***INTRODUCCIÓN A LA INGENIERÍA DEL SOFTWARE***

***Relevancia del Software en nuestro mundo***

-Las economías de todos los países

-Cada vez más sistemas importantes se controlan con software

-El software es un elemento clave

-Las inversiones en software representan una fracción significativa del PIB

-Los costes del software dominan los costes de los sistemas informáticos

-La ingeniería del software es la disciplina que incluye las técnicas, los métodos y las herramientas para el desarrollo del software

-La ingeniería del software potencia el desarrollo de software rentable

***¿Qué es el Software?***

-**Instrucciones** que cuando se ejecutan, proporcionan la función y comportamiento deseados

-**Estructuras** de datos que facilitan a los programas manipular adecuadamente la información

-**Documentos** que describen la operación y el uso de los programas

-Un producto software se puede desarrollar para un cliente particular o para el mercado general

-Los productos software pueden ser: Genéricos para venderlos a un grupo o a medida para un cliente único

-Desarrollando sistemas, parametrizando o reutilizando software se pueden crear nuevos software

***¿Qué es la Ingeniería del Software?***

-Es un **enfoque sistemático, disciplinado y cuantificable** para el desarrollo, operación y mantenimiento de software (IEEE 90), una actividad bien definida que contribuye a la satisfacción de los objetivos del proyecto.

-Los ingenieros necesitan **producir productos correctos y económicos realizando prácticas** específicas en distintos procesos

-El proceso implica traducir las necesidades del usuario en requisitos de software, transformar los requisitos de software en diseño, implementar el diseño en código, probar el código y, a veces, instalar y verificar el software para su uso operativo. Estas actividades pueden superponerse o realizarse iterativamente.

-Una buena práctica es una práctica que ha demostrado que conduce de manera confiable a un resultado deseado y se considera prudente y recomendable en una variedad de contextos. Por todo ello, una buena práctica no es lo mismo que una “ley científica”.

***Prácticas, Procesos y Modelos***

-Las prácticas de desarrollo de software son actividades bien definidas que contribuyen a la obtención de objetivos.

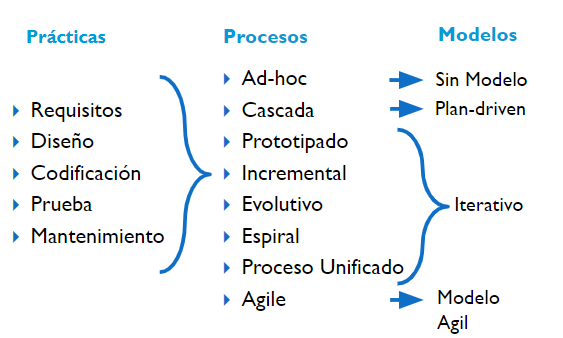
-El proceso de desarrollo de software las necesidades del usuario se traducen en un producto de software

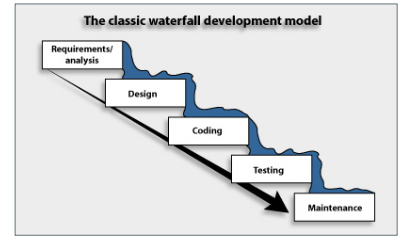
-Un modelo de proceso de software es una descripción abstracta y simplificada de un proceso de desarrollo.

Modelos “Plan-Driven”: Acotados y detallados desde el inicio.

Modelos Iterativos: Menos información al inicio, más flexibles.

Modelos “Ágiles”: Enfocados a proyectos dinámicos y cambiantes.

-Varias metodologías de desarrollo de software pueden compartir un modelo de proceso. Los metodólogos de software incorporan las características generales de los modelos de desarrollo de software en procesos específicos. Las metodologías de desarrollo de software tienen prácticas específicas que deben seguirse.

***El Proceso (Modelo) Cascada***

-Proceso en una serie de etapas

-Inflexible

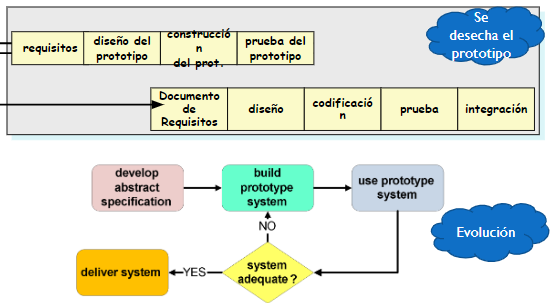
-Separación poco realista de las especificaciones de los requisitos del diseño

-No se adapta al prototipado

-Pasa mucho tiempo de que esté disponible la versión funcional del producto de software

-El modelo de cascada toma una vista estática de los requisitos, ignora las necesidades cambiantes

***Prototipado (De usar y tirar)***



-El prototipado se basa en construir una versión simplificada del sistema propuesto

-El cliente vuelve a refinar los requisitos para incorporar la información adicional.

-Tener en cuenta las prácticas

-Se usa para:

Comprender los requisitos de interfaz de usuario

Examinar la viabilidad de un enfoque de diseño propuesto

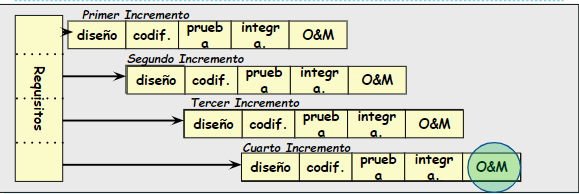
Explorar aspectos del rendimiento del sistema

-Debe ser rápido y barato; programas completamente nuevos desarrollados una vez que se identifican los requisitos.

-Los usuarios suelen ver el prototipo como la solución o sistema final, lo que crea falsas expectativas y hay que dejar claro que es sólo una especificación parcial

-El prototipo Evolutivo = prototipo robusto a medida que evoluciona hasta convertirse en la solución final.

***Proceso Incremental***

******

-Cada incremento/versión añade funcionalidad al sistema, asume todos los requisitos y no los cambia

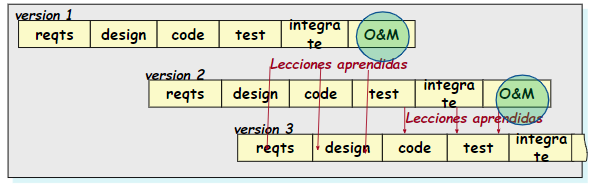
-Los “incrementos” se diseñan, prueban y entregan individualmente en puntos sucesivos en el tiempo el diseño siempre permanece sin cambios a partir de entonces.

-La naturaleza iterativa de este proceso solo está relacionada con la parte de desarrollo y entrega; no a los pasos iniciales de planificación y requisitos.

-Cuándo usar:

Grandes aplicaciones empresariales de misión crítica que consisten preferiblemente en partes poco acopladas

***Proceso evolutivo***



-Las primeras versiones son prototipos que ayudan a obtener las nuevas versiones

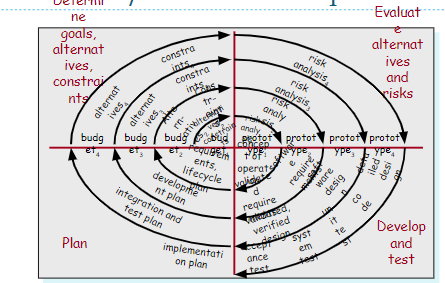
-Necesita la participación del usuario

-Especialmente interesante cuando se quiere empezar a usar el producto

-Puede generar nuevas demandas del cliente durante el desarrollo

-Difícil planificar para las versiones más allá de la primera;   
-Las lecciones se pueden aprender demasiado tarde

***El Proceso / Modelo en Espiral***

******

-Pasos en cascada y creación de prototipos

-Centrarse en la evaluación de riesgos

-Se necesita una participación intensiva del cliente

-No puede hacer frente a cambios imprevistos

-Cuándo usar:

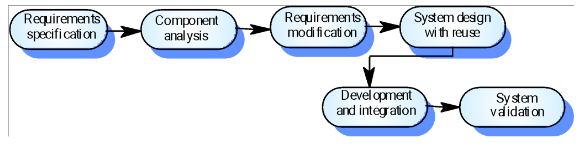
Proyectos con necesidades comerciales poco claras o requisitos demasiado ambiciosos/innovadores.

Proyectos que son grandes y complicados.

Actividad de investigación y desarrollo (I+D) o la introducción de un nuevo servicio o producto.

***Modelo orientado a la reutilización***

-Se construyen integrando componentes ya existentes o sistemas COTS



***Proceso Unificado***

-Modelo de Proceso Híbrido: mejora las características de cascada, iterativo e incremental

-4 Fases en serie (Cascada): etapas por las que pasa un proyecto→ Inicio, elaboración, construcción y transición

-Iterativo en lo pequeño: 9 prácticas→ Actividades Lógicas.

Cuando se finaliza se incrementa un incremento para el usuario como una versión

***Proceso Unificado: Fases***

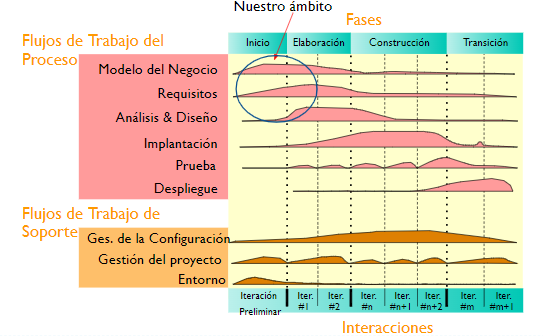
-Inicio: Objetivos y Necesidades desde el punto de vista del Negocio

-Elaboración: Requisitos, arquitectura y plan

-Construcción: Desarrollo y Pruebas

-Transición: Participación del usuario y entrega de producto

***Estructura del Proceso Unificado***

-Proceso Unificado Modelo de proceso moderno con 4 fases que es considerado como un híbrido al reunir elementos de los modelos de proceso precedentes

***Desarrollo Ágil de Software - “Abrazar” el Cambio***

-Iteraciones muy cortas

-Participación activa del Cliente

-Dirigido basándose en las prioridades de los hechos

-Proceso adaptativo

-Equipos potentes de personas centradas

-inconvenientes(no apto, dificultades para redactar contactos, equipos hábiles….)

***Agilidad - Tipos de Proyectos***

-Conejo (proyecto ágil): Interacciones frecuentes, incrementos pequeños, documentación mínima, vida corta.

-Caballo (rápido, fuerte, fiable): común en proyectos corporativos, cierta formalidad y orden, vida media.

-Elefante (sólido, fuerte, larga vida, gran memoria): Acuerdos contractuales, comunicación formal, muchas personas, proyectos distribuidos.

***¿Qué es CASE (Computer - Aided Software Engineering)?***

-Sistemas software creados para proporcionar soporte automatizado

-Las herramientas CASE se suelen utilizar para dar soporte a los métodos utilizados

-Upper-CASE (CASE de alto nivel): herramientas como Visual Paradigm

Desarrollo de modelos de sistemas gráficos en especificación y diseño

Generación de código a partir de los modelos gráficos

Producción de interfaces de manera interactiva

-Lower-CASE (CASE de bajo nivel): Herramientas que dan soporte a las actividades finales como programación, prueba y depuración.

-IDE: Integra diferentes herramientas CASE especialmente enfocadas a actividades de programación y validación

***Responsabilidad Ética y Profesional***

-La práctica profesional de la ingeniería del Software implica responsabilidades más amplias que la simple aplicación de conocimientos técnicos.

-Los ingenieros de software deben comportarse de una manera honesta y éticamente responsable si quieren ser respetados como profesionales

-El comportamiento ético es más que cumplir y hacer cumplir la ley

-Principios de Código Ético

-Dilemas Éticos