CC51A – Ingeniería de Software

Modelos de Desarrollo de Software

Sergio Ochoa D.

Parte de este material ha sido cedido por el Prof. Jaime Navón, PhD.
(www.ing.puc.cl/~jnavon)

Clase 3 - Versión 3

Estructura de la Presentación

- Definición y objetivos de los Modelos de Desarrollo.
- Evolución de estos modelos.
- Conclusiones.

Objetivo de la Presentación:

- Historiar las diferentes formas de organizar el desarrollo de software según el avance de la informática.
- Realizar un estudio crítico de los “ciclos de vida” más representativos.

Definición y Objetivos.

Proceso de producción de software:

- Actividades que se realizan para la construcción, liberación y evolución de un producto de software, comenzando con el estudio de una idea y finalizando con el retiro final del sistema.

Ciclo de Vida:

- Los **modelos estructurados** de procesos de software son conocidos como *ciclo de vida* del software.

Definición y Objetivos (cont...)

Objetivos de un modelo de proceso:

"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?" (B.Bohem 1988)

Definición y Objetivos (cont...)

Beneficios de los modelos:

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

- Dan un marco o estructura para gerenciar, desarrollar y mantener un desarrollo, lo cual nos permite: estimar recursos, definir hitos intermedios y monitorear los progresos.

- Anticipan y controlan el proceso para alcanzar las cualidades deseadas del software.

Definición y Objetivos (cont…)

Beneficios de los modelos:

- Automatizan el proceso con el uso de estándares, metodologías y herramientas.

- Hacen la producción de software **confiable, predecible y eficiente.**

Evolución de Modelos

Modelos de Desarrollo:

- Code & Fix Model (~1960)
- Waterfall Model (~1970)
- Evolutionary Model (~1985)
- Transformation Model (~1975)
- Spiral Model (~1988)
- Component Model (~1992)
- RUP (~1997)
- Webe (~1998)

Modelos de Desarrollo de Soft.

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 de Desarrollo Genéricos

- aquí caen todos los modelos de desarrollo de software tradicionales: cascada, espiral, prototipos, etc.
- presentan una estrategia muy general para efectuar el proceso de desarrollo
- en la realidad casi nunca es posible efectuar el desarrollo adhiriendo completamente a uno de estos modelos

Modelos de Desarrollo de Soft. (cont...)

Modelos de Desarrollo Genéricos

- permiten un entendimiento de tipo general del proceso pero no es fácil llevarlos a un nivel de detalle más fino, lo cual es necesario para guiar el trabajo de los profesionales
- no representan en forma precisa lo que realmente se hace

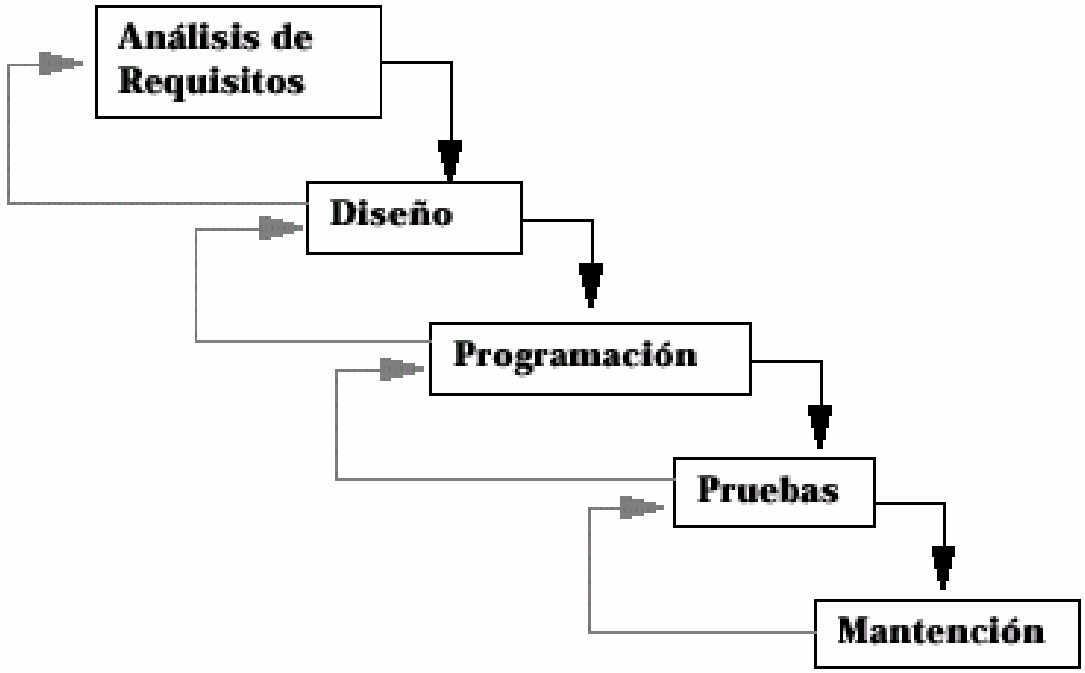
Modelos de Desarrollo Específicos

- Aparecen las *actividades mayores* involucradas en el proceso
- Brindan una especificación precisa del orden entre ellas
- Establecen relaciones entre las diversas tareas, los artefactos producidos en ellas, las personas que las realizan y las herramientas utilizadas
- Usan formalismos gráficos (por ejemplo Redes de Petri)

Code & Fix Model

- Usado en los comienzos de la computación, cuando ésta era una actividad personal y artesanal.
- Consta de 2 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.
- Es una práctica habitual en el desarrollo Web

Waterfall Model



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.

Waterfall Model (cont…)

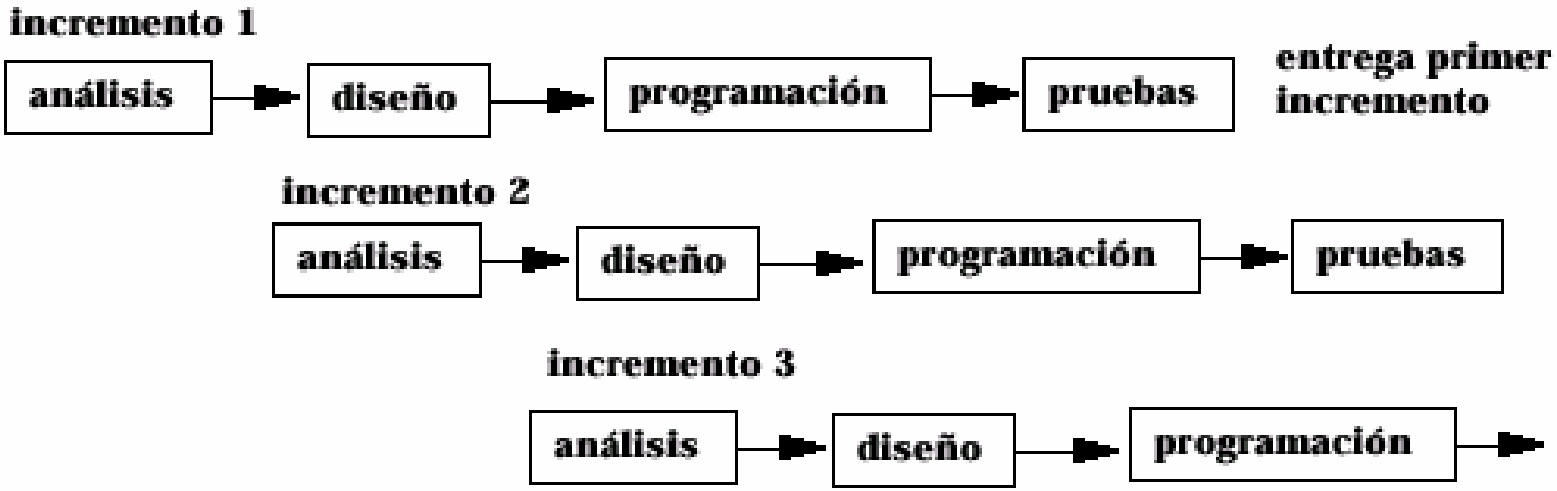
- En el modelo de cascada puro no hay realimentación entre las distintas etapas
- También se usa el modelo con feedback hacia atrás
- Dependiendo de la versión, posee entre 5 y 10 actividades fundamentales
- Utilidad mayor es la identificación de las actividades fundamentales (que aparecen en cualquier proceso) y la definición de los documentos de entrega (deliverables)

Los problemas más importantes de este modelo son: (a) que no maneja los riesgos y (b) que obtiene un producto demasiado tarde.

Waterfall Model (cont...)

Actividad	Documento Entrega
estudio factibilidad	resultado estudio
análisis de requisitos	documento de requisitos de usuario
especificación	especificación funcional test de aceptación borrador manual de usuario
diseño arquitectónico	especificación arquitectónica plan de pruebas de sistema
diseño de alto nivel	especificaciones de interfaces plan de pruebas de integración
diseño detallado	especificación del diseño plan de prueba de unidades
programación	código fuente
prueba unidades	informe pruebas
prueba subsistema	informe pruebas subsistema
prueba integración	informe prueba integración manual de usuario final
prueba del sistema	informe prueba sistema
prueba de aceptación	sistema final con documentación

Evolutionary o Incremental Model



La idea es combinar los elementos del tradicional modelo de cascada con la filosofía de prototipos (usada principalmente en la etapa de definición de requisitos)

Evolutionary o Incremental Model (cont...)

Características:

- Secuencias lineales en forma escalonada
- Cada secuencia genera un incremento del software
  - Ejemplo: Word Processor
- Primer incremento: edición y formateo básicos
- Segundo incremento: párrafos y documento
- Tercer incremento: spelling y gramática
- Cuarto incremento: layout avanzado, etc.

Evolutionary o Incremental Model (cont...)

Características:

- Cada incremento representa una versión reducida del producto final
- Puede ser validada de inmediato por el usuario
- Es una buena preparación para la vida evolutiva del producto
- Es difícil obtener una buena segmentación del producto.
- Es difícil incorporar cambios (respecto a lo que se pretende obtener), una vez que el proyecto está en marcha.

Transformation Model

Representación Gráfica del Modelo

Spiral Model

Modelo Espiral de Desarrollo de Software - Boehm - 1988

Objetivo: *Proveer una estructura para el diseño de procesos de producción de software, guiado por los* ***niveles de riesgo*** *del proyecto a desarrollar.*

Spiral Model (cont...)

El modelo en espiral puede ser visto como un "*metamodelo"* pues cualquier modelo de proceso puede ser enmarcado en él.

Definiciones:

- *Riesgo:* Circunstancias potencialmente adversas que pueden perjudicar el proceso de desarrollo o la calidad del producto.
- *Manejo del Riesgo:* "Disciplina cuyos objetivos son identificar, manejar y eliminar los elementos de riesgo del software, antes que se conviertan en problemas" B.W. Boehm.

Spiral Model (cont...)

- El modelo en espiral se basa en la identificación y eliminación de problemas de alto riesgo a través de un proceso de diseño cuidadoso.
- El modelo en espiral tiene como característica principal que es *cíclico y no lineal*.
- El radio del espiral marca el costo acumulado en el proceso, mientas que la dimensión angular representa el progreso dentro del proceso.
- Cuando no existe riesgo, el modelo en espiral puede reducirse a un modelo en cascada.

Spiral Model (cont...)

Elementos de Riesgo	Técnicas de manejo Riesgo
1- Carencias del personal.	- Staff calificado, equipos, entrenamiento, etc.
2- Cronogramas y presupuestos no realistas.	- Estimación de cronogramas y costos detallados, reuso de experiencia previas.
3- Desarrollo de funciones Equivocadas.	- Análisis organizado, prototipación, etc.
4- Desarrollo de interfases equivocadas.	- Prototipos, escenarios, caracterización de usuario.
5- “Gold plating”...	- Análisis costo-beneficio.
6- Continuos cambios de Requisitos.	- Ocultamiento de información, desarrollo incremental.

Spiral Model (cont...)

Elementos de Riesgo	Técnicas de manejo Riesgo
7- Carencias en componentes externos (soft, hard, etc).	- Análisis de compatibilidad, benchmarks, inspecciones.
8- Carencias en tareas desarrolladas en forma externa.	- Control de referencias, contratos cuidadosos, selección de subcontratados.
9- Carencias de performance en tiempo real.	- Simulación, prototipos, benchmarks.
10- Forzar la computación.	- Análisis técnico, prototipos y honestidad.

Component Model

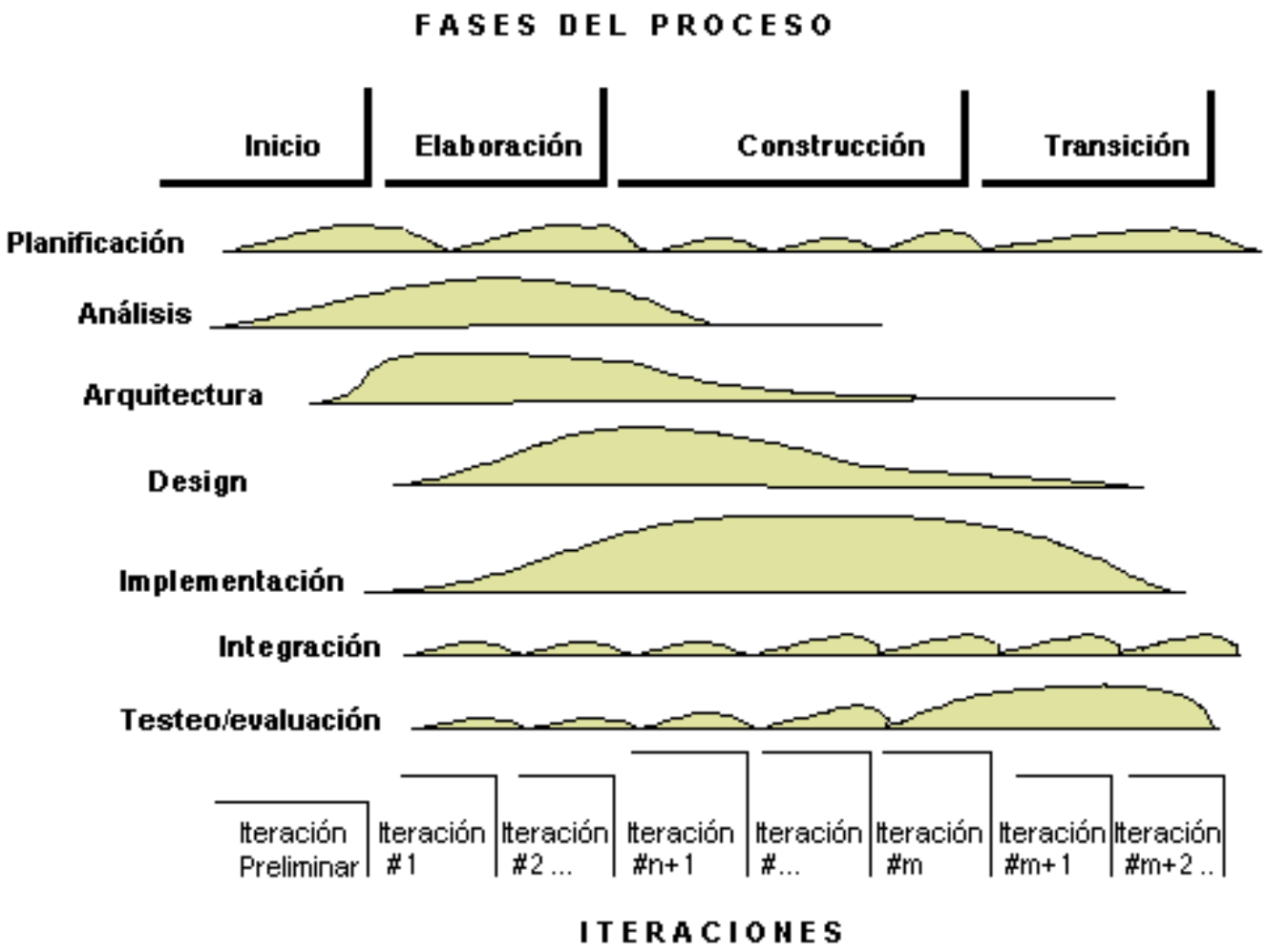
Modelo Espiral de Desarrollo de Software - Boehm - 1988

Modelo Espiral basado en Componentes - Nierstrasz - 1992.

Component Model

- El desarrollo basado en componentes contempla la construcción de sistemas, a través del ensamble de módulos predefinidos (componentes).
- Estos módulos son genéricos (sirven para más de un sistema), configurables, y pueden ser desarrollados por cualquiera.
- Para que esto sea factible, los módulos deben adherir a una especificación de componentes de software como: JavaBeans, ActiveX, EJB, COM, DCOM, etc.
- El desarrollo basado en componente tiene un sin número de ventajas, en lo relacionado con el tiempo y costo de desarrollo, y también en lo que respecta a la calidad del producto final.

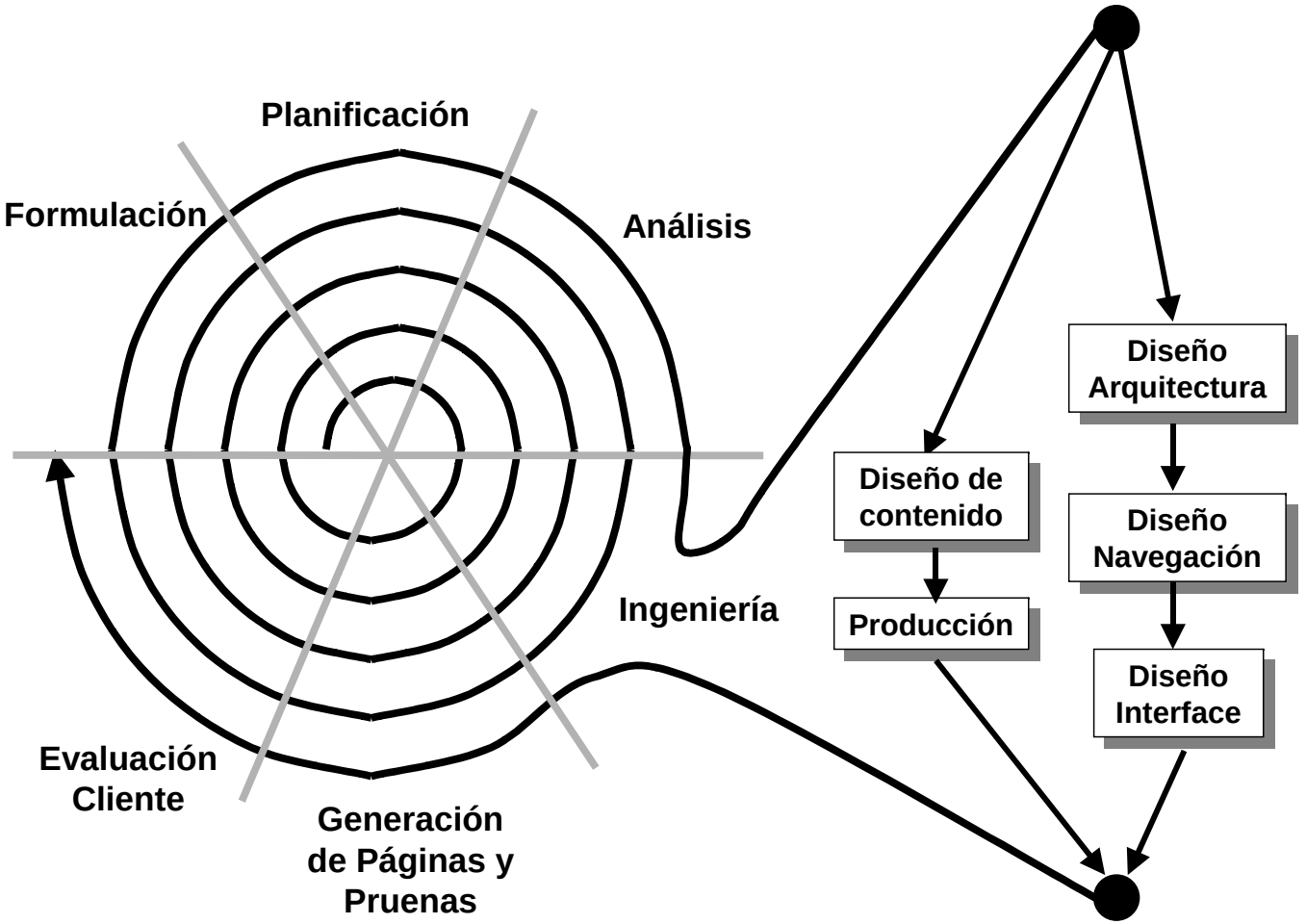
RUP (Rational Unified Process)



Component Model

- Es un modelo moderno, que incorpora las recomendaciones (buenas prácticas) de la ingeniería de software.
- Tiene mucho respaldo, pero es un tanto complejo (se requiere entrenamiento par usarlo).
- Está muy atado a UML (Unified Modeling Language).
- Hasta el momento, sólo ha demostrado ser uno más.
- A diferencia del resto, RUP considera la arquitectura como una pieza clave del desarrollo.
- No es demasiado aplicable a proyectos Web.

WebE (Web Engineering)



WebE (Web Engineering)

- Es un modelo moderno, aplicable a proyectos Web.
- Definido por R. Pressman en 1998.
- Es el primer modelo diseñado específicamente para desarrollar proyectos Web.
- Aunque hay otros como RMM, y OOHDM que proponen alternativas a WebE, pero que vienen del lado de los sistemas hipermediales.
- WebE aún no ha sido probado lo suficiente, y por ahora es sólo una buena propuesta.

Conclusiones

- El desarrollo de software debe ser guiado por un modelo, como forma de disciplinar, organizar y gerenciar las actividades.
- El modelo debe ser definido por la organización y adaptado a cada proyecto en particular.
- Las actividades que deben cumplirse en el proceso de desarrollo son básicamente las que establece el modelo en cascada.

Conclusiones (cont...)

- El modelo debe ser lo suficientemente flexible como para incorporar el principio de ANTICIPACION AL CAMBIO.
- En lo posible debe utilizarse un desarrollo incremental, con entregas parciales al usuario (modelo evolutivo, incremental o espiral).
- Para los productos que así lo ameriten, realizar análisis de riesgo (modelo espiral)