Implementación de Web Services

Ingeniería de Sistemas

UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA
Escuela de Ingeniería Informática

Plataforma Java EE

- Java Platform Enterprise Edition se basa en la especificación Java SE y está destinada a desarrollar aplicaciones empresariales en el lenguaje de programación Java que se ejecutan en un servidor de aplicaciones como Tomcat, weblogic, jboss, etc.
- Se trata de un conjunto de especificaciones que permiten soluciones para el desarrollo, despliegue y gestión de aplicaciones multicapa centradas en servidor.
- Por especificación se entiende el detalle de cada una de las tecnologías dentro de la plataforma Java EE. Un conjunto de reglas que dictan cómo debe desarrollarse ese producto de tal forma que se pueda garantizar que una aplicación desarrollada siguiendo las especificaciones de Java EE pueda desplegarse y ejecutarse.

Tecnologías de Java EE

- Enterprise JavaBeans (EJB)
- Java Servlet
- JavaServer Page (JSP)
- JavaServer Pages Standard Tag Library (JSTL).
- JavaServer Faces (JSF)
- Java Message Service (JMS)
- Java Transaction API (JTA)
- JavaMail API y JavaBeans Activation Framework (JAF)
- Tecnologías XML (JAXP, JAX-RPC, JAX-WS, JAXB, SAAJ, JAXR)
- JPA, JDBC API
- Java Naming and Directory Interface (JNDI)
- Java Authentication and Authorization Service (JAAS)

Aplicación multi-capa

- La plataforma Java EE está destinada a desarrollar aplicaciones distribuidas con una arquitectura multi-capa. El desarrollo de aplicaciones se puede separar en diferentes capas según su función.
- Las aplicaciones Java EE se consideran aplicaciones de tres capas porque se distribuyen en tres localizaciones: ordenadores clientes, el sistema donde se ejecuta el servidor de aplicaciones y el sistema donde reside la base de datos.

Descripción de capas

- La capa del cliente (Client-tier) que es la capa destinada a mostrar la interfaz gráfica de usuario. Las aplicaciones Java EE pueden ser una aplicación Java Swing normal, o una aplicación Web renderizada en un navegador. Esta capa se ejecuta en el ordenador cliente.
- La capa de la lógica de negocio (Business-tier) y la capa de la lógica de presentación (Web-tier). Estas capas se ejecutan en el servidor de aplicaciones.
- La capa de los datos (Data-tier) que es la capa destinada a la gestión de los datos. Esta capa puede separarse a su vez en una o más capas.

Servidor de aplicaciones

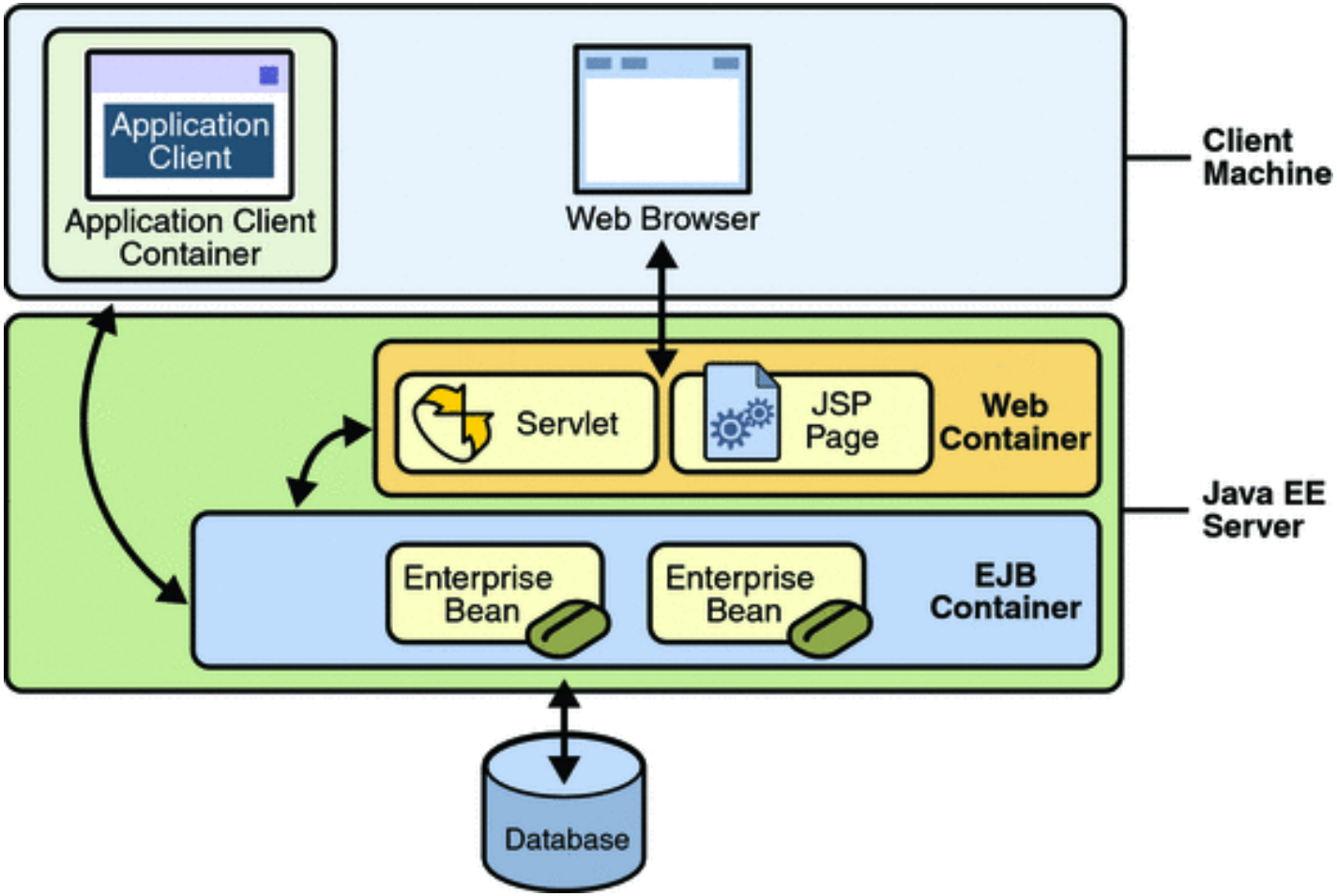
- Las aplicaciones empresariales Java EE se ejecutan en un servidor de aplicaciones y están formadas por un conjunto de módulos donde cada módulo es un conjunto de uno o más componentes que se ejecutan en el mismo contenedor.
- Un componente es una unidad de software, puede ser un componente web como una página JSP o un servlet, un componente EJB, etc. Estos componentes se ejecutan dentro de su correspondiente contenedor dentro del servidor de aplicaciones.
- El contenedor es un entorno de ejecución que gestiona los componentes. Los componentes deben cumplir el contrato que establece el contenedor. El contrato es un conjunto de métodos que debe implementar el componente y que permite al contenedor interactuar con él.

Contenedores

- Son los encargados de gestionar el ciclo de vida de los componentes, realizar la reserva de recursos, etc.
- Proporcionan una serie de servicios que el componente puede utilizar. Algunos de estos servicios se declaran en vez de programarse.
- La declaración se realiza mediante descriptores de despliegue. Cada módulo dispone de un descriptor de despliegue. El descriptor de despliegue es un archivo XML que describe cómo se deben desplegar esos componentes en el contenedor del servidor de aplicaciones.

Tipos de contenedores

Contenedor WEB	Es el encargado de gestionar los componentes servlets y páginas JSP
Contenedor EJBs	Es el encargado de gestionar los componentes EJBs


Application Client
Application Client Container
Web Browser
Client Machine
Servlet
JSP Page
Web Container
Java EE Server
Enterprise Bean
Enterprise Bean
EJB Container
Database

Tipos de módulos

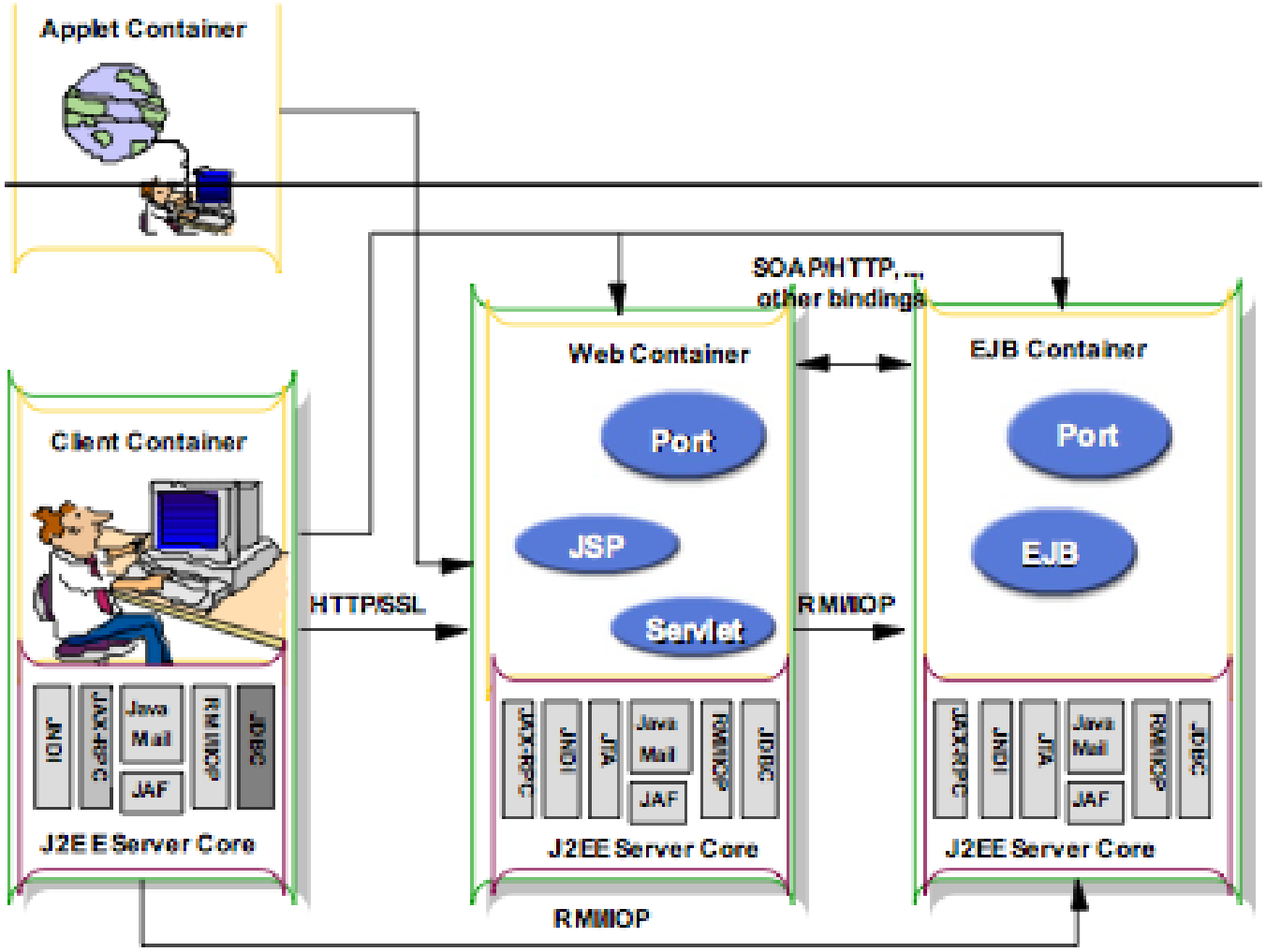
Archivos JAR (Java Archive)	Empleados para empaquetar componentes EJBs, permiten agrupar distintos archivos .java en uno solo.
Archivos WAR (Web Application Archive)	Permiten empaquetar en una sola unidad aplicaciones web completas (servlets, páginas JSPs, contenido estático como imágenes y otros recursos Web).
Archivos EAR (Enterprise Application Archive)	Desplegables en servidores de aplicaciones JEE, contienen archivos WAR y EJBs empaquetos en ficheros JAR.

Tipos de aplicaciones

- Aplicaciones Web JAVA
- Objetos distribuidos EJBs
- Aplicaciones empresariales que engloba a las dos anteriores

WS para Java EE

- La especificación JSR-109 (Implementing Web Services), que define el modelo de programación y arquitectura del soporte de ejecución (run-time) para implementar servicios Web en Java, define una serie de relaciones arquitectónicas requeridas para dichos servicios.
- Como añadido para la plataforma Java EE se incluye un componente Port que pueda ser referenciado desde un cliente, así como desde los contenedores web y EJB.


Applet Container
SOAP/HTTP, ...
other bindings
Web Container
EJB Container
Client Container
Port
Port
JSP
EJB
Servlet
HTTP/SSL
RMI/IIOP
JNDI
JAXR/SAAJ
Java Mail
JAF
RMI/IIOP
JDBC
J2EE Server Core
JAX-RPC
JNDI
JTA
Java Mail
JAF
RMI/IIOP
JDBC
J2EE Server Core
JAX-RPC
JNDI
JTA
Java Mail
JAF
RMI/IIOP
JDBC
J2EE Server Core
RMI/IIOP

Implementación de WS

- Los servicios pueden implementarse de dos formas: como una clase Java que se ejecuta en un contenedor Web o como un EJB de sesión *stateless* o *singleton* en un contenedor EJB.
- El contenedor del servicio debe proporcionar soporte para la gestión del ciclo de vida de la implementación del servicio, de la concurrencia de la invocación de los métodos del servicio, y de la seguridad.

Interoperabilidad de WS

- Metro (http://metro.java.net) es la propuesta de Sun para la interoperabilidad de los WS utilizando la plataforma Java.
- WCF (*Windows Communication Foundation*) es la aportación de Microsoft para la plataforma .NET.

Metro

Metro constituye la implementación de la *web service stack* (colección de tecnologías de servicios Web) y forma parte del proyecto GlassFish, un servidor de aplicaciones Java EE de código abierto.

Metro se estructura en tres componentes:

- WSIT es el conjunto de tecnologías (Web Services Interoperabe Technologies) que permiten la interoperabilidad con .NET
- JAX-WS RI es la implementación de referencia del estándar JAX-WS (especificación JSR 224: *Java API for XML-Based Web Services*)
- JAXB RI es la implementación de referencia del API para la capa de enlazado de datos (JAXB: *Java Architecture for XML binding*).

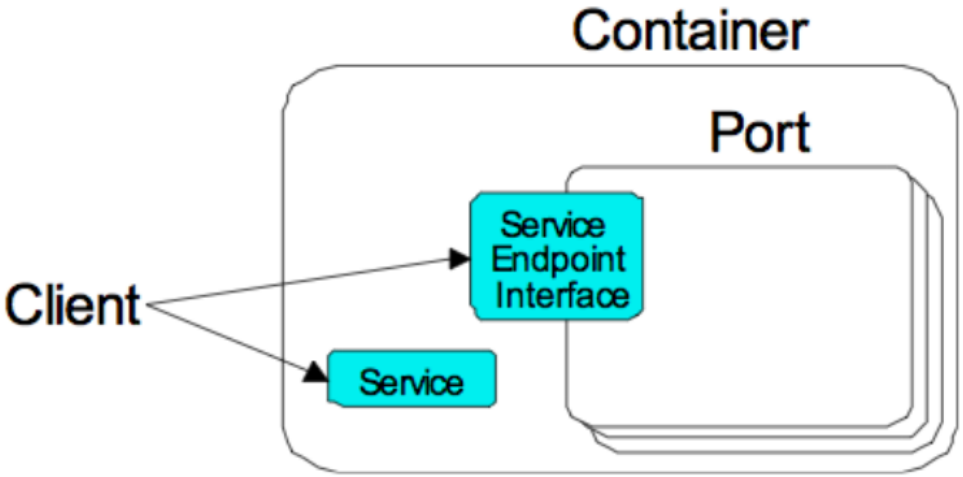
JAX-WS

- JAX-WS (Java API for XML Web Services) es una interfaz de programación de aplicaciones (API) de Java en Extensible Markup Language (XML) para la creación de servicios web .
- Es parte del Java Web Services Development Pack y al igual que las otras API de Java EE, utiliza anotaciones introducidas en Java SE 5 para simplificar el desarrollo y despliegue de los clientes y endpoints de servicios web.
- JAX-WS RI es la implementación de referencia de JAX-WS.

JAX-WS

- JAX-WS permite leer o generar mensajes SOAP para la invocación de métodos remotos.
- Una invocación de una operación a un servicio web se representa con un protocolo basado en XML (p.e. SOAP). Aunque los mensajes SOAP son complejos, la API JAX-WS oculta esta complejidad al desarrollador.
- En la parte del servidor, el desarrollador especifica las operaciones del servicio web definiendo métodos en una interfaz escrita en lenguaje Java y también codifica una o más clases que implementan dichos métodos.
- Los programas cliente también son fáciles de codificar. Un cliente crea un proxy (un objeto local que representa el servicio) y luego invoca los métodos sobre el proxy.
- Con JAX-WS, el desarrollador no necesita generar o "parsear" mensajes SOAP. El runtime de JAX-WS convierte las llamadas y respuestas del API en y desde los mensajes SOAP.

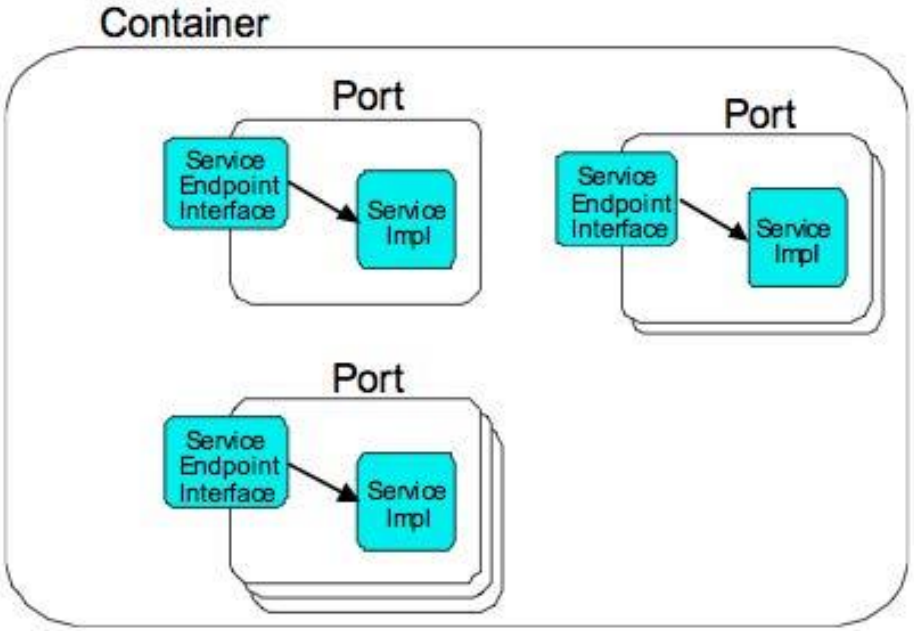
WS desde la vista del cliente



WS desde la vista del cliente

- La clase/interfaz **service** (SI) define los métodos que un cliente puede utilizar para acceder a un **Port** de un servicio Web. El cliente la utiliza para obtener el acceso a un Port.
- La clase/interfaz Service se define en la especificación JAX-WS, pero su comportamiento viene definido en el documento WSDL proporcionado por el proveedor del servicio Web.
- Las herramientas de despliegue del contenedor proporcionan una implementación de los métodos de la clase/interfaz service generada por JAX-WS.
- El cliente accede a una implementación de un servicio web utilizando el SEI. Dicho SEI es especificado por el proveedor del servicio. Las herramientas del despliegue y el *run-time* del contenedor proporciona las clases en la parte del servidor que van a atender las peticiones SOAP sobre la implementación de dicho servicio de los métodos especificados en el SEI.

Desde la vista del servidor



WS desde la vista del servidor

- Un componente **Port** define la vista del servidor de un servicio web. Cada *Port* proporciona un servicio en una dirección física particular. La implementación del servicio (*Service Implementation Bean*) depende del contenedor del componente Port, pero en general es una clase Java que puede implementar los métodos definidos en el SEI (*Service Endpoint Interface*).
- Un servicio web es un conjunto de *Ports* que difieren únicamente en su dirección física, y son mapeados en componentes Port separados, cada uno con su potencialmente único, pero probablemente compartido, Service Implementation Bean.
- Una implementación de un servicio web JAX-WS reside en un contenedor web, y por tanto puede desplegarse en un servidor web o un servidor de aplicaciones, una implementación EJB, reside en un contenedor EJB y sólo podrá desplegarse en un servidor de aplicaciones.

JAX-RS

- Java API for RESTful Web Services es una API del lenguaje de programación Java que proporciona soporte en la creación de servicios web de acuerdo con el estilo arquitectónico Representational State Transfer (REST).
- JAX-RS usa anotaciones, introducidas en Java SE 5, para simplificar el desarrollo y despliegue de los clientes y endpoints de los servicios web.
- A partir de la versión 1.1 en adelante, JAX-RS es una parte oficial de Java EE 6.
- Jersey es la implementación de referencia de JAX-RS.

Anotaciones principales

JAX-RS proporciona algunas anotaciones para ayudar a mapear una clase recurso (un POJO) como un recurso web. Entre estas anotaciones se incluyen:

- @Path especifica la ruta de acceso relativa para una clase recurso o método.
- @GET, @PUT, @POST, @DELETE y @HEAD especifican el tipo de petición HTTP de un recurso.
- @Produces especifica los tipos de medios MIME de respuesta.
- @Consumes especifica los tipos de medios de petición aceptados.

Anotaciones adicionales

- @PathParam enlaza el parámetro a un segmento de ruta.
- @QueryParam enlaza el parámetro al valor de un parámetro de consulta HTTP.
- @MatrixParam enlaza el parámetro al valor de un parámetro de matriz de HTTP.
- @HeaderParam enlaza el parámetro a un valor de cabecera HTTP.
- @CookieParam enlaza el parámetro a un valor de cookie.
- @FormParam enlaza el parámetro a un valor de formulario.
- @DefaultValue especifica un valor por defecto para los enlaces anteriores cuando la clave no es encontrada.
- @Context devuelve todo el contexto del objeto. (Por ejemplo: @Context HttpServletRequest request).