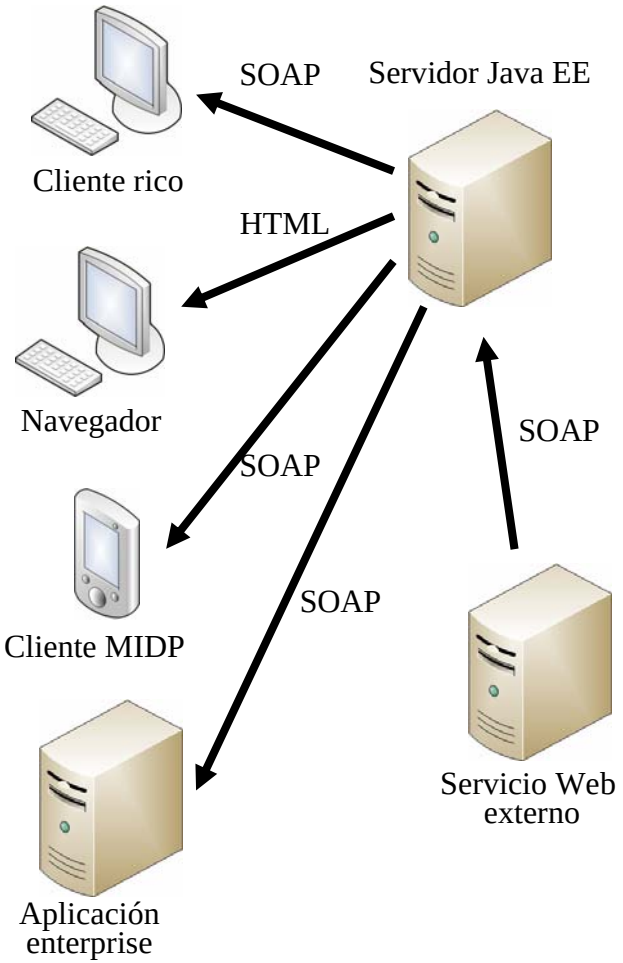
Componentes EJB

- Un componente EJB (o Enterprise JavaBean) es un objeto remoto que reside en un contenedor EJB de un servidor de aplicaciones
- Proporciona acceso a un conjunto de servicios definidos por su interfaz de negocio
- El contenedor EJB:
  - Recubre la interfaz de negocio con un conjunto de servicios añadidos (seguridad, transaccionalidad, concurrencia, escalabilidad)
  - Proporciona acceso a un conjunto de recursos (BD, colas de mensajes, ...)
- El desarrollo de componentes EJB posibilita la construcción de aplicaciones débilmente acopladas



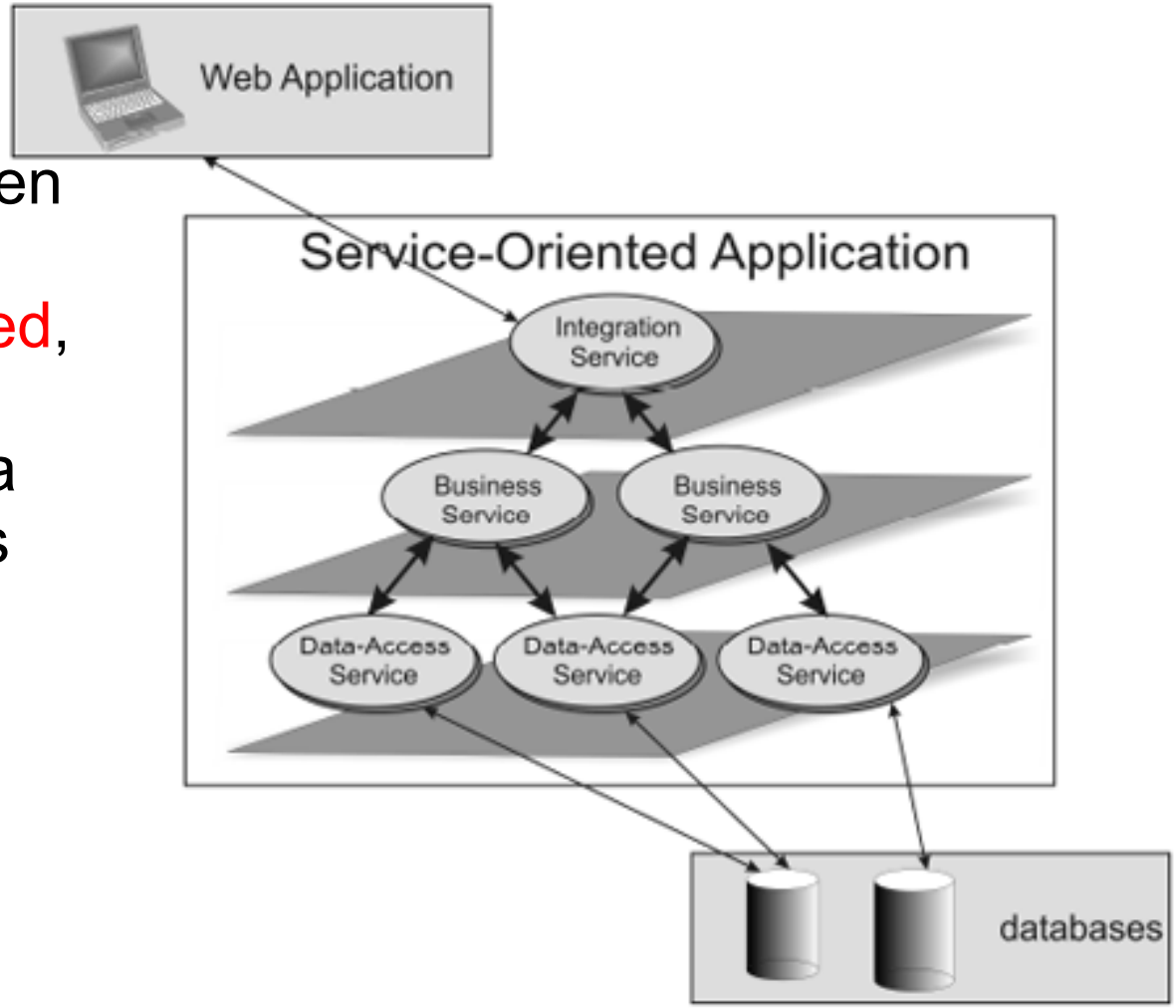
Servicios Web

- Componentes Java EE pueden ser exportados como Servicios Web
  - Servicios accesibles a través de Internet mediante protocolos Web estándar.
  - Similar a RPC con independencia del lenguaje
  - No conflictivo con *firewalls*
  - Aplicaciones distribuidas en Internet
  - Se invocan mediante protocolo HTTP
- Mensajes codificados en XML
  - SOAP: Llamada y respuesta de un
  - servicio
  - WSLD: Descriptor de servicios
  - UDDI: Localización de servicios

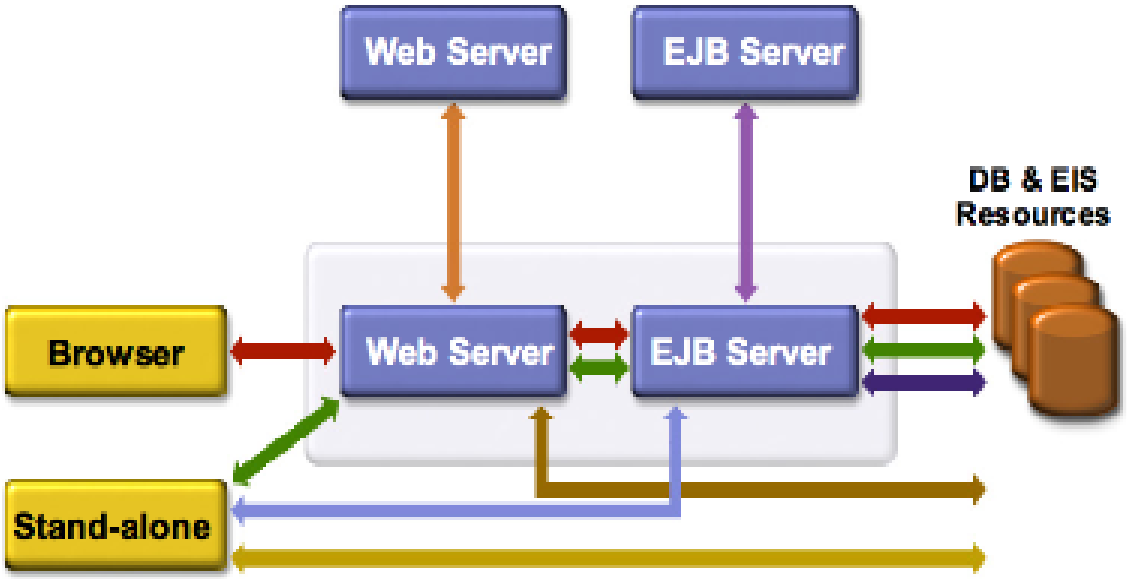


Service-Oriented Architecture

- Es una forma de organizar el software, basada en servicios que se ejecutan en una red, que facilita una respuesta rápida a los requerimientos cambiantes del mercado



Arquitectura con Java EE

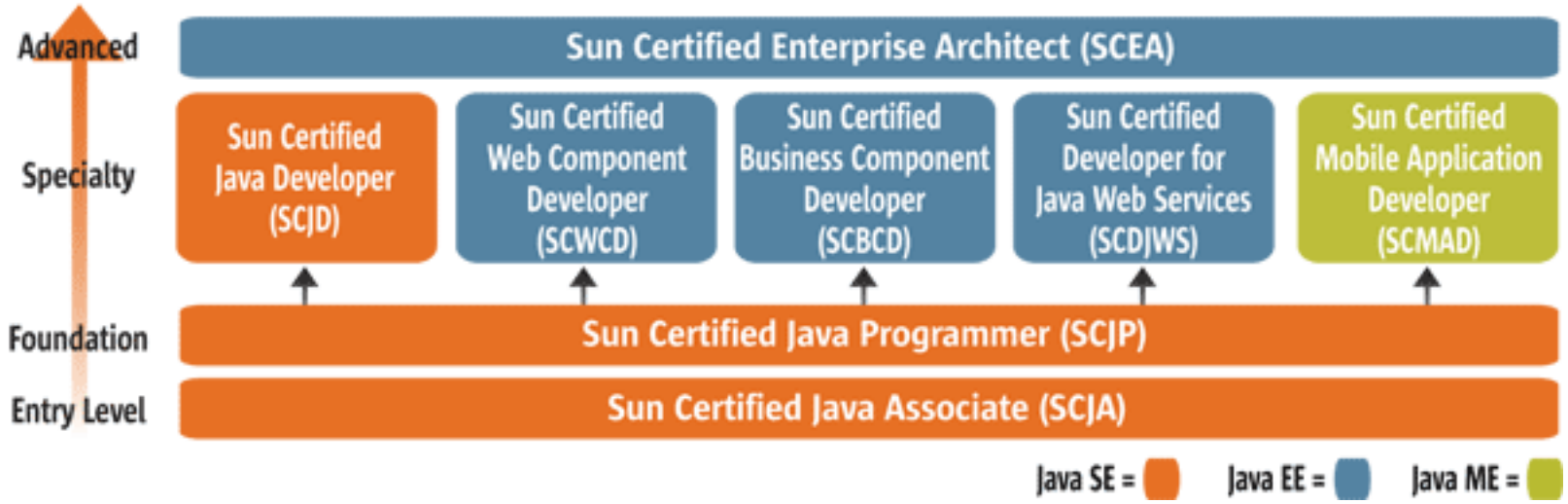


- La plataforma Java EE permite múltiples configuraciones en la arquitectura de la aplicación
- Tarea del arquitecto de proyectos: decidir qué arquitectura es la más adecuada para cada ocasión

RoadMap Java: enlaces

- Sun
  - http://java.sun.com
  - http://java.sun.com/javaee/
  - http://java.net/
- Java Hispano
  - http://www.javahispano.org/
- Java Lobby
  - http://www.javalobby.org/
- The Server Side
  - http://www.theserverside.com/

Roadmap Java: certificación



- La certificación Java cada vez está más valorada en el mercado profesional
- Nuestro Especialista te da la base para abordar con éxito los exámenes de certificación: SCWCD y SCBCD
- Precio de cada examen: alrededor de 250 €

¿Preguntas...?