TEMA 1. Introducción

Modelado del Software

Raquel Martínez España

Grado en Ingeniería Informática

Raquel Martínez España- Tlf: (+34) 968 27 86 57
Universidad Católica San Antonio de Murcia - Tlf: (+34) 968 27 88 00 info@ucam.edu -
www.ucam.edu

Evolución

Los primeros años

- Grandes computadores
- SW especializado
- Escasa movilidad

La segunda era

- Entornos multiusuario
- Mantenimiento de sw externo
- Tiempo real, bases de datos

La tercera era

- Sistemas distribuidos
- Hardware de bajo coste
- Complejidad creciente

La cuarta era

- IA, Redes Neuronales
- Entornos Cliente/Servidor
- Seguridad y fiabilidad
- Paralelismo
- Orientación a objeto

Software (SW)

El software son **instrucciones** que cuando se ejecutan proporcionan la función y el rendimiento deseados, **estructuras de datos** que permitan a los programas manipular adecuadamente la información y **documentos** que describen la operación y el uso de programas

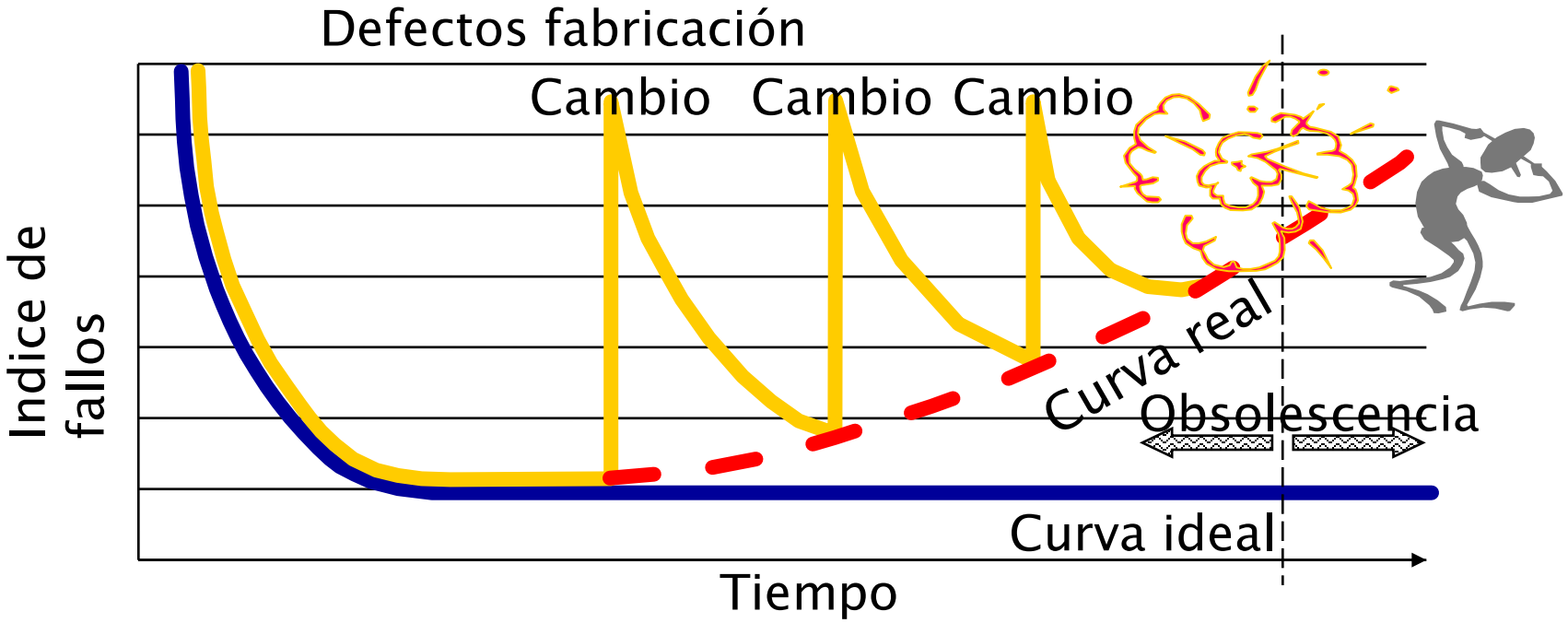
Características del SW

Es un producto lógico no físico por tanto:

1. Se desarrolla, no se fábrica
2. Se construye a medida, no ensamblando partes
3. No se estropea, se deteriora con los cambios

Características del SW

- Curva de fallos del software



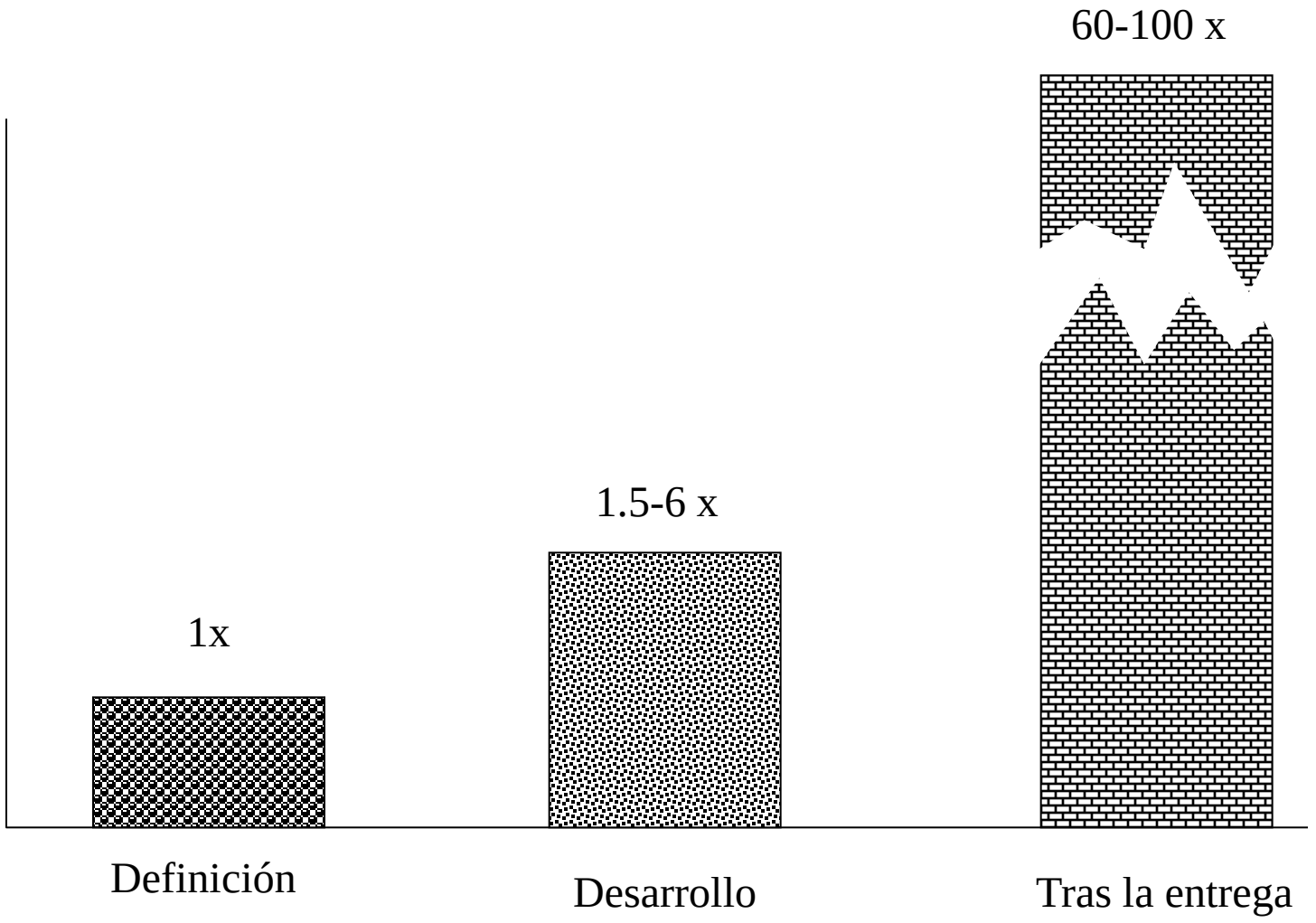
Clasificación

1. De sistemas
   - Dar servicio a otros programas
   - Compiladores, sw redes, etc.
2. De aplicación
   - Resolver una necesidad puntual
   - TPV, banco, gestión matrículas, etc.
3. De tiempo real
   - Operaciones críticas
   - Control central nuclear, planes vuelo compañías aéreas, etc.

Clasificación

4. Ingeniería y científico
   - Gran capacidad de cómputo
   - Simuladores, análisis numérico, etc.
5. Empotrado
   - Embebido en un producto
   - Horno, alarma, etc.
6. Software basado en web
7. Inteligencia Artificial
   - Resolución de problemas complejos
   - Robótica, redes neuronales, etc.

El coste del cambio



Ingeniería del SW

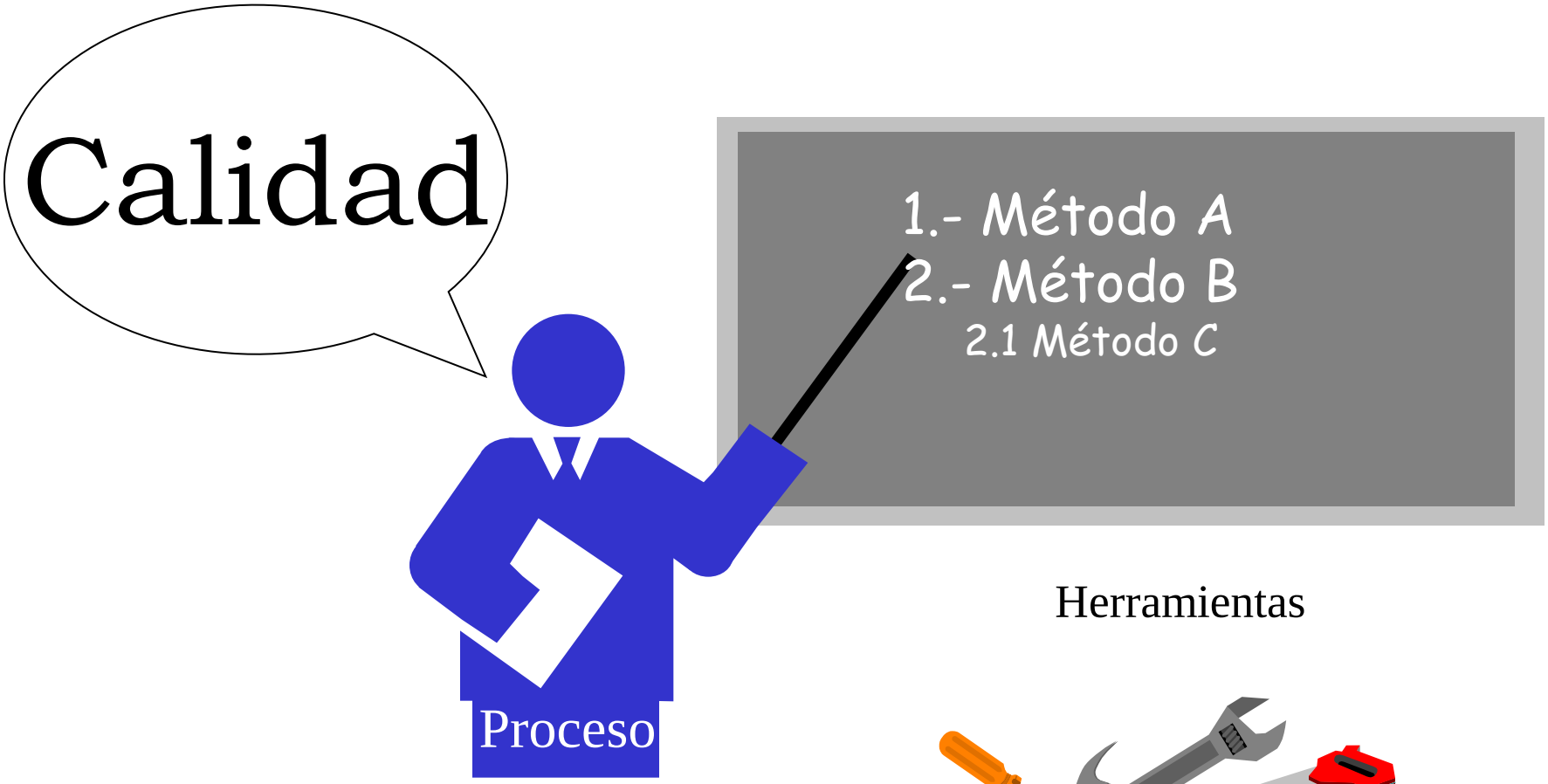
"La ingeniería del software es el establecimiento y uso de principios robustos de la ingeniería a fin de obtener económicamente software que sea fiable y que funcione eficientemente sobre máquinas reales"

⇒ **Principios robustos**

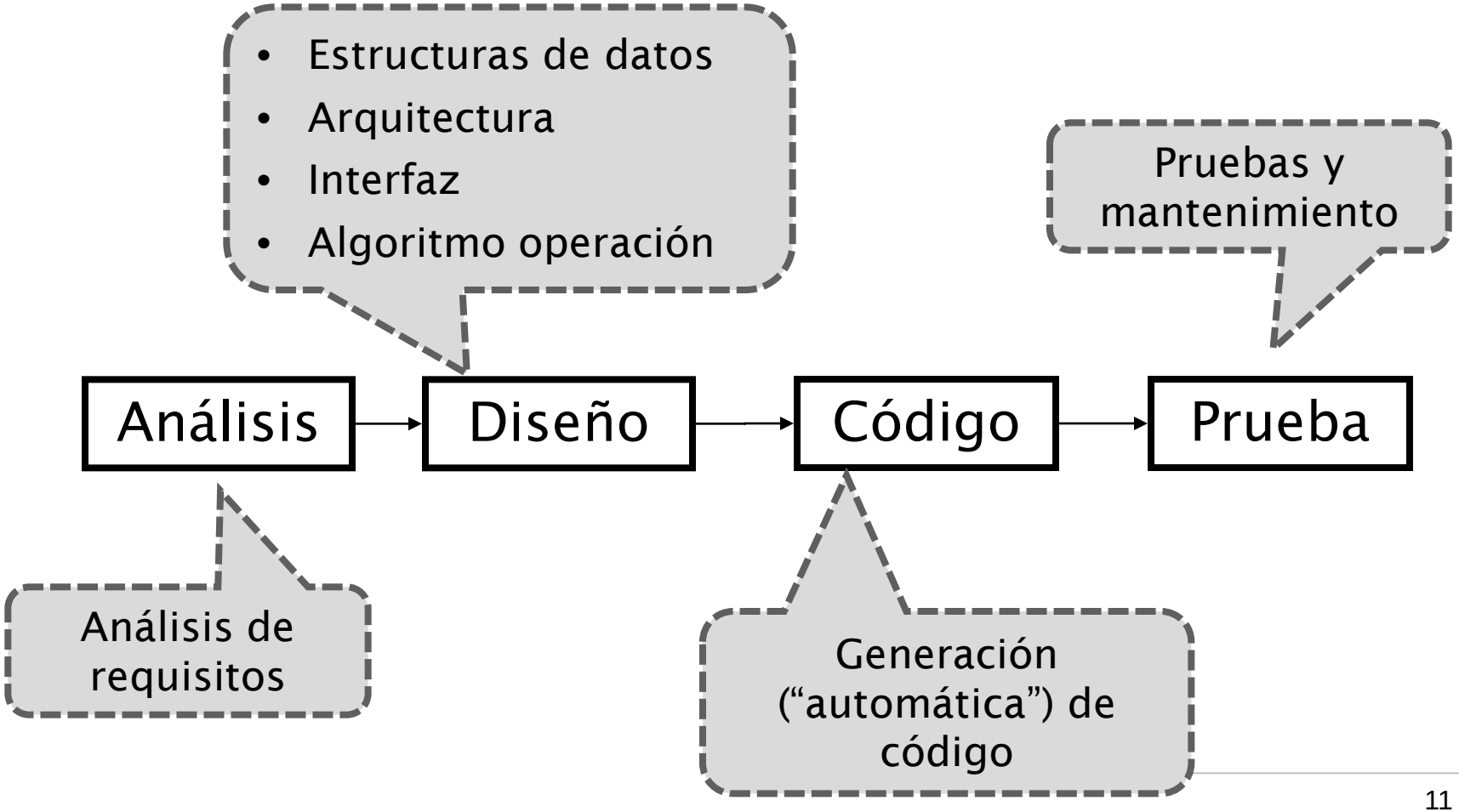
⇒ **Económicamente**

⇒ **Máquinas reales**

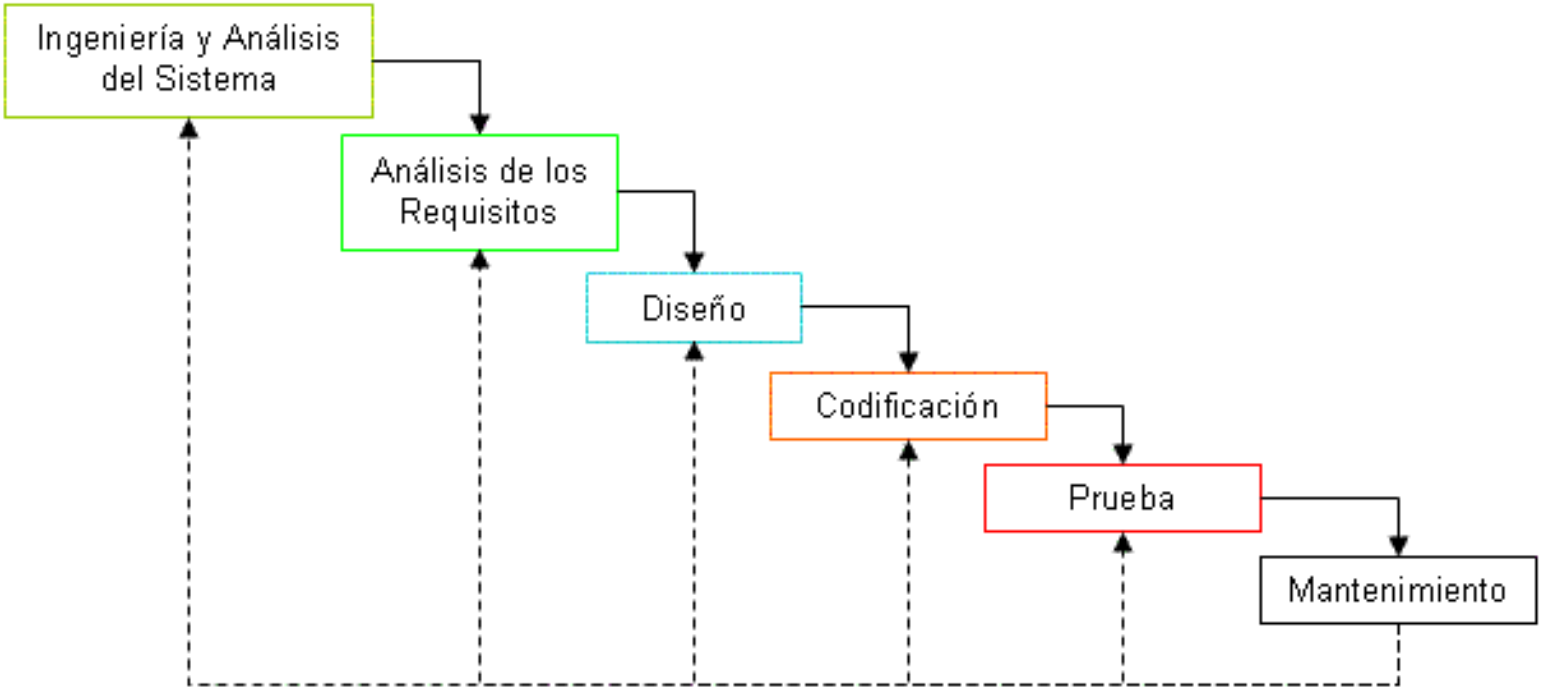
Tecnología multicapa



Modelo de proceso lineal (cascada)



Modelo de proceso lineal (cascada)



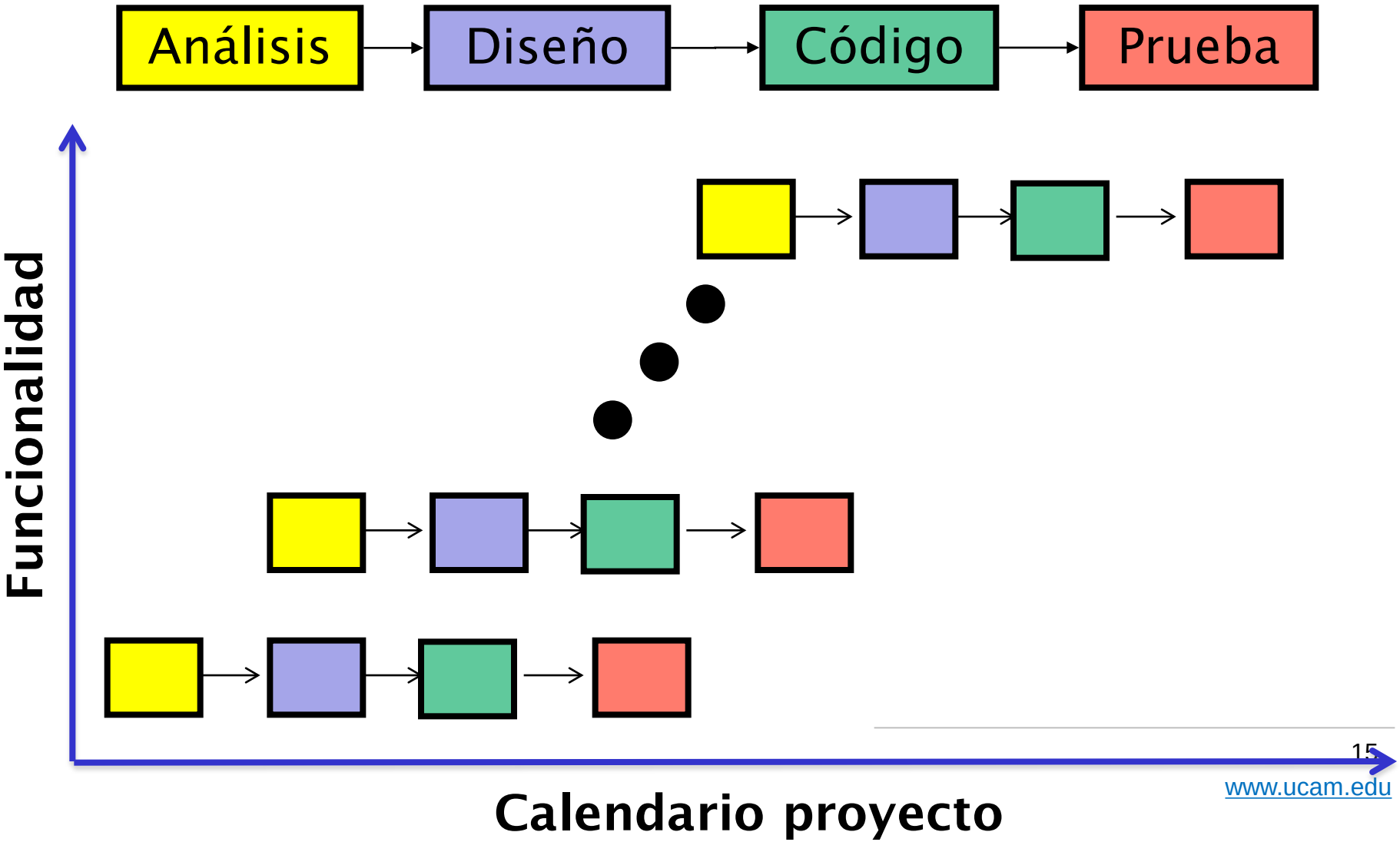
Proceso lineal: problemas

- No es un modelo realista
- No es normal conocer todos los requisitos al principio
- Se tarda en tener una versión de trabajo
- Tiempo de espera para tareas dependientes

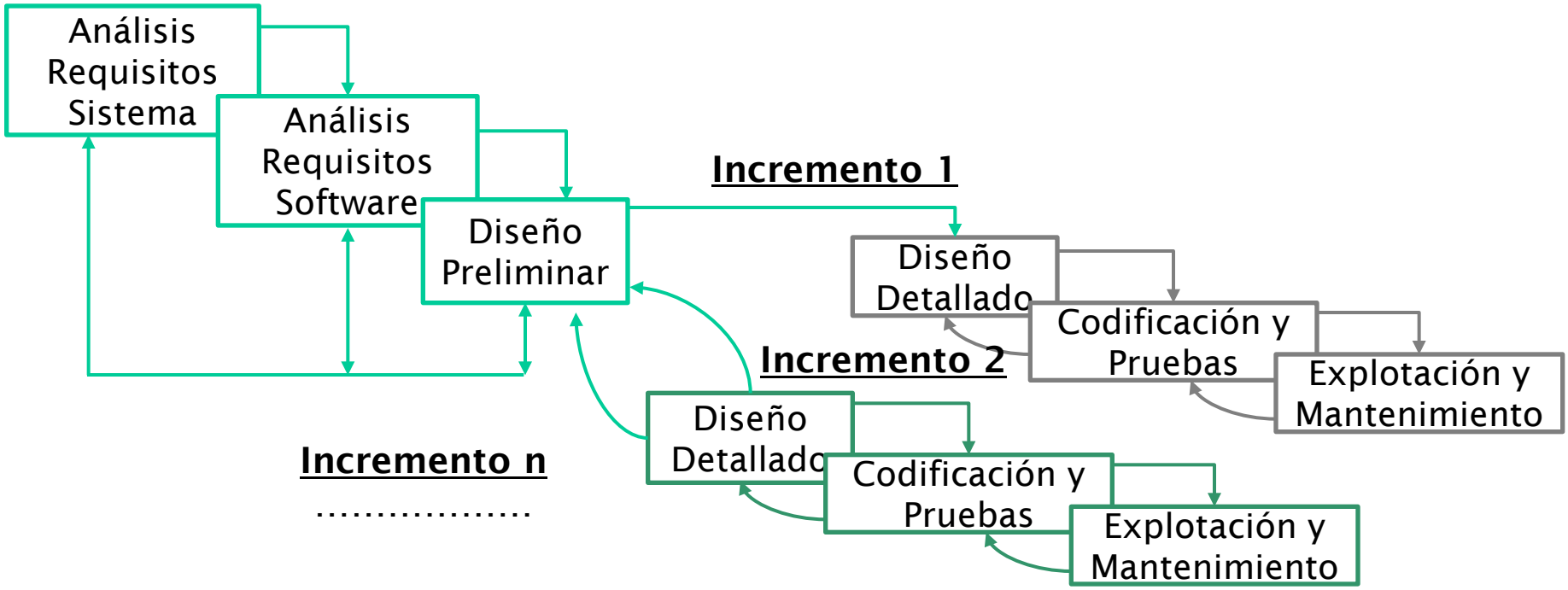
Modelo incremental

- A veces no es posible aplicar directamente un desarrollo lineal.
- Prototipos funcionales = incrementos
- Cada incremento añade nueva funcionalidad
- El cliente trabaja desde el principio con una versión funcional del producto final

Modelo incremental



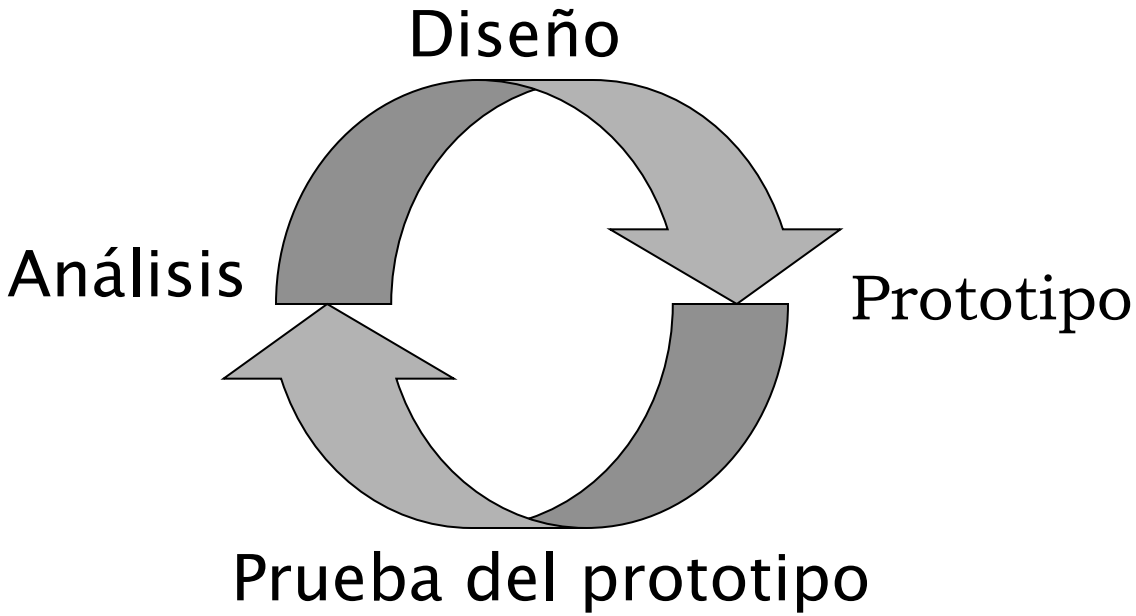
Modelo incremental



Modelos evolutivos

- El software evoluciona
- Los requisitos a menudo cambian y no están claros desde el comienzo
- Necesario un modelo de proceso que soporte esta "evolución"
- **Modelos evolutivos**
  - Iterativos
  - Identificación de requisitos
  - Modelo de prototipado y modelo espiral

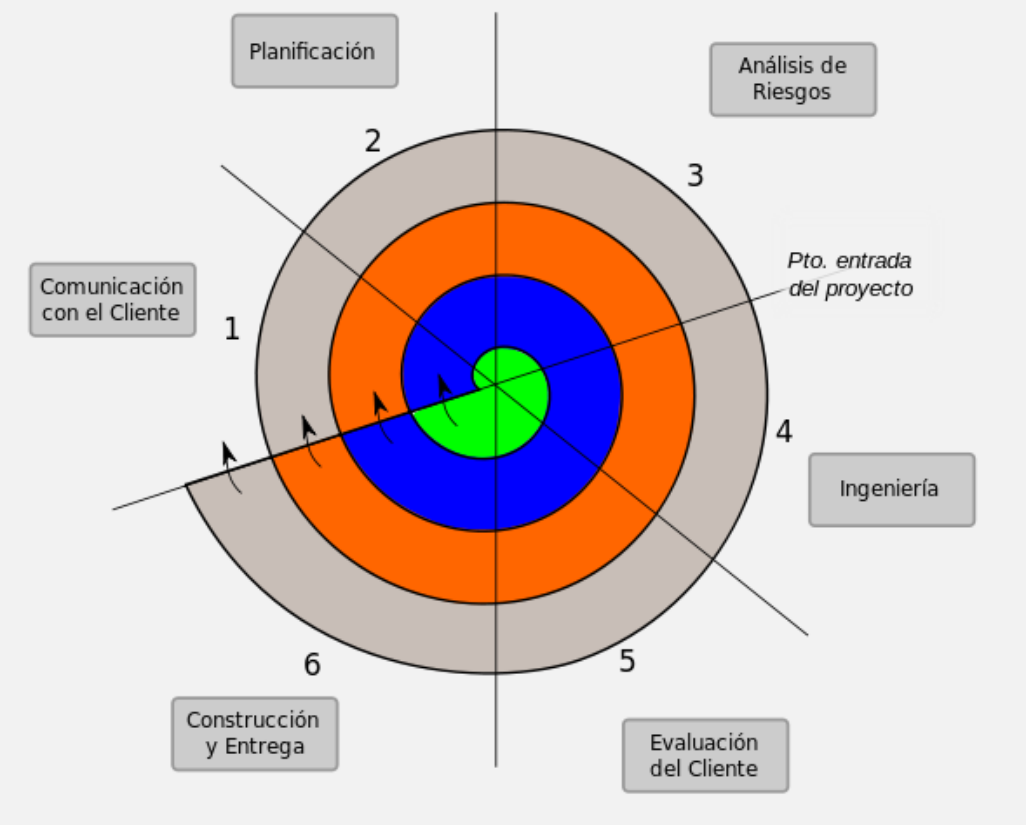
Modelo de prototipado



Prototipado: problemas

- El cliente no concibe el concepto de "prototipo"
- Arrastre de decisiones tomadas de forma rápida para la construcción inmediata del prototipo.
- Es una buena manera de establecer las condiciones iniciales y ayudar a la definición de requisitos del producto final

Modelo en espiral



Desarrollo basado en componentes

- Orientación a objeto
- Identificación de las clases candidatas
- Extracción la biblioteca de clases
- Inserción en la biblioteca de clases:
  - Aumenta la productividad
  - Depende de la biblioteca de clases

Modelo de métodos formales

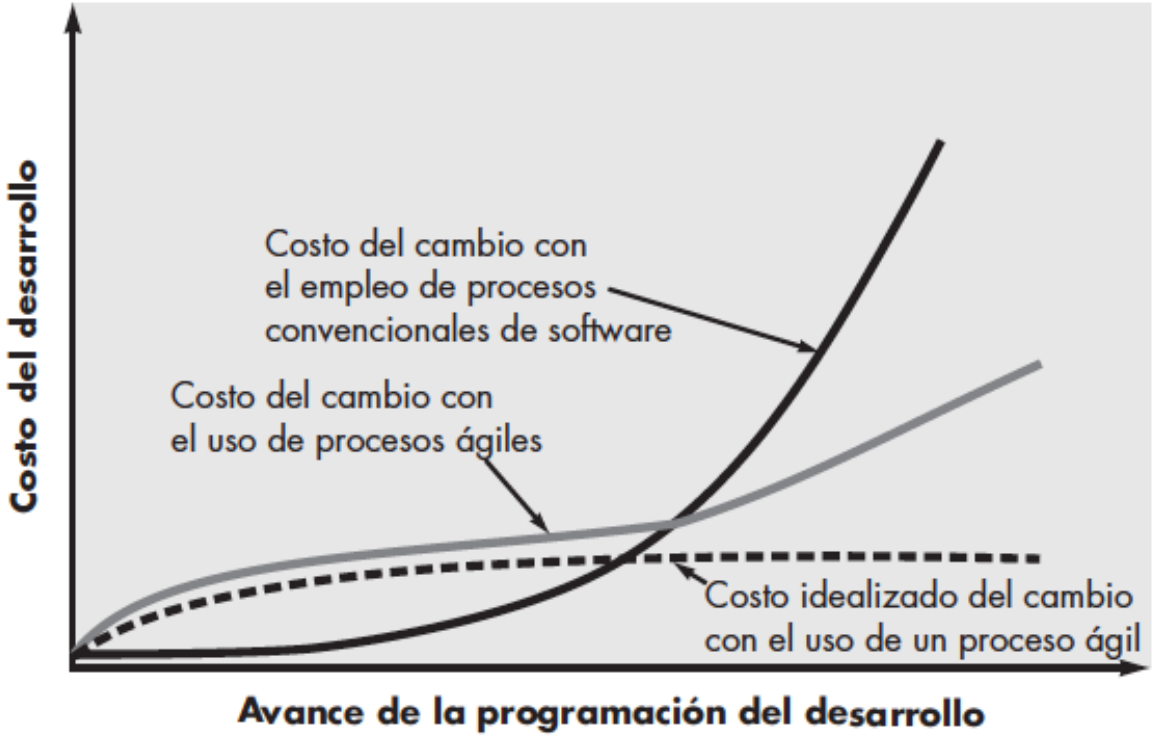
- Aplicación de las matemáticas a la especificación del SW.
- Especificación formal del software.
- Pruebas algebraicas de su corrección.
- Precisa de conocimientos técnicos tanto en el cliente como en el desarrollador.
- Se basan en el uso de **lenguajes formales.**

Procesos Ágiles

- Objetivo: permitir cambios en fase tardía de desarrollo con un reducido coste.
- Características principales:
  - *Adaptable e incremental*
- Tres suposiciones clave:
  1. Difícil **predecir** los requisitos que persistirán y cuáles cambiarán.
  2. El diseño y la construcción están intercalados. Es difícil **predecir** cuanto diseño se necesita antes de que se use la construcción para probar el diseño.
  3. Análisis, diseño y construcción no son **predecibles**.

Coste del cambio usando métodos ágiles

- Realizar cambios en fases tardías sin gran coste económico ni tiempo.



Programación extrema (I)

- O más conocida como *eXtreme Programming (XP)*.
- Se enmarca dentro de los denominados *Procesos Ágiles*.
- En el desarrollo software son habituales los cambios en los requisitos.
- *Un proceso que trata de manejar lo impredecible.*
- Es mejor adaptarse a los cambios:

Adaptabilidad vs. **Previsibilidad**

Programación Extrema (II)

- Cuatro variables: *coste*, *tiempo*, *ámbito y calidad* .
  - Las tres primeras sólo pueden ser establecidas por fuerzas externas.
  - La última es establecida por los programadores en función de las otras.
  - Aunque pueda parecerlo no existe una relación directa entre ellas.

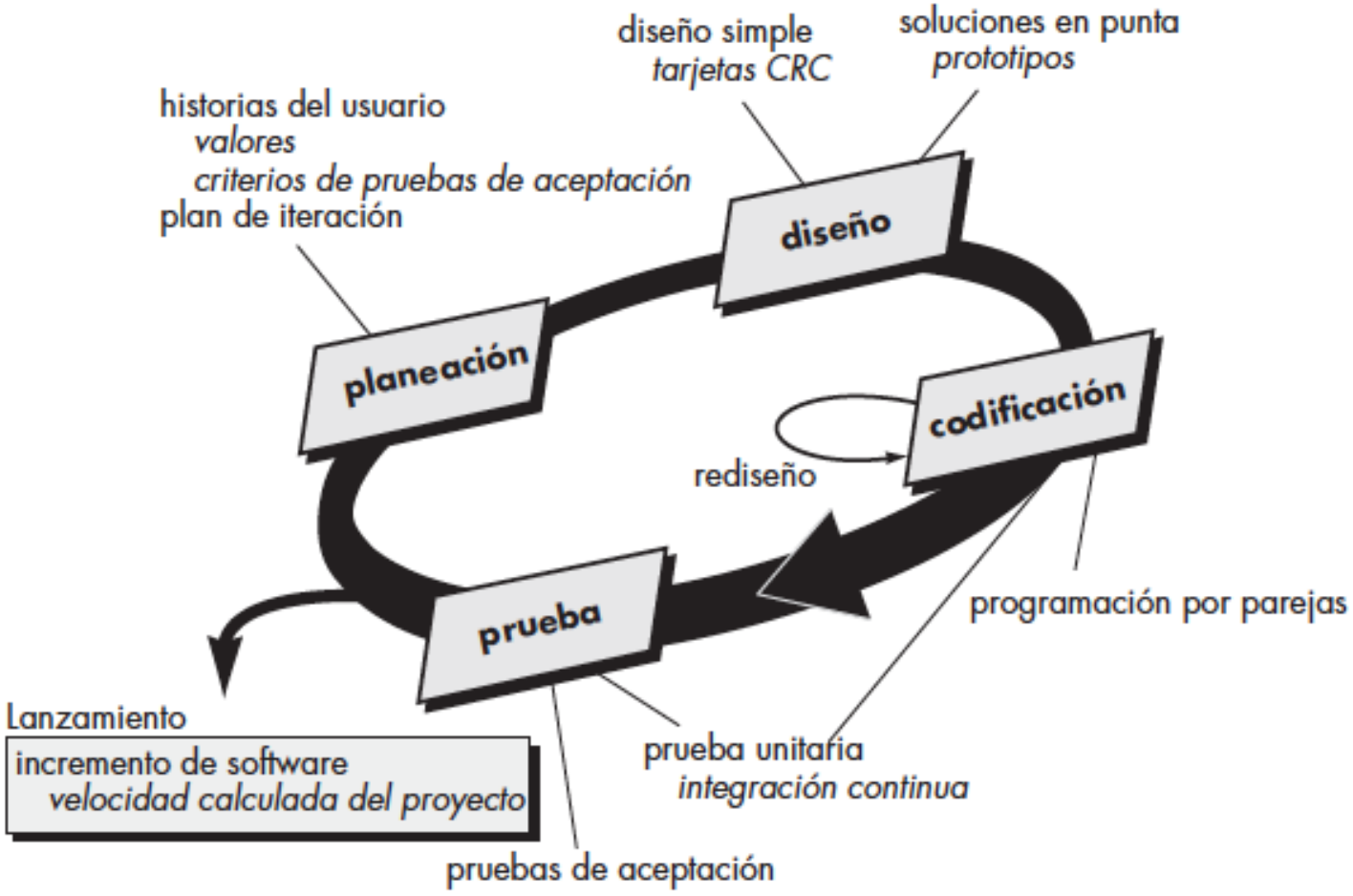
Programación Extrema (III)

- Cuatro valores:
  - Comunicación: continua entre ingenieros y clientes.
  - Sencillez: el diseño cubrirá necesidades inmediatas. Las mejoras más adelante.
  - Retroalimentación: "*no me preguntes a mi, pregúntale al sistema*"
  - Valentía: asumir retos, ser valientes ante los problemas y afrontarlos. No parches.

Programación Extrema (IV)

- El proceso XP incluye las siguientes actividades:
  1. **Planeación:** escuchar al cliente y recabar *historias del usuario*
  2. **Diseño:** sencillez ante la complejidad. Escalabilidad y fácil extensión.
  3. **Codificación**: programación *por parejas*.
  4. **Pruebas:** *pruebas unitarias* y *pruebas de aceptación*

Programación Extrema (IV)

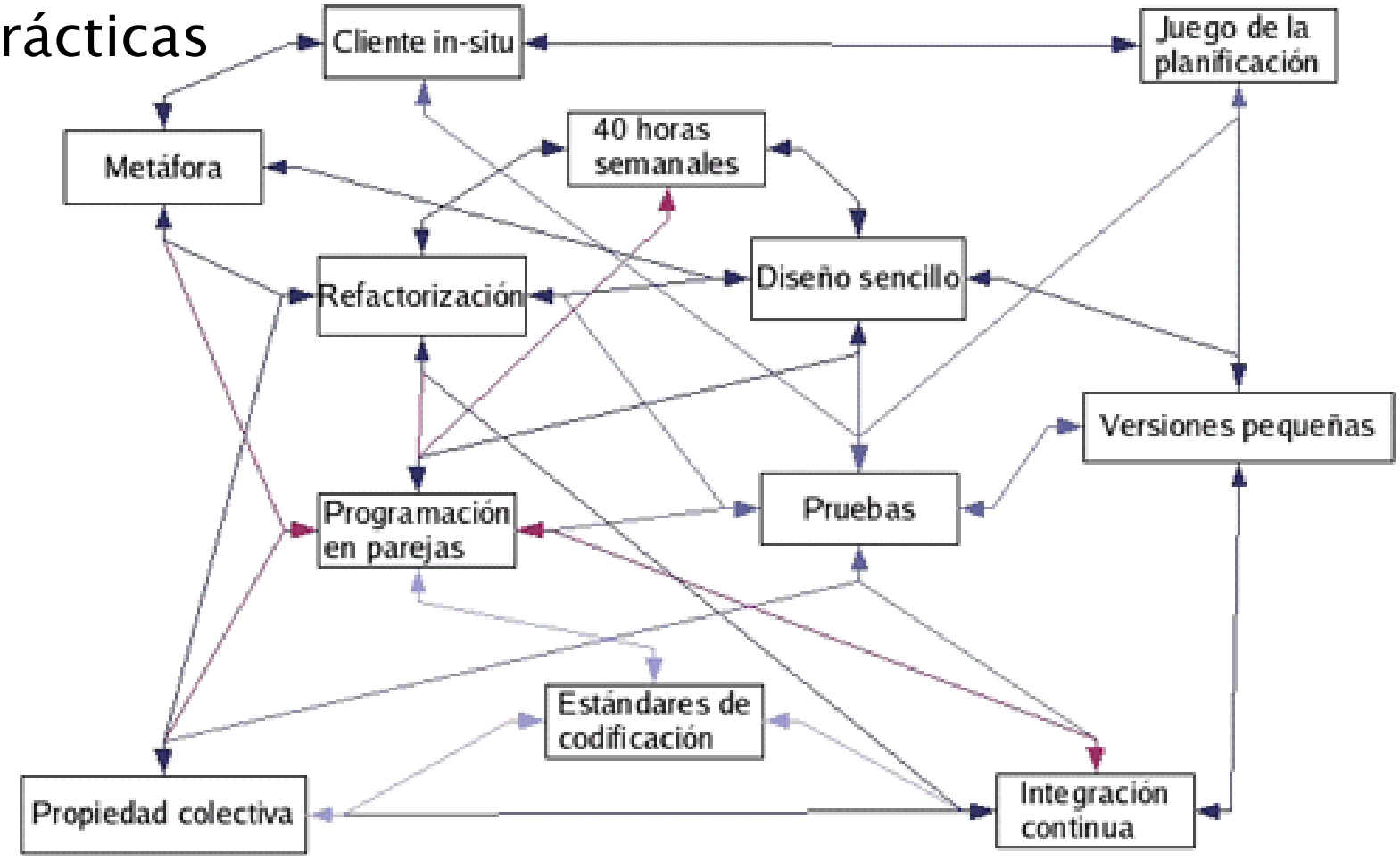


Programación Extrema (V)

- Prácticas:
  - Retroalimentación a escala fina
    - Desarrollo guiado por pruebas.
    - Juego Planificación.
    - On site Customer.
    - Programación en parejas.
  - Proceso continuo en lugar de por lotes
    - Integración continúa.
    - Recodificación.
    - Pequeñas versiones.
  - Entendimiento compartido
    - Diseño simple.
    - Metáfora del sistema.
    - Propiedad colectiva código.
    - Estándares de codificación.
  - Bienestar del programador
    - Semana de 40 horas.

Programación Extrema (VI)

12 Prácticas



Referencias

- **Ingeniería del software, un enfoque práctico (Roger S. Pressman). Séptima edición.**

- Piattini Velthuis, M.; Garcia Rubio, F.; Garzas Parra, J.; Genero Bocco, M. Medición y Estimación de Software: Técnicas y Métodos para mejorar la Calidad y Productividad. Madrid; Ed. Ra-Ma, 2008.