A1 (***)

65. Arquitectura de desarrollo en la web. Desarrollo web front-end. Scripts de cliente. Frameworks. UX. Desarrollo web en servidor, conexión a bases de datos e interconexión con sistemas y servicios.

66. Entorno de desarrollo Microsoft.NET.

67. Entorno de desarrollo JAVA.

68. Entorno de desarrollo PHP.

72. Sistemas CRM (Customer Relationship Management) y ERP (Enterprise Resource Planning). Generación de informes a la dirección.

77. Lenguajes y herramientas para la utilización de redes globales. HTML, CSS y XML. Navegadores web y compatibilidad con estándares.

85. Características del blockchain, glosario de Szavo, tipos de redes y algoritmos de consenso, Smart contract. El consorcio europeo EBP y la construcción de la infraestructura EBSI.

107. Las tecnologías emergentes. Concepto. Clasificación, aspectos jurídicos y aplicaciones.

A2 (***)

T3. Características técnicas de los lenguajes y paradigmas actuales de programación.

T9. Lenguajes de marca o etiqueta. Características y funcionalidades. SGML, HTML, XML y sus derivaciones. Lenguajes de script.

D10. La arquitectura Java EE/Jakarta EE. Características de funcionamiento. Elementos constitutivos. Productos y herramientas. Persistencia. Seguridad.

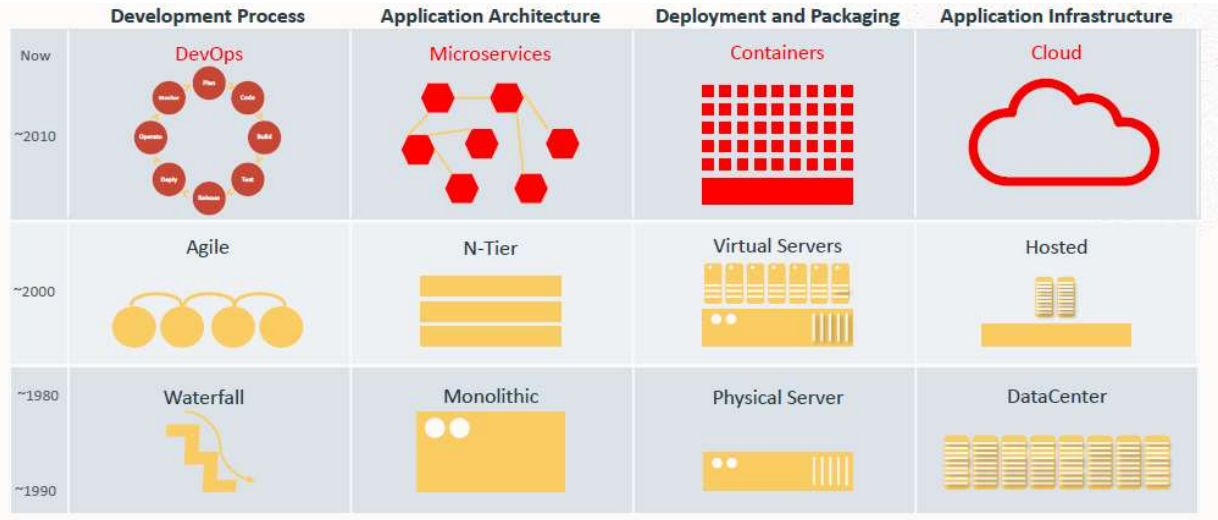
D11. La plataforma.NET. Modelo de programación. Servicios. Herramientas. Persistencia. Seguridad. D

D12.Aplicaciones web. Diseño web multiplataforma/multidispositivo. Desarrollo web front-end y en servidor. Componentes de tecnologías de programación. Servicios web: estándares, protocolos asociados, interoperabilidad y seguridad. Internacionalización y localización.aplicaciones.

Contenido

1. PLATAFORMA JAVA

1.1. INTRODUCCIÓN



- Lenguaje de alto nivel que, al compilarse, genera código binario intermedio **Bytecode**.

- Posteriormente se interpretara en cada CPU en lo que se denominó una máquina virtual Java o **JVM (Java Virtual Machine)**.

- Cuando se habla de Java se habla de "**write once, run everywhere**", aunque esto **sólo es cierto si se cumplen restricciones** en cuanto a la compatibilidad entre el entorno de desarrollo utilizado y la máquina virtual instalada en las plataformas en las que se requiere ejecutar el código objeto generado.

- El código compilado en una máquina determinada **puede ser ejecutado sin ser recompilado** en cualquier otra máquina para la que **exista una JVM**, sea cual sea la arquitectura y tecnología de esa máquina.

- Hay tres **características** que definen a la tecnología Java:
  - **MULTIPLATAFORMA:** la independencia de la plataforma no es completamente cierta ya que es necesaria la compatibilidad entre el entorno de desarrollo JDK y el entorno de ejecución instalado, JRE.
  - **ORIENTADO A OBJETOS PURO:** todo método debe pertenecer a una clase.
  - **FUERTEMENTE TIPADO:** es un lenguaje seguro de forma intrínseca.

- Cada entorno de desarrollo Java (JDK) ha tenido un entorno de ejecución Java (JRE) asociado.

- El JRE es distinto para cada plataforma (Intel-Linux, Intel-Windows ... etc). JRE es gratuito pero no es libre y, en su interior, incluye la JVM (incompatible con la MVM).

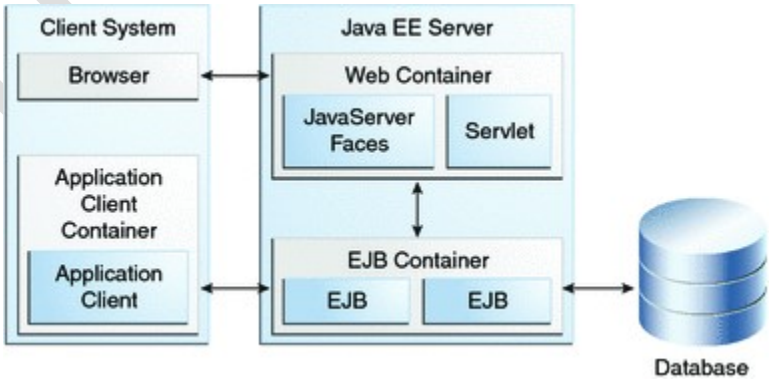
- El **Java Community Process (JCP)** utiliza documentos formales que describen las especificaciones y tecnologías propuestas para que sean añadidas a la plataforma Java. Estos documentos reciben el nombre de **JAVA SPECIFICATION REQUEST (JSR).**

- Java abarca varias plataformas de desarrollo:
  - **Java SE**: para el desarrollo de aplicaciones en clientes y servidores (Versión 21)
  - **Java EE**: para software empresarial.(Jakarta EE 10)
  - Java Card

- **GRAALVM**: GraalVM es una Maquina Virtual que es capaz de ejecutar código de diversos lenguajes de programación. Esto en principio nos puede sonar un poco raro porque ya existen maquinas virtuales que ejecutan código . Sin ir más lejos la propia maquina virtual de Java JVM es capaz de ejecutar código Java y otros lenguajes basados en su propia plataforma.

1.2. ARQUITECTURA JEE

- J2EE 1.2 (December 12, 1999)
- J2EE 1.3 (September 24, 2001)
- J2EE 1.4 (November 11, 2003)
- Java EE 5 (May 11, 2006)
- Java EE 6 (December 10, 2009)
- Java EE 7 (May 28, 2013)
- Java EE 8 (2017) -> JAKARTA EE 9.1 -> **2023: Jakarta EE 10**



- Las especificaciones JEE definen un **contenedor** como el software responsable de ofrecer una serie de **servicios de ejecución a los componentes JEE de aplicación**, a través de las API que define JEE.

1.3. CONTENEDOR WEB

- Encargado de ejecutar los componentes web (**Servlets y JSP**).
- Un **contenedor web** es la implementación que hace cumplimiento del contrato de componentes web de la arquitectura JEE.
- Este contrato especifica un entorno de ejecución para componentes web que incluye seguridad, concurrencia, gestión del ciclo de vida, procesamiento de transacciones, despliegue y otros servicios.
- Un contenedor web se suministra incluido en un servidor web.
- Un servidor web está optimizado para servir páginas HTML estáticas mientras que un **contenedor web** actúa como una **capa intermedia de software entre el servidor web y el servlet.**

1.3.1. SERVLETS.

- Una clase en el lenguaje de programación Java, utilizada para ampliar las capacidades de un servidor, que extienden la interfaz **HttpServlet**, invocadas a través de una **URL**.
- Utilizadas para **generar contenido dinámico**, normalmente texto con formato HTML o XML. Este contenido es devuelto al cliente mediante HTTP.
- Los paquetes **javax.Servlet y javax.Servlet.http** proporcionan todo lo necesario para escribir Servlets.
- Cuando se implementa un servicio genérico, se puede extender la clase GenericServlet de la API Servlet pero para servicios específicos, debe extenderse la interfaz HttpServlet.
- Cada servlet debe sobrecargar al menos un método, que normalmente es uno de los siguientes:
  - **doGet**(). Se ejecuta cuando se recibe una petición HTTP GET. Se envía dentro del URL.
  - **doPost**(). Se ejecuta cuando se recibe una petición HTTP POST. Se envía dentro del documento HTML.
  - **doPut**(). Se ejecuta cuando se recibe una petición HTTP PUT.
  - **doDelete**(). Se ejecuta cuando se recibe una petición HTTP DELETE.
  - **init**() y **destroy**(). Crean y destruyen el servlet.
  - **getServletInfo**(). Lo usa el servlet para proporcionar información sobre sí mismo.
  - **service**(). Maneja la petición HTTP para redirigir al método doXXX que corresponda.
- Los métodos doXXX reciben como parámetros objetos de la clase **HttpServletRequest** con la información de la solicitud HTTP y **HttpServletResponse**, donde se incluye la información de respuesta a la petición