



Dpto. Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur

ELEMENTOS DE BASES DE DATOS

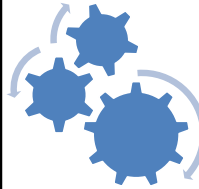
Segundo Cuatrimestre 2013

Clase 26: Ingeniería de Software – Proceso de Desarrollo

Mg. María Mercedes Vitturini
[mvitturi@cs.uns.edu.ar]



El proceso de desarrollo de software



El objetivo de un proceso de
producción es garantizar
productos confiables,
predecibles y eficientes.

El Proceso de Desarrollo

- El **proceso** define un *marco de trabajo* para un conjunto de tareas claves en la producción de software.

Ingeniería de Software [IEEE93]

1. La aplicación de un *enfoque sistemático, disciplinado y cuantificable* al desarrollo, operación y mantenimiento del software, esto es, **aplicación de ingeniería al software**.
2. Estudio de enfoques que permitan alcanzar 1.

EBD2013_26 - Mg. Mercedes Vitturini

El proceso

- El **proceso** forma la base para *el control de la gestión de los proyectos de software*.
- Establece el contexto en el cual se aplican:
 - los métodos, técnicas y metodologías,
 - se generan los productos de trabajo (modelos, documentos, datos, código, etc.)
 - se asegura calidad,
 - el cambio se puede manejar.

EBD2013_26 - Mg. Mercedes Vitturini

Ciclo de Vida

Ciclo de vida de un sistema – comprende desde la concepción de la idea de un desarrollo del software, hasta que es implementado, entregado, y aún después hasta que deje de usarse.

- Cada sistema tiene un *ciclo de vida* compuesto por varias etapas.
- Existen varios patrones o *modelos de procesos*, el ciclo de vida de un sistema puede estar conducido por alguno de ellos.

EBD2013_26 - Mg. Mercedes Vitturini

Ciclo de Vida del Sistema



ISO/IEC 12207
Information Technology / Software Life Cycle Processes

“Un marco de referencia que contiene los procesos, las actividades y las tareas involucradas en el desarrollo, la explotación y el mantenimiento de un producto de software, abarcando la vida del sistema desde la definición de los requisitos hasta la finalización de su uso”

EBD2013_26 - Mg. Mercedes Vitturini

El primer modelo de proceso de desarrollo

Code-Fix

- Fue la primera aproximación de desarrollo.
- No es un modelo de proceso.

Características

- El desarrollo de software era una tarea unipersonal.
- El problema era sencillo y bien entendido.
- Los lenguajes de programación bajo nivel.

EBD2013_26 - Mg. Mercedes Vitturini

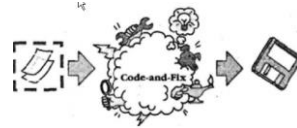
Code & Fix

• Actividades

- Escribir código
- Corregirlo (mejorar o agregar funcionalidades).

• Observaciones

- Después de algunas evoluciones se torna no confiable.
- Posible en aplicaciones cortas y bien entendidas.



Hitos

Cambios importantes:

- Se separan los roles *programador* y *usuario*.
 - El producto no satisface las necesidades del usuario.
- Necesidad de *aplicaciones más complejas*.
 - Se requiere mayor calidad.
- El *desarrollo* se convierte en una *actividad grupal*.
 - Se necesitan modelos de comunicación.



EBD2013_26 - Mg. Mercedes Vitturini

Actividades

- Una **organización tradicional** para las actividades de un proceso de desarrollo (no excluyentes):
 1. Estudio de factibilidad
 2. Elicitación, análisis y especificación de requerimientos - *¿qué?*
 3. Diseño - *¿cómo?*
 - de arquitectura
 - de programas.
 4. Codificación y testeo de módulos.
 5. Integración y testeo de sistemas.
 6. Distribución, entrega y mantenimiento.
 7. Otras actividades.

Ingeniería de Software -
C. Quezgy. Ed. 2002
Distintos autores usan
diferentes actividades

EBD2013_26 - Mg. Mercedes Vitturini

1- Estudio de Factibilidad

- En general para cualquier desarrollo la primera actividad es el *estudio de factibilidad*.
 - Se *evalúan costos y beneficios* del desarrollo.
 - Cuanto más conocimiento se tenga del problema mejor. Sin embargo, *no se le destina demasiado tiempo*.
 - Incluye:
 - Una definición general del problema.
 - Soluciones alternativas y beneficios esperados.
 - Recursos requeridos, costos, tiempos, fechas de entrega para cada solución alternativa.
- Según el marco de trabajo este estudio debe ser más o menos formal.

EBD2013_26 - Mg. Mercedes Vitturini

2 – Elicitación, análisis y especificación de requerimientos

- Identificar las **calidades** (necesidades) requeridas por la aplicación: funcionales y no funcionales.
- Identificar **qué?** y **cómo?**. La clave:

Separar, abstraer y particionar

- Descripción en **distintos niveles de abstracción**.
 - Separar en partes analizables.
 - Separar en distintas visiones del software:
 - Datos - funciones - restricciones - control (comportamiento)

EBD2013_26 - Mg. Mercedes Vitturini

3 – Definición de arquitectura y diseño detallado del sistema

- **Diseño de arquitectura**
 - Es de alto nivel.
 - Partición de funciones entre clientes y servidor.
- **Diseño de módulos**
 - Descomposición modular.
 - Definición de componentes, interfaces e interconexiones.
- **Documentar!**

EBD2013_26 - Mg. Mercedes Vitturini

4 – Codificación y Testeo de Módulos

- Se **escriben programas** en un lenguaje de programación.
- Una buena práctica es **definir y adoptar convenciones** de programación y testeo.
- Puede incluir revisiones de código (debugging y testeo y chequeos de performance).
- **Resultado:** módulos implementados y testeados.

EBD2013_26 - Mg. Mercedes Vitturini

5- Integración y Testeo de Sistema

- Se **ensamblan los componentes** desarrollados y testeados separadamente.
- En la etapa final se puede **incluir testeos con:**
 - Datos reales.
 - Clientes reales (alpha test).
- Aplicar estándares (*top down* o *bottom-up*).
- **Documentar** los casos de prueba.

EBD2013_26 - Mg. Mercedes Vitturini

6 – Distribución, entrega y mantenimiento de sistema

- Cuando se completa el desarrollo de la aplicación.
- Primero se entrega a un **grupo reducido de clientes** antes de la entrega oficial (**beta testing**).
 - Objetivos: experimentación y retroalimentación.
- Pueden hacerse cambios antes de la entrega oficial.
- El **mantenimiento** incluye las actividades realizadas una vez que el producto es entregado al cliente.

EBD2013_26 - Mg. Mercedes Vitturini

Mantenimiento

- El costo de mantenimiento en muchos casos representa más del 60% del costo del producto.
 - Corregir errores encontrados después de la entrega (**mantenimiento correctivo**)
 - Adaptar la aplicación a cambios de entorno (**mantenimiento adaptativo**)
 - Cambiar o agregar funciones o calidades del sistema (**mantenimiento perfectivo**)
- El 50% de los costos de mantenimiento se atribuyen a cambios en los requerimientos.

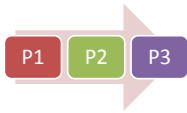
EBD2013_26 - Mg. Mercedes Vitturini

7 – Otras actividades

- Dependiendo del tipo de desarrollo puede ser necesario incorporar nuevas actividades:
 - Documentación.
 - Control de calidad
 - Walk-through.
 - Validación.
 - Verificación.
 - Administración
 - Definición de estándares.
 - Tratamiento con recursos (personas).

EBD2013_26 - Mg. Mercedes Vitturini

Modelos de Procesos Genéricos



Un modelo de proceso provee una representación abstracta del proceso de desarrollo de software

Modelos de Procesos

- Vamos a estudiar algunos *modelos de procesos* clásicos para la construcción de software.
- Los modelos de procesos *evolucionan* en la medida que evoluciona la *tecnología y los requerimientos a satisfacer*.
- Existen cientos de modelos, la mayoría son variaciones de otros
 - Cascada.
 - Evolutivos.
 - Transformacionales.

TIP! Cada proyecto particular define su propio ciclo de vida

EBD2013_26 - Mg. Mercedes Vitturini

Modelo de CASCADA (MC)

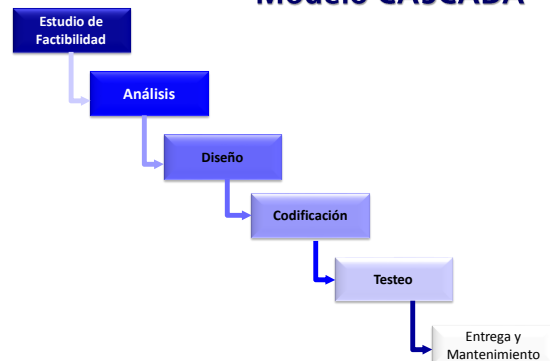
- Las tareas o actividades de desarrollo se organizan en **cascada**, una después de la otra.
- La **salida (output)** de una etapa es la **entrada (input)** de la siguiente etapa.
 - Las etapas se pueden dividir en actividades ejecutadas concurrentemente.

Origen

- Creado en los años '60 en un proyecto de defensa de EEUU.
- Se usó para las primeras aplicaciones es crítica y compleja.

EBD2013_26 - Mg. Mercedes Vitturini

Modelo CASCADA



EBD2013_26 - Mg. Mercedes Vitturini

Modelo Cascada

Características:

- Es un *modelo secuencial*. Cada fase debe estar terminada antes de pasar a la siguiente.
- La definición de las **actividades depende del tipo de proyecto y la relación equipo de desarrollo cliente**.
- Cada actividad **produce un documento de salida** que es el ingreso para la siguiente actividad.
- Las empresas que usan este modelo de proceso deben definir

EBD2013_26 - Mg. Mercedes Vitturini

MC – Conclusiones

- Fue el primer modelo.
- Es un modelo *conducido por la documentación*.

Contribuciones:

- Introduce *disciplina* al proceso.
- *Postpone la implementación* hasta que los objetivos estén claros.
- Es el paradigma más antiguo y más extensamente usado en ingeniería de software.
- *Impone puntos de control* claros.

EBD2013_26 - Mg. Mercedes Vitturini

MC – Conclusiones

Desventajas:

- Demasiado rígido.
- Retrasa la detección de problemas críticos.
- Difícil la estimación de recursos.
- Es lineal y los proyectos reales raras veces siguen un modelo lineal.
- El usuario interviene al principio y al final del proyecto.
- No provee anticipación al cambio.
- Basado en documentación. Parece burocrático.
- Los costos se trasladan al mantenimiento fácilmente.
- No permite implementaciones parciales.

EBD2013_26 - Mg. Mercedes Vitturini

El paradigma evolutivo

Motivación – las fallas de la primer versión de una aplicación conducen a la necesidad de rehacerla.

HACERLO DOS VECES.

- **Problema** – *gap* entre la definición de requerimientos y la distribución de la aplicación.
- **Estrategia** –
 - Entregar algo y medir el valor agregado.
 - Ajustar diseño y objetivos.
 - Iterar: desarrollo de versiones cada vez más completas del software.

EBD2013_26 - Mg. Mercedes Vitturini

Modelos Evolutivos

- Proceso evolutivos:
 - Prototipo
 - Iterativo incremental.
 - Espiral



EBD2013_26 - Mg. Mercedes Vitturini

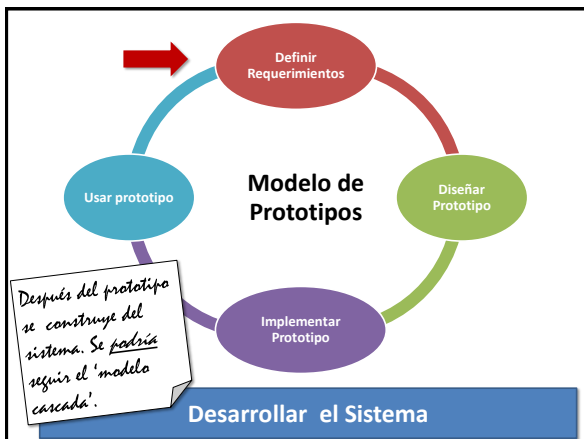
Modelo de PROTOTIPO (MP)

- Primer desarrollo: elaborar *un prototipo para descartar*. El prototipo sirve para validar requerimientos y resolver aspectos críticos del diseño.

Objetivos

- Investigación repetida de requerimientos y diseño.
- **Reducir el riesgo** que la aplicación no se ajuste a las necesidades del cliente.
- La revisión del prototipo puede resultar en revisar los requerimientos anteriores.

EBD2013_26 - Mg. Mercedes Vitturini



Modelo de Prototipos

• Actividades

- **Recolección requerimientos del sistema:** desarrollador y cliente definen los objetivos globales.
- Se **elabora un diseño rápido**, centrado en los aspectos del software que son **visibles a los usuarios** (enfoques de entrada y formatos de salida).

Desventajas

- El cliente ve *el prototipo* y lo *confunde con el sistema real*.
- Se toman **decisiones rápidas** para poder construir el prototipo, que **son difíciles de revertir** (por ejemplo el lenguaje de programación).
- El prototipo para descartar *nunca se descarta*.

EBD2013_26 - Mg. Mercedes Vitturini

Modelo Iterativo Incremental

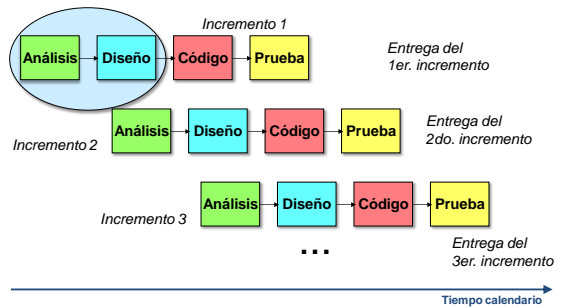
Iterativo – hacemos varias veces lo mismo.

Incremental – el producto se “incrementa” a medida que se avanza. También se los llamados “evolutivos”.

- Consisten en de *incrementos expandidos de un producto de software operativo*, conducidos por la evolución determinada según la experiencia operativa.
- Los incrementos se distribuyen a medida que se completan.
- **Incremento integrado**: es una unidad de software auto-contenida que realiza algún propósito útil.

EBD2013_26 - Mg. Mercedes Vitturini

Modelo Iterativo Incremental



EBD2013_26 - Mg. Mercedes Vitturini

Ejemplo

- Un ejemplo de desarrollo siguiendo el modelo iterativo incremental para un procesador de textos:
 - **Incremento #1:** administración de archivos básicos y producción de documentos.
 - **Incremento #2:** funciones de edición más sofisticadas (gráficos, tablas, etc.)
 - **Incremento #3:** corrección gramatical y ortográfica.
- Al primer incremento se lo *denomina producto esencial*.

EBD2013_26 - Mg. Mercedes Vitturini

Paradigma Iterativo Incremental

Ventajas

- El usuario ve algo rápidamente.
- Se admite que lo que se está construyendo es el sistema, y por lo tanto se piensa en su calidad desde el principio.
- Los ciclos van mejorando con las experiencias de los anteriores.
- Los trabajadores del equipo de desarrollo se especializan en ciertas actividades.

Desventajas

- Es difícil mantener el foco. Los incrementos entran en mantenimiento mientras se está desarrollando nuevos.
- Es difícil seguir procesos puramente iterativos incrementales.

EBD2013_26 - Mg. Mercedes Vitturini

Modelo Espiral (MEs)

- El modelo *combina las actividades del desarrollo con la gestión del riesgo*, para minimizar y controlar el riesgo.

- **Riesgo:** circunstancias potencialmente adversas que pueden perjudicar el proceso de desarrollo y la calidad de los productos.
- **Administración del riesgo:** disciplina cuyos objetivos son manejar y eliminar los ítems de riesgo del software antes que se transformen en amenaza.
- Es cíclico.
- Es un meta-modelo.
- Estrategia: comenzar por los desarrollos de mayor riesgo.

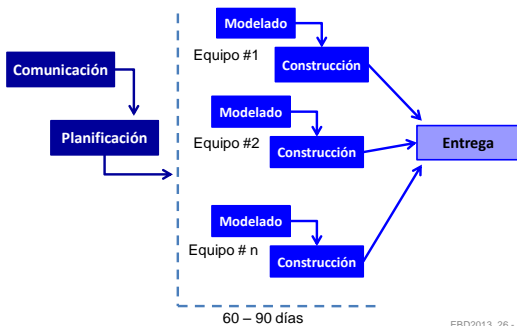
EBD2013_26 - Mg. Mercedes Vitturini

Desarrollo Rápido de Aplicaciones (DRA)

- Es un modelo de proceso que apunta a proyectos rápidos que finalizan **en 60-90 días**.
- Utiliza un enfoque de construcción **basado en componentes**.
- Requiere muchos recursos humanos como para crear varios **equipos DRA**.
- Desarrolladores y clientes deben estar **comprometidos** en las actividades necesarias para completar un sistema en un **marco de tiempo abreviado**.

EBD2013_26 - Mg. Mercedes Vitturini

Desarrollo Rápido de Aplicaciones



DRA

Desventajas

- Tiene requerimientos fuertes de personal de desarrollo.
- No es un modelo adecuado para sistemas que requieren de muchas actividades de mantenimiento.
- No es adecuado cuando los riesgos técnicos son altos.
- No funciona si no existe un compromiso real del cliente.

EBD2013_26 - Mg. Mercedes Vitturini

Generalidades

¿Cuál es la importancia de los modelos de proceso?

- Organizan la **tarea de desarrollo** de software.
- Dan el marco para la **planificación y control del desarrollo**.
- Permiten **asegurar calidad** en el producto.
- Dan **visibilidad** al proceso de desarrollo

EBD2013_26 - Mg. Mercedes Vitturini

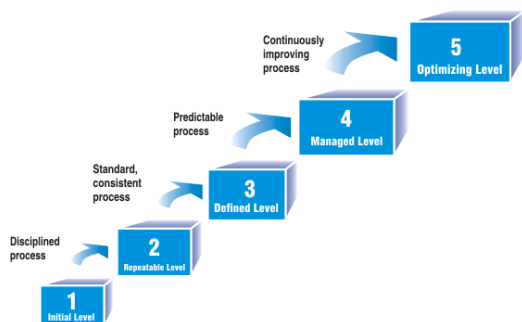
Madurez en Desarrollo de SW

CMMI– Modelo integrado de Capacidad y Madurez

- El desarrollo de sistemas en una organización madura **disminuye costos y tiempo de producción** y **aumenta la productividad y la calidad del producto**.
- El **Modelo de Integrado de Capacidad y Madurez (CMMI)**, Desarrollados por Software Engineering Institute (SEI) provee un estándar de referencia para evaluar el nivel de madurez del desarrollo de sistema de una organización.

EBD2013_26 - Mg. Mercedes Vitturini

Nivel de Madurez



Temas de la Clase de hoy

- Ciclo de vida
- El modelos de proceso
 - Cascada.
 - Evolutivos: Prototipo, incremental, espiral.
 - Desarrollo rápido de aplicaciones.

Bibliografía

- *Fundamentals of Software Engineering* – C. Ghezzy. Capítulo 7.
- *Ingeniería del Software – Un enfoque Práctico*. R. Pressman. Capítulo 3.

EBD2013_26 - Mg. Mercedes Vitturini

Próximas fechas de examen y coloquio

- **Fechas de examen y coloquio previstas: 4-12, 12-12 y 23-12** horario y aula a confirmar.
- **Coloquios: inscribirse a la mesa y notificar 48 hs. antes por mail a la cuenta mvitturi@cs.uns.edu.ar** con asunto **Coloquio EBD.**
- Horario de consulta: lunes 15:00hs.
- ¿Aportes?

EBD2013_26 - Mg. Mercedes Vitturini



Dpto. Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur

Gracias!

FIN