Preguntas Ingeniería de software

1) ¿Qué es software?

Muchas personas asocian el término de Software con los programas de PC.

Sin embargo, software no son sólo programas, sino también todos los documentos asociados y la configuración de datos que se necesitan para hacer estos programas operen de manera correcta.

Programas de cómputo y documentación asociada, como ser documentos de requerimientos, arquitectura y modelos de diseño y manuales de usuario.

2) ¿Qué tipo de productos de software pueden ser?

Genéricos: desarrollados para ser vendidos a una gama de diversos clientes. Ejemplo Word

Hechos a medida: desarrollado para un cliente particular acorde a sus requerimientos y especificaciones.

Un nuevo software puede ser creado, desarrollando nuevos programas, configurando sistemas de software genérico o reutilizando software existente.

3) ¿Qué es ingeniería de software?

Es una disciplina de la ingeniería que concierne a todo lo referente con la producción de software

Los ingenieros de software deberían adoptar un enfoque sistemático y organizado con respecto a su trabajo, utilizar herramientas y técnicas apropiadas con relación al problema planteado, las restricciones del desarrollo de los recursos disponibles.

4) ¿Indique las diferencias entre la ingeniería de software y la ciencia de la computación?

La computación comprende teorías y fundamentos de cualquier sistema de cómputo.

Las teorías de la computación aún son insuficientes para respaldar completamente a la ingeniería de Software como ocurre en otras ingenierías como la eléctrica.

La ingeniería de software concierne los aspectos prácticos del desarrollo y la entrega de software útil.

5) ¿Indique las diferencias entre la ingeniería de software y la ingeniería de sistemas?

Ingeniería de sistemas compete todos los aspectos de desarrollo de sistemas basados en cómputos incluyendo el hardware y el software y procesos de ingeniería.

Los ingenieros de sistema están involucrados con la especificación del sistema, diseño arquitectónico, integración y despliegue de sí mismo.

Ingeniería de software es parte de este proceso, haciendo referencia al desarrollo de la infratuctura del software, aplicaciones y bases de datos en el sistema.

6) ¿Qué es proceso de software?

Un conjunto sistemático de actividades cuya meta es el desarrollo o la evolución de software

7) ¿Cuáles son las actividades genéricas que hacen todos los procesos?

Especificación: Es lo que el sistema debería hacer y restricciones del sistema

Desarrollo: Producción del sistema de software

Validación: Comprobando que el software es lo que el cliente quiere

Evolución: Cambios y mantenimiento en el software con relación a los cambios en los requerimientos y demanda

8) ¿Qué es modelo de software?

Una representación simplificada de un proceso de software, presentada desde una perspectiva especifica.

9) ¿De ejemplos de perspectiva de software?

Perspectiva de flujo de trabajo de trabajo- secuencia de actividades

Perspectiva de flujo de datos - flujos de información

Perspectiva de casos de uso - quien hace que funcione

10) ¿Indique cuáles son los modelos genéricos?

Modelo cascada

Desarrollo interactivo

Ing. de sw basada en componentes

11) ¿Describa los costos de la ingeniería de software?

Rigurosamente el 60% de los costos son de desarrollo, 40% de los costos son de pruebas.

Para el software hecho a medida, a menudo los costos de evolución exceden a los costos de desarrollo.

Los costos varían acorde al sistema a desarrollar y a los requerimientos con respecto a los atributos del mismo, como lo pueden ser su funcionamiento y la confiabilidad del sistema

La distribución de los costos depende del sistema empleado para el desarrollo de software

12) ¿Describa la metodología de desarrollo de software?

Existen algunos acercamientos estructurados -> al desarrollo de software que incluyen -> modelos del sistema, notaciones, reglas, pautas de diseños y pasos a seguir

Descripciones de modelo

Modelos gráficos: que deben ser producidos

Reglas

Restricciones: aplicadas a modelos de sistema

Recomendaciones

Pautas: para un buen diseño en la práctica

Pasos a seguir: Que actividades deben seguirse

13) ¿Qué son las herramientas CASE?

Son sistemas de sw cuya finalidad es proveer soporte para para actividades o procesos, también sirve para el desarrollo de software.

Son usados frecuentemente como soporte de la metodología de desarrollo.

Herramientas Case tempranas o superiores

Herramientas encargadas de soportar requerimientos de actividades y diseño.

Herramientas case tardías o inferiores

Herramientas para soportar etapas posteriores en el desarrollo como lo son la programación, depuración y pruebas.

Ejemplos de herramientas case inferiores tenemos los IDE como netbeans, DEVc, visual studio, etc.

14) ¿Cuáles son los atributos de un buen software?

El software debe proveer una funcionalidad y el funcionamiento que el usuario requiriera.

Debe ser: mantenerle, confiable y aceptable

Mantenibilidad: el sw debe evolucionar para cubrir todas aquellas necesidades que varían con el tiempo

Confiabilidad: Debe ser confiable

Eficiencia: El sw no debería hacer un mal uso de los recursos del sistema

Aceptabilidad: El sw debe aceptar las necesidades de los usuarios para lo cual fue diseñado. Ser compatible y que funcione con otros sistemas

15) ¿Cuáles son los principales desafíos de la ingeniería de software?

La heterogeneidad de los ing de sw

Esto se refiere a la percepción que tienen las personas sobre la labor de un ingeniero informático; para ellos uno debe poseer conocimiento de todo lo relacionado a la tecnología

El desafío de uno como futuro profesional es dejar bien claro la especialización de la función que uno realiza como informático en este caso como ingeniero de software

Para ello tenemos técnicas de desarrollo para la construcción de software que puedan encararse con plataformas heterogenias y ambientes cambiantes

Entrega

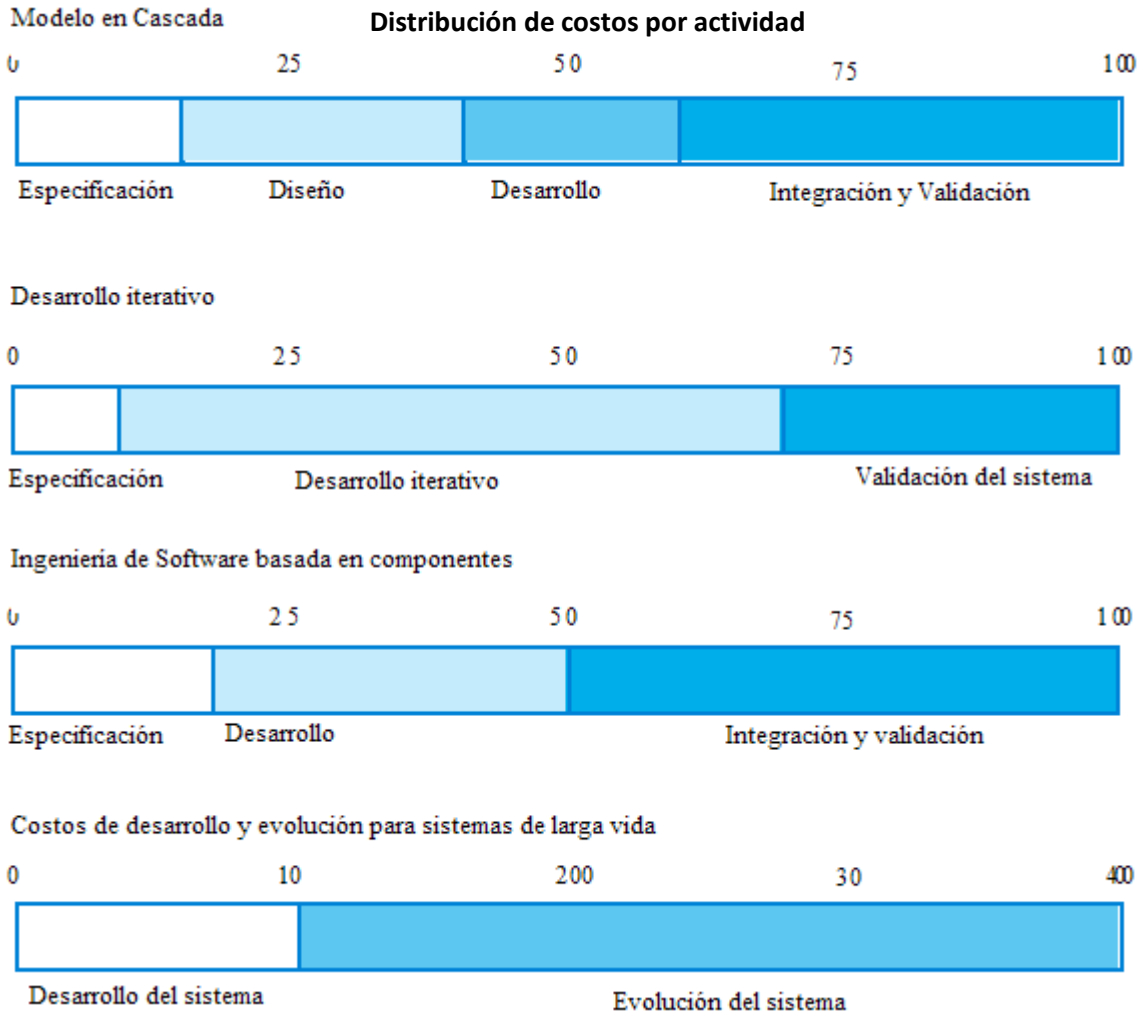
Ya que por la complejidad de los sistemas de la ingeniería de software, la fecha de término nunca es tan exacta.

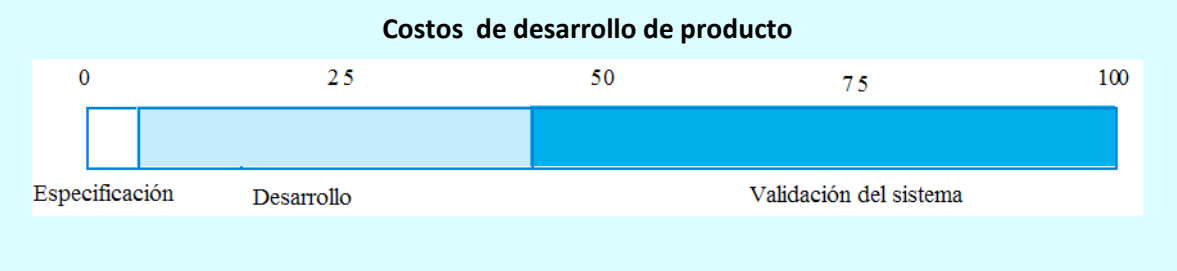
Para ello también existen técnicas que lleven a una entrega de software mucho más rápido

Confianza

Al ser un producto intangible en su etapa de desarrollo, como ingenieros de sw se hace un desafío presentar algún avance al cliente

Para ello también existen técnicas en las cuales demuestre que el software es de confianza para con sus usuarios





¿Cuáles son los aspectos de responsabilidad profesional?

La ingsw implica responsabilidades > que el simple uso de habilidades técnicas

Los ing de sw deben comportarse de manera honesta y ética para así ser respetado como profesionales

Actuar éticamente es mucho + que actuar dentro de los márgenes de la ley.

Confidencialidad

Siempre los ingsw deben respetar este punto, hayan firmado un contrato de con fidelidad o no

Capacidad

Los ingsw no deberían pretender tener > capacidad (intelectual y laboral) de la que tienen, debe aceptar trabajos que estén dentro de su capacidad

Derechos de propiedad intelectual

Los ingsw deben estar conscientes de las leyes que gobiernan el uso de propiedad intelectual, como patentes y derechos reservados

Garantizar que la propiedad intelectual de los clientes este protegida

Mal uso de la computadora

Los ingsw no deberían usar sus técnicas y habilidades para hacer mal uso de las computadoras de otras personas.

Ya sea jugar en la computadora de un cliente hasta por ejemplo diseminar un virus

Código de Ética ACM/IEEE

Las sociedades profesionales en los Estados Unidos cooperaron para producir un código de práctica ética.

Miembros de estas organizaciones fueron consecuentes con el código de práctica ética cuando se asociaron.

El código contiene ocho principios relacionados al comportamiento y las decisiones hechas por ingenieros de software profesionales, incluyendo a los que lo ejercían, educadores, encargados, supervisores y editores, así como aprendices y estudiantes de la profesión.

Código de Ética - Principios

PUBLICO

Los ingenieros de software actuarán constantemente con el interés público.

CLIENTE Y EMPLEADOR

Los ingenieros de software actuarán acorde al mayor interés de sus clientes y empleadores en constancia con el interés público.

PRODUCTO

Los ingenieros de software garantizarán que sus productos y relativas modificaciones van acorde a los estándares profesionales más altos posibles

COLEGAS

Los ingenieros de software serán condescendientes y brindarán apoyo a sus colegas**.**

UNO MISMO

Los ingenieros de software participarán en la formación continua con respecto a la práctica de su profesión y promoverán un acercamiento ético a la práctica de la profesión.

Dilemas Éticos

Desacuerdo en principio con las políticas de administración superior.

El empleador actúa de un modo no ético y lanza un sistema de seguridad crítico sin terminar la prueba del sistema.

Participación en el desarrollo de armamento militar o sistemas nucleares.

Resumen

La Ingeniería de Software es una disciplina de la ingeniería concerniente a todos los aspectos de la producción de software.

Los productos software están consistidos de programas desarrollados y su documentación asociada. Atributos esenciales del producto son la mantenibilidad, formalidad, eficiencia y utilidad.

El proceso de software consiste en actividades involucradas en el desarrollo de productos software. Actividades básicas son la especificación del software, desarrollo, validación y evolución.

Las metodologías son maneras organizadas de producir software. Estas incluyen sugerencias para el proceso a seguir, notaciones a usar, reglas que rigen las descripciones del sistema que son pautas para el desarrollo.

Las herramientas CASE son sistemas de software diseñadas para soportar actividades rutinarias en el proceso de software, como ser: editar diagramas de diseño, verificaciones de la consistencia de diagramas y seguir el rastro de las pruebas del programa que han sido ejecutadas.

Los ingenieros de software tienen responsabilidades para con la profesión y la sociedad. No deberían simplemente preocuparse de aspectos técnicos.

Las sociedades profesionales publican códigos de conducta que establecen los estándares de comportamiento que se esperan de sus miembros.

¿Cuál es la función de un administrador de proyecto?

Es el coordinador y responsable del trabajo de su equipo.

Entre sus responsabilidades está:

– coordinar el trabajo de los distintos miembros del equipo

– interactuar con el cliente

– planificar y administrar el proyecto

– velar por el cumplimiento de los plazos y los costos programados.

¿Cuál es la función de un analista?

Es el encargado de especificar los requisitos del problema a desarrollar.

Entre sus tareas está:

– entrevistar al cliente

– generar los documentos de requisitos de software (SRD) y de usuario (URD)

– velar porque el diseño cumpla con los requisitos (funciones de tester)

– velar porque el producto final cumpla con los requisitos (funciones de tester).

¿Cuál es la función de un administrador de un diseñador?

Es el encargado de general el diseño del sistema.

Entre sus funciones está: genera el diseño arquitectónico y diseño detallado de sistema basándose en los requisitos;

Generar un prototipo rápido del sistema (con analistas e implementadores) para chequear los requisitos;

Velar porque los requisitos del cliente cuenten con una solución factible (función de tester);

Velar porque el producto final se ajuste al diseño realizado (función de tester).

¿Cuál es la función de un administrador de un implementador?

Es el encargado de implementar el sistema.

Entre sus funciones está:

– implementar los diseños del sistema (BD y código ejecutable);

– implementar los prototipos rápidos para chequea los requisitos;

– coordinar la tarea de todas las personas que colaboren con la implementación.

¿Cuál es la función de un administrador de un tester?

Es el encargado de asegurar la calidad de cada uno de los productos (documentos, prototipos, programas, producto).

Entre sus tareas está:

– coordinar las inspecciones y caminatas;

– velar por la adhesión al estándar adoptado para el desarrollo;

– velar por la completitud, exactitud y no ambigüedad de los documentos;

– velar por la calidad del producto final (cumplimiento de los requisitos).

Introducción a la Ingeniería de Software

Generalidades de la Ingeniería de Software.

Problemas con la ejecución de grandes proyectos de software.

¿Qué se requiere?

Características de la Ingeniería de Software.

Conclusiones.

¿Cuáles son las características del software?

SOFTWARE considerado como ARTE

TRANSICIÓN Y RECONOCIMIENTO (Casas de Software)

ERA DEL SOFTWARED del problema de INGENIO, a… …… uso de METODOLOGÍA

NATURALEZA DEL SOFTWARE

Producto Lógico

Costos en el desarrollo y no en la producción

No se desgasta

Confiabilidad determinada por atributos lógicos

CARACTERISTICAS DEL SOFTWARE

Variedad de funciones

Variedad de Implementación

Evolución (Aceptar cambios y mejoras)

Visibilidad

Continuidad (pequeños cambios, requieren Grandes cambios en el proyecto)

PRODUCTOS DE SOFTWARE "EQUIVALEN A" PRODUCTOS DE INGENIERIA

CICLO DE VIDA

PRODUCTO CONCEPTUAL Y NO FÍSICO (Debe tener un mayor componente Humano)

DESEQUILIBRIO ENTRE COMPONENTES

-Hardware, principios físicos y matemáticos

- Software

PROCESO DE SOLUCION DE PROBLEMAS

A través de COMPUTADORES

MODELOS para describir Algoritmos Mecanizables

Algoritmos de Markov

Máquinas de Turing

Funciones recursivas

¿Qué es la ingeniería de software? por varios autores

1)"La Ingeniería de Software es el área de las ciencias de la computación que trata con la construcción de

Sistemas de software, los cuales son tan grandes y complejos que se construyen con equipos de ingenieros [Ghezzi91].""Construcción multi-persona de software multi-versiones [Parnas87]."

2)"Es un proceso definido paso a paso, que facilita la especificación, el diseño, la implementación y las pruebas de una solución de software, para un conjunto de requisitos explícitos, de modo eficiente y eficaz".

Esto requiere que antes de empezar se tenga:

Objetivos claros

Planes para lograr los objetivos,

Procedimientos que implementen los planes,

Un ambiente conducente al logro de los objetivos

3 Es la aplicación práctica del conocimiento científico, en el diseño y construcción de programas de computador, y la documentación requerida para el desarrollo, operación y mantenimiento de él.

¿Cuáles son los pasos de para construir un producto de ingeniería?

1. PLANIFICACIÓN
2. ANÁLISIS
3. DISEÑO
4. IMPLEMENTACIÓN
5. PRUEBA
6. OPERACIÓN
7. MANTENIMIENTO

¿Estrategias generales de la ingeniería de software?

Analizar un problema antes de plantear una solución. No plantear soluciones tratando de identificar los problemas.

Separar los problemas complejos en sub-problemas más simples:

– acotar las relaciones entre los sub-problemas.

No evitar el cambio, administrarlo.

Especificar claramente la calidad esperada.

En cuanto al cliente…

Nunca perder de vista al cliente y al usuario.

Tener en cuenta:

– los objetivos particulares del cliente y el usuario,

– los temores del usuarios,

– los costos de aprender el uso del sistema,

– los costos de usar el sistema (incluyendo actividades que no implican el uso del computador),

– agrado/desagrado de usar el software.

¿A qué se refiere con la factibilidad?

Diseñar algo implementable:

– con tecnología accesible,

– con costos aceptables,

– con plazos adecuados,

– con una buena relación costo-beneficio,

– con un nivel de calidad suficiente y abordable

¿Qué es un proyecto grande?

La Ingeniería de Software tiene que ver con la cconcepción, diseño, implementación y mantenimiento de sistemas GRANDES basados en computador.

– El software debe ser terminado a tiempo y dentro del presupuesto;

– el software debe tener niveles de desempeño y usabilidad aceptables;

– el software debe ser correcto, confiable y robusto.

Lo más difícil es lograr todo esto en grandes proyectos

¿Cuán grande es un proyecto grande?

– Windows NT 4.0 tiene entre 6 y 10 millones de líneas de código.

– El código del bombardero B2 tiene 3.5 millones.

– Un switch telefónico típico tiene 2 millones. 1 millón de líneas de código (1 MLOC) es:

– 13.000 páginas (75 líneas por página);

– 26 tomos de 500 páginas cada uno;

– 22 horas para imprimir (10 pp/min);

– 12 horas para recompilar (...depende de la máquina).

Consecuencias

En casos extremos:

– quiebra del productor de software;

– quiebra de los clientes que dependen del producto;

– pérdida de vidas humanas.

¿Qué calidad presenta nuestro software?

Fallos en el desarrollo.

Accidentes.

Software de baja calidad.

¿Cuáles son los problemas con el desarrollo de software?

Estudio llevado a cabo por IBM en el año 1994:

El 55% de los sistemas costaron más de lo previsto.

El 68% excedieron el tiempo previsto para su desarrollo.

El 88% se tuvo que volver a diseñar por completo.

Sistema de Automatización Avanzada (FAA, 1982-1994)

El promedio de producción industrial era de 100 dólares/línea, y se preveía pagar 500 dólares/línea.

Se terminó por pagar 700-900 dólares/línea.

Trabajos cancelados por valor de 6.000 millones de dólares.

Departamento de Estadística Laboral (1997)

2 de cada seis nuevos sistemas que se ponen en funcionamiento sufren cancelaciones.

Los grandes sistemas tienen aproximadamente un 50% de probabilidad de ser cancelados.

La media de tiempo empleado en un proyecto se excede en un 50% con respecto al plazo previsto.

Las ¾ partes de los sistemas se consideran “fracasos operativos”.

Accidentes

La mayor parte de los expertos coinciden en señalar que :

“la manera más probable de destruir el mundo es por accidente”.

Y aquí es donde entramos en juego nosotros, los profesionales de la informática: “nosotros somos los que provocamos los accidentes".

Nathaniel Borenstein, creador de MIME en: *Programming as if People Mattered: Friendly Programs, Software Engineering and Other Noble Delusions,* Princeton University Press, Princeton, NJ, 1991.

Historias de Horror

El Bank of America proyectó US$23 M para un proyecto de 5 años. Se terminó gastando más de US$60 M para finalmente abandonar el proyecto. Pérdidas totales estimadas en más de US$1000 M.
El bombardero B1 requirió US$1000 M más de lo proyectado para mejoras su sistema de defensa.
All State (seguros) comenzó en 1982 un proyecto de automatización integral de sus operaciones de 5 años de duración y US$8 M de presupuesto. Fue abandonado en 1993 después de gastar US$100 M.

Como para preocuparse

BlueCross (isapre norteamericana) perdió más de US$60 M en pagos incorrectos debido a un error en el software por el que habían pagado más de US$200 M.
El 4 de julio de 1996, un cohete Ariane se desvió de su curso y explotó a 3700 m de altura. La causa: un error de software. Parte de las conclusiones de la comisión investigadora son:
“...en el escenario de la falla, la principal causa técnica es el error de operando al convertir la variable BH, y la falta de protección en esta conversión causó que el computador SRI dejara de funcionar.”

Casos límite

Entre 1987 y 1995 a lo menos 6 pacientes fueron severamente heridos o muertos por un error en los sistemas del Therac-25(acelerador lineal usado para administración de radioterapia).

Caso Therac-25 (1985-87)

Se trataba de un aparato de radioterapia dotado de controlador de software.
Se retiró el interbloqueo del hardware, pero el software no tenía dispositivo de interbloqueo.
El software falló al mantener las constantes vitales: un flujo de electrones o bien un flujo más intenso de radiación mediante placa para generar rayos X.
A consecuencia de ello se produjeron varias muertes por quemaduras.
El programador no tenía experiencia en programación concurrente.
Véase: http://sunnyday.mit.edu/therac-25.html

Cabría pensar que se aprendería de la experiencia y que un desastre de este tipo no volvería a suceder jamás.
Sin embargo... La Agencia Internacional de Energía Atómica anunció una “emergencia radiológica” el 22 de Mayo del 2001 en Panamá.
28 pacientes sufrieron sobreexposición: 8 murieron, 3 de ellos como consecuencia directa de la sobreexposición; con la probabilidad de que ¾ de los 20 que sobrevivieron desarrollaran “serias complicaciones que en algunos casos, a la larga, podrían resultar mortales”
Los expertos anunciaron que el equipo de radioterapia “funcionaba perfectamente”; la razón de la emergencia tuvo que ver con la entrada de datos.
Si los datos se introdujeron en varios bloques protegidos dentro de un lote, la dosis se computó de manera incorrecta.
Al menos, la FDA llegó a la conclusión de que “la interpretación de los datos del bloqueo de flujo por el software” fue uno de los factores causantes del desastre.
Visite la web: http://www.fda.gov/cdrh/ocd/panamaradexp.html

Ariane-5 (Junio de 1996)

Agencia Espacial Europea.
Pérdida absoluta de misiles no tripulados poco después del despegue.
Causada por una excepción en el código de Ada.
Ni siquiera se precisó el código defectuoso después del despegue.
Debido a un cambio en el entorno físico: se infringieron supuestos no documentados.
Visite la web: http://www.esa.int/htdocs/tidc/Press/Press96/ariane5rep.html
En los desastres provocados por fallos de software, son más comunes los accidentes como el del Ariane, que los causados por aparatos de radioterapia. No es muy probable que los errores en el código sean la causa; normalmente, el problema se remonta al análisis de las necesidades; en este caso, a un error al articular y evaluar presunciones claves sobre el entorno.

Servicio de Ambulancias de Londres (1992)

Pérdida de llamadas, doble servicio por llamadas duplicadas.
Mala selección del programador: experiencia insuficiente.
Visite la web: http://www.cs.ucl.ac.uk/staff/A.Finkelstein/las.html
El desastre del Servicio de Ambulancias de Londres se debió en realidad a fallos de gestión. Los informáticos que desarrollaron el software pecaron de ingenuidad y aceptaron una oferta de una empresa desconocida que era bastante peor que las de otras compañías más acreditadas.
Cometieron el terrible error de saltar a la red repentinamente, sin pararse a contrastar la ejecución del sistema nuevo y la del ya existente. A corto plazo, estos problemas se acentuarán debido al uso generalizado del software en nuestra infraestructura cívica.

En el informe PITAC se hacía eco de esta realidad, y los argumentos que en él se exponían han servido para incrementar los fondos destinados a la investigación en software:
"La demanda de software ha crecido mucho más rápido que nuestra capacidad para crearlo. Además, el país requiere un tipo de software más práctico, fiable y robusto que el que se está desarrollando hoy en día. Nos hemos hecho peligrosamente dependientes de los grandes sistemas de software, cuyo comportamiento no es del todo comprensible, y que a menudo fallan de forma imprevista".
Investigación en tecnología de la información: Invirtiendo en nuestro futuro
Comité Consultor en Tecnología de la Información del Presidente (PITAC) Informe al Presidente, 24 de febrero de 1999

¿Características del software grande?

Escala- - - - - - - – >una sola persona no puede entenderlo todo.

Complejidad –> requiere trabajo en equipo,

– >problemas de manejo de personas,

– >coordinación, egos, motivación, rotación, etc.

Vida - - - - - - - – >años y décadas.

Cambio

– las necesidades cambian desde que se comienza el desarrollo,

– el sistema es *"obsoleto"* no bien se termina.

Especificaciones imprecisas

– ambigüedad, contradicciones, requisitos cambiantes,

– distintos usuarios piden distintas cosas del sistema.

¿Qué se requiere?

Burocracia: útil y efectiva, tediosa pero vital.

Manejo formal del proceso de desarrollo;

Documentación formal detallada tanto interna como externa

Puede pensarse en términos de contratos cliente-productor;

– Trazabilidad:

¿Quién escribió este código?

¿Cuándo se agregó esta parte?

¿Quién autorizó el cambio?

– Manejo de configuración y control de versiones.

Técnicas usadas

Análisis riguroso del problema:

– interacción con el usuario;

Diseño cuidadoso usando principios que han demostrado ser útiles:

– abstracción, encapsulamiento, ocultamiento, etc. Implementación disciplinada;

Pruebas rigurosas:

– procedimientos bien definidos de antemano,

– resultados de las pruebas documentados;

Planificación a largo plazo (mantenimiento).

¿Productos de desarrollo de software?

Más que código: en un proyecto de software se genera además del código, muchos otros documentos: requisitos formales, diseño de alto nivel, diseño detallado, plan y casos de prueba, documentación de usuario, estudios de factibilidad, informes de marketing, planes de mantenimiento, informes de errores y correcciones.

Se requieren personas

Ingeniero de procesos

Analista de requisitos

Programadores

Gerente de proyecto

Ingeniero de calidad

Diseñador de software

Verificador-testeador

Gestor de configuración de software

IS como una Ingeniería

Ataca problemas prácticos reales

– la gente realmente quiere o necesita resolver esos problemas.

Genera soluciones razonables

– en términos de costos, uso de recursos, etc.

Tiene su base en la ciencia

– sus resultados son repetibles,

– usa modelos matemáticos,

– es un área técnica bien entendida.

Codificación del conocimiento

– la experiencia de generaciones se plasma en manuales para ser reutilizada.

Responsabilidad profesional

– código de conducta,

– ética profesional,

– acreditación y monitoreo por parte de sociedades profesionales.

Pero el software es diferente

No es posible aplicar metodologías usadas en la ingeniería de otros productos porque hay varias diferencias importantes:

– el software no se manufactura sino que se desarrolla o crea

Control de calidad de distinta naturaleza, recursos humanos con distinto perfil, estructura de costos radicalmente distinta.

– El software no se desgasta (idealmente), pero sufre envejecimiento.

Aún faltan cosas

La mayor parte de los proyectos implica partir casi de cero:

– no hay catálogos de componentes a disposición de los ingenieros.

Falencias habituales importantes:

– procesos no repetibles,

– modelos y descripciones imprecisas,

– baja confiabilidad,

– la experiencia no se comparte, codifica ni re-usa,

– entrenamiento insuficiente de los profesionales.

Proceso de Desarrollo de Software

Para mejorar la situación es necesario mejorar el proceso de desarrollo de software.

Para mejorar el proceso, debe ser más visible.

La administración y control de proyectos de software implica:

– métricas, recursos y tiempos,

– manejo de riesgo,

– manejo de cambios,

– control de calidad.

Conclusiones

Análisis y especificación de requisitos

– no queremos software que no se use,

– no queremos usuarios descontentos.

Diseño de software

– queremos diseños que respeten los requisitos,

– queremos diseños realistas.

Verificación y validación

– queremos productos confiables,

– queremos productos que satisfagan las especificaciones.

Apoyo del computador

– queremos hacer todo de la forma más eficiente posible

¿Por qué es importante la ingeniería de software?

Aporte del software a la economía de EE.UU (datos del año 1996):

Principal fuente de superávit por exportaciones en la balanza comercial.

24.000 millones de dólares de ingresos por exportaciones de software y 4.000 millones gastados en importaciones, arrojan un superávit anual de 20.000 millones de dólares.

(Datos tomados de: *Software Conspiracy,* Mark Minasi, McGraw Hill, 2000).

Papel del software en la infraestructura:

No sólo tiene un papel importante en Internet; también en sectores como transportes, energía, medicina y finanzas.

El software se halla cada vez más presente como elemento incorporado a otros mecanismos arraigados.

Los automóviles modernos, por ejemplo, poseen entre 10 y 100 procesadores para dirigir todo tipo de funciones, desde el reproductor de música hasta el sistema de frenado.

¿Qué es coste de software?

Coste total de la propiedad del software: 5 veces el coste del hardware.

El grupo Gartner calcula que el coste de mantenimiento de un PC durante 5 años asciende en la actualidad a 7 dólares por cada K de memoria del ordenador).

¿Calidad de software?

Sistema de medición de la calidad: errores/kloc.La medición se realiza después de la entrega del software.

La media en la industria es de aproximadamente 10. De alta calidad: 1 o menos.

¿Qué son los procesos de la producción de software?

¿Objetivos de los modelos de proceso?

¿Beneficios de los modelos de proceso?

¿Evolución de los modelos de proceso?

Proceso de Producción de Software

Son las actividades que se realizan para la construcción, liberación y evolución de un producto de software, comenzando con una idea y finalizando con el retiro del sistema.

Los modelos estructurados de procesos de software se conocen como *ciclo de vida* del software.

Objetivos de Modelos de Procesos

La meta de un modelo estructurado de proceso es determinar el orden de las etapas que componen el desarrollo y la evolución de un software, estableciendo los "criterios de transición" para progresar de una etapa a la siguiente.

Los modelos están hechos para contestar las siguientes preguntas en un proyecto de software:

¿Qué debemos hacer a continuación?

¿Cuánto tiempo debemos continuar haciéndolo?

Beneficios de los Modelos

Proveen guías a los ingenieros de software acerca del orden en el cual deben ser llevadas a cabo las variadas actividades técnicas.

Dan un marco o estructura para gerenciar, desarrollar y mantener el software, el cual nos permite:

– estimar los recursos,

– definir hitos intermedios,

– monitorear los progresos.

Evolución de los Modelos

Codificar & Corregir (>1960)

Modelo de Cascada (>1970)

Modelo de Transformaciones (>1975)

Modelo Evolutivo (>1985)

Modelo de Espiral (>1988)

RUP (>1997)

WebE (>1998)

Programación Extrema (>2000)

¿Indique algunos modelos genéricos y específicos?

No basta con seleccionar un modelo genérico de desarrollo de software y tratar de adaptarse a él.

Se requiere una definición más precisa de cómo se llevan a cabo las distintas tareas.

Modelos genéricos:

Son todos los modelos de desarrollo tradicional de software: cascada, espiral, prototipos, etc.

Presentan una estrategia general para realizar el desarrollo,

Casi nunca es posible realizar un desarrollo adhiriendo completamente al modelo.

Permiten un entendimiento de tipo general del proceso pero no es fácil llevarlos a un nivel de detalle más fino, necesario para guiar el trabajo de los profesionales;

No representan en forma precisa lo que realmente se hace.

Modelos Específicos:

– aparecen las *principales actividades* involucradas en el proceso;

– brindan una especificación precisa entre las mismas;

– establecen relaciones entre Las diversas tareas, Los artefactos producidos en ellas, las personas que las realizan, las herramientas utilizadas;

– usan formalismos particulares.

Codificar y Corregir

Usado en los comienzos de la computación cuando ésta era aún una actividad personal y artesanal.

Consta de dos etapas:

– codificar

– eliminar errores en el código.

Fuente de dificultades y deficiencias:

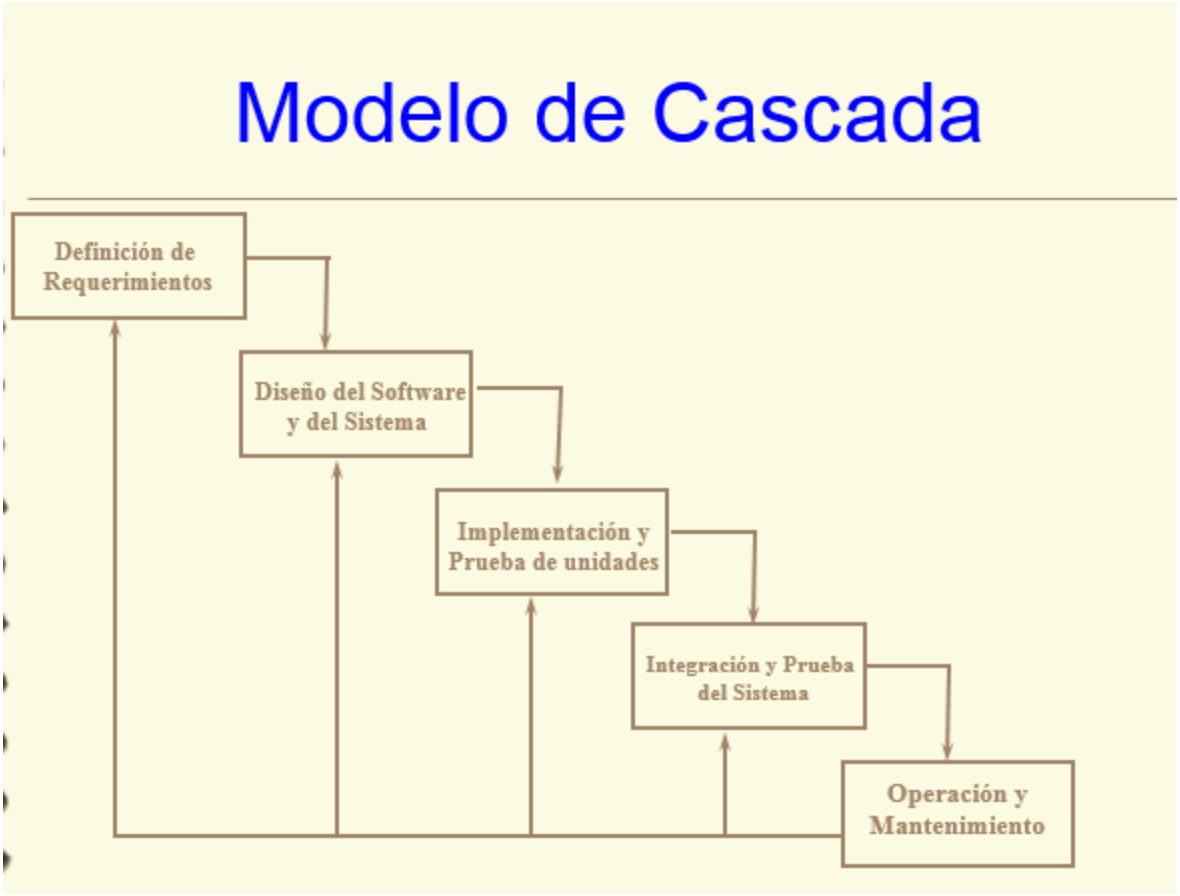
– luego de una secuencia de cambios, el código era tan enredado que eliminar errores era una tarea pesada y muy difícil de realizar;

– cuando el desarrollo de sistemas dejó de ser una actividad personal y artesanal, el modelo no podía manejar la complejidad de los sistemas;

– no acepta la rotación de personal en un proyecto.

¿Explique cómo funciona el modelo cascada?

El modelo de cascada, aunque algo desprestigiado, continúa siendo usado para visualizar las etapas de desarrollo (¿cómo debería avanzarse?) y funciona bien si no hay grandes sorpresas.



¿Cuáles son las actividades en el modelo cascada?

En el modelo de cascada puro no hay retroalimentación entre las actividades.

Existen diferentes versiones que incluyen entre 5 y 10 actividades:

– estudio de factibilidad, integración, instalación, etc.

En todo caso es importante:

– identificar de las actividades principales,

– definición de los documentos que debe entregarse como hito al fin de la tarea.

Documentos del modelo cascada

Actividad	Documentos Producidos
Análisis de Requerimientos	Documento de Requerimientos
Definición de Requerimientos	Documento de Requerimientos.
Especificación del Sistema.	Especificación Funcional, Plan de Pruebas de Aceptación.
Diseño Arquitectural	Especificación de la Arquitectura, y Plan de Pruebas del Sistema
Diseño de Interfaces	Especificación de la Interfaces y Plan de pruebas de Integración.
Diseño Detallado	Especificación del diseño y Plan de prueba de Unidades.
Codificación	Código de Programa
Prueba de Unidades	Reporte de prueba de unidades
Prueba de Módulos	Reporte de prueba de módulos
Prueba de Integración	Reporte de prueba de integración y Manual de usuario final
Prueba del Sistema	Reporte de prueba del sistema
Prueba de Aceptación	Sistema final mas la documentación.

¿Fases del modelo de cascada?

Análisis de requerimientos y definición.

Diseño del sistema y del software.

Implementación y prueba de unidades

Integración y prueba del sistema.

Operación y mantenimiento.

La dificultad en esta modelo reside, en la dificultad de hacer cambios entre etapas.

¿Explique el modelo evolutivo o incremental?

La idea es combinar los elementos del tradicional modelo de cascada con la filosofía de prototipos

Incremento 1

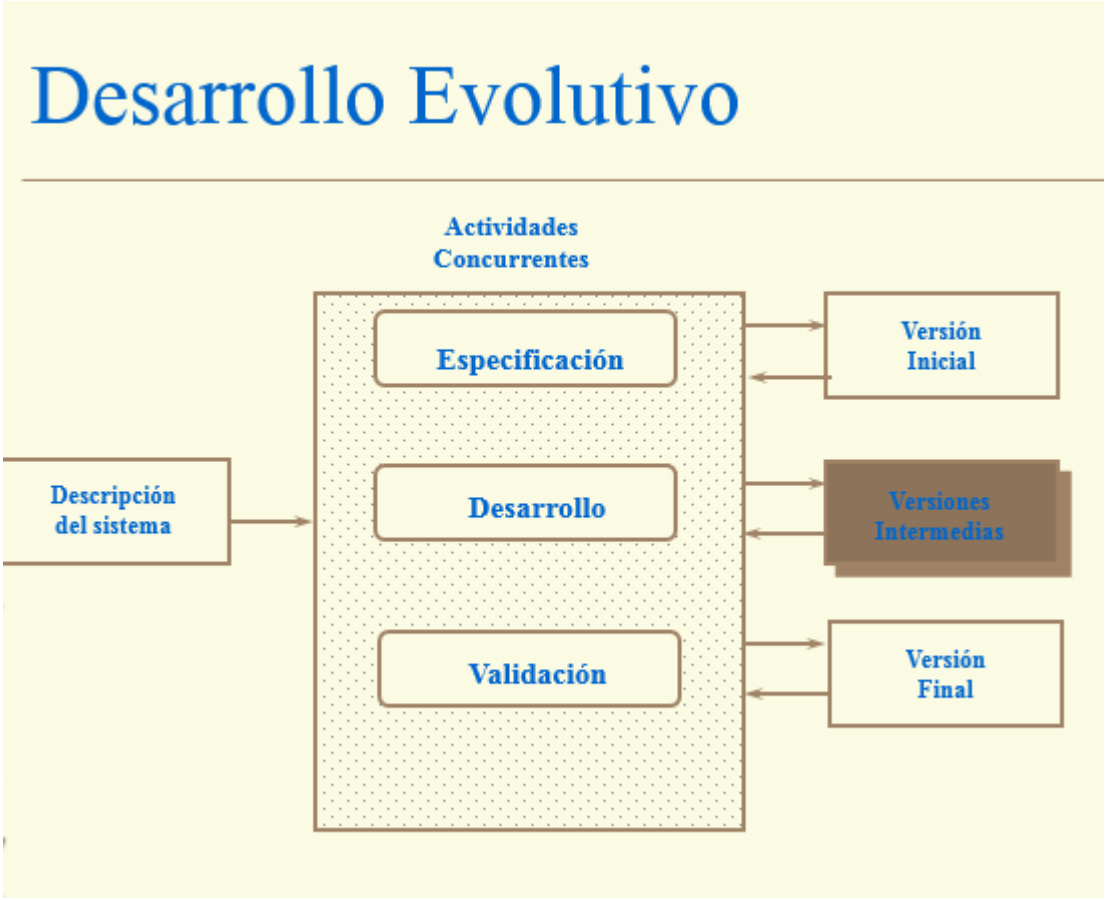
Modelo de la cascada

Incremento 2

Modelo de la cascada

etc.

¿Características del desarrollo evolutivo?



Características del desarrollo evolutivo

Secuencias lineales en forma escalonada.

Cada secuencia genera un incremento del software.

– Ejemplo:

Diseño e implementación de interfaces usuarias;

Diseño, implementación y carga de la base de datos;

Diseño e implementación de los procesos de ingreso y consulta de información en línea;

Diseño e implementación de los procesos batch.

Cada incremento es una versión reducida del producto final que puede ser validada por el usuario.

Es una buena preparación para el mantenimiento evolutivo del sistema.

Problemas con el desarrollo evolutivo

Poca visibilidad en el proceso

Los sistemas están pobremente especificados

Se requieren habilidades especiales.

Aplicabilidad con el desarrollo evolutivo

Para sistemas interactivos pequeños o medianos.

Para partes de sistemas grandes (p.ej. la interfaz de usuario).

Para sistemas de corta vida.

¿Qué es un prototipo?

Prototipo exploratorio

El objetivo es trabajar con clientes hasta evolucionar a un sistema final, a partir de una especificación inicial. Se debe comenzar con unas especificaciones bien entendidas.

Prototipo de "throw-away". (Cambio)

El objetivo es entender los requerimientos del sistema. Se puede comenzar con especificaciones poco entendidas.

¿Qué problemas y riesgos tienen los modelos?

Cascada.

Alto riesgo en sistemas nuevos debido a problemas en las especificaciones y en el diseño.

Bajo riesgo para desarrollos bien comprendidos utilizando tecnología conocida.

Prototipo.

Bajo riesgo para nuevas aplicaciones debido a que las especificaciones y el diseño se llevan a cabo pasó a paso.

Alto riesgo debido a falta de visibilidad

Evolutivo.

Alto riesgo debido a la necesidad de tecnología avanzada y habilidades del grupo desarrollador.

¿Manejo de riesgos?

La tarea principal del administrador consiste en minimizar riesgos.

El “riesgo” inherente en una actividad se mide en base a la incertidumbre que presenta el resultado de esa actividad.

Las actividades con alto riesgo causan sobre-costes en cuanto a planeación y costos

El riesgo es proporcional al monto de la calidad de la información disponible. Cuanto menos información, mayor el riesgo.

Manejo de Riesgos

La tarea principal del administrador consiste en minimizar riesgos.

El “riesgo” inherente en una actividad se mide en base a la incertidumbre que presenta el resultado de esa actividad.

Las actividades con alto riesgo causan sobre-costes en cuanto a planeación y costos

El riesgo es proporcional al monto de la calidad de la información disponible. Cuanto menos información, mayor el riesgo.

¿Qué son los modelos híbridos?

Los sistemas grandes están hechos usualmente de varios subsistemas.

No es necesario utilizar el mismo modelo de proceso para todos los subsistemas.

El prototipo es recomendado cuando existen especificaciones de alto riesgo.

El modelo de cascada es utilizado en desarrollos bien comprendidos.

¿Cuáles son las fases del modelo en espiral?

Planteamiento de Objetivos: Se identifican los objetivos específicos para cada fase del proyecto.

Identificación y reducción de riesgos: Los riesgos clave se identifican y analizan, y la información sirve para minimizar los riesgos.

Desarrollo y Validación: Se elige un modelo apropiado para la siguiente fase del desarrollo.

Planeación: Se revisa el proyecto y se trazan planes para la siguiente ronda del espiral.

Plantilla para una ronda del espiral

Objetivos.
Restricciones.
Alternativas.
Riesgos.
Resolución de riesgos.
Resultados.
Planes.
Garantías

¿Cuál es la flexibilidad del modelo en espiral?

Para sistemas bien comprendidos utiliza el Modelo de Cascada. La fase de análisis de riesgos es relativamente fácil.

Con requerimientos estables y sistemas de seguridad críticos, utiliza modelos formales.

Con especificaciones incompletas, utiliza el modelo de prototipo.

Pueden utilizarse modelos híbridos en distintas partes del desarrollo.

¿Cuáles son las ventajas del modelo en espiral?

Centra su atención en la reutilización de componentes y eliminación de errores en información descubierta en fases iniciales.

Los objetivos de calidad son el primer objetivo.

Integra desarrollo con mantenimiento.

Provee un marco de desarrollo de hardware/software.

¿Cuáles son los problemas del modelo en espiral?

El desarrollo contractual especifica el modelo del proceso y los resultados a entregar por adelantado.

Requiere de experiencia en la identificación de riesgos.

Requiere refinamiento para uso generalizado.

¿Explique la visibilidad de procesos?

Los sistemas de software son intangibles por lo que los administradores necesitan documentación para identificar el progreso en el desarrollo.

Esto puede causar problemas..

El tiempo planeado para entrega de resultados puede no coincidir con el tiempo necesario para completar una actividad.

La necesidad de producir documentos restringe la iteración entre procesos.

El tiempo para revisar y aprobar documentos es significativo.

El modelo de cascada es aún el modelo basado en resultados más utilizado.

Visibilidad de Procesos

Modelo de Proceso	Visibilidad del Proceso
Modelo de Cascada	Buena visibilidad, cada actividad produce un documento o resultado
Desarrollo Evolutivo	Visibilidad pobre, muy caro al producir documentos en cada iteración.
Modelos Formales	Buena visibilidad, en cada fase deben producirse documentos.
Desarrollo orientado a la reutilización	Visibilidad moderada. Importante contar con documentación de componentes reutilizables.
Modelo de Espiral	Buena visibilidad, cada segmento y cada anillo del espiral debe producir un documento.

¿Explique que es la responsabilidad profesional?

Los Ingenieros de software no solo deben considerar aspectos técnicos. Deben tener una visión más amplia, en lo ético, social y profesional.

No existen estatutos para ninguno de estos aspectos.

Desarrollo de sistemas militares.

Piratería.

Que es mejor para la profesión de Ingeniero de Software.

Aspectos Éticos

Confidencialidad.
Competencia.
Derechos de propiedad intelectuall.
Mal uso de la computadora.

Resumen

Los productos de software consisten de programas y documentación.

El proceso de software consiste en aquellas actividades involucradas en el desarrollo de software.

Los productos de software consisten de programas y documentación.

El proceso de software consiste en aquellas actividades involucradas en el desarrollo de software.

El modelo de cascada considera cada actividad del proceso como una actividad discreta.

El modelo de desarrollo evolutivo considera actividades del proceso en forma concurrente.

El modelo de espiral se basa en análisis de riesgos.

La visibilidad del proceso involucra la creación de documentos o resultados de las actividades.

Los Ingenieros de software deben tener responsabilidades éticas, sociales y profesionales.

¿Qué es un proceso de software?

El modelo de cascada

Separadas y distintas fases de la especificación y el desarrollo.

Desarrollo evolutivo

Especificación, desarrollo y validación están intercalados.

Ingeniería de Software Basada en Componentes

El sistema está montado a partir de los componentes existentes.

Hay muchas variantes de estos modelos, por ejemplo, el desarrollo formal donde se toma un proceso similar al de cascada, pero la especificación es una especificación formal que se perfecciona a través de varias etapas hacia un diseño implementable.

¿Cuáles son los modelos genéricos de proceso de software?

El modelo de cascada

Separadas y distintas fases de la especificación y el desarrollo.

Desarrollo evolutivo

Especificación, desarrollo y validación están intercalados.

Ingeniería de Software Basada en Componentes

El sistema está montado a partir de los componentes existentes.

Hay muchas variantes de estos modelos, por ejemplo, el desarrollo formal donde se toma un proceso similar al de cascada, pero la especificación es una especificación formal que se perfecciona a través de varias etapas hacia un diseño implementable.

¿Cuáles son las fases del modelo de cascada?

Análisis de requerimientos y definición diseño del sistema y del software

Ejecución y unidad de pruebas

Implementación y pruebas del sistema

Operación y mantenimiento

El principal inconveniente del modelo de cascada es la dificultad de acomodar el cambio después de que el proceso está en marcha. Una fase tiene que ser completada antes de pasar a la siguiente fase.

¿Cuáles son los problemas del modelo cascada?

¿Explique de qué se trata el desarrollo evolutivo?

¿Explique que es la ingeniería de software basada en componentes?

¿Desarrollo orientado a la reutilización?

¿Qué es interacción de procesos?

¿Qué es la entrega incremental?

¿Ventajas del desarrollo incremental?

¿Qué es la programación extrema?

¿Qué es el desarrollo en espiral?

¿Cuáles son los sectores del modelo en espiral?

¿A qué se refiere con especificación de software?

¿A qué se refiere diseño e implementación de software?

¿Cuáles son las actividades del proceso de diseño?

¿Indique algunos métodos estructurados?

¿Qué es la programación y la depuración?

¿Qué es la validación de software?

¿Proceso de pruebas?

¿Qué es una evolución de software?

¿Qué es un proceso unificado rational y sus fases?

¿Cuáles son las buenas prácticas del RUP?

¿Explique la ingeniería de software basada en computadora?

¿Qué es la tecnología CASE?

¿Qué es la gestión de proyectos?

¿Cuáles son las decisiones de la gestión de proyectos?

¿Cuáles son las actividades de la gestión de proyectos?

¿Cuáles son los aspectos comunes de la gestión de proyectos?

¿Explique la dotación de personal del proyecto?

¿Explique cómo se lleva a cabo la planificación de proyectos?

¿Cuál es la estructura del plan de proyecto?

¿La organización de las actividades en un proyecto?

¿Qué es la calendarización de proyecto?

¿Cuáles son los problemas de la calendarización?

¿Gráficos de barras y redes de actividades?

¿Qué es la gestión de riesgo?

¿Qué es el proceso de gestión de riesgos?

¿Qué es el análisis de los riesgos?

¿Qué es la planificación de los riesgos?

¿Qué es la supervisión de riesgos?

¿Qué es la ingeniería de requerimientos?

¿Qué es un requerimiento?

¿Qué es la abstracción de requerimientos?

¿Cuáles son los tipos de requerimientos?

¿Qué es el sistema LIBSYS?

¿Algunos ejemplos de requerimientos funcionales?

¿Qué es la imprecisión de requerimientos?

¿Qué es la integridad de requerimientos y consistencia?

¿Qué son los requerimientos no funcionales?

¿Cuáles son los tipos de requerimientos no funcionales?

¿Hable de las metas y requerimientos?

¿Qué es la interacción de requerimientos?

¿Qué es el requerimiento de dominio?

¿Cuáles son los problemas de los requerimientos de dominio?

¿Explique que son los requerimientos de usuario?

¿Cuáles son los problemas con el lenguaje natural?

¿Cuáles son los requerimientos de usuario par un sistema de compatibilidad LIBSYS?

¿Cuáles son los requerimientos de usuario para un editor de cuadricula?

¿Cuáles son los problemas de requerimientos?

¿Cuáles son las directrices para la redacción de requerimientos?

¿Qué es requerimientos del sistema?

¿Cuáles son los problemas con las especificaciones en lenguaje natural?

¿Qué son las especificaciones basadas en formulario?

¿Qué es la especificación tabular?

¿Qué es el modelo grafico?

¿Qué son los diagramas de secuencia?

¿Qué es la especificación de la interfaz?

¿Qué es el documento de requerimientos?