Diseño Orientado al Objeto

Agustín J. González

ELO-326: Seminario de Computadores II

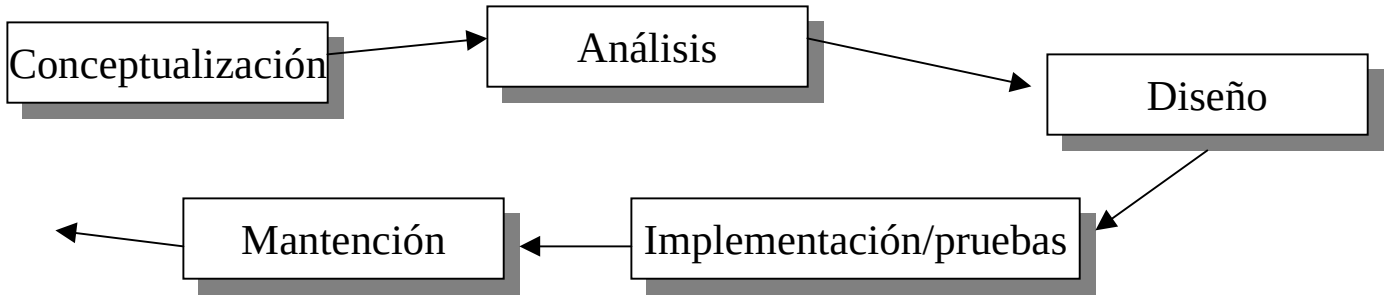
2do. Sem. 2001

Generalidades

- ¿Qué es la programación orientada al objeto?. La respuesta normalmente apunta a enfatizar características de los lenguajes como C++ en comparación con C, por ejemplo. Se habla de clases, herencia, paso de mensajes y métodos virtuales y estáticos.
- Un aspecto de de gran importancia (tal vez el más importantes) en OOP es una técnica de diseño, la cual se caracteriza por la determinación y delegación de responsabilidades.
- Responsabilidad implica no interferencia. Cuando un objeto se hace responsable por acciones específicas, esperamos cierto comportamiento.
- Programación en pequeño versus en grande:
  Pequeño: Una sola persona o un pequeño grupo. Una persona puede comprender todo.
  El mayor problema es el diseño y el desarrollo de algoritmos.
  Grande: Hay un gran equipo de personas. Los diseñadores pueden ser distintas personas a los implementadores.
  El mayor problema es la administración de los detalles y la comunicación entre las partes del proyecto.

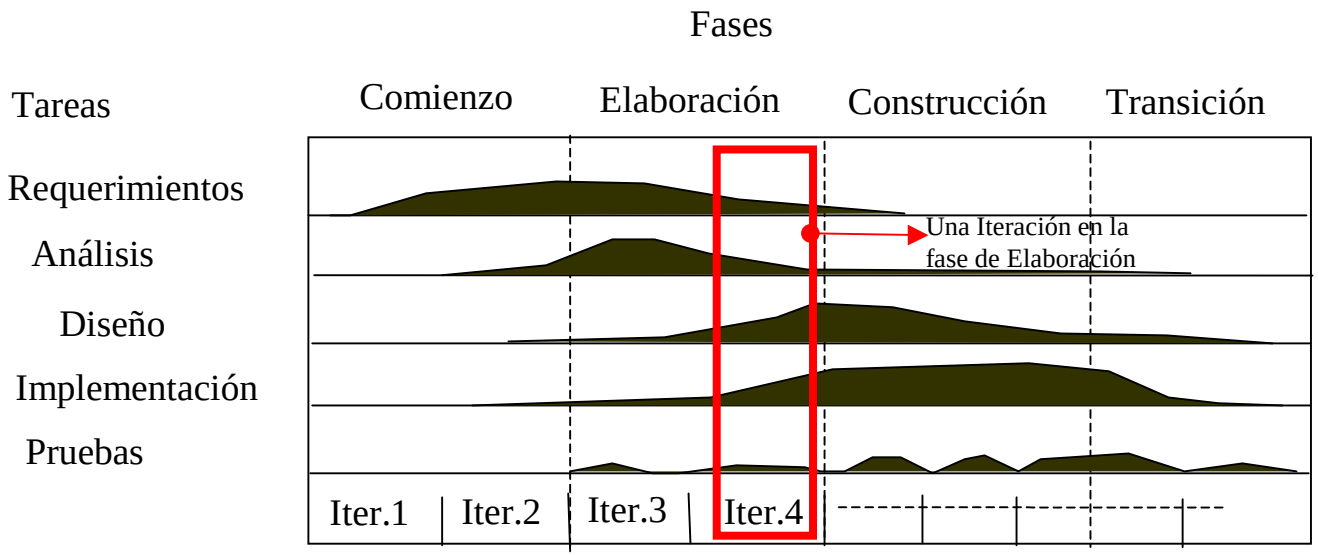
Proceso de desarrollo de software

- Un proceso de desarrollo de software es una serie estructurada de actividades que cubren la manera en la cual una organización desarrolla proyectos de software.
- Hay varios modelos para desarrollo de software: modelo de cascada, modelo por flujo de tareas, modelo unificado.
- El modelo por flujo de tareas identifica las actividades que el equipo del proyecto debe realizar. Éstas son:
  - Establecimiento de Requerimientos (Conceptualización)
  - Desarrollo de un modelo para el comportamiento deseado (Análisis)
  - Crear una arquitectura (Diseño)
  - Implementación (Evolución)
  - Pruebas
  - Administrar la evolución posterior a la entrega (Mantención)



Proceso de desarrollo de software

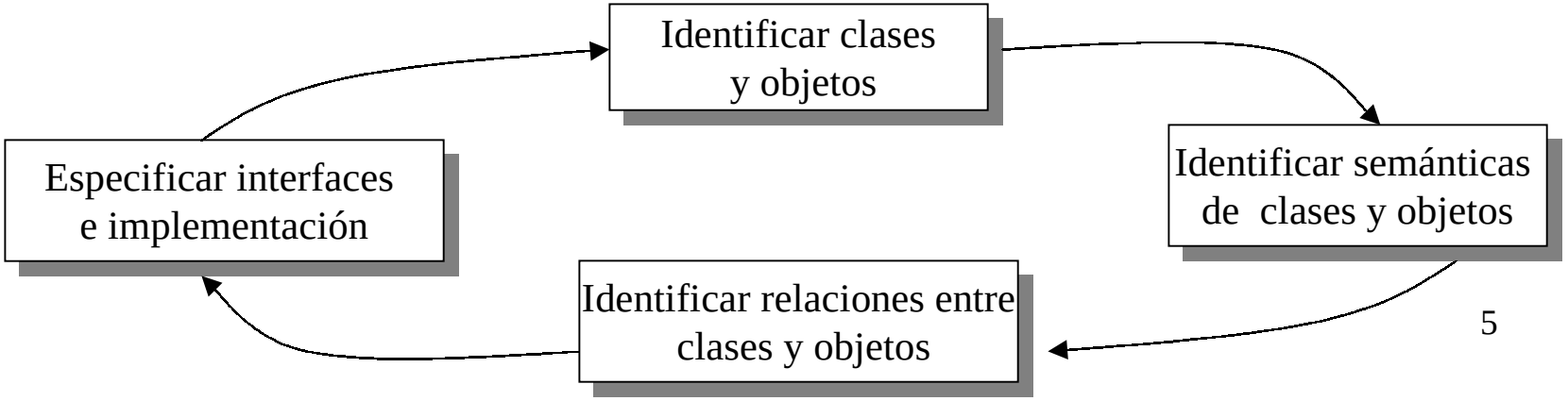
- Por otro lado, en una visión del modelo Unificado, las mayores fases son: Comienzo, Elaboración, Construcción y Transición.
- Estas fases están mas en relación con el avance temporal del proyecto. Cada fase puede ser dividida en incrementos que cubren las actividades del modelo por flujo de trabajo.



Tareas

- Requerimientos: el propósito es establecer los requerimientos básicos para un sistema nuevo o las modificaciones mayores de uno ya existente.
- Análisis: el propósito es desarrollar un modelo de los comportamientos deseados del sistema.
- Diseño: el propósito es crear una arquitectura para desarrollar la implementación.
- Implementación: El propósito es hacer evolucionar la implentación a través de sucesivos refinamientos de la arquitectura del sistema.
- Pruebas: Su propósito es verificar el cumplimiento de los requerimientos.
- Mantención: su propósito es administrar la evolución posterior ala entrega del producto.

- Una forma de trabajar en cada iteración menor es la siguiente:



Actividades y resultados

- Primero que todo: En mi opinión no existe una técnica de diseño universalmente aceptada. Mas bien creo en el estudio de estas técnicas para que cada uno tome lo que le parece más útil en su caso.
- Estudiaremos algunas de estas técnicas y las aplicaremos en un ejemplo.
- Requerimientos: normalmente se obtienen en comunicación con el usuario solicitante del sistema. Normalmente es en lenguaje natural (español). Se debe esperar gran informalidad y bajo nivel de detalles.
- Análisis: Se sugiere
  1.- hacer estudio de algunos escenarios representativos.
  2.- hacer un análisis del dominio.
  Una herramienta usada en esta etapa son las tarjetas CRC (Class, Responsabilities, Collaborators).

Nombre de la Componente	
Responsabilidades: *Aquí va la descripción de las responsabilidades asignadas a esta componente*	**Colaboradores:** *Lista de otras componentes con las cuales se relaciona*

Para hacer un análisis del dominio puede ser necesario entrevistar expertos en el dominio o área del problema. ¿Qué sabemos nosotros de medicina, en el diseño de un equipo médico?

Actividades y resultados

- Análisis (cont.) Como resultado del análisis se generan:
  Una descripción del contexto del sistema. Define los bordes del sistema, las cosas que están dentro y que queda fuera del sistema.
  Una colección de escenarios que definen el comportamiento del sistema. Cada escenario específica algún(os) comportamiento(s) del sistema. Un escenario es una instancia de un caso de uso del sistema.
  Un modelo del dominio. Es un modelo del mundo que el sistema computacional está creando. Éste captura nuestro entendimiento de cómo las cosas trabajan (o deberían trabajar). Colección de tarjetas CRC, Diagramas de interacciones (paso de mensajes), diagrama de clases.