

# Ingeniería de Software

Agustín J. González

EIO329: Diseño y Programación Orientados a Objeto

Adaptado de: material asignatura CS169, Software Engineering, UC Berkeley, entre otras fuentes.

# Ingeniería de software

- Definición: Aplicación del arte del desarrollo software junto con las ciencias matemáticas y computadores para **diseñar**, **construir**, y **mantener programas computacionales** **eficientes** y **económicos** que **logran sus objetivos**.
  - Se busca: Resolver el problema a costo y en tiempo controlados.
- Estado del arte: Se encuentra en estado muy primitivo
  - A lo más una serie de “mejores prácticas”, desarrolladores de software construyen software y si éstos funcionan, entonces nosotros estudiamos cómo ellos lo hicieron.
  - Si éstos funcionan por un largo tiempo, entonces estudiamos sus procesos de software aún más cuidadosamente.

# Construcción de una casa para “fido”



La puede hacer una sola persona

Requiere:

Modelado mínimo

Proceso simple

Herramientas simples

# Construcción de una casa

Construida eficientemente y en un tiempo razonable por un equipo

Requiere:

- Modelado

- Proceso bien definido

- Herramientas más sofisticadas



# Construcción de un rascacielos



- Moraleja: No es lo mismo
  - hacer una tarea en programación de 1er. año que
  - desarrollar un software comercial
- ¿Cómo debe cambiar el proceso de desarrollo?

# Claves en Desarrollo de Ingeniería de Software (IS)

## Notación (UML)



**Herramientas**  
(Ej: Rational Rose,  
Umbrello, IDEs)

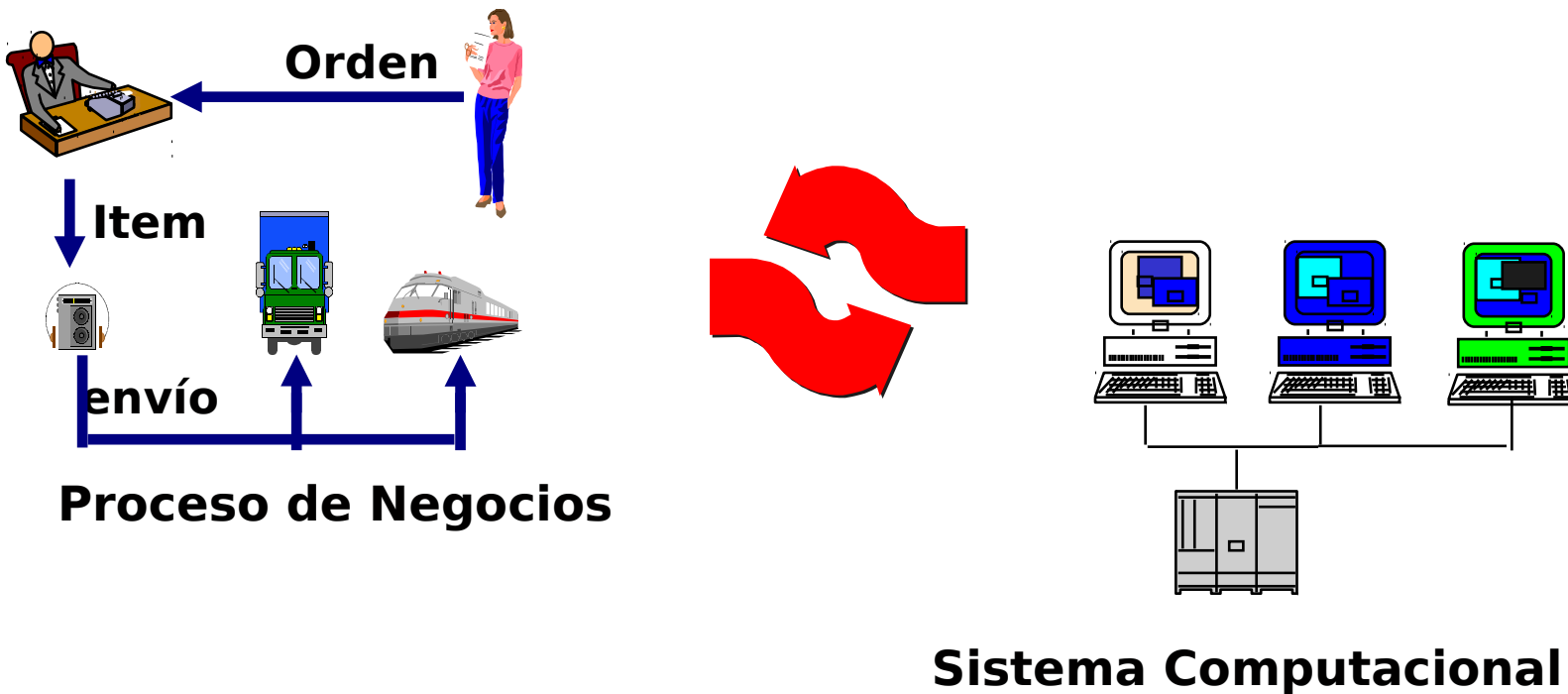
**Proceso**  
(Metodologías  
Ej: ITIL, SCRUM, Extreme Programming,  
RUP: Rational Unified Process,  
Personal Software Process)

# Abstracción - Modelado Visual (MV)

*“El modelado captura las partes esenciales del sistema”*

*En otras disciplinas se conoce el beneficio de tener representaciones visuales del modelo.*

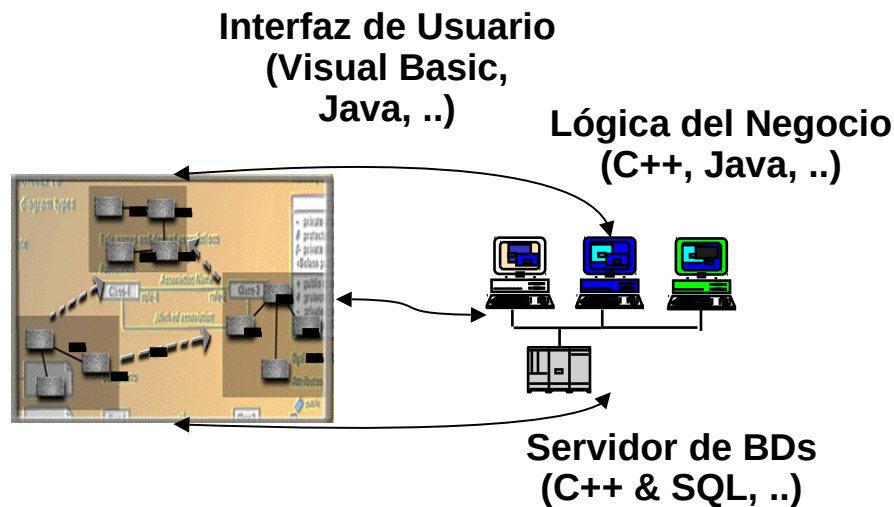
*Ej. Plano circuito, plano arquitectura, diagrama cuerpo libre*



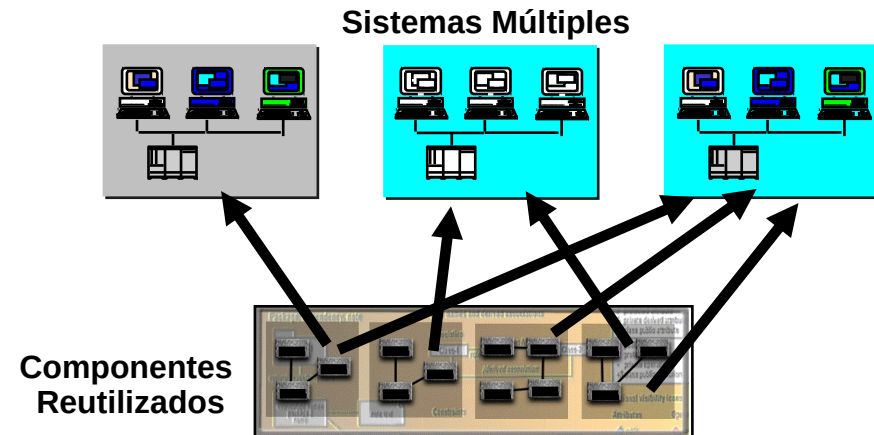
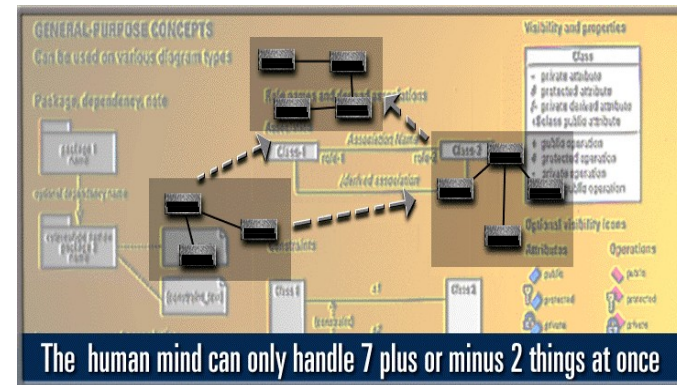


# Notación (Visual) - Beneficios

## Manejar la complejidad



“Modelar el sistema independientemente del lenguaje de implementación”



Promover la Reutilización

# ¿Por qué la Orientación a Objetos?

- Parte importante del proceso de desarrollo de software es el modelado.
- La orientación a objetos entrega una forma intuitiva de modelar objetos del mundo real (estado) y sus interacciones en objetos de software por su proximidad a las entidades del mundo real que surgen del modelado.
  - Mejora la captura y validación de requisitos
  - Acerca el “espacio del problema” al “espacio de la solución”
- Su diseño facilita:
  - la creación de **Abstracciones** (Ignorar detalles)
  - la **Modularización** (separación en módulos)
  - **Ocultar información** (separar la implementación del uso)

# Problemas en OO

- Podemos distinguir dos tipos de objetos degenerados:
  - Un objeto sin datos (que sería lo mismo que una biblioteca de funciones). Si los métodos son estáticos, “peor” aún.
  - Un objeto sin “operaciones”, con sólo atributos públicos (que equivaldría a las estructuras de datos tradicionales)
- Un sistema construido con objetos degenerados no es un sistema verdaderamente orientado a objetos.