Análisis y Diseño de Sistemas (3303)

Teoría 1:

- Introducción a la Ingeniería de Software
- Conceptos
- Características del Software
- Modelos de Desarrollo de Software
- Bibliografía Consultada

Miércoles 16 de marzo de 2016

Introducción

• En la ingeniería clásica existen técnicas formales y estándares que permiten resolver la mayoría de los problemas que se presentan.

• **El desarrollo de software es una actividad de ingeniería**, que presenta mayor dificultad para encontrar estándares.

El Software se caracteriza por:

- Es un elemento de un sistema lógico, es inmaterial
- Un pequeño error puede causar un gran efecto
- Se modifica fácilmente, Sufre permanentes cambios
- Es difícil de medir

✓ Se desarrolla con intelecto, no se manufactura

✓ No se desgasta

✓ Se construye principalmente para un uso individualizado

Grafico 2 Distribución por Grupos etarios

Donde se aplica Software ?

El software puede aplicarse a numerosas y diversas situaciones del mundo real

- ✓ problemas con un conjunto específico de acciones que lleven a su resolución,
- ✓ problemas que se describen formalmente indicando la solución deseada,
- ✓ problemas que los humanos resolvemos utilizando una multitud de reglas heurísticas posiblemente contradictorias,
- ✓ problemas de los cuales no se tiene una idea clara de cómo se resuelven, pero se conoce la solución apropiada para algunos ejemplos de los datos de entrada.

Software de base, Software de tiempo real, Software de gestión, Software científico y de ingeniería, Software de ordenadores personales, Software empotrado o embebido, Software de inteligencia artificial, Software Educativo, Sistemas Colaborativos, Workflow.....

Dominios de Aplicación del Software

Software de Sistemas

Conjunto de programas escritos para dar servicios a otros programas

Software de Aplicación

Programas que resuelven una necesidad específica de negocio

Software de Ingeniería

Aplicaciones para la astronomía, vulcanografía, biología molecular, simulación de sistemas. Sistemas de cálculo en gral.

Software Incrustado

Reside dentro de un elemento y sirve para implementar y controlar funciones para un usuario final y el sistema en si.

Dominios de Aplicación del Software

Software de Línea de productos

Proporciona una capacidad específica para uso de muchos consumidores diferentes. Dirigido a un mercado particular (Control de Inventario) o a un mercado masivo (hoja de cálculo)

Aplicaciones Web WebApps

Conjunto de archivos de hipertexto vinculados. También pueden integrarse con bases de datos corporativas y aplicaciones de negocios

Software de Inteligencia Artificial

Algoritmos no numéricos para resolver problemas complejos. Robótica, sistemas expertos, reconocimiento de patrones (imagen y voz), redes neurales artificiales, juegos.

La necesidad del Análisis y Diseño de Sistemas

- El análisis y diseño de sistemas tiene el propósito de analizar sistemáticamente el flujo de datos de entrada, procesarlos y producir la salida de información esperada.

- El análisis de sistemas estudia, diseña e implementa mejoras para dar soluciones a un problema planteado.

- La implementación de un sistema sin una planificación adecuada lleva a grandes fracasos.

La necesidad del Análisis y Diseño de Sistemas

Como cualquier disciplina de la ingeniería,

la **ingeniería de software** requiere de compromiso,

y el ingeniero de software debe tomar decisiones

correctas respecto a los recursos que utilizará, en cuanto

a software, a hardware, a personas, a herramientas, y

dependiendo de cada problema particular.

Introducción

La Ingeniería de Software es el campo de las Ciencias de la Computación que se dedica al estudio de la construcción de sistemas de software de gran tamaño, y que requieren de uno o más equipos de ingenieros de software para su desarrollo.

Programación e *Ingeniería de Software*.

- Un *programador* escribe programas completos.
- Un *ingeniero de software* construye componentes de software (CS) para ser combinados con otros CS, realizados por otros ingenieros o equipos de ingenieros. (Un CS puede ser usado, modificado y reusado)

Porqué surge la Ingeniería de Software?

✓ Los primeros programas se escribían en un lenguaje entendible por la computadora. Usuario + Máquina.

✓ Se evolucionó, el usuario daba su especificación al programador para que escribiera sus programas. Se obtenían buenos resultados.

El software se realizaba de forma artística (experiencia, ensayo-error)

✓ Con el surgimiento de proyectos cada vez más grandes y complejos, el avance del HW y SW, y su aplicación en diversos contextos, llevó a:

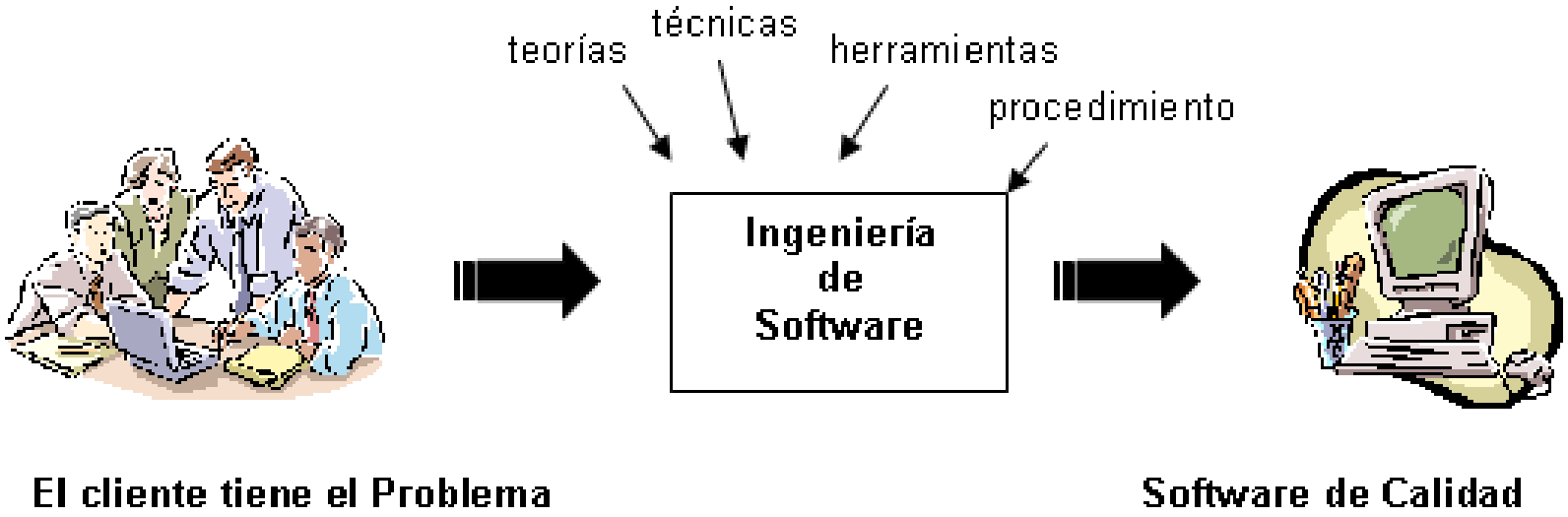
- software de baja calidad
- incumplimiento de tiempos y de presupuestos
- incrementos dramáticos en los costos de mantenimiento.

✓ A fines de los 60's se produce la denominada Crisis del Software.

✓ En los 70's surgen métodos estructurados (Jourdon, DeMarco).

✓ En los 80's y 90's el paradigma estructurado evolucionó hacia otros paradigmas, como los orientados a objetos (RUP, Diseño por contratos).

Ingeniería de Software



*La **Ingeniería de Software** es una disciplina que se basa en un **Proceso**, y aplica **Métodos** y **Herramientas** al desarrollo de software, con el fin de producir **Productos** fiables y que funcionen eficientemente en máquinas reales con costos razonables.*

Conceptos- Desarrollo de Software

- **PROCESO o MODELO DE DESARROLLO DE SW:** conjunto de actividades y resultados asociados que producen un Producto de Software. Define un marco de trabajo (Pasos a seguir).

- **METODOS:** indican “cómo” construir técnicamente el SW.

- **HERRAMIENTAS:** brindan un soporte automático al Proceso y a los Métodos.

- **PRODUCTO DE SW:** sistema que se realiza para un usuario con la documentación correspondiente. Genérico o a medida.

- **ATRIBUTOS DEL PRODUCTO DE SW:** características que distinguen y describen al producto.

Capas de la Ingeniería de Software

El rol del Ingeniero de Software

Para pequeños o medianos proyectos de software **("programming-in-the-small")**

- Requiere educación y experiencia.
- Deber ser buen programador, independiente del lenguaje de programación, con sólida formación en estructura de datos y algoritmos.

El rol del Ingeniero de Software

Para proyectos de software más grandes y más complejos (**"programming-in-the-large"**), requerirá además:

- Conocer distintas métodos de modelado y diseño.
- Comprender y traducir los requerimientos del usuario en especificaciones precisas.
- Habilidad para definir e interpretar los distintos niveles de abstracción en los diferentes estados del proyecto.
- Construir modelos correctos, completos y precisos.
- Buena comunicación con los miembros del equipo.
- Capacitación p/ planificar tareas y manejar equipos de trabajo.
- En la actualidad, el IS obtiene una especialidad.

Características de Calidad del SW (I)

- **Exactitud (Correctness):** habilidad del SW para ejecutar sus tareas correctamente de acuerdo a la especificación.
- **Robustez (Robustness):** reacción apropiada a condiciones anormales. Complementa la exactitud. Normal y anormal relativos a la especificación.
- **Extensibilidad (Extensibility)**: facilidad del software de adaptar los cambios a la especificación.
- **Reusabilidad (Reusability):** habilidad de los elementos de software para ser usados en aplicaciones diferentes.
- **Compatibilidad (Compatibility):** facilidad de combinación entre elementos de software.

Características de Calidad del SW (II)

- **Eficiencia (Efficiency):** habilidad de un sistema para hacer la menor cantidad de demandas posibles a los recursos de hardware, como el tiempo de procesador, la cantidad de memoria interna y externa.

- **Portabilidad (Portability):** facilidad de transferencia del software.

- **Facilidad de uso (Ease to use):** facilidad con la que las personas pueden usar un producto de software y aplicarlo para resolver problemas. Facilidad de instalación.

- **Funcionalidad (Functionality):** es la extensión de las posibilidades que provee un sistema.

- **Oportuno (Timeliness):** habilidad del software de estar resuelto cuando (o antes) se acordó con los usuarios. Es una de las grandes frustraciones de la industria del software.

Características de Calidad del SW (III)

- **Verificable (Verifiability):** procedimientos de aceptación, prueba de datos y procedimientos para detectar fallas.
- **Integridad (Integrity):** proteger a sus programas y datos de accesos y modificaciones no autorizadas.
- **Reparable (Repairability):** reparar defectos con facilidad.
- **Economía (Economy):** habilidad de ser completado de acuerdo al presupuesto pactado. Relacionado con el tiempo.
- **Documentación**: para que los usuarios comprendan las funcionalidades del sistema, para que los desarrolladores comprendan la estructura e implementación de un sistema.
- **Acerca del Mantenimiento del Software:** modificaciones por cambios en requerimientos del usuario, formato de los datos, documentación, Hardware, cuestiones externas, corrección de errores.

Modelos de Proceso

Actividades estructurales

Comunicación

Planeación

Modelado

Construcción

Despliegue

Actividades Sombrilla

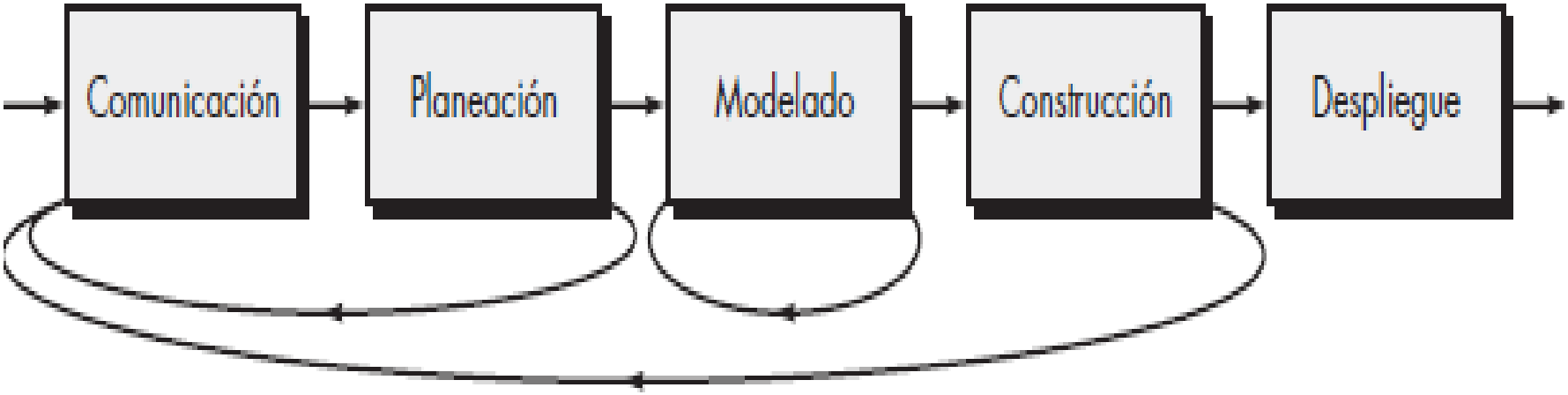
Seguimiento y control del proyecto, Administración de Riesgos, Aseguramiento de la Calidad, Medición, Gestión de Cambios, Administrar la reutilización, Revisiones Técnicas

La Práctica de la Ingeniería de Software

1. **Entender el problema (comunicación y análisis)**
   - *¿Quiénes son los participantes?*
   - *¿Cuáles datos, funciones y características se requieren para resolverlo?*
   - *¿Puede fraccionarse? ¿Es posible representarlo gráficamente?*
2. **Planear la solución (modelado y diseño del software)**
   - *¿Ha resuelto problemas similares? ¿son reutilizables sus elementos?*
   - *¿Algún software existente implementa características, datos y funciones que se requieren? ¿Pueden definirse problemas más pequeños?*
3. **Ejecutar el plan (generación del código)**
   - *¿El código fuente se corresponde con el modelo del diseño?*
   - *¿Cada parte componente de la solución es correcta? ¿El diseño y código se han revisado?*
4. **Examinar la exactitud del resultado (probar y asegurar la calidad)**
   - *¿Puede probarse cada parte componente de la solución? ¿Se ha implementado una estrategia* razonable para hacer pruebas?
   - *¿La solución produce resultados respecto de los datos, funciones y características que se requieren? ¿El software se ha validado contra todos los requerimientos de los participantes?*

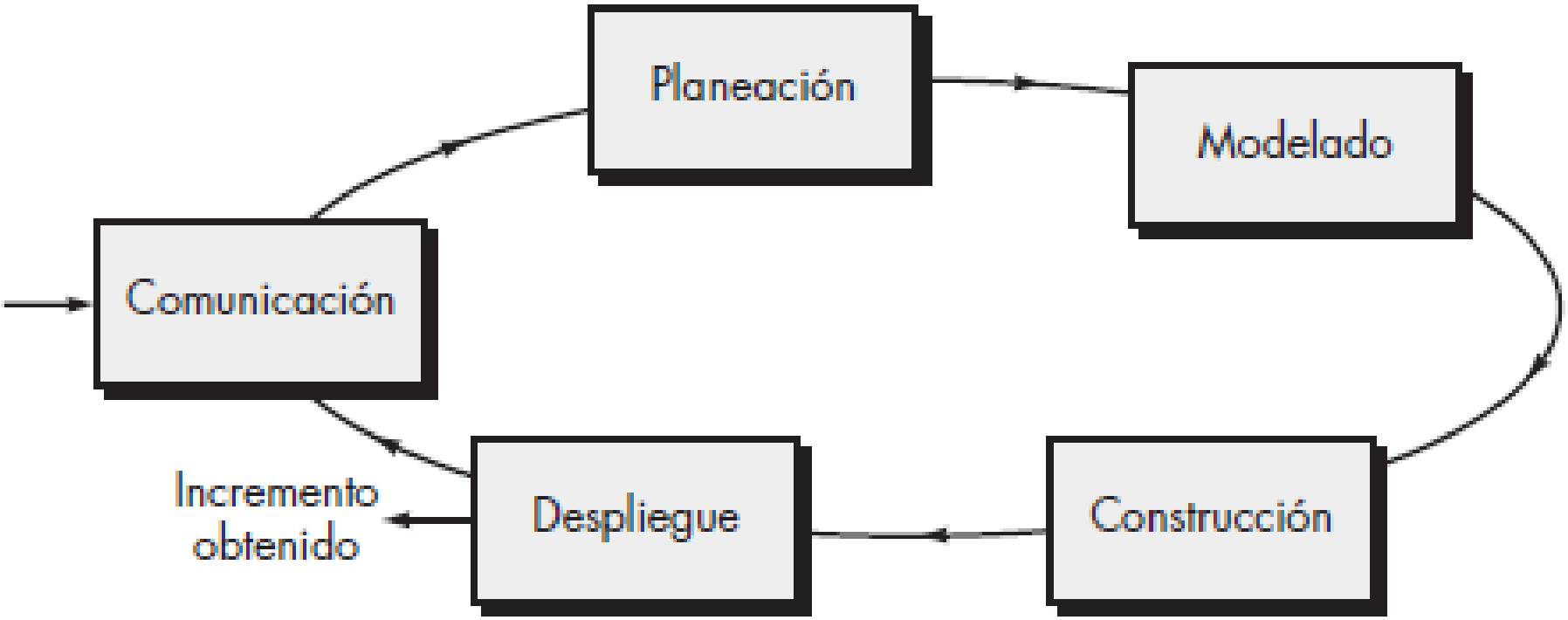
Flujos de Procesö: pasos a seguir

a) Flujo de proceso lineal



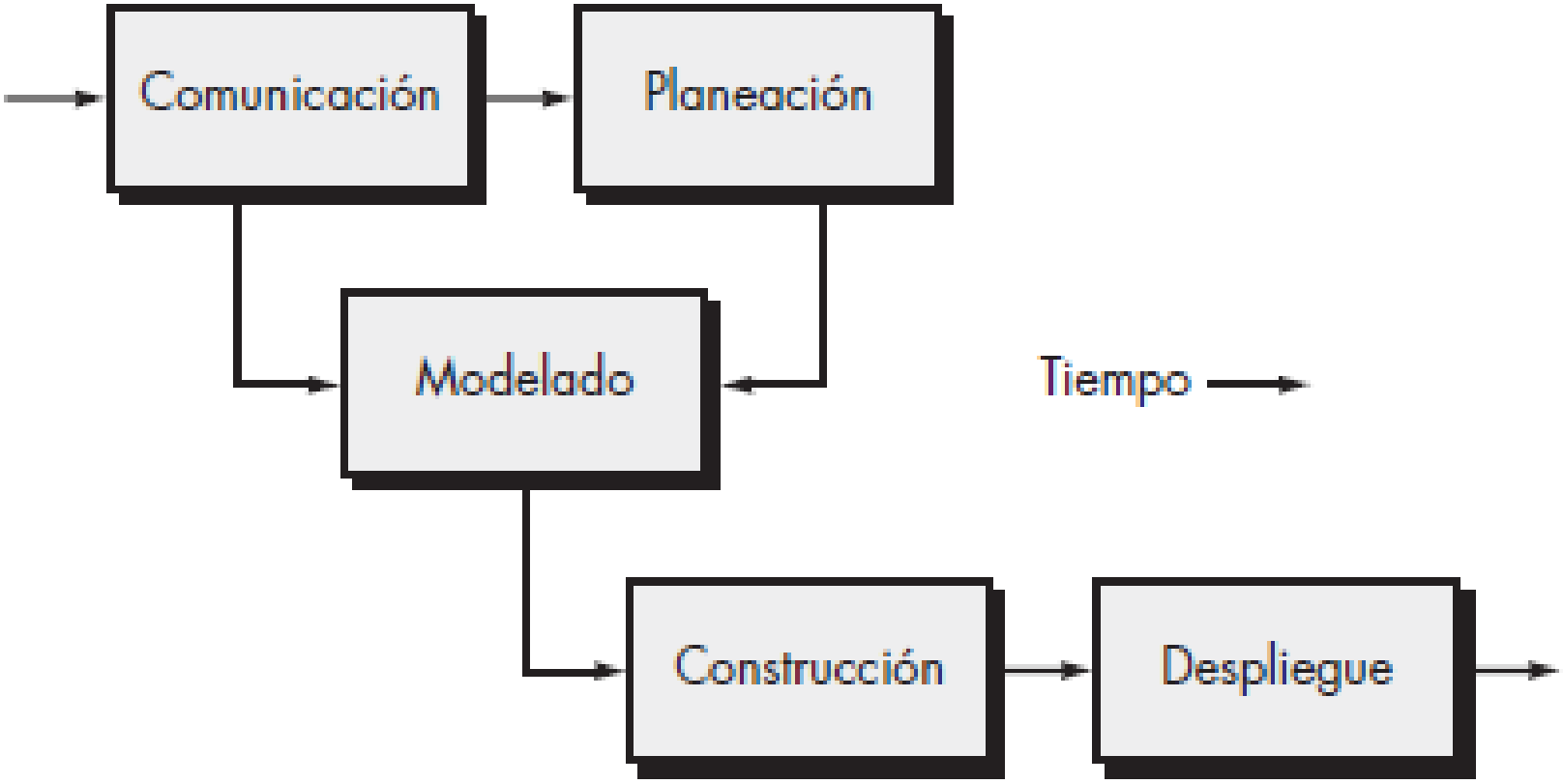
b) Flujo de proceso iterativo

Flujos de Proceso



c) Flujo de proceso evolutivo

Flujos de Proceso



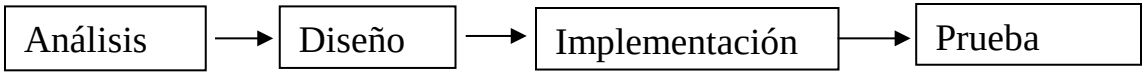
d) Flujo de proceso paralelo

Modelos o Procesos para el desarrollo de SW

- MODELO LINEAL SECUENCIAL O EN CASCADA.
- MODELO DE CONSTRUCCION DE PROTOTIPOS.
- MODELO EN ESPIRAL.
- ENSAMBLAJE DE COMPONENTES.
- MODELO DE LOS METODOS FORMALES.
- DISEÑO POR CONTRATOS.
- El PROCESO UNIFICADO
- SCRUM

MODELO LINEAL SECUENCIAL O EN CASCADA

- Enfoque sistemático.
- Se Presenta con actividades por separado:
  1. Especificación de requerimientos – Análisis
  2. Diseño
  3. Implementación – Código
  4. Testeo - Prueba



- Es el paradigma más antiguo y más utilizado.
- Es útil cuando deben hacerse adaptaciones o mejoras bien definidas a un sistema ya existente.

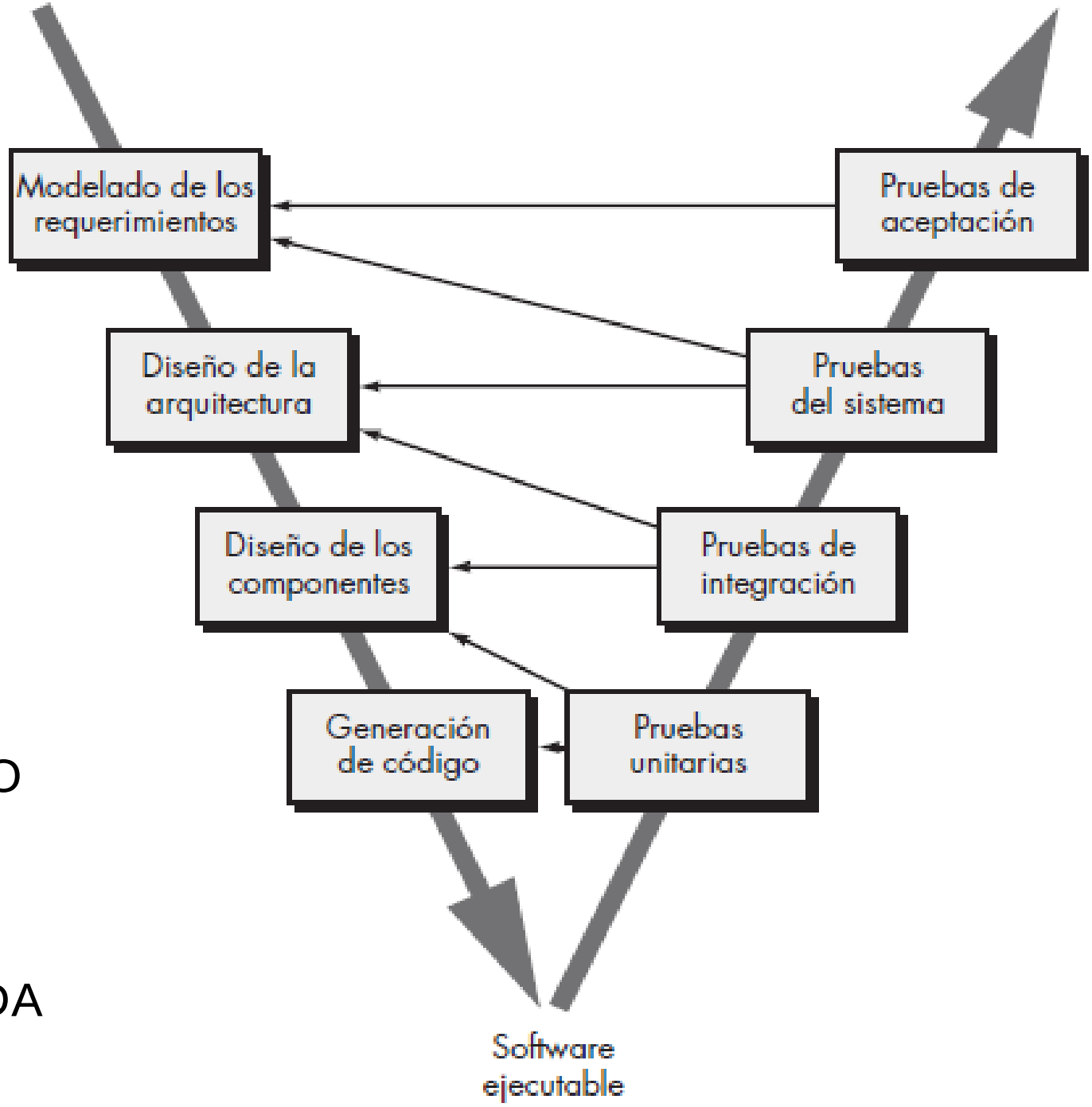
PROBLEMAS:

- Los proyectos rara vez siguen un modelo secuencial.
- Es difícil que el cliente exponga todas las necesidades al comienzo.
- Se requiere de mucha paciencia del cliente.
- Un error detectado en el programa podría ser catastrófico.
- Es difícil respetar los tiempos.

MODELO

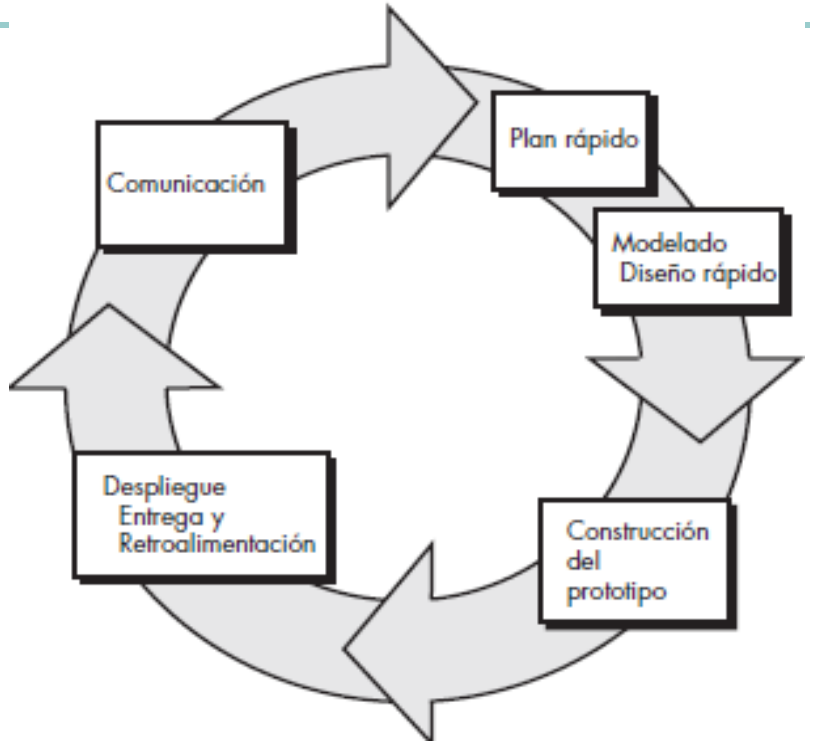
EN

CASCADA



MODELO DE CONSTRUCCION DE PROTOTIPOS

Se hace una versión rápida, se implementa, y luego se va aplicando refinamiento hasta llegar a un sistema robusto y mantenible.
Las etapas se van entrelazando.

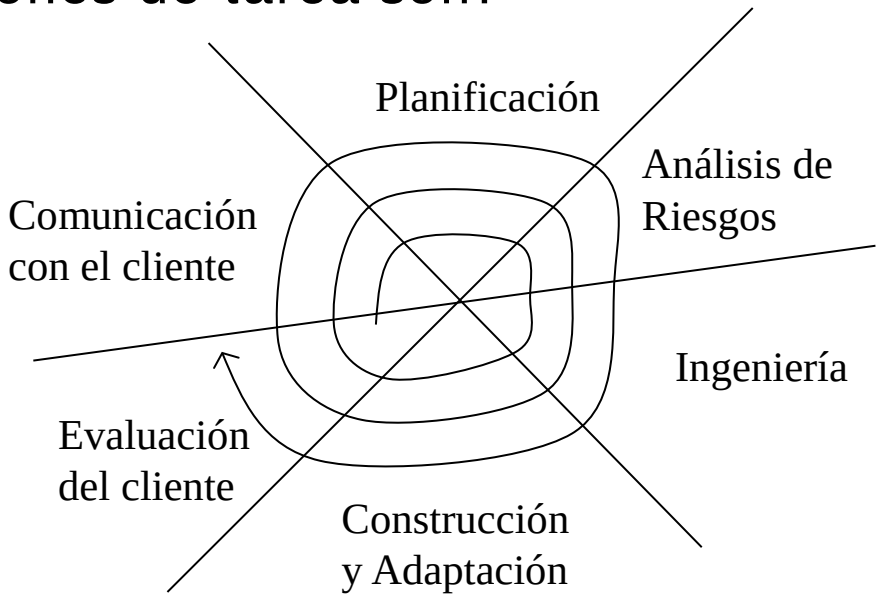


PROBLEMAS:

- El cliente pide pequeños ajustes al prototipo y quiere comenzar a operar inmediatamente con el sistema.
- Para el desarrollador, hacer que el sistema funcione rápidamente lo puede llevar a realizar malas selecciones.
- Aunque desechar el prototipo en algunas es imposible, puede servir cuando es un proceso.

MODELO EN ESPIRAL

- Los espacios de trabajo que componen el proceso de desarrollo lineal se ejecutan de manera iterativa.
- En cada iteración se produce una nueva versión más compleja y completa del sistema.
- Es un modelo de proceso de software evolutivo.
- Los espacios de trabajo o regiones de tarea son:
  1. Comunicación con el cliente
  2. Planificación
  3. Análisis de riesgos
  4. Ingeniería
  5. Construcción y adaptación
  6. Evaluación del Cliente



PROBLEMAS:

- Es un método poco utilizado.
- Difícil convencer al cliente de un desarrollo evolutivo y controlable.
- Requiere de mucha habilidad para descubrir riesgos potenciales.

ENSAMBLAJE DE COMPONENTES

- Basado en la Tecnología de Objetos.
- Es un proceso evolutivo que también usa una espiral.
  - Se crean clases que incluyen tanto datos como los algoritmos para manejar esos datos.
  - Las clases creadas se almacenan en un repositorio o depósito o biblioteca de clases o componentes.
  - En base a los requerimientos de los usuarios, se seleccionan los componentes del repositorio y se ensamblan entre si para producir el sistema final.

PROBLEMAS:

- Complejidad para construir y mantener el repositorio.
- Los componentes deben ser catalogados y clasificados con criterios bien definidos para su organización.

MODELO DE LOS METODOS FORMALES

Se produce una especificación matemática formal del sistema y se van aplicando transformaciones a esa especificación preservando la correctitud.

PROBLEMAS:

- Es un proceso muy caro
- Consume mucho tiempo
- Requiere de profesionales expertos
- No favorece la comunicación con el cliente

- Los métodos formales se utilizan en combinación con otros métodos.

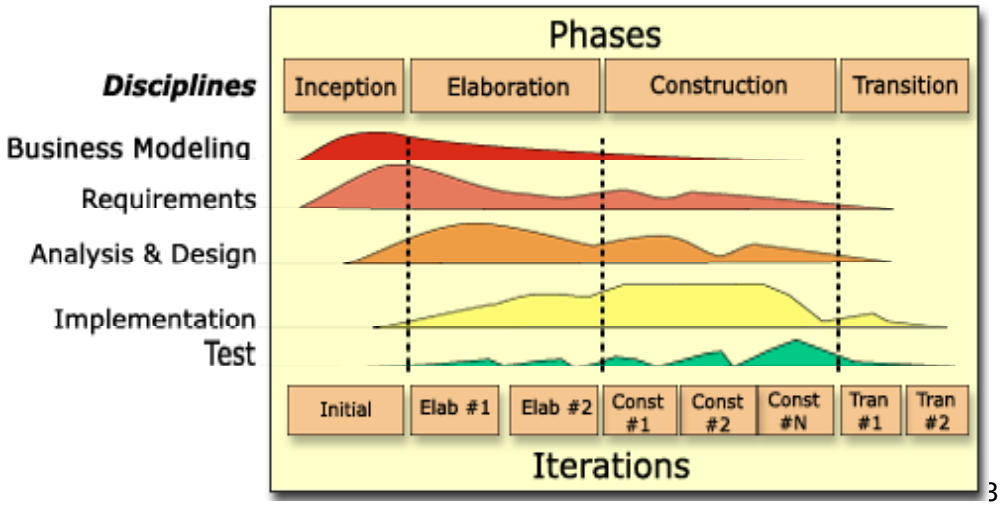
- En sistemas con funcionalidades críticas son una alternativa que asegura un producto de calidad.

DISEÑO POR CONTRATOS (Meyer)

- Metodología de diseño de software OO, basada en *Contratos*.
- El diseño por *Contratos*, tiene sus raíces en los métodos formales para la construcción de software.
- El sistema es visto como un conjunto de *elementos de software* que cooperan entre si.
- Los elementos juegan un rol principal: *Proveedores* o *Clientes*.
- La cooperación establece claramente *obligaciones* y *beneficios*, y su especificación son los *Contratos*.
- Los *Contratos* especifican expresiones lógicas o *aserciones*.
  - Precondición: requisitos para que una rutina sea aplicable.
  - Poscondición: propiedades garantizadas a la salida de una rutina.
  - Invariante de Clase: expresa las restricciones semánticas de la clase. Se añade implícitamente a cada pre o poscondición.
  - Pre + Poscondición = Contrato entre la clase y sus clientes.

PROCESO UNIFICADO

- Es una metodología de desarrollo de software OO.
- **Basado en Casos de Uso**
  - Caso de Uso: funcionalidad del sistema que da un resultado de valor para un usuario.
- **Centrado en la Arquitectura**
  - Arquitectura: forma del sistema. Diferentes vistas en las distintas etapas del proyecto.
- **Iterativo e Incremental**
  - Iteración: división del proyecto en miniproyectos. Pasos en el flujo de trabajo.
  - Incremento: crecimiento del producto.



Metodologías Tradicionales vs Metodologías Ágiles

- Las **metodologías tradicionales** se centran especialmente en el control del proceso, mediante una rigurosa definición de roles, actividades, artefactos, herramientas y notaciones para el modelado y documentación detallada. Efectivas y necesarias en proyectos grandes.
- Las **metodologías ágiles**, dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas. Poca documentación.

  XP (eXtreme Programming), SCRUM, Kanbam, Crystal Methodologies, ASD (Adaptive Software Development, FDD (Feature-Driven Development), DSDM (Dynamic Systems Development Method).

Bibliografía Consultada

- Capítulo 1:“Fundamentals of Software Engineering”. Carlo Ghezzi.
- Capítulo 1, 2: “Ingeniería de Software: Un Enfoque Práctico”, 7ma. Edición. Roger Pressman.
- Capítulo 1: “Object Oriented Software Construction”. Bertrand Meyer.
- www.agilemanifesto.org - http://www.agile-spain.com/