
# INTRODUCCIÓN

Ingeniería de Software 2

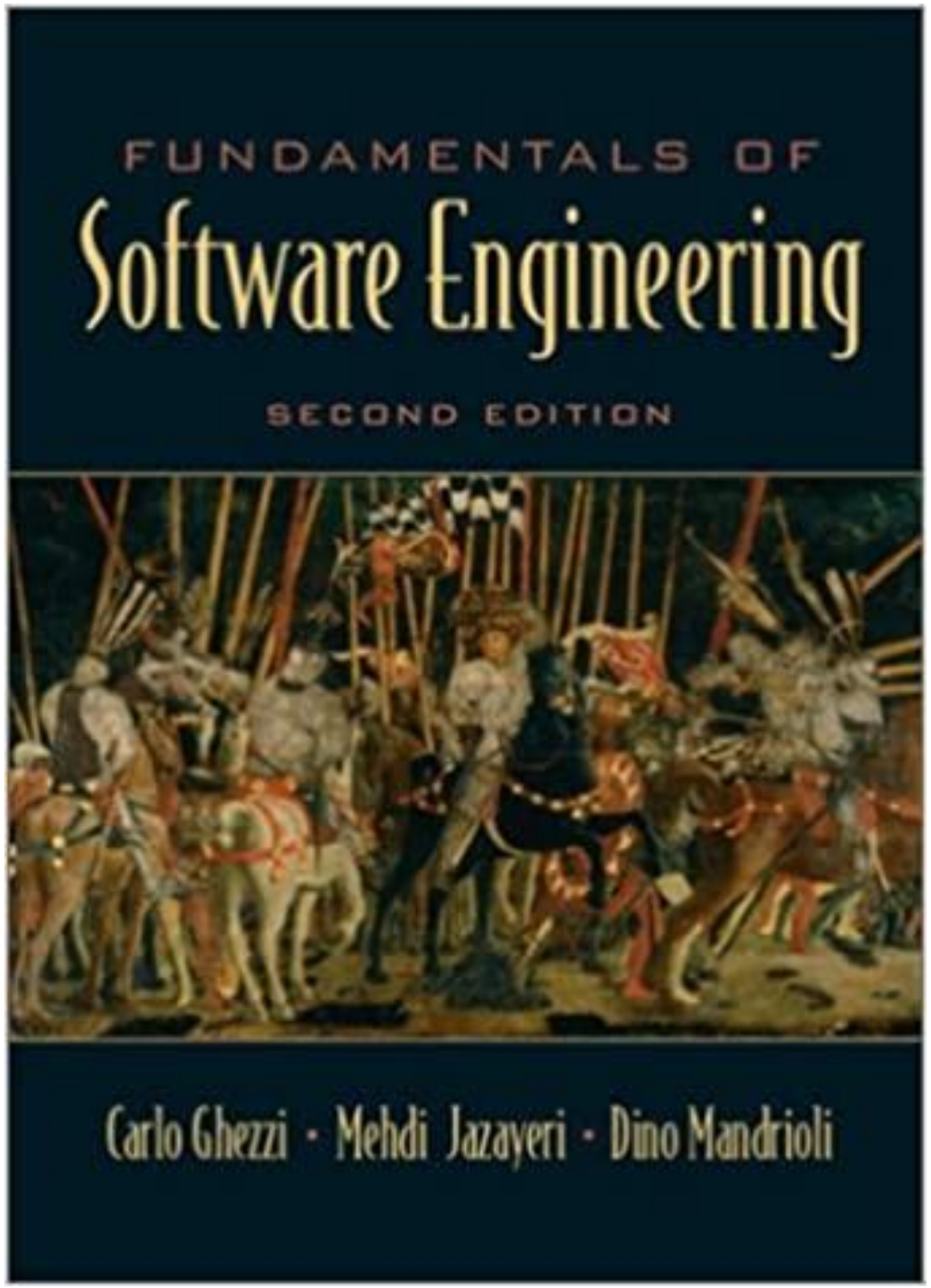

# INTRODUCCIÓN: PRINCIPIOS

Ingeniería de Software 2


Herramientas
Metodologías
Métodos
y técnicas
Principios

# PRINCIPIOS

1. Rigor y Formalidad
2. Separación de Intereses
3. Modularidad
4. Abstracción
5. Generalidad
6. Anticipación al Cambio
7. Incrementalidad

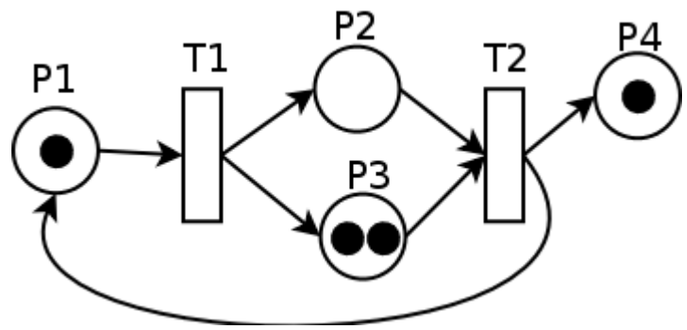


# 1. RIGOR Y FORMALIDAD

- ✓ El Desarrollo de Software es un proceso Creativo
- ✓ El RIGOR es un necesario complemento a la CREATIVIDAD
- ✓ RIGOR fomenta la práctica sistemática de la Ingeniería de Software
- ✓ RIGOR aumenta confianza en los resultados
- ✓ RIGOR fomenta la repetibilidad y evita incurrir en errores pasados

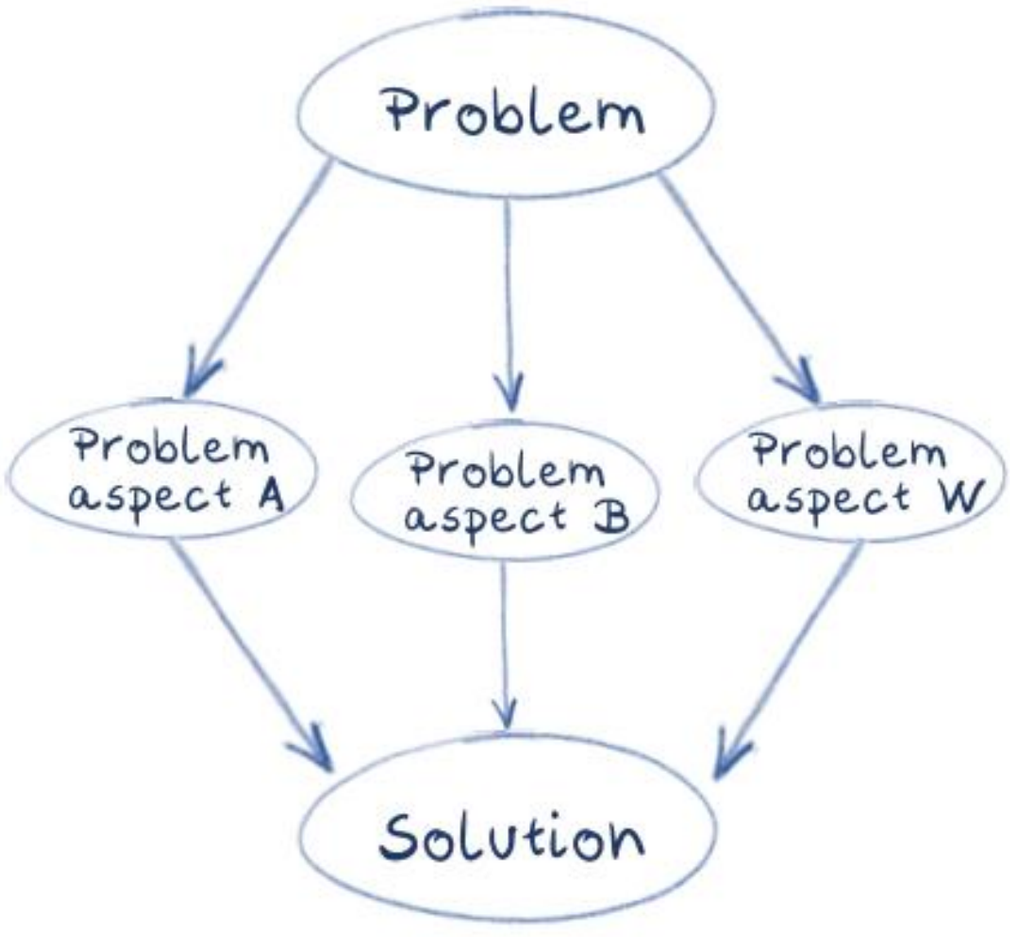
# 1. RIGOR Y FORMALIDAD

- ✓ La FORMALIDAD lleva el RIGOR a su máxima expresión
- ✓ Especificaciones y métodos FORMALES fomentan:
  - ✓ Verificación por leyes matemáticas
  - ✓ Especificaciones no ambiguas (Redes de Petri, Z)
  - ✓ Generación automática de código



# 2. SEPARACIÓN DE INTERESES

- ✓ Divide et Impera (Divide y Vencerás)
- ✓ Identifica diferentes aspectos de un problema, en los cuales podemos enfocarnos individualmente

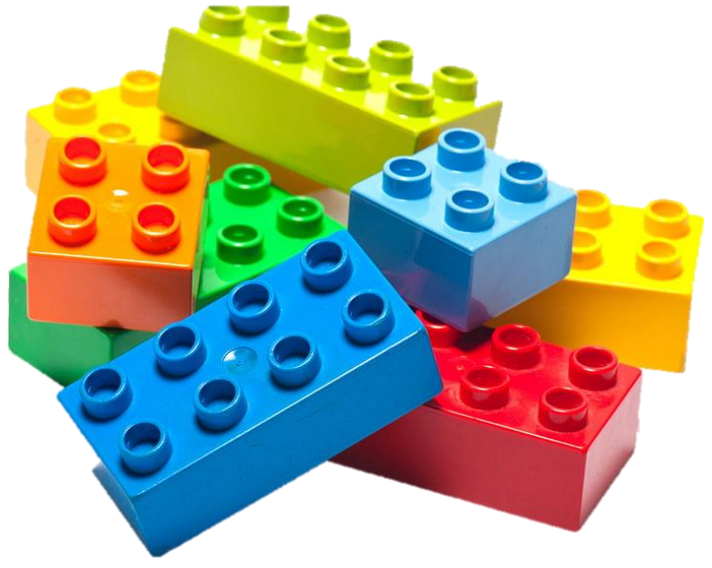


# 2. SEPARACIÓN DE INTERESES

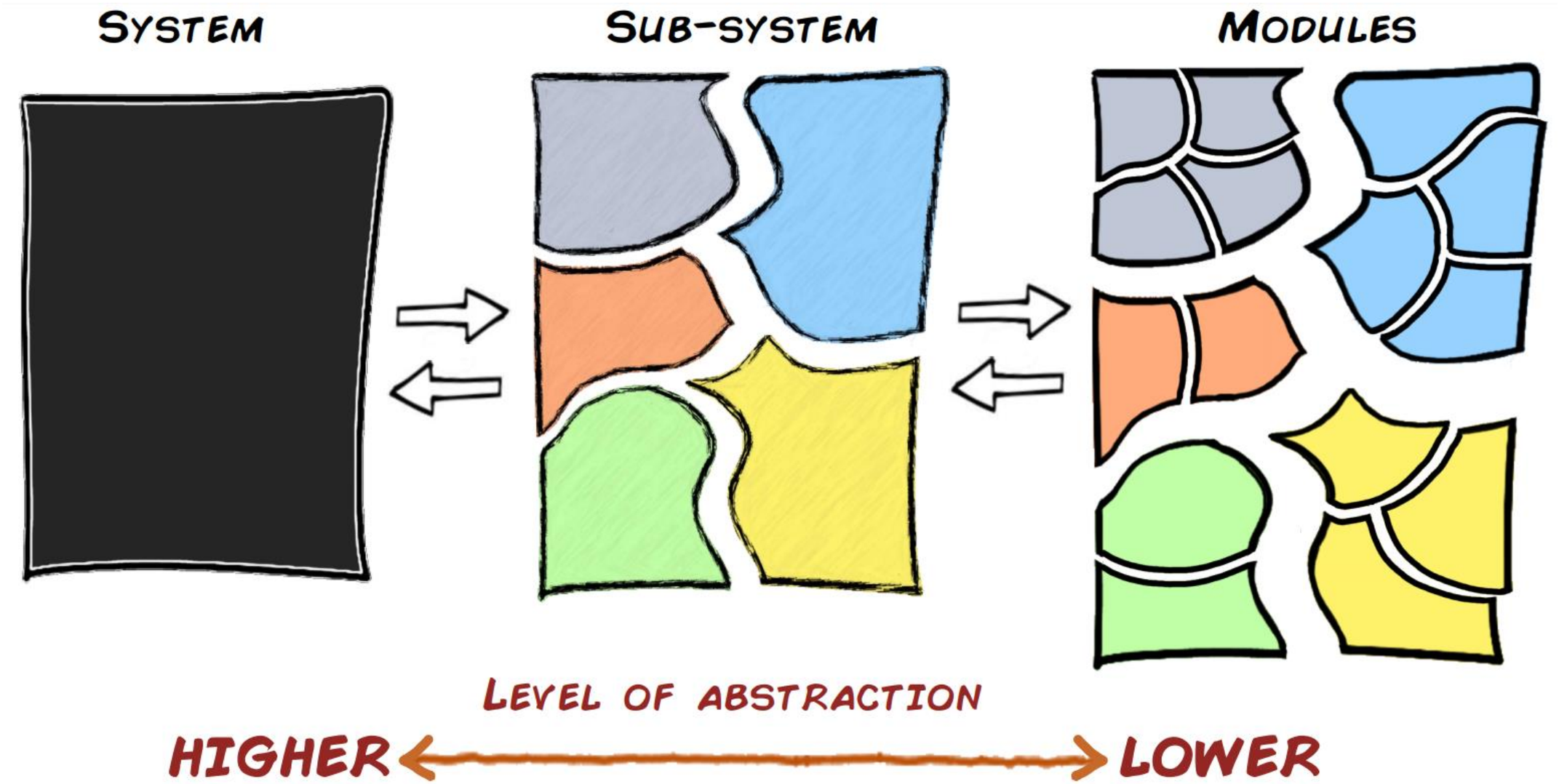
- ✓ Gestionar la complejidad de un proyecto software, separando:
  - ✓ **Tiempo**: fases, iteraciones, actividades
  - ✓ **Calidad**: funcional, eficiencia, usabilidad, etc...
  - ✓ **Vistas**: estructural, funcional, interfaz
  - ✓ **Partes**: módulos, subsistemas, componentes
  - ✓ **Responsabilidades y habilidades**: jefe de proyecto, analista, diseñador, etc…

# 3. MODULARIDAD

- ✓ Un sistema complejo puede ser particionado en unidades o partes más pequeñas y simples

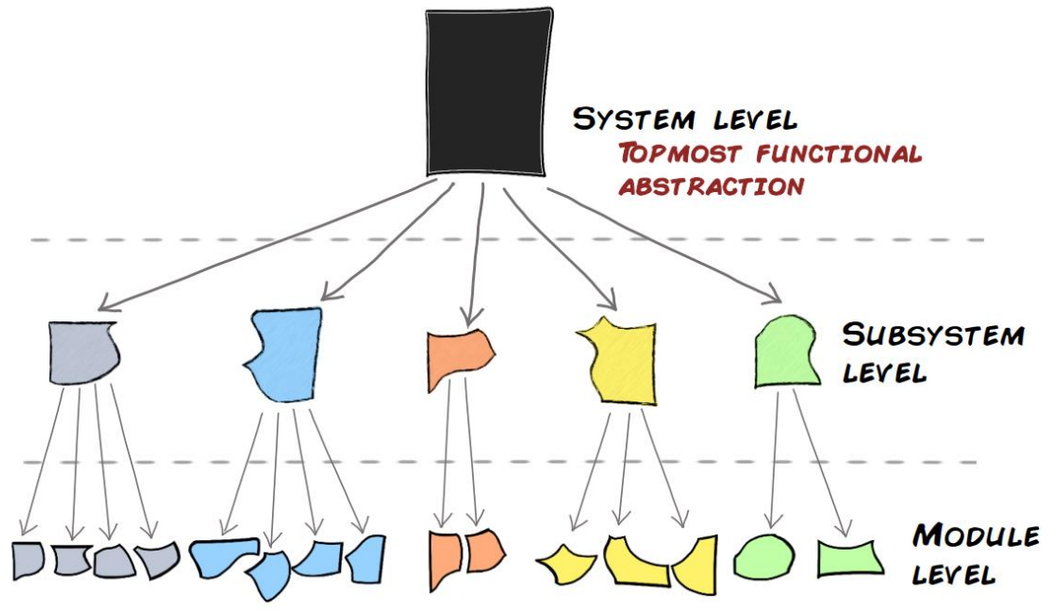


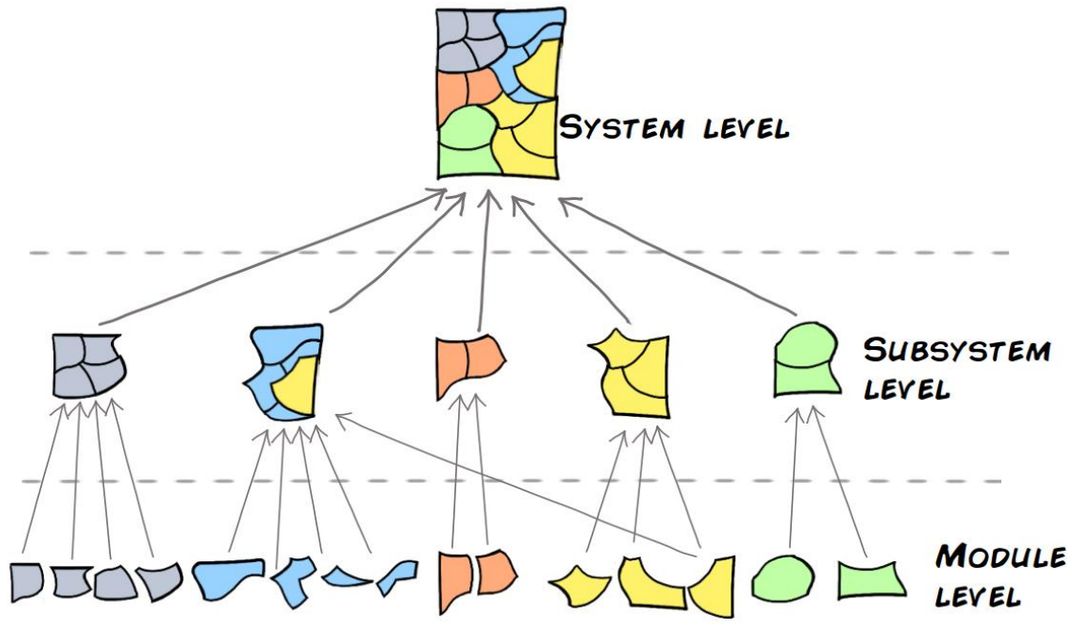
# 3. MODULARIDAD



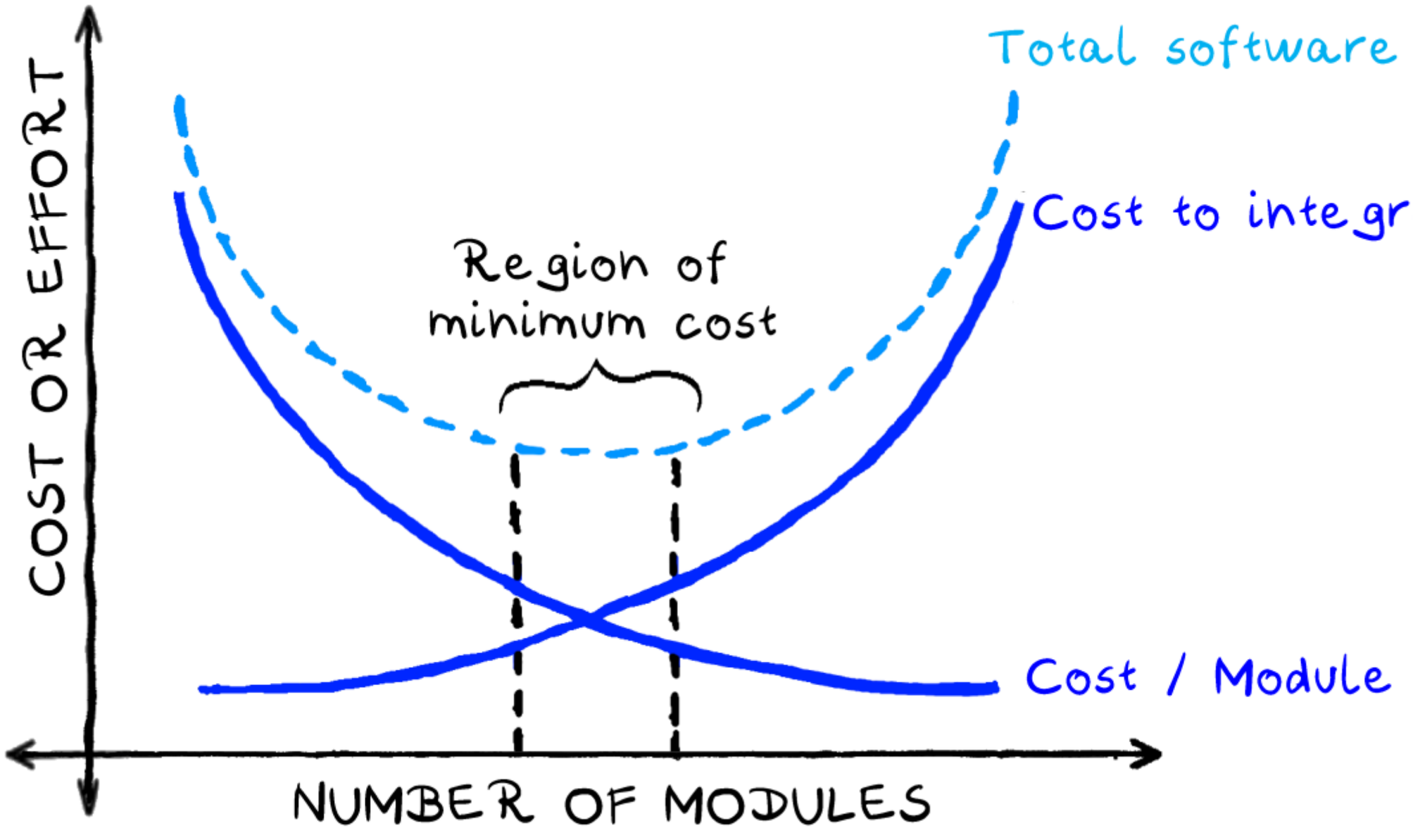
# 3. MODULARIDAD

- ✓ Lidiar con un módulo a la vez, ignorando detalles de los restantes
- ✓ Se habla de un sistema modular cuando está compuesto por módulos claramente identificables
- ✓ Enfoques Top-Down y Bottom-Up



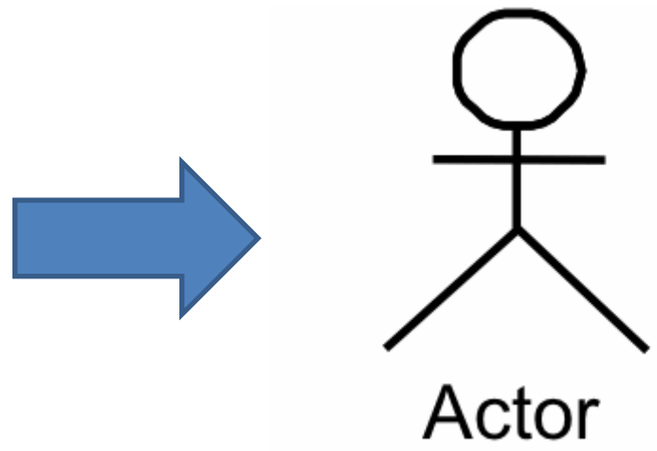



# 3. MODULARIDAD



# 4. ABSTRACCIÓN

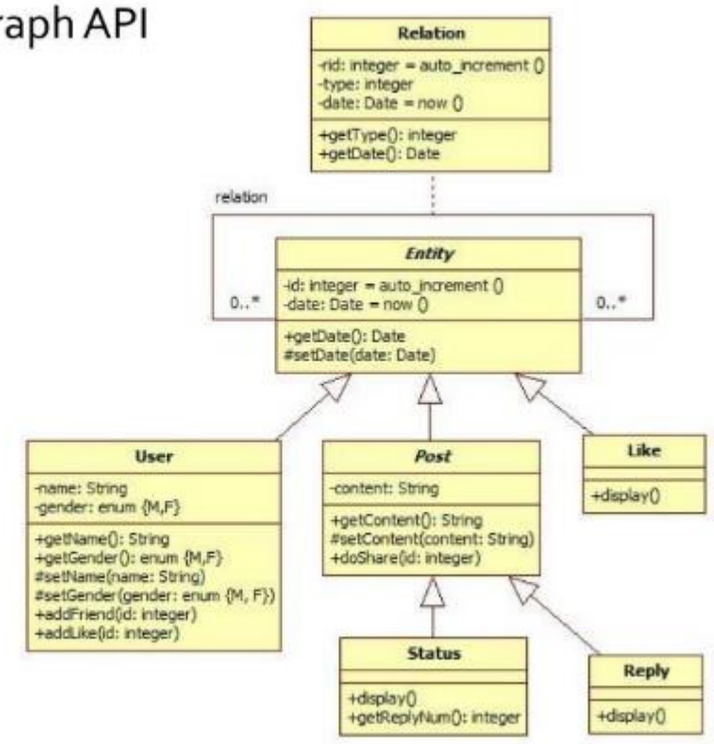
- ✓ Identificación de los aspectos esenciales, ignorando los menos importantes



# 4. ABSTRACCIÓN

- ✓ Lo que dejamos de lado depende del observador y del problema
- ✓ Modelos, Diagramas, Prototipos, Ecuaciones, etc.

Facebook Graph API




# 5. GENERALIDAD

- ✓ Observar el problema y su contexto con una mirada amplia
- ✓ Explorar si el problema es una instancia de un problema más general

# 5. GENERALIDAD

- ✓ Puede suceder que el problema generalizado no sea más complejo.
- ✓ Siendo más general, la solución será más reusable.

# 6. ANTICIPACIÓN AL CAMBIO

- El software sufre cambios permanentemente
  - Nuevas características
  - Mejoras en las características existentes
  - Mejoras en performance
  - Corrección de defectos
  - Adaptación a cambios en leyes, regulaciones, ambiente

# 6. ANTICIPACIÓN AL CAMBIO

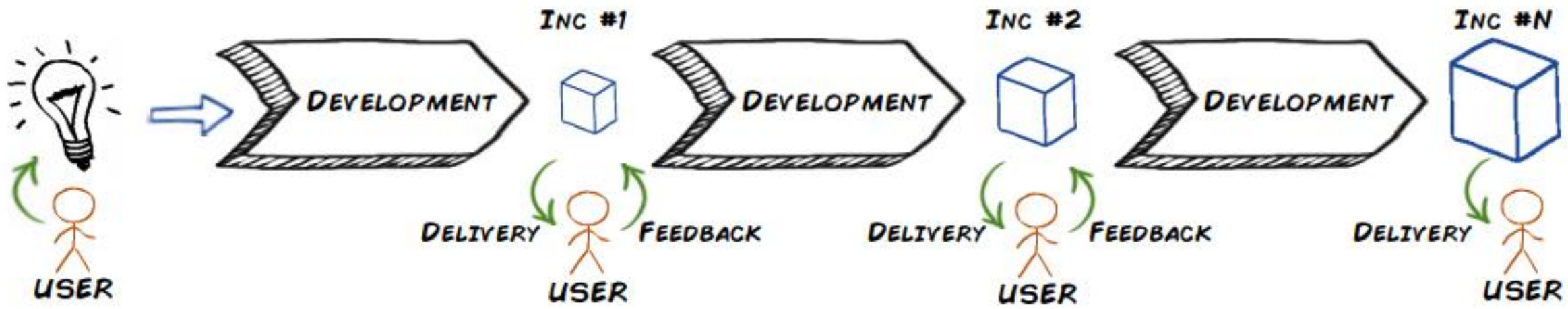
- ✓ Se requiere un esfuerzo especial en las fases iniciales para anticipar cómo y dónde será probable que se den los cambios.
- ✓ Los cambios probables deben ser aislados en porciones específicas del software (módulos sujetos a cambios)

# 7. INCREMENTALIDAD

- ✓ Identificar tempranamente subconjuntos útiles de una aplicación para rápido feedback.
- ✓ Requerimientos cambian a medida que se desarrolla el producto.

# 7. INCREMENTALIDAD

- ✓ Feedback temprano, acciones correctivas, añadir nuevas características incrementalmente
- ✓ De forma incremental, el prototipo se transforma en el producto final




# INTRODUCCIÓN: PRINCIPIOS

Ingeniería de Software 2


# INTRODUCCIÓN

Ingeniería de Software 2