{ Java Enterprise Edition }

Arquitectura de componentes JEE

Formador: Ezequiel Llarena Borges

Herramientas de software

- JDK 1.8+
- IDE Eclipse Juno (JEE developer)
- Apache Tomcat 7.0+
- Maven 3.0+
- MySQL Server 6.3.10

Breve…, muy breve historia de Java

- Se inició como proyecto en 1991, como una herramienta de programación para ser usada en un proyecto desarrollado por Sun Microsystems.
- El lenguaje, llamado inicialmente *Oak*, pasó a denominarse *Green*.
- Finalmente se renombró a **Java**, posiblemente por la variante de café que tomaban los desarrolladores de Oak.
- Sun publicó la primera implementación, llamada JDK (Java Development Kit) 1.0, en 1995, y JDK 1.1 en 1997, centrado principalmente en los applets.
- Versión mejorada, ampliada y con nuevo nombre: J2SE (Java 2 Standard Edition) 1.2, nombre clave *playground*, que se publicó a fines de 1998.
- En 1999, nace J2EE (Java 2 Enterprise Edition) 1.2, la versión empresarial de Java.

James Gosling

- Oracle adquirió Sun en 2009-2010, heredando la plataforma Java. Luego adquirió BEA, haciéndose dueño de WebLogic, un importante servidor que implementa los estándares empresariales de Java EE.
- En 2009 se publicó Java EE 6, una mejora significativa a los estándares empresariales.
- Pasaron casi cinco años, hasta 2011, para que se publicara la siguiente versión, Java SE 7, nombre clave *Dolphin*, que incorpora nuevas características.

- El lema de Java es:

WORA: Write once, run anywhere

(escríbelo una vez, ejecútalo en cualquier parte)

* Java funciona básicamente con el siguiente esquema:

fuente

Crear un archivo fuente (**.java**), utilizando la especificación del lenguaje Java.

compilador

Pasar el archivo fuente a través de un compilador, que sólo funciona cuando no hay errores de codificación.

salida (binario)

El compilador crea un nuevo archivo (**.class**), llamado el **bytecode**. Cualquier dispositivo que sea capaz de **interpretarlo** puede ejecutarlo.

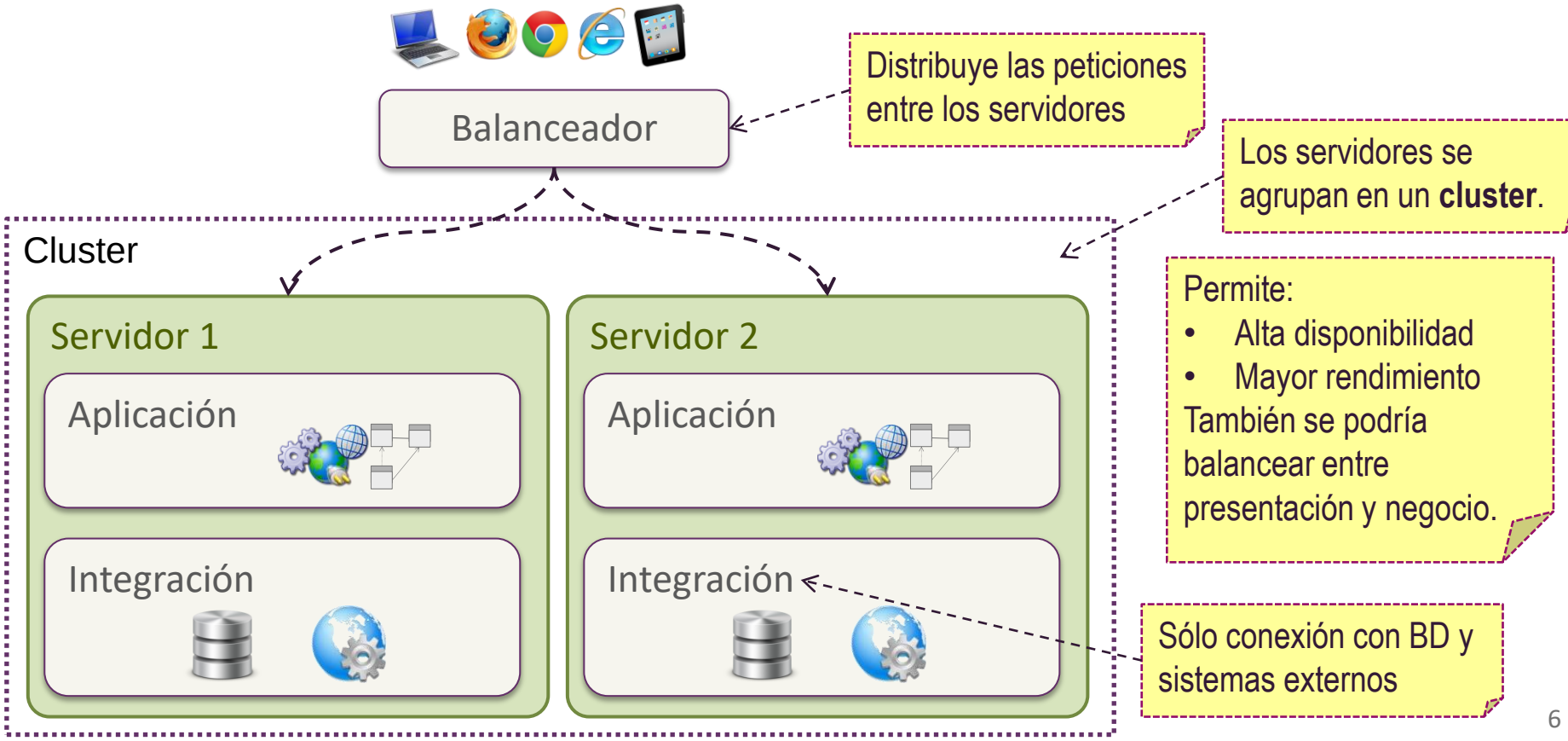
Java Virtual Machine

La máquina virtual Java (**JVM**) de la plataforma/S.O. donde se ejecuta **interpreta** el bytecode y lo ejecuta.

funcionamiento básico

Despliegue en Cluster

Hay varias topologías que se pueden utilizar. Ejemplo:



Fundamentos de arquitectura

Las ventajas de la utilización de una arquitectura multicapa en un entorno con cluster son las siguientes:

- Escalabilidad horizontal: Se puede aumentar la capacidad del cluster agregando máquinas en las capas que lo necesiten.
- Rendimiento: La separación de funcionalidades y el balanceo de carga permiten que las peticiones se respondan en menor tiempo.
- Alta disponibilidad: Si un servidor falla, los demás pueden responder hasta que se reponga.

Ejemplos de soluciones basadas en arquitectura multicapa:

- JEE
- Framework Spring

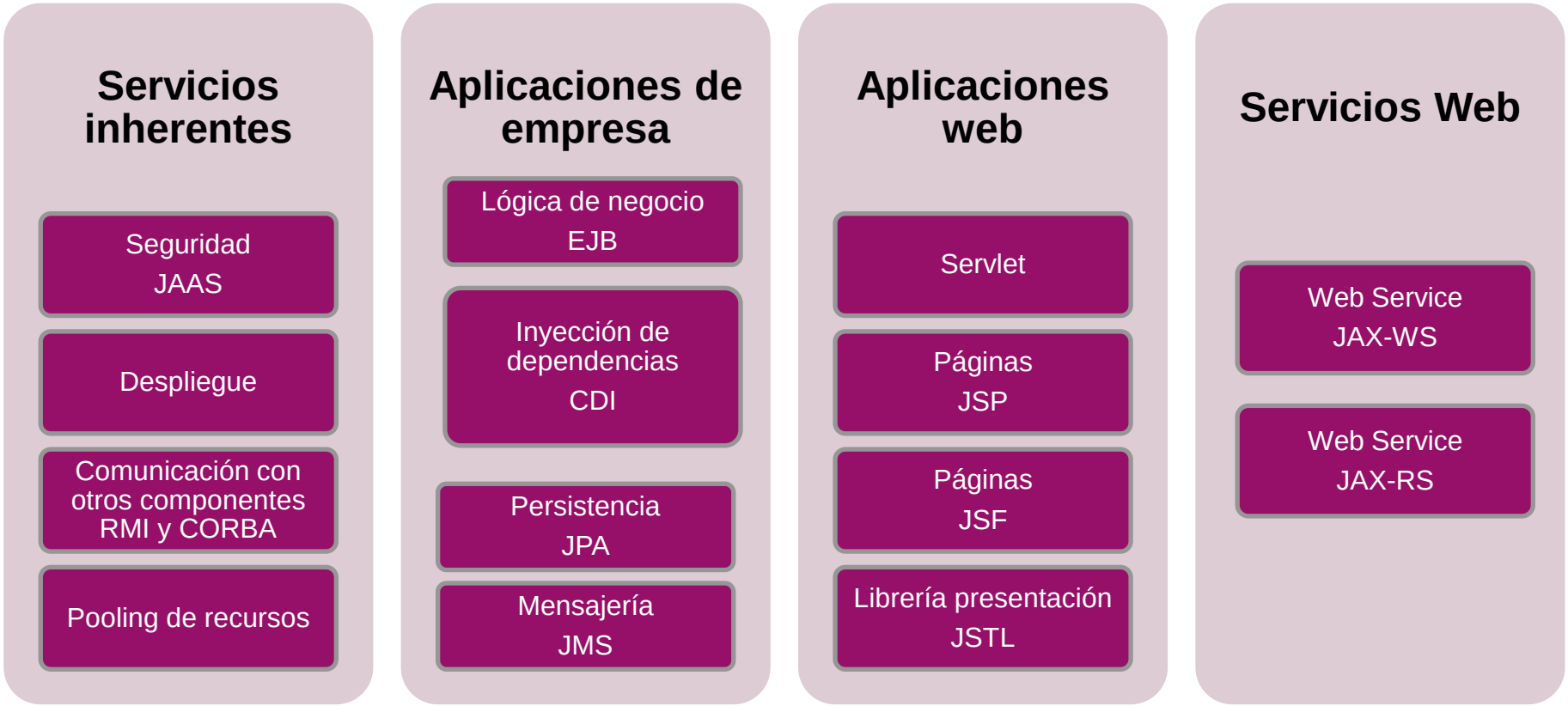
Java EE

Java Platform Enterprise Edition

- Java EE es un conjunto de estándares y APIs para la construcción de aplicaciones empresariales.
- **No** es una implementación. Sólo la especificación.
- Se llama Java EE desde la versión 5. Antes se llamaba J2EE.
- Las APIs más utilizadas de Java EE son:
    - JSP, Servlet, JSF: componentes de presentación web.
    - EJB: para lógica de negocio.
    - JMS: Mensajería.
    - JAX-RS, JAX-WS, JAX-RPC: Web Services.
    - JPA: Persistencia. Complementa a EJB.
    - JTA: Transacciones.
    - JNDI: Localización de recursos.
    - JAAS: Java Authentication and Authorization Service.

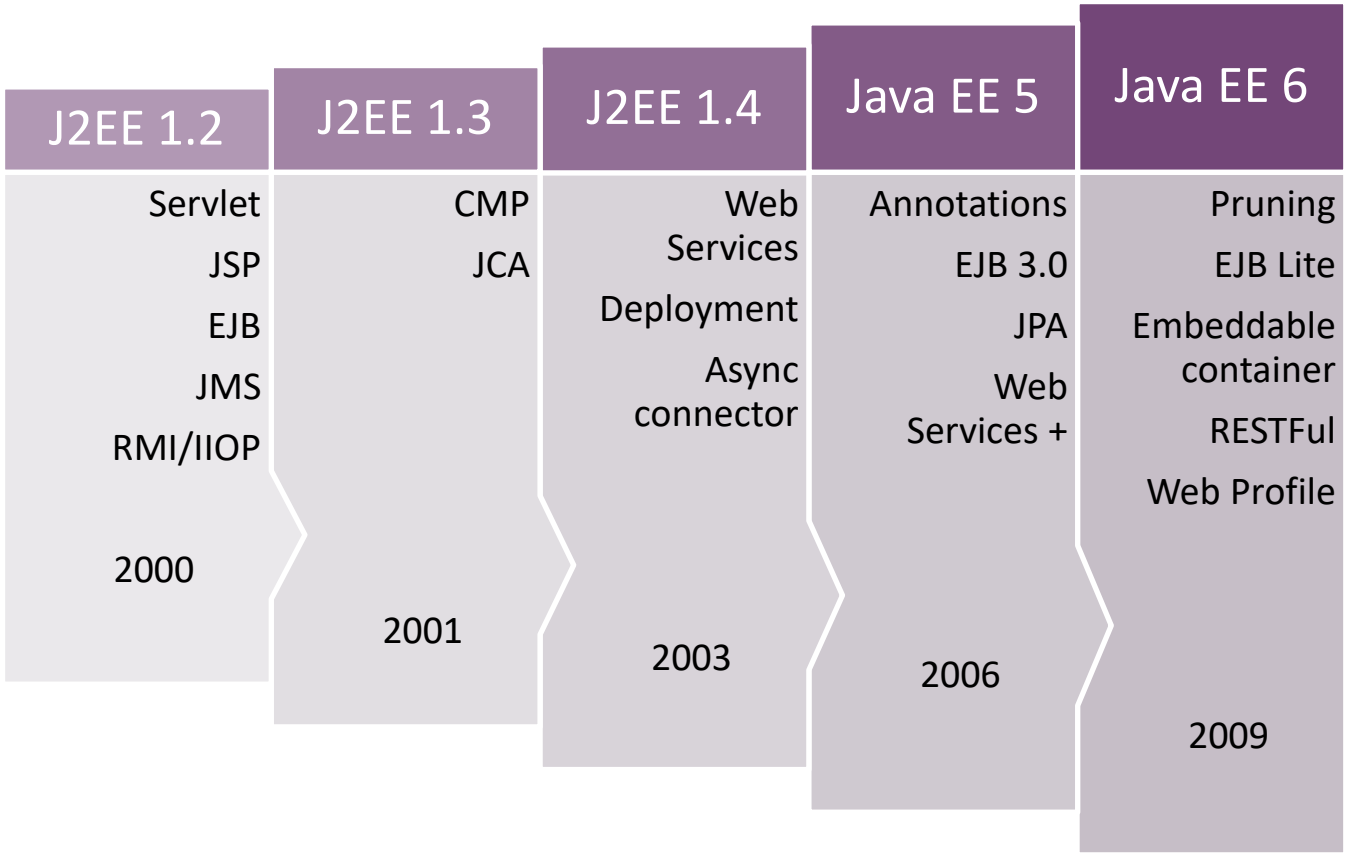
Java EE

Agrupando los servicios **Java EE**:



Java EE

Evolución desde **J2EE** a **Java EE**:



Servidores de Aplicaciones

Un **servidor de aplicaciones**, en el ámbito Java EE, es un producto que implementa las APIs especificadas en Java EE. Por ejemplo:

- Permiten configurar servicios que incluyen algunos de los estándares. Por ejemplo, se puede configurar un DataSource, para acceder a base de datos, y publicarlo en JNDI.
- Permiten desplegar aplicaciones web que tienen componentes que implementan estándares, como JSP, Servlet o JSF.
- Permiten desplegar módulos que contienen componentes que implementan EJB. Los EJB se publican en JNDI.
- Permiten desplegar aplicaciones que publican web services, utilizando el estándar JAX-WS.
- El propio servidor publica las aplicaciones a través de puertos, que pueden ser accedidos por URL desde un browser (caso web), o desde aplicaciones cliente (caso EJB o Web Service).

Servidor de aplicaciones

Los servidores de aplicaciones Java EE más conocidos son:

- Corporativos de pago:
    - WebLogic, de Oracle (heredado de BEA).
    - WebSphere, de IBM.
    - JBoss Enterprise Platform, de Red Hat (heredado de JBoss company).
- Corporativos libres:
    - GlassFish, de Oracle (hededado de Sun).
    - JBoss Application Server.
    - WebSphere Community Edition
- Libres:
    - TomEE, de Apache. Versión Java EE de Tomcat.
    - JOnAS, de Object Web.
    - Geronimo, de Apache.

Contenedor Web

Existen servidores que sólo implementan la parte web de Java EE (JSP, Servlet, JSF).

También son conocidos como servlet container o web container, y que son más livianos de ejecutar, lo que permite utilizarlos para desarrollo de aplicaciones web.

Los más conocidos son:

- Tomcat, de Apache.
- Jetty, de Eclipse Foundation.

Normalmente se utiliza como servidor Tomcat, embebido en Eclipse, y controlado por un plugin del propio Eclipse, excepto en el de EJB.

	Tomcat	TomEE	TomEE JAXRS	TomEE+	TomEE PluME	OpenEJB
Java Servlets	✔	✔	✔	✔	✔	
Java ServerPages (JSP)	✔	✔	✔	✔	✔	
Java ServerFaces (JSF)		✔	✔	✔	✔	
Java Transaction API (JTA)		✔	✔	✔	✔	✔
Java Persistence API (JPA)		✔	✔	✔	✔	✔
Java Contexts and Dependency Injection (CDI)		✔	✔	✔	✔	✔
Java Authentication and Authorization Service (JAAS)		✔	✔	✔	✔	✔
Java Authorization Contract for Containers (JACC)		✔	✔	✔	✔	✔
JavaMail API		✔	✔	✔	✔	✔
Bean Validation		✔	✔	✔	✔	✔
Enterprise JavaBeans		✔	✔	✔	✔	✔
Java API for RESTful Web Services (JAX-RS)			✔	✔	✔	✔
Java API for XML Web Services (JAX-WS)				✔	✔	✔
Java EE Connector Architecture				✔	✔	✔
Java Messaging Service (JMS)				✔	✔	✔
EclipseLink					✔	
Mojarra					✔	

Framework

Otro concepto importante en el ámbito de aplicaciones es el de **framework**.

- Un framework es un conjunto de componentes de uso general y reutilizable, para el desarrollo de aplicaciones, productos o soluciones.
- Dependiendo de su propósito, puede incluir:
    - Componentes de ejecución
    - Librerías
    - Herramientas
    - APIs para su utilización
- Normalmente, debería ser posible extenderlo, pero no modificarlo.
- Ejemplos de frameworks:
    - Spring
    - Hibernate
    - Hibernate es un framework ORM que implementa la especificación JPA*.

Aplicación Empresarial

Una **aplicación empresarial** es un conjunto de componentes que cumplen los estándares Java EE, y que se ejecuta desplegándola en un servidor de aplicaciones.

Los tipos de aplicaciones, de acuerdo a su estructura, son:

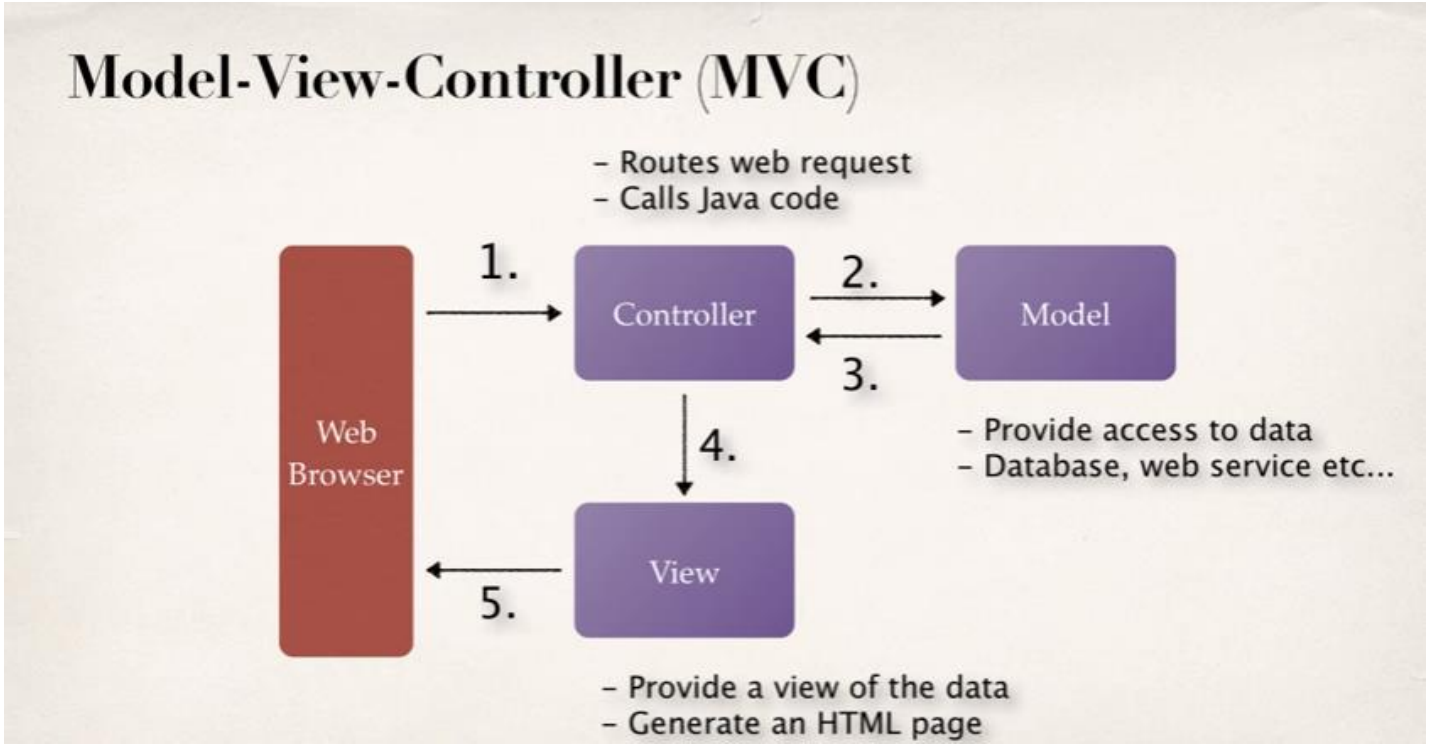
- Web: contiene páginas, elementos de presentación, y en general componentes que pueden ser accedidos por URL. Se almacenan en un archivo WAR (Web Archive).
- EJB: contiene servicios publicados como EJB. Se almacenan en un archivo JAR, con formato de EJB.
- Enterprise: agrupa una o más aplicaciones, Web o EJB, en una sola. Se almacenan en un archivo EAR (Enterprise Archive).

Existe además otro tipo de aplicaciones, llamados Resource Adapter, que conectan servidores utilizando JCA y se almacenan en archivos RAR (Resource Archive).

Conceptos fundamentales

- API (Application Programming Interface)
- ORM (Object-Relational Mapping)
- XML (eXtensible Markup Language)
- JSON (JavaScript Object Notation)
- EJB (Enterprise Java Bean)
- POJO (Plain Old Java Object)
- Java Bean

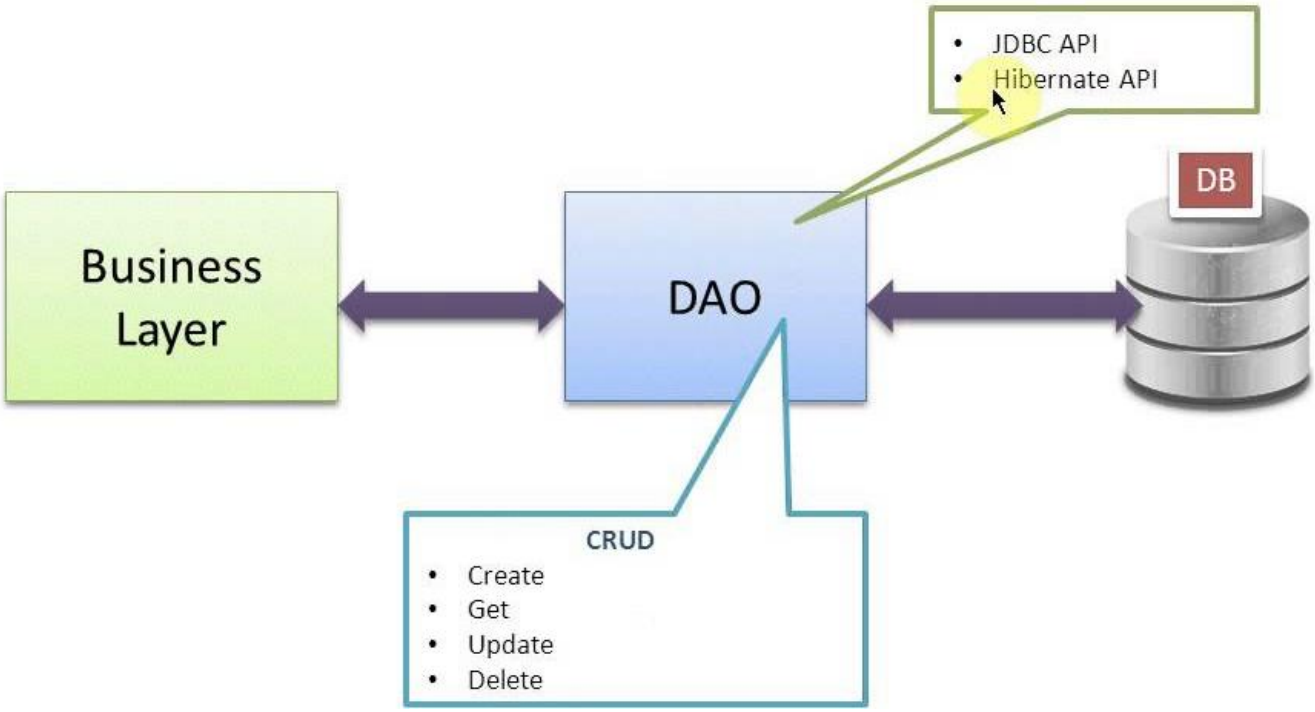
Patrón de diseño MVC



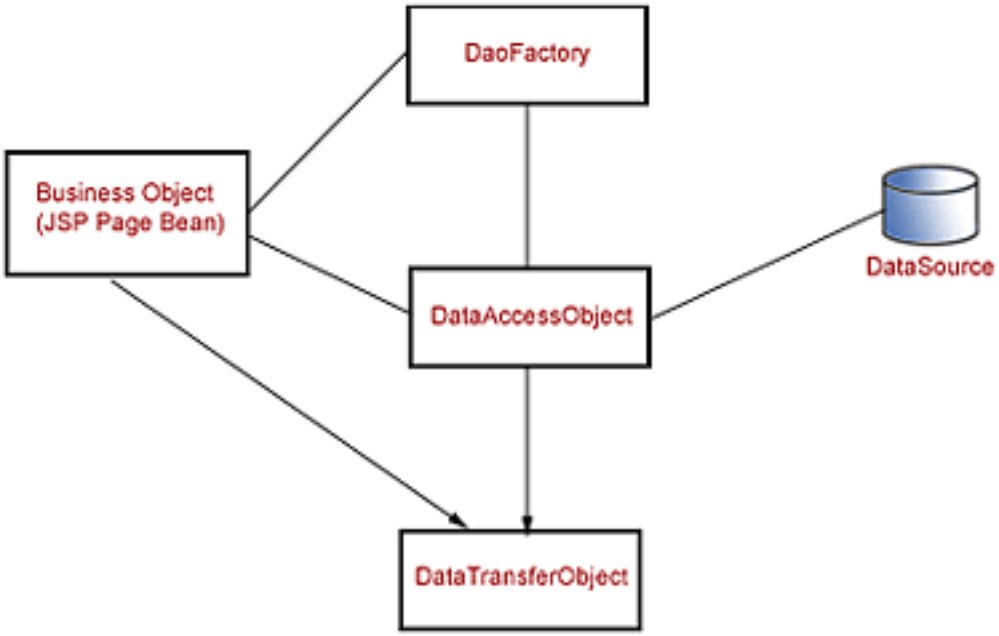
Patrón de diseño DAO

Data Access Object pattern

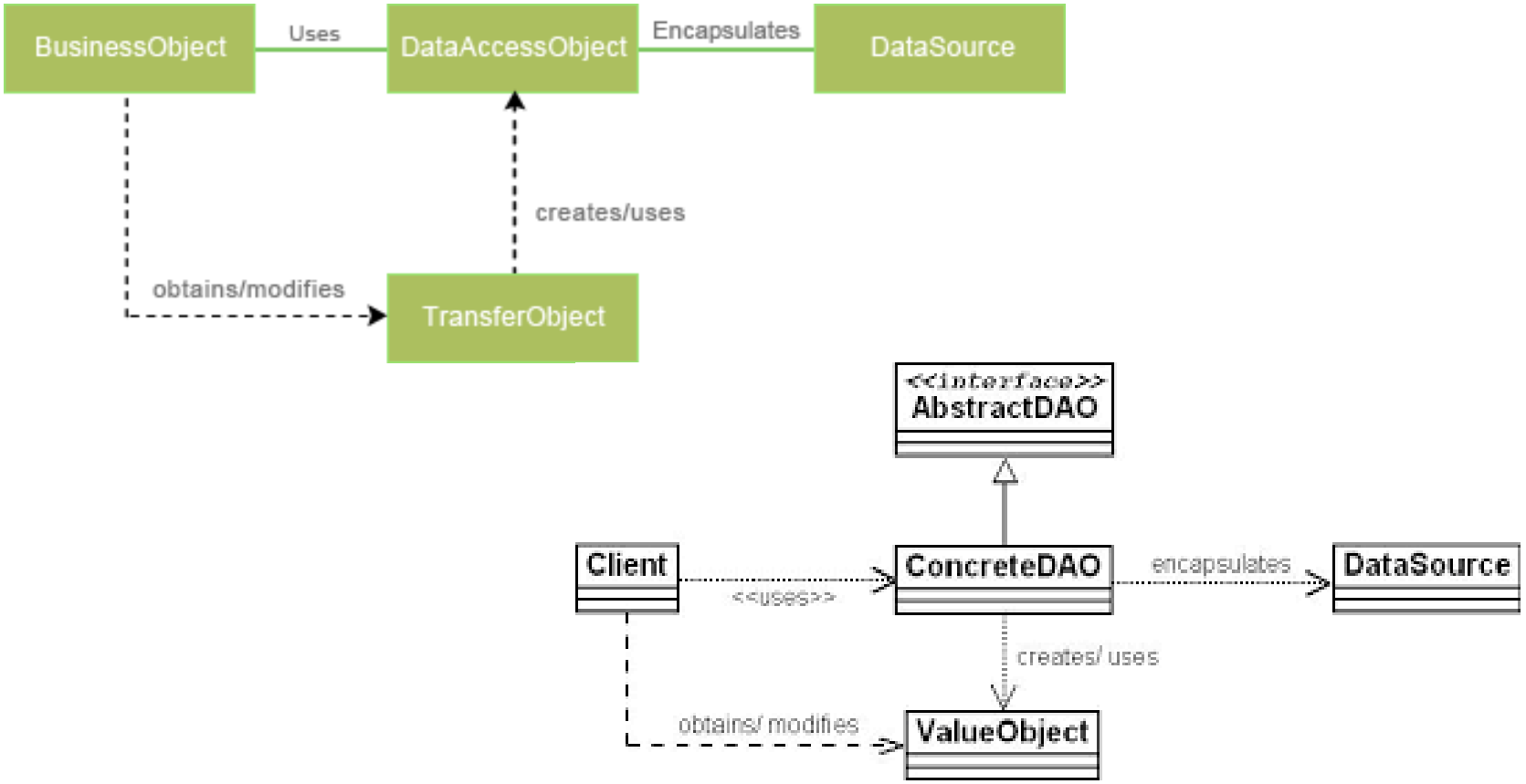
- ✓ Data Access Object Pattern or DAO pattern is used to separate low level data accessing API or operations from high level business services.
- ✓ DAO layer is responsible for Data access from the persistence storage[DB/LDAP/File system] and manipulation of Data in in the persistence storage
- ✓ Decouple the persistent storage implementation from the rest of your application.



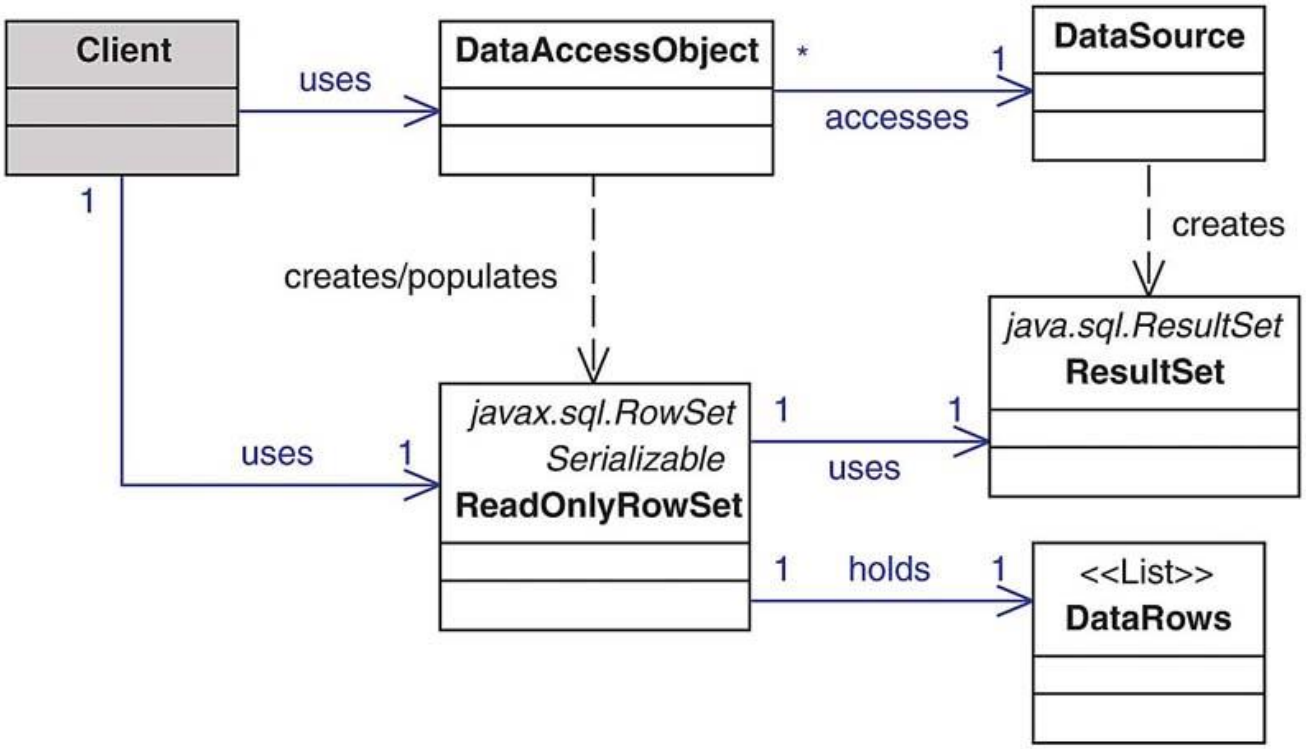
DAO Factory


DaoFactory
Business Object
(JSP Page Bean)
DataAccessObject
DataSource
DataTransferObject

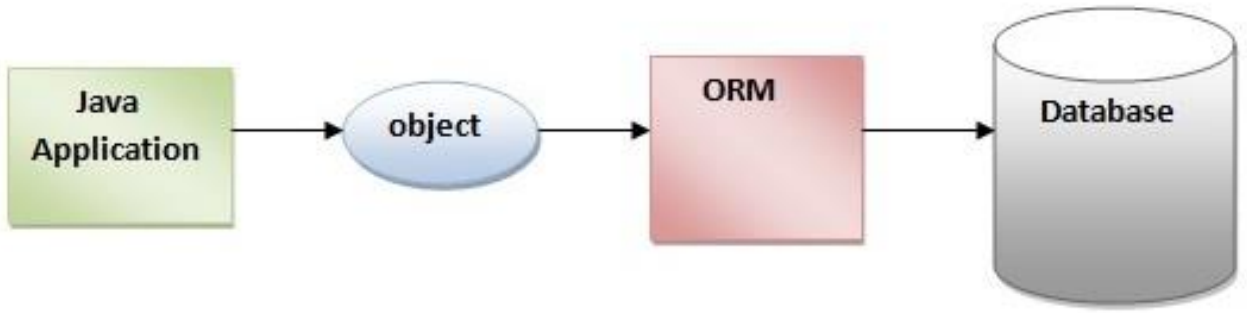
DAO, Data Source, DTO



DAO en Java


Client
uses
DataAccessObject
*
1
DataSource
accesses
1
creates
creates/populates
java.sql.ResultSet
ResultSet
javax.sql.RowSet
Serializable
ReadOnlyRowSet
uses
1
1
1
uses
1
holds
1
<<List>>
DataRows

ORM (Object-Relational Mapping)



CAS (Central Authentication Service)

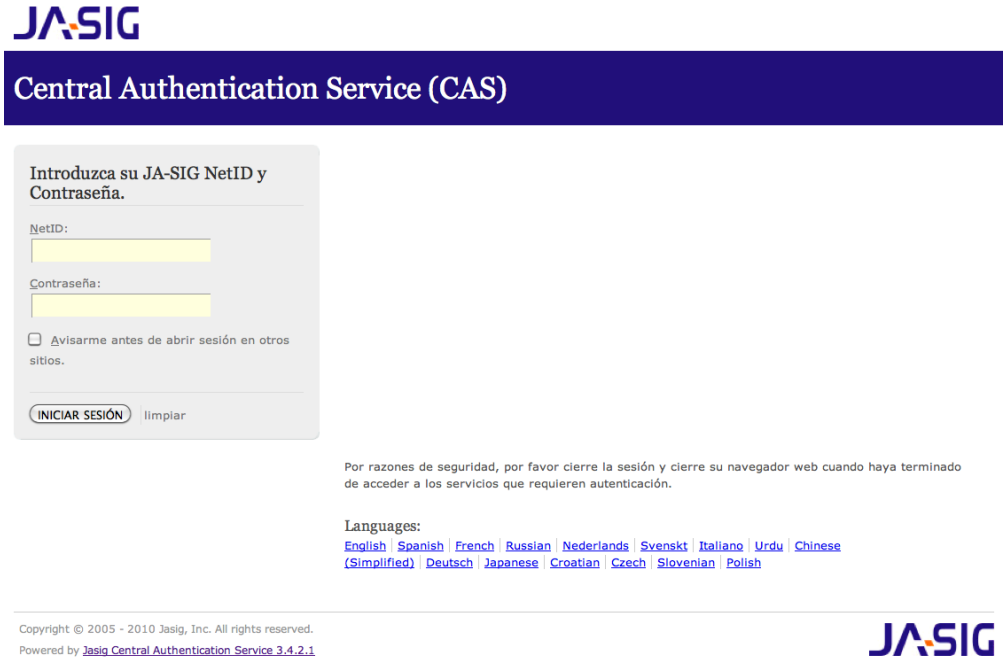
Procedimiento de Autenticación

- Permite implementar procedimiento de Autenticación **SSO (Single Sign On)**
- Aplicación web con **login**
- En distintos **dominos** y distintos **servidores**
- Simplemente comprobación si usuario ya registrado: `request.getRemoteUser()`
- NO se encarga de la Autorización
- CAS se integra fácilmente con Spring Security (p.e. para añadir Autorización)

CAS (Central Authentication Service)

Requisitos entorno:

- CAS Server: https://www.apereo.org/projects/cas
- Maven instalado en el Server
- Descomprimir carpeta CAS_HOME
- Ejecutamos “mvn clean install” en CAS_HOME/cas-server-webapp…
- …CAS_HOME/cas-server-webapp/target/cas.war
- Desplegar cas.war en Apache **Tomcat:**



CAS (Central Authentication Service)

Ventajas

- Evita creación formulario login en todas las aplicaciones web
- Delegar la autenticación en CAS
- Involucrada capa de Presentación: View y Controller
- Soporta varias fuente de datos: base de datos, LDAP, ActiveDirectory
- Permite integración con Spring Security para Autorización

CAS (Central Authentication Service)

provides enterprise single sign-on service for the Web:

- An open and well-documented protocol
- An open-source Java server component
- Pluggable authentication support (LDAP, database, X.509, 2-factor)
- Support for multiple protocols (CAS, SAML, OAuth, OpenID)
- A library of clients for Java, .Net, PHP, Perl, Apache, uPortal, and others
- Integrates with uPortal, BlueSocket, TikiWiki, Mule, Liferay, Moodle and others
- Community documentation and implementation support
- An extensive community of adopters

Spring Cloud module

Spring Cloud module