# Ingeniería de Software: Resumen

by Juan Suazo Verger

## Clase 1: Introducción a la Ingeniería de Software

**Introducción a la Ingeniería de Software (IS)**: La ingeniería de software (IS) es la disciplina que se encarga del desarrollo, operación y mantenimiento del software mediante un enfoque sistemático, disciplinado y cuantificable. Este campo surge como respuesta a la creciente complejidad de los sistemas de software y su importancia en la sociedad moderna.

**Importancia de la IS**: La IS es crucial porque permite crear software de alta calidad, fiable y seguro, que cumple con los requisitos de los usuarios y se adapta a sus necesidades cambiantes. Sin un enfoque estructurado, el software puede presentar fallas, ser costoso de mantener o resultar difícil de evolucionar.

**Evolución de la IS**: Desde sus inicios, la IS ha pasado de ser un proceso rudimentario a una disciplina madura que utiliza metodologías ágiles, integración continua y prácticas de DevOps, garantizando ciclos de desarrollo más rápidos y eficientes.

**Ejemplo práctico**: Imagina que una empresa debe desarrollar una aplicación de banca en línea. Sin aplicar principios de ingeniería de software, la aplicación podría ser vulnerable a ataques, tener tiempos de inactividad prolongados y resultar insatisfactoria para los clientes.

**Estado actual y tendencias**: Hoy en día, la ingeniería de software ha incorporado tecnologías emergentes como la inteligencia artificial (IA), el desarrollo ágil, la automatización y el uso de la nube, lo que facilita la implementación de software más rápidamente y con mayor calidad.

**Proceso del software**: El ciclo de vida del desarrollo de software incluye varias fases clave: análisis de requerimientos, diseño, implementación, pruebas, despliegue y mantenimiento. Cada una de estas fases es crítica para asegurar que el producto final funcione según lo esperado y sea escalable en el futuro.

**Definición IEEE**: Según la IEEE, la IS es "la aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento del software". Esta definición subraya la necesidad de un enfoque formal en todas las etapas del ciclo de vida del software.

**Bibliografía**: Pressman, R. (Cap. 1) *Ingeniería del Software, Un enfoque práctico.*

## Clase 2: Dominios de Aplicación y Procesos de Ingeniería

**Dominios de Aplicación del Software**:

El software puede clasificarse en diferentes dominios de aplicación, cada uno con características y requisitos específicos. A continuación, se detallan algunos de los más relevantes:

1. **Software de sistemas**: Comprende sistemas operativos, controladores de dispositivos y otros programas que gestionan directamente el hardware de la computadora. Ejemplo: Windows, Linux.
2. **Software de aplicación**: Incluye programas que ayudan al usuario a realizar tareas específicas, como procesadores de texto, hojas de cálculo o navegadores web. Ejemplo: Microsoft Word, Google Chrome.
3. **Software de ingeniería y ciencias**: Se utiliza para realizar cálculos avanzados y simulaciones en campos como la ingeniería y las ciencias. Ejemplo: MATLAB, AutoCAD.
4. **Software incrustado**: Software embebido en dispositivos electrónicos para controlar su funcionamiento. Se encuentra en electrodomésticos, automóviles, dispositivos médicos, entre otros. Ejemplo: Software en un sistema ABS de un automóvil.
5. **Software de línea de productos**: Plataformas de software reutilizables que permiten la creación de diferentes productos en una misma familia, optimizando el desarrollo. Ejemplo: Variantes de teléfonos móviles de una misma marca.
6. **Aplicaciones web (Webapps)**: Software accesible a través de un navegador, como plataformas de comercio electrónico, aplicaciones de redes sociales o herramientas colaborativas. Ejemplo: Google Drive, Amazon.
7. **Software de inteligencia artificial (IA)**: Sistemas que pueden aprender, adaptarse y tomar decisiones basadas en datos. Se utiliza en áreas como el reconocimiento de voz, visión por computadora, y sistemas de recomendación. Ejemplo: Alexa, sistemas de recomendación en Netflix.

**Ejemplo práctico**: En el sector automotriz, el software incrustado permite que los vehículos modernos funcionen de manera más eficiente y segura. Desde el sistema de frenado ABS hasta la conectividad de entretenimiento, todo está controlado por software.

**Procesos de ingeniería**: Para desarrollar software, se sigue un conjunto de procesos fundamentales:

- **Comunicación**: Entender las necesidades del cliente o usuario.
- **Planeación**: Establecer un plan para el desarrollo, tiempos y recursos.
- **Modelado**: Definir la estructura y el diseño del software.
- **Construcción**: Implementar el software.
- **Despliegue**: Entregar el software al usuario final y asegurar su correcta instalación y funcionamiento.

**Actividades sombrilla**: Estas actividades se realizan a lo largo de todo el proceso de desarrollo:

- **Control de proyecto**: Monitorear el progreso del proyecto para asegurar que se cumplan los objetivos.
- **Gestión de riesgos**: Identificar posibles problemas y desarrollar estrategias para mitigarlos.
- **Control de calidad**: Asegurar que el producto final cumple con los estándares establecidos.
- **Revisiones técnicas**: Evaluar el software en diferentes fases para identificar posibles fallos.
- **Administración de la configuración**: Gestionar los cambios en el software de manera controlada.

**Bibliografía**: Pressman, R. (Cap. 1).

## Clase 3: Scripting y Automatización

**Instalación de paquetes de software**: Existen diversos métodos para instalar software en sistemas complejos:

- **Instalación directa**: Se instala el software de una sola vez.
- **Instalación en paralelo**: Se instala el nuevo software junto con la versión anterior para comparar resultados.
- **Instalación piloto**: Se implementa en una pequeña parte del sistema antes de hacer el despliegue completo.
- **Instalación en fases**: Se instala gradualmente, probando el software en diferentes etapas del proceso.

**Scripting**:

- Los **scripts** son programas pequeños diseñados para automatizar tareas repetitivas, mientras que los **programas** son más complejos y estructurados. Los scripts pueden ejecutarse sin necesidad de compilación y son usados comúnmente para automatizar procesos rutinarios.
- **Tipos de scripts**: Línea de comandos, scripts para aplicaciones web y scripts de automatización que ayudan a ejecutar tareas administrativas en servidores o entornos de trabajo.

**Ejemplo práctico**: Los administradores de sistemas usan scripts de línea de comandos para realizar copias de seguridad automáticas o reiniciar servicios cuando un servidor falla.

**Balanceo de carga**: Es una técnica que distribuye las solicitudes de usuarios entre varios servidores para garantizar que ninguno esté sobrecargado. Algunos métodos incluyen:

- **Round Robin**: Distribuye las solicitudes de manera equitativa entre todos los servidores.
- **Least connections**: Envía las nuevas solicitudes al servidor con menos conexiones activas.
- **IP hashing**: Asigna solicitudes basadas en la IP del cliente.

**Reliability (Confiabilidad)**: La confiabilidad de un sistema se mide a través de dos conceptos importantes:

- **MTBF (Mean Time Between Failures)**: El tiempo promedio entre fallos.
- **MTTR (Mean Time to Repair)**: El tiempo promedio que toma reparar un fallo.

## Clase 4: Pruebas y Operaciones

**Migración de Datos**:

- **Planificación**: La migración debe planearse cuidadosamente, mapeando los datos desde el sistema origen al destino, asegurando la consistencia de las relaciones y estructuras.
- **Ejecución**: Una vez completada la migración, se realizan pruebas exhaustivas para verificar la integridad y calidad de los datos.

**Ejemplo práctico**: En una migración de datos de una base de datos local a la nube, la planificación minuciosa es esencial para evitar la pérdida de información crítica.

**Gestión de Cambios y Problemas**:

- **Gestión de cambios**: Documentar y autorizar cualquier modificación en el software antes de implementarla.
- **Gestión de problemas**: Registrar, analizar y resolver incidentes que afecten la operación del sistema.

**Control de Operaciones**: Monitorear los sistemas en tiempo real para asegurar que operen correctamente, identificar problemas rápidamente y responder de manera oportuna.

**Bibliografía**: Pressman, R. (Cap. 11).

## Clase 5: Proceso de Tercerización

**Tercerización (BPO)**: La tercerización o Business Process Outsourcing (BPO) consiste en delegar procesos o servicios a un proveedor externo especializado. Esto permite a las empresas enfocarse en sus actividades principales mientras el proveedor externo se encarga de tareas no esenciales como soporte técnico, procesamiento de nóminas o atención al cliente.

**Service Level Agreement (SLA)**: Un acuerdo de nivel de servicio (SLA) es un contrato que establece los estándares de calidad, disponibilidad y responsabilidad entre el proveedor y el cliente. Define parámetros clave como el tiempo de respuesta ante problemas y la calidad del servicio ofrecido.

**Ejemplo práctico**: Una empresa que terceriza su soporte técnico firma un SLA con un proveedor externo, el cual especifica que cualquier problema debe ser atendido en menos de 4 horas para asegurar una operación fluida.

**Software Factory**: Este concepto se refiere a una metodología industrializada para la creación de software, donde se emplean técnicas de automatización y mejora de procesos para aumentar la productividad y calidad del producto final. Las fábricas de software permiten gestionar varios proyectos en paralelo y escalar el desarrollo.

**Bibliografía**: Varios recursos sobre BPO y SLAs.