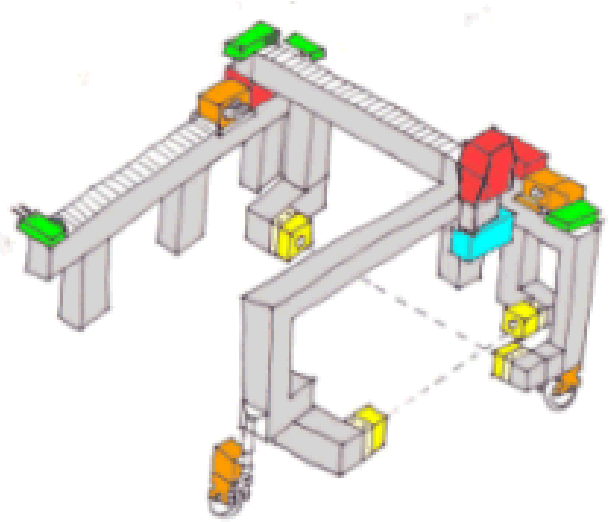
Tema 1: Introducción a la Ingeniería del Software



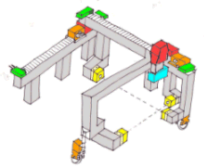
Contenido

1. El producto Software
2. El concepto de Ingeniería del Software
3. El proceso de desarrollo del Software

Contenido

1 El producto Software

1.1. Definición y tipos de software.
1.2. Propiedades del software.
1.3. Proceso de producción.
1.4. Problemas durante el proceso de producción.
1.5. Desastres ocasionados por el software.

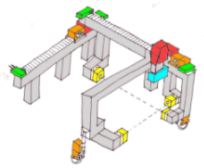


Definición de Software

- El software es un transformador de información, para ello: adquiere, gestiona, modifica, produce o trasmite información.

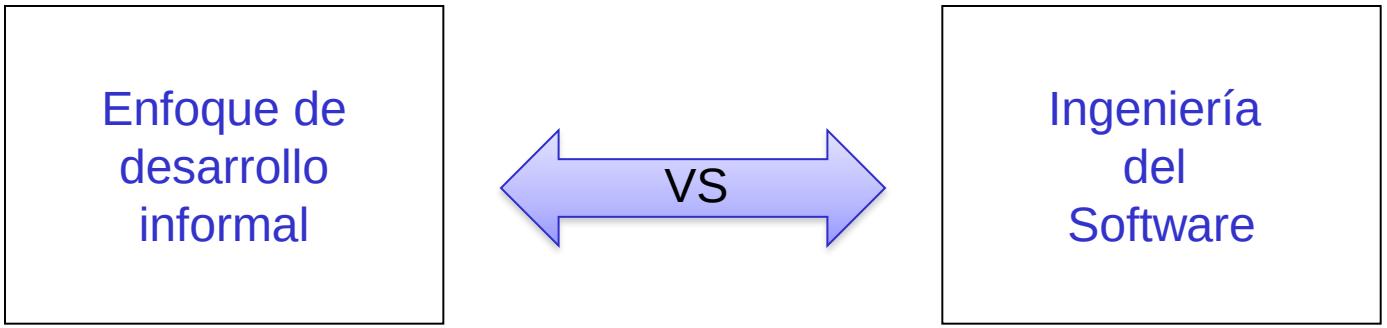
Software = Programa de computadora

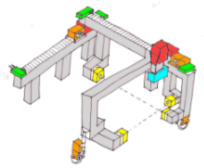
- No es sólo código y datos, también forma parte del software la documentación producida durante su desarrollo.



Problemas del Software

- Mal funcionamiento (Calidad)
- Hay que mantener el volumen de software existente
- Mantenimiento
- Hay una demanda creciente de nuevo software
- Adaptación a las nuevas tecnologías
- Incremento de la complejidad
- ¿Como desarrollamos software?

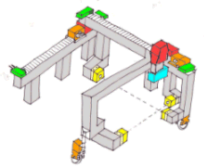




Tipos de software

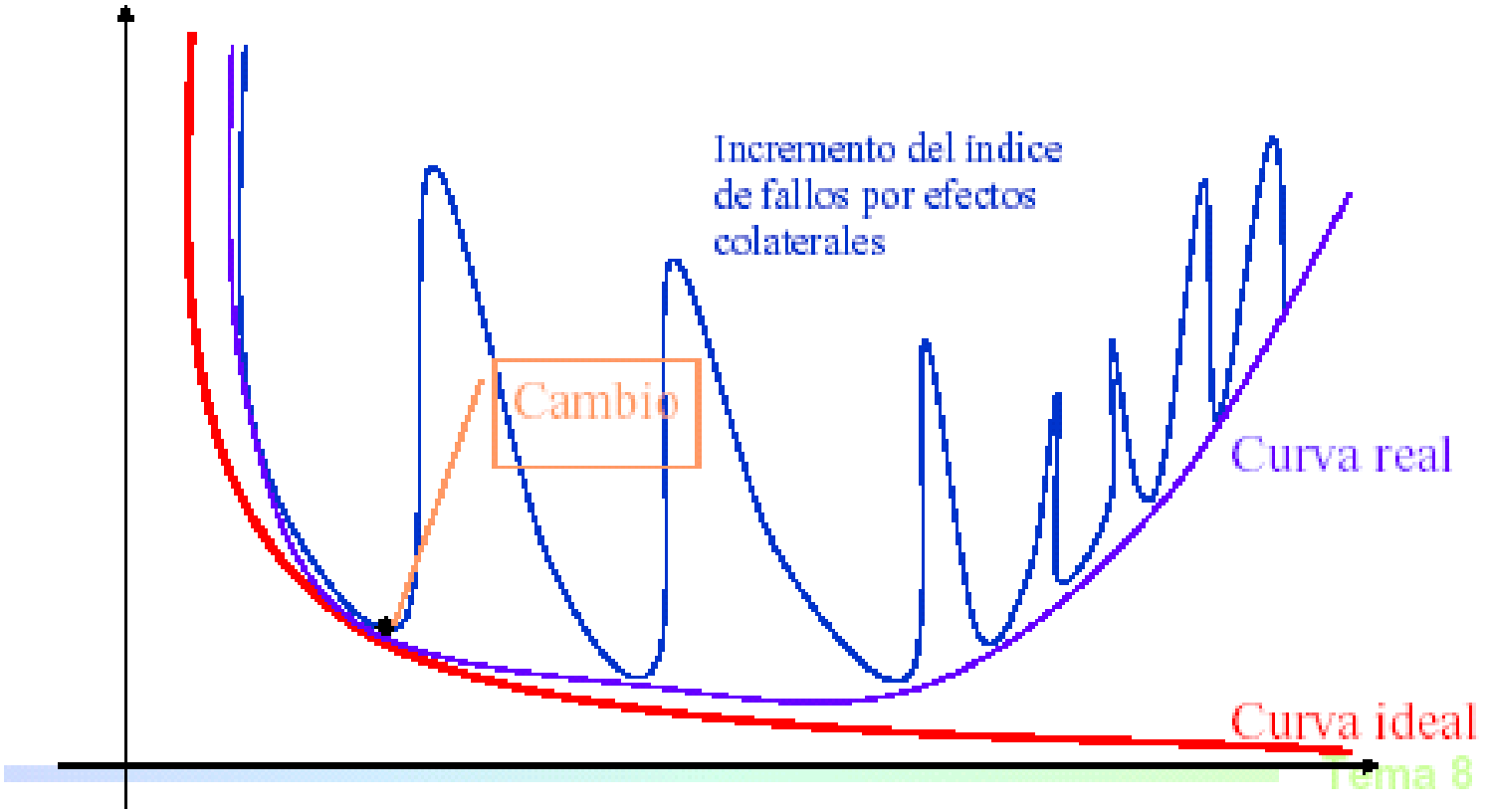
- Según el campo de aplicación
  - Software de sistemas
  - Software de tiempo real
  - Software de gestión
  - Software de ingeniería científica
  - Software empotrado
  - Software de inteligencia artificial

- Software genérico: Productos que son producidos por una organización para ser vendidos al mercado.
- Software hecho a medida: Sistemas que son desarrollados bajo pedido a un desarrollador específico.



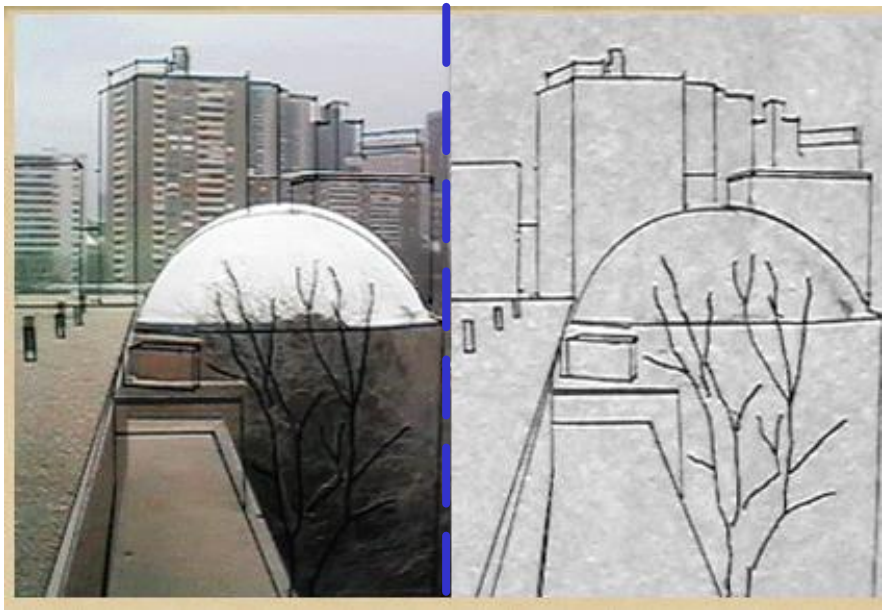
Características principales

1) El software es un producto lógico: **se desarrolla**, no se fabrica; **se deteriora**, no se estropea.



Características principales

2) El software crea modelos de la realidad.



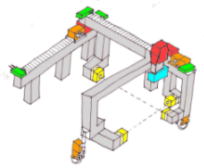
La realidad

El modelo

Dominio del Problema

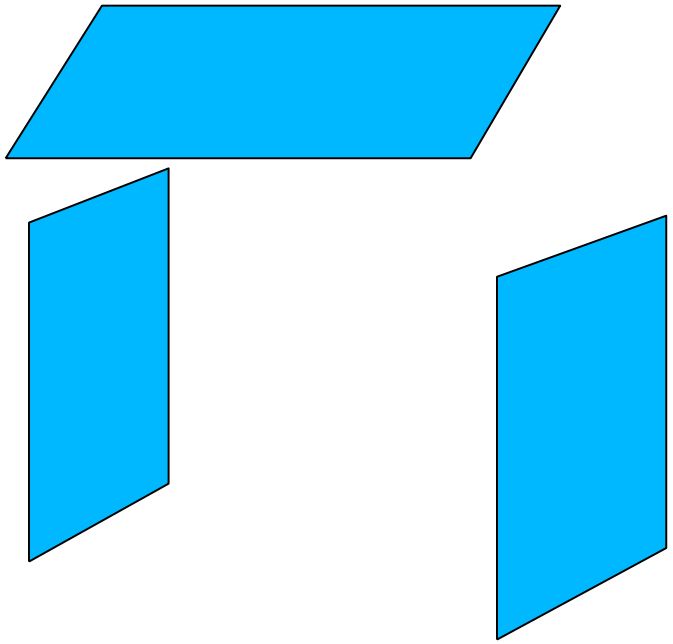
Dominio de la Solución

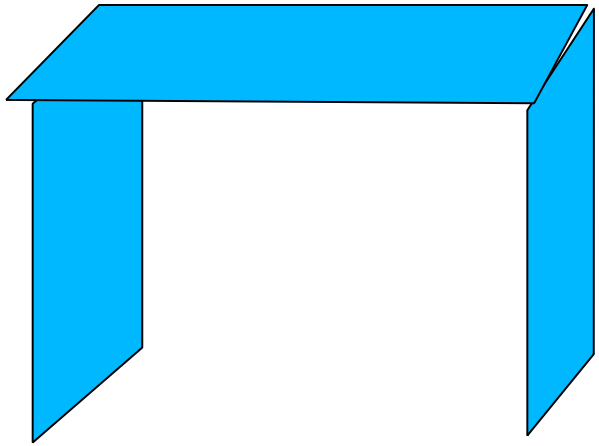
Modelo de funcionamiento
Modelo de la Información
…

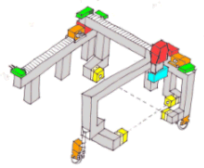


Características principales

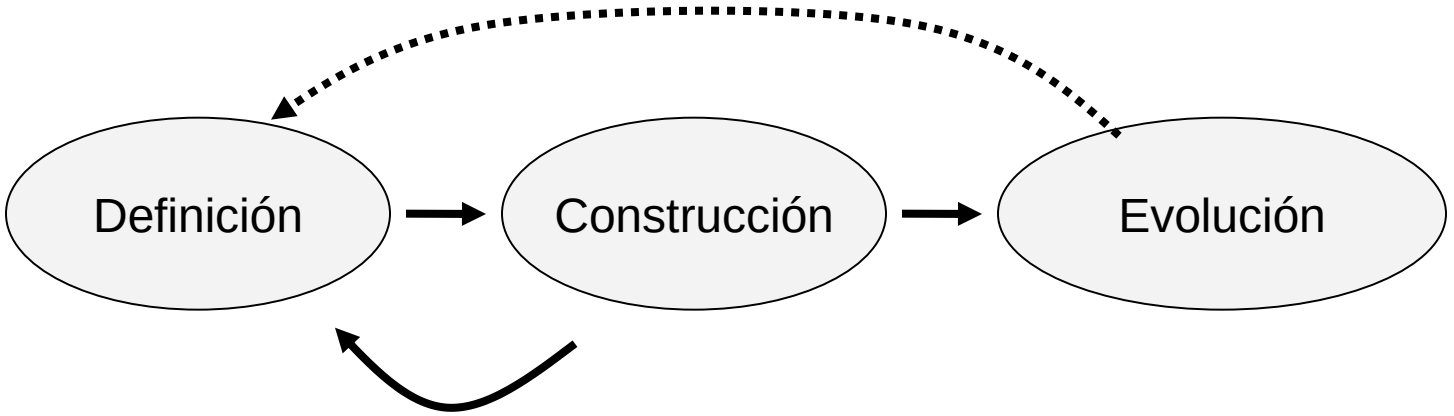
3) El software está formado por multiples piezas que deben **encajar perfectamente.**



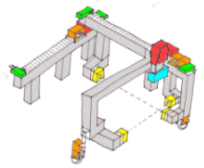




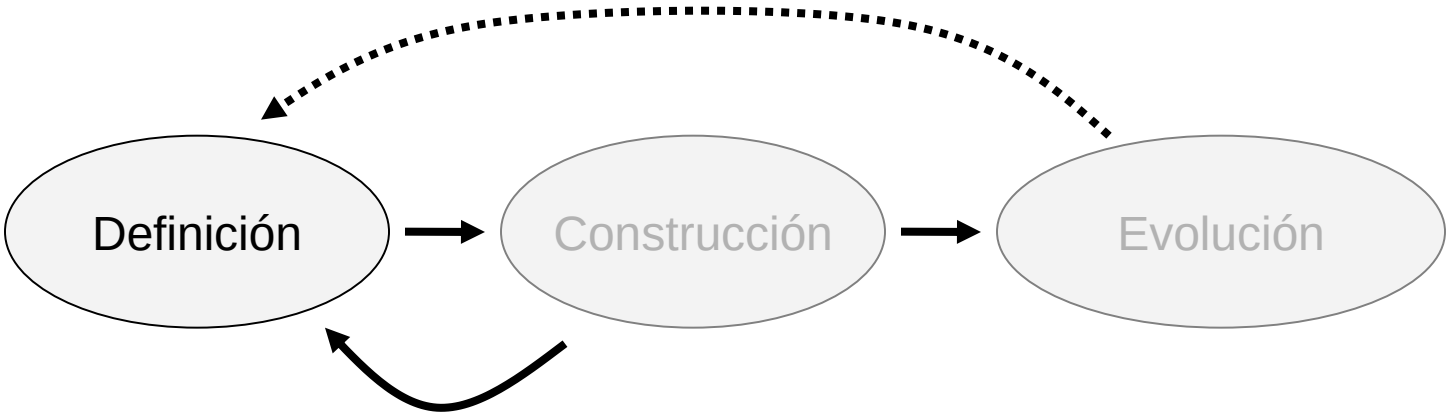
Proceso de producción



¿Qué? ¿Cómo? Cambios !

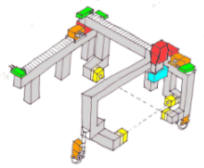


Proceso de producción

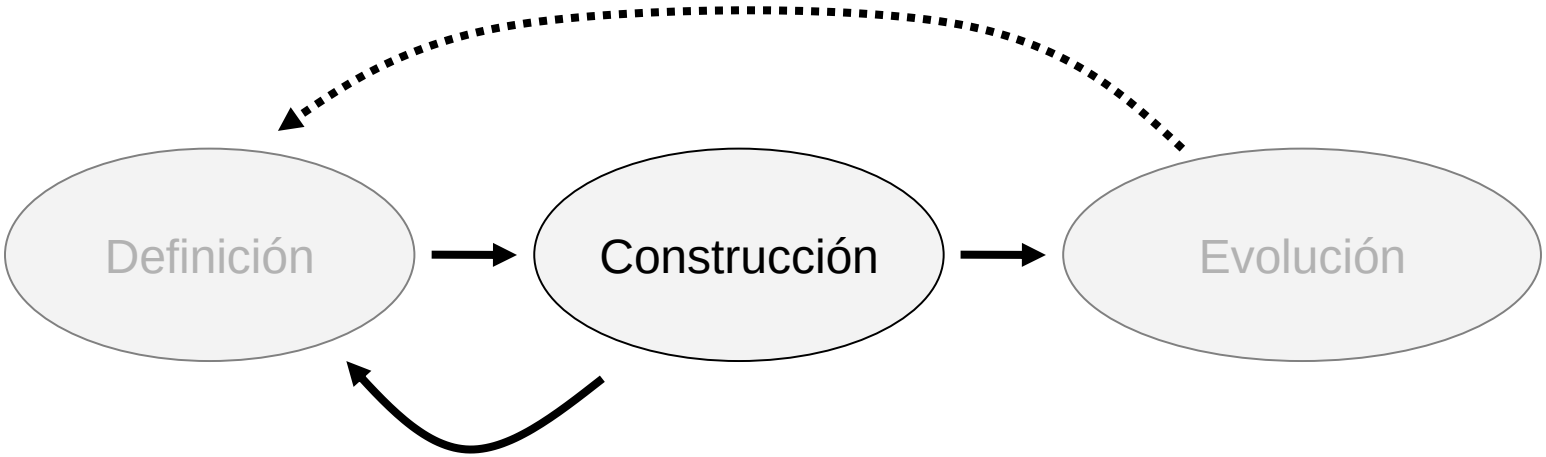


¿Qué vamos a construir?

Tareas a realizar: Ingeniería de sistemas
Ingeniería de requisitos
Planificación de proyectos

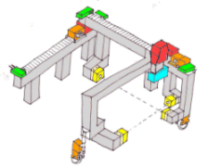


Proceso de producción

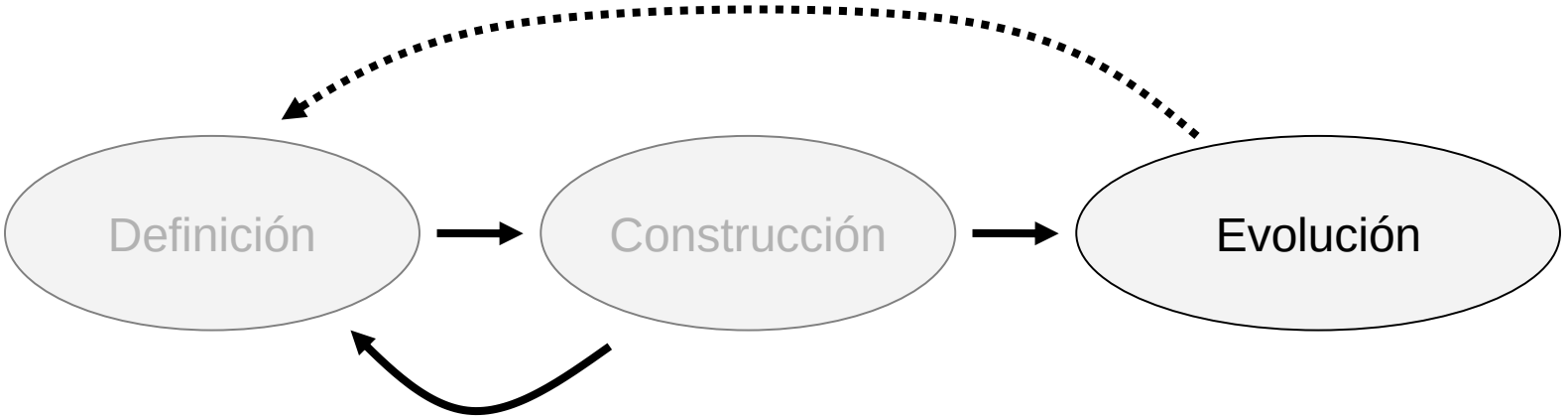


¿Cómo lo vamos a construir?

Tareas a realizar:
- Diseño del software
- Generación del código
- Prueba del software



Proceso de producción

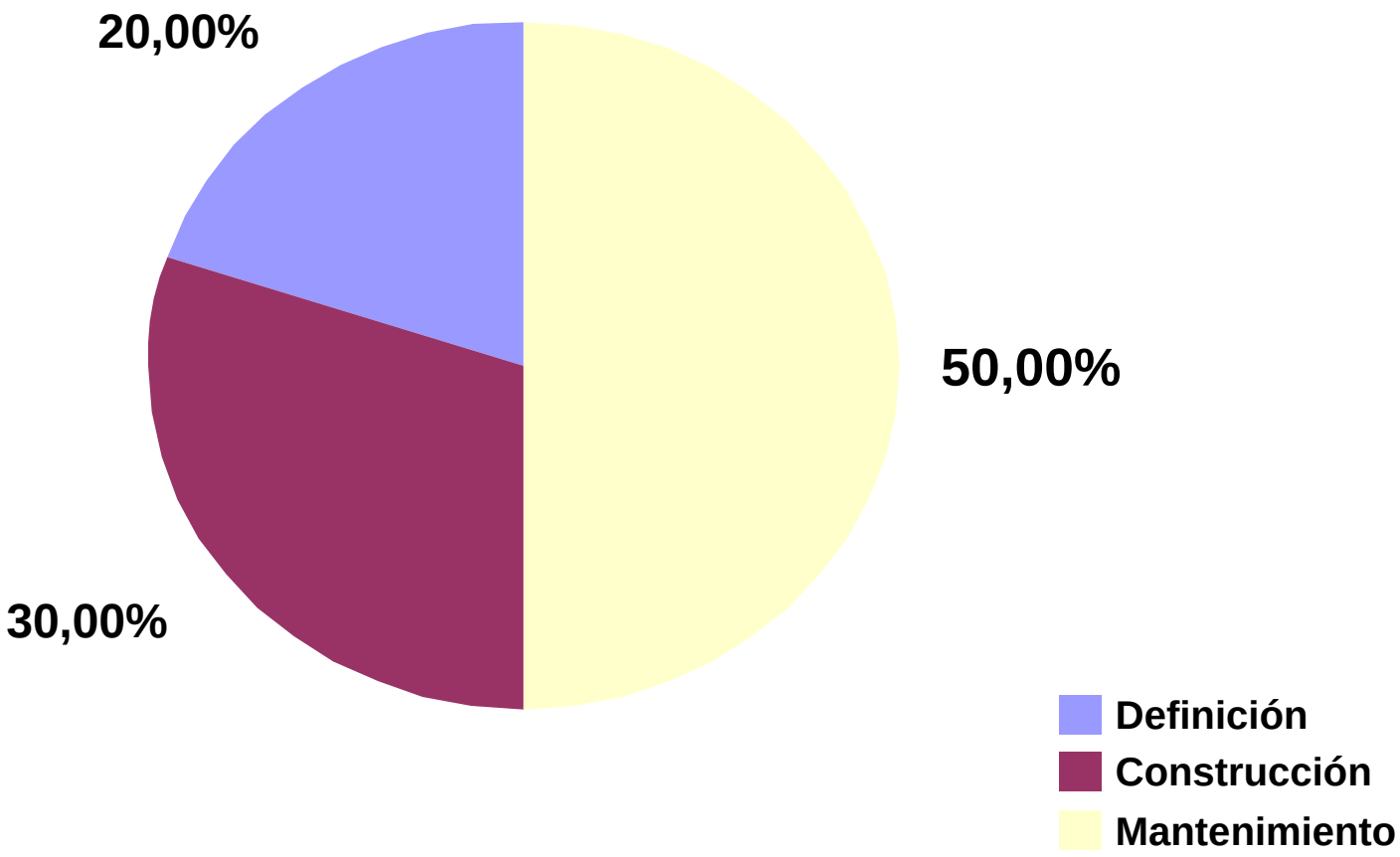


¿Que va a cambiar en el software?

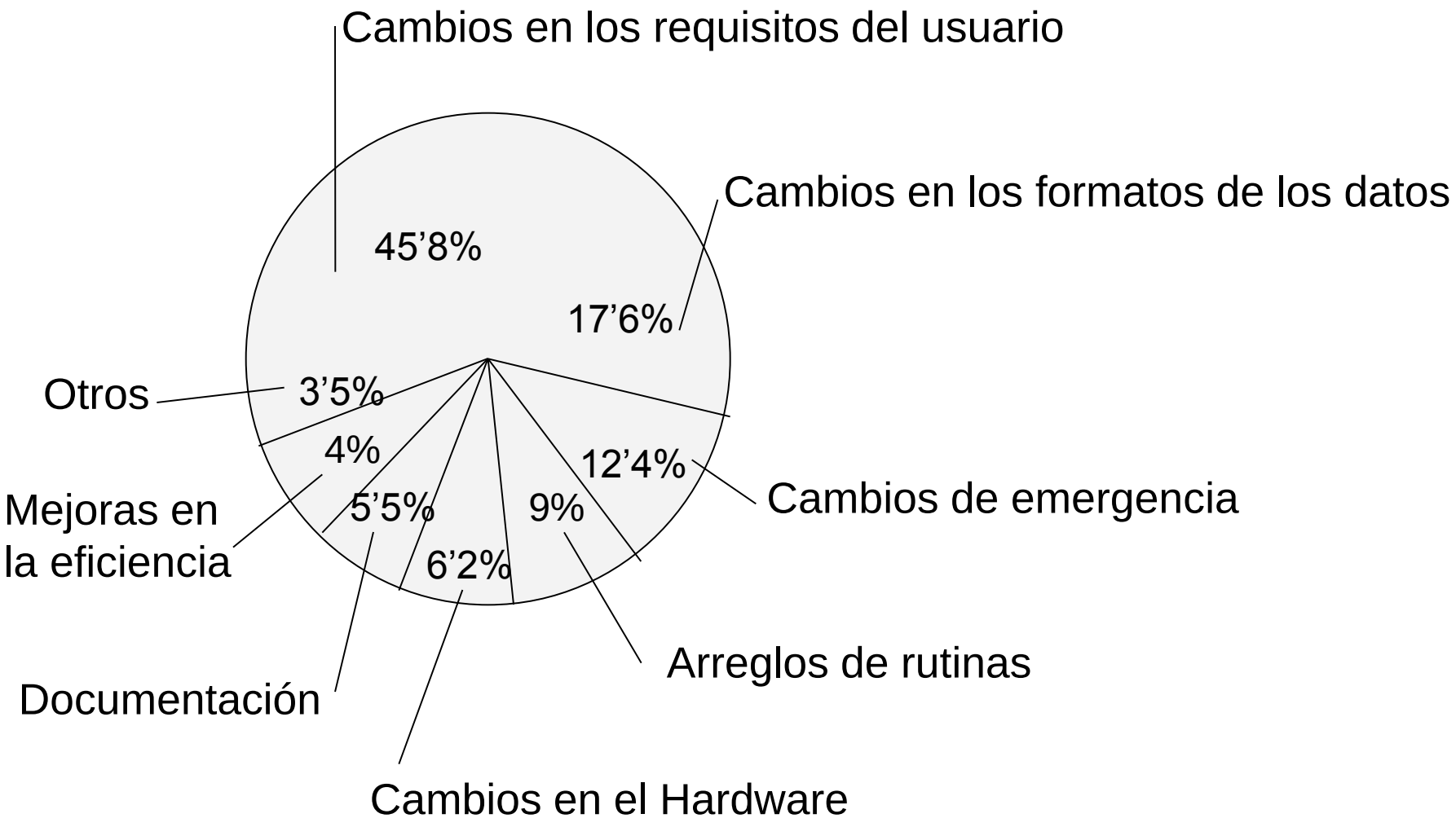
Tareas a realizar:
- Corrección
- Adaptación
- Mejora
- Prevención

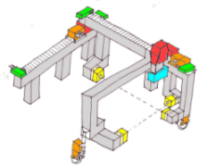
Proceso de producción

- Esfuerzo requerido por etapas



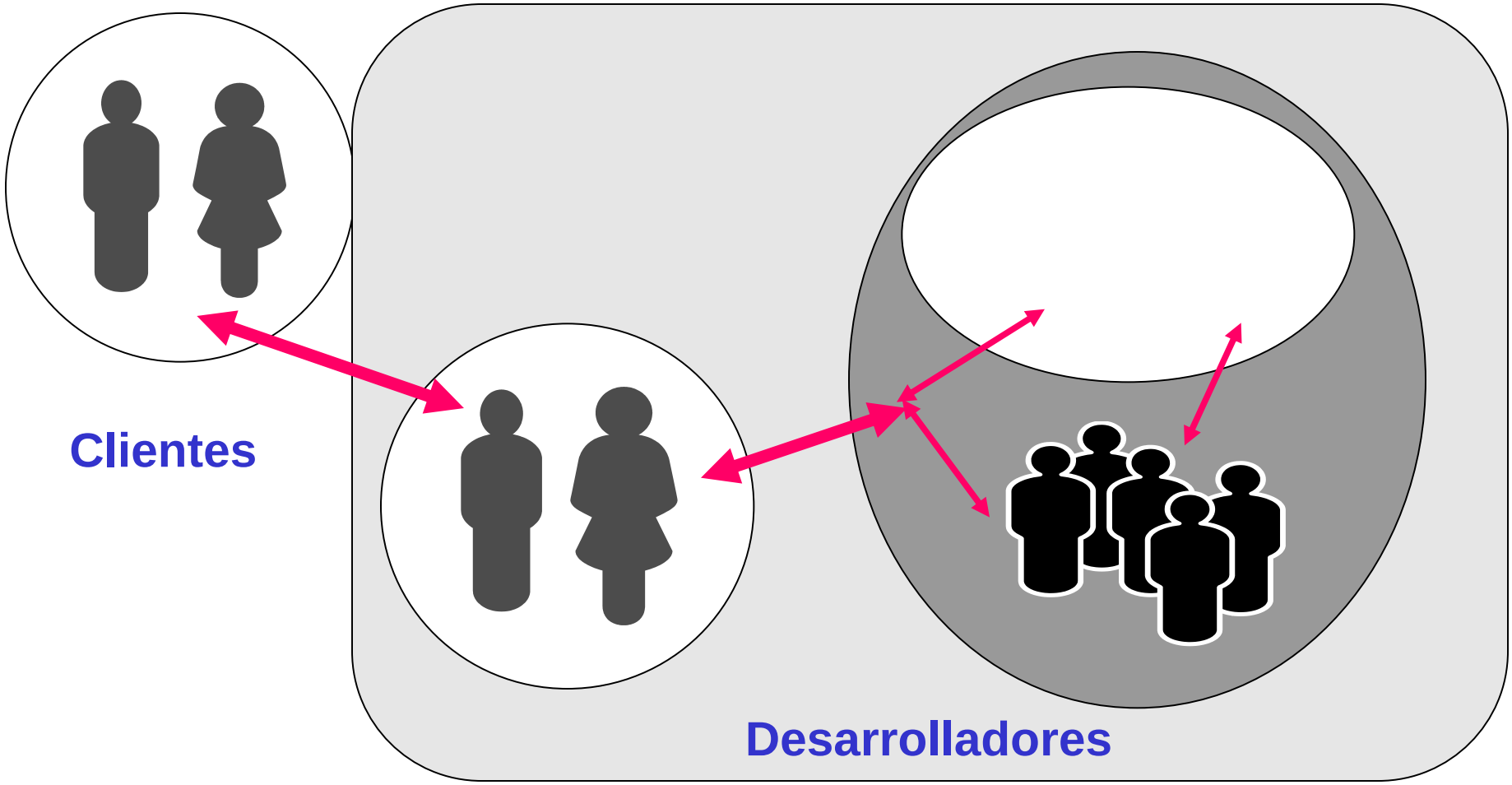
Mantenimiento





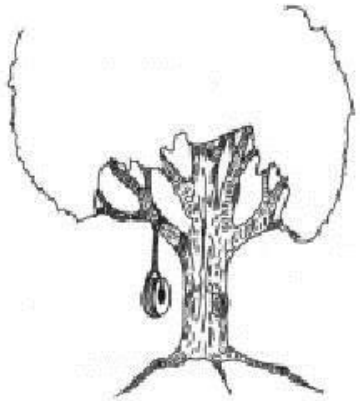
Problemas

1) Comunicación entre personas.

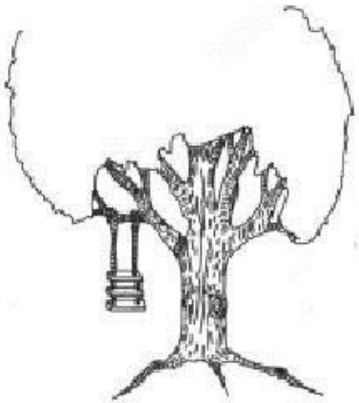


Problemas

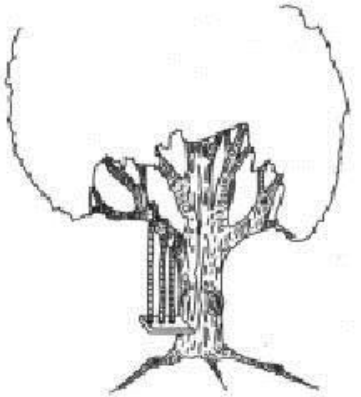
1) Comunicación entre personas.



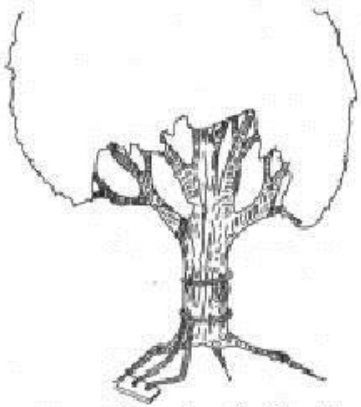
Lo que necesita el usuario



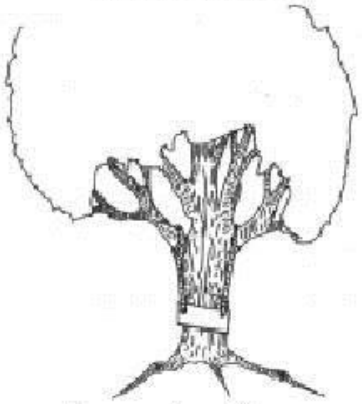
Lo que entiende el gestor de proyectos



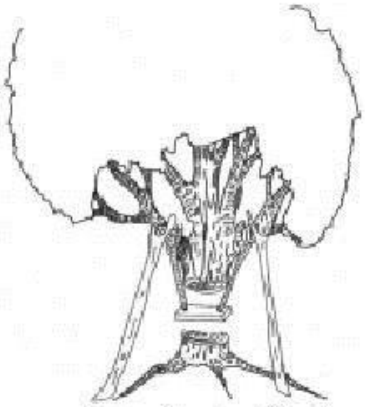
Lo que entiende el analista



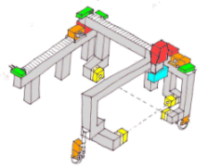
Lo que entiende el diseñador



Lo que hacen los Programadores



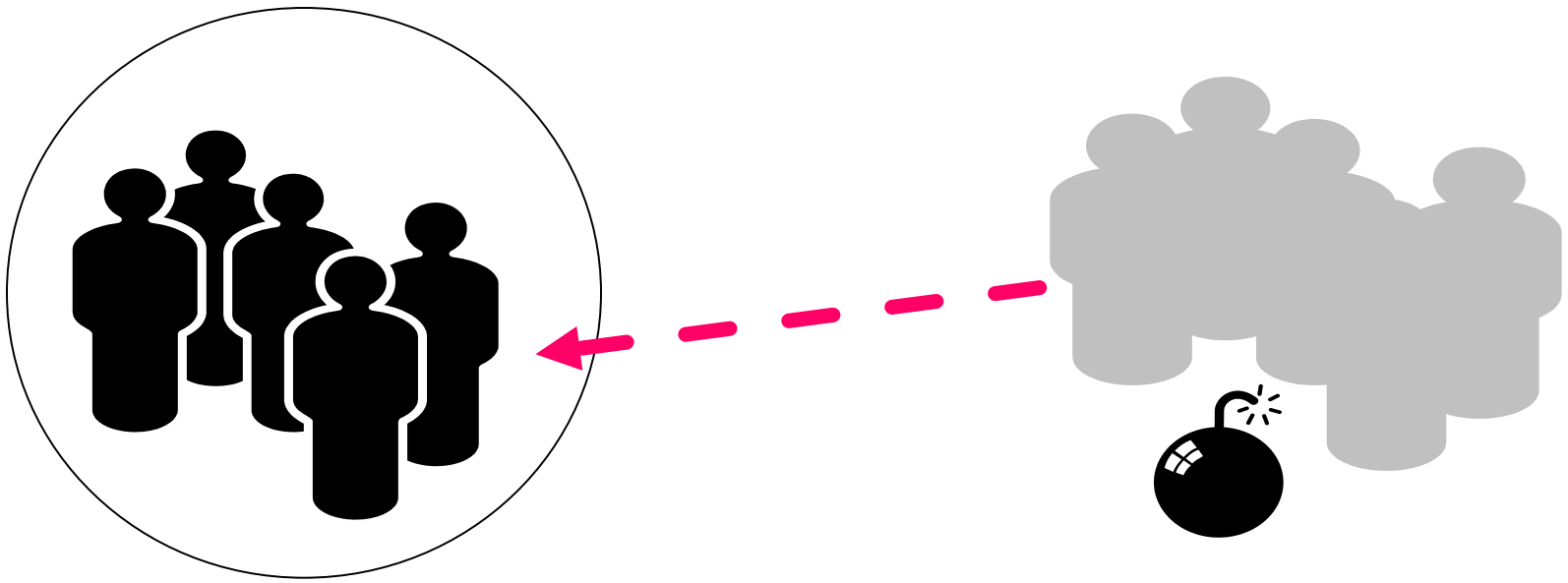
Resultado Final



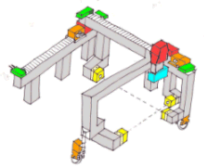
Problemas

2) Incumplimiento de la planificación.

¿ solución: *horda mongoliana ?*

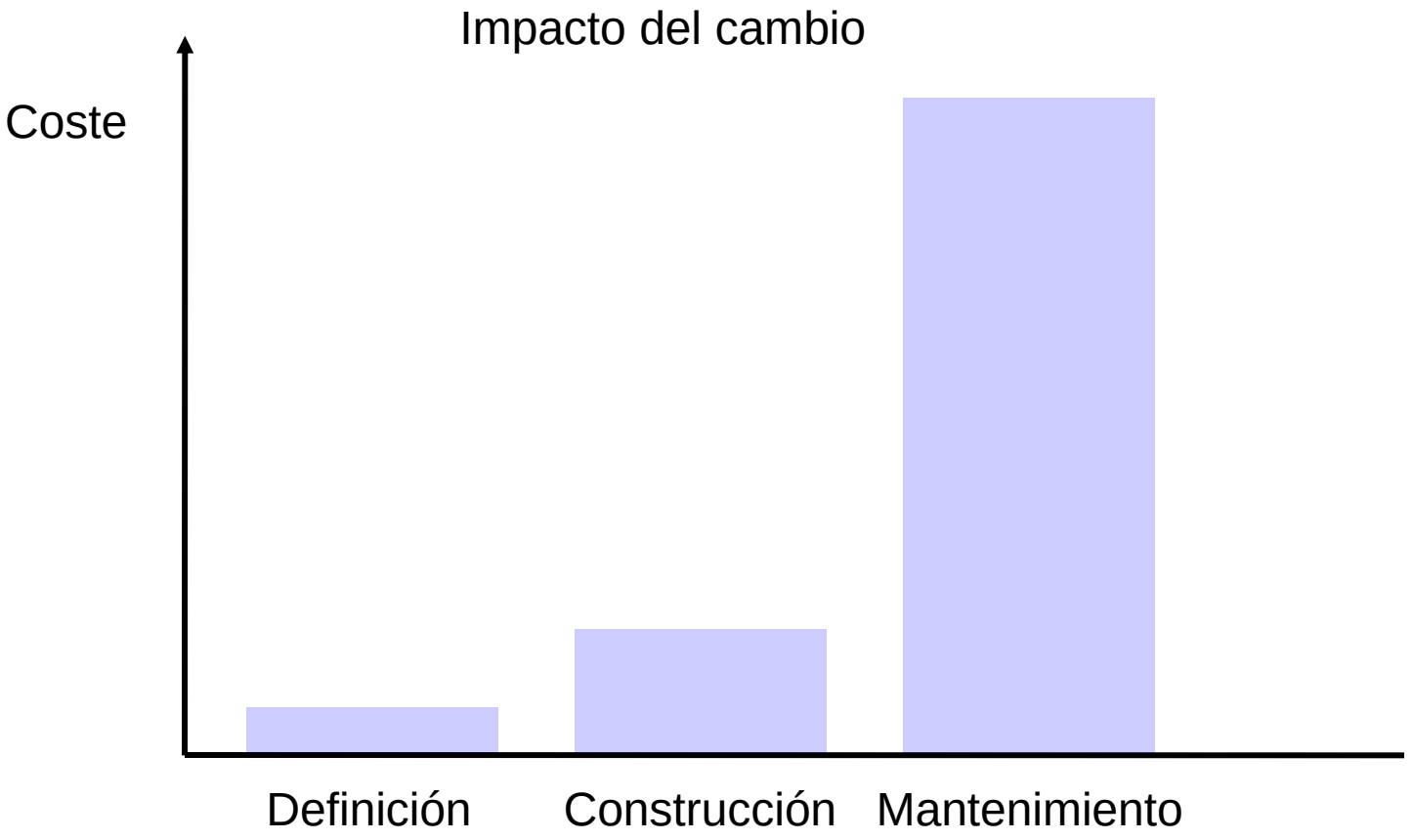


Equipo de desarrollo



Problemas

3) Incorporar cambios durante etapas avanzadas del proceso.

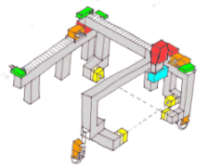


Contenido

2 El concepto de Ingeniería del Software

2.1. Definición de IS.

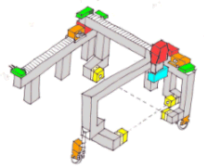
2.2. Terminología usada en IS.



Definición de IS

- "Establecimiento de los principios y métodos de la ingeniería a fin de obtener software de modo rentable que sea fiable y trabaje en máquinas reales". (Friz Bauer, 1972).

- "Aplicación práctica del conocimiento científico en el diseño y construcción de programas de computadora y la documentación asociada y requerida para el desarrollo, operación y mantenimiento del programa". (B. Bohem, 1976).



Definición de IS

- "Estudio de los principios y métodos para el desarrollo y mantenimiento de sistemas software". (M.V. Zelkowitz, 1978).

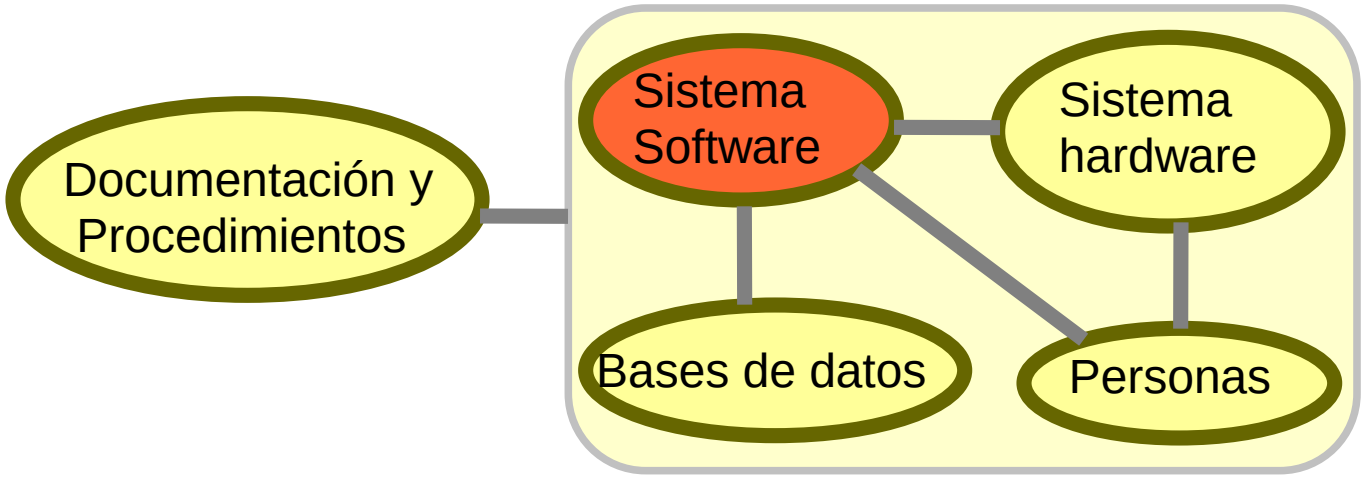
- "Aplicación de un enfoque sistémico, disciplinado y cuantificable al desarrollo, operación y mantenimiento del software; es decir, aplicación de la ingeniería al software". (standard - IEEE, 1993).

- Conjunto de teorías, métodos e instrumentos (tecnológicos y organizativos) que permitan construir sistemas software con las características de calidad deseadas". (M. Marré, 2002 apuntes de clase).

Terminología usada en IS

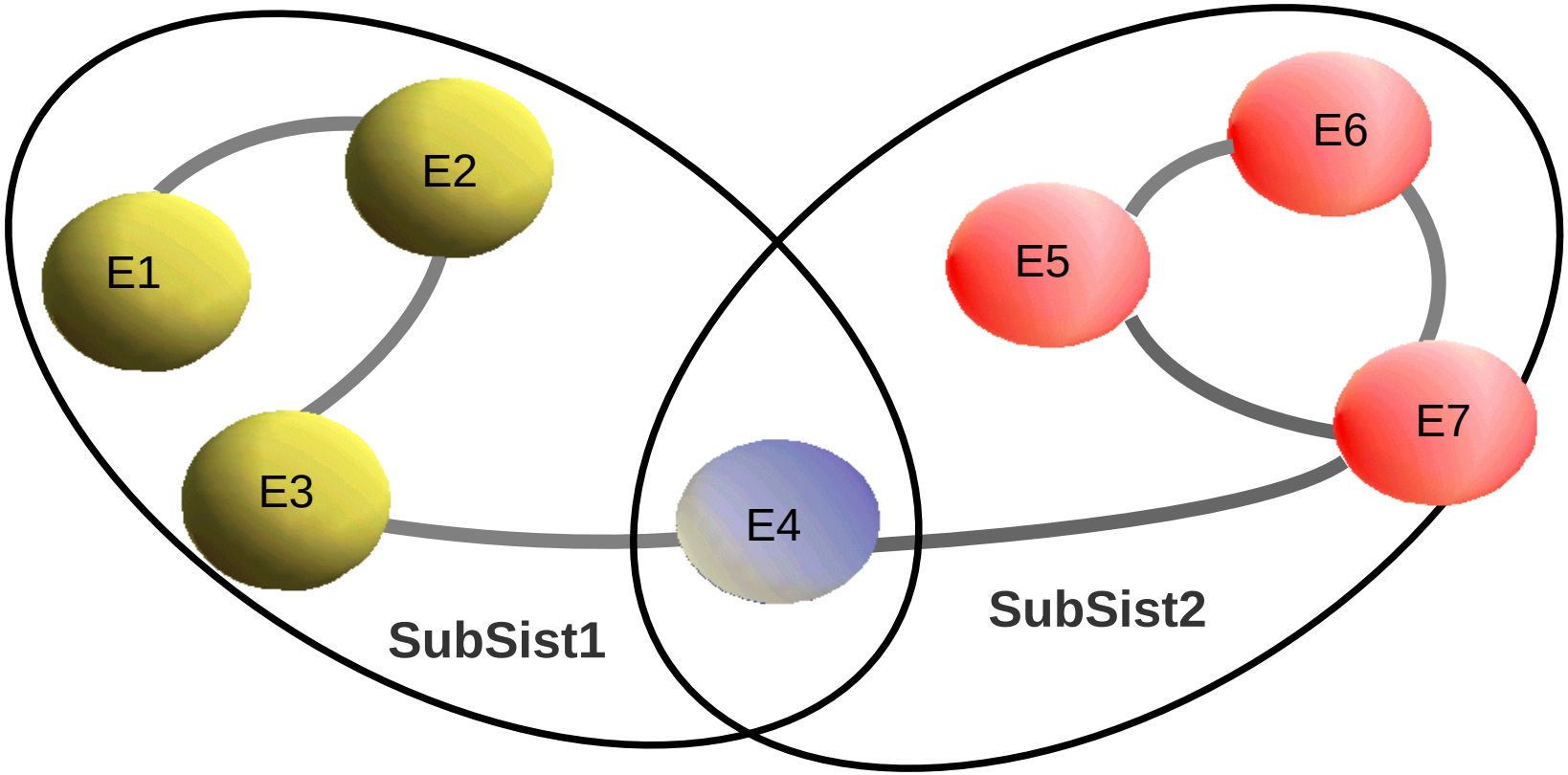
Sistema**:** Conjunto de elementos relacionados entre si y con el medio, que forman una unidad o un todo organizativo.

Sistema basado en computadora: Conjunto o disposición de elementos organizados para cumplir una meta predefinida al procesar información.



Terminología usada en IS

Sistema Software**:** Conjunto de piezas o elementos software relacionados entre si y organizados en subsistemas.

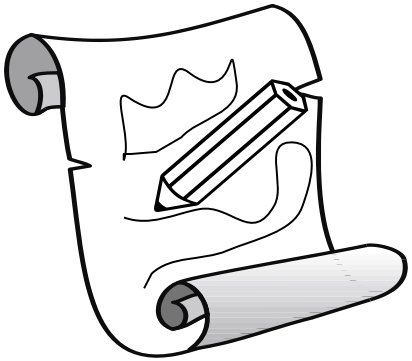


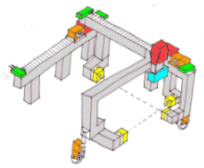
Terminología usada en IS

- Modelo: Representación de un sistema en un determinado lenguaje. De un mismo sistema se pueden construir muchos modelos.

Terminología usada en IS

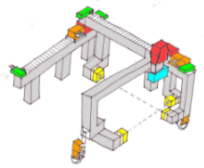
- Principio: Elementos adquiridos mediante el conocimiento, que definen las características que debe poseer un modelo para ser una representación adecuada de un sistema.“
- Herramienta: Instrumentos que permiten la representación de modelos.
- Técnica: Modo de utilización de las herramientas.



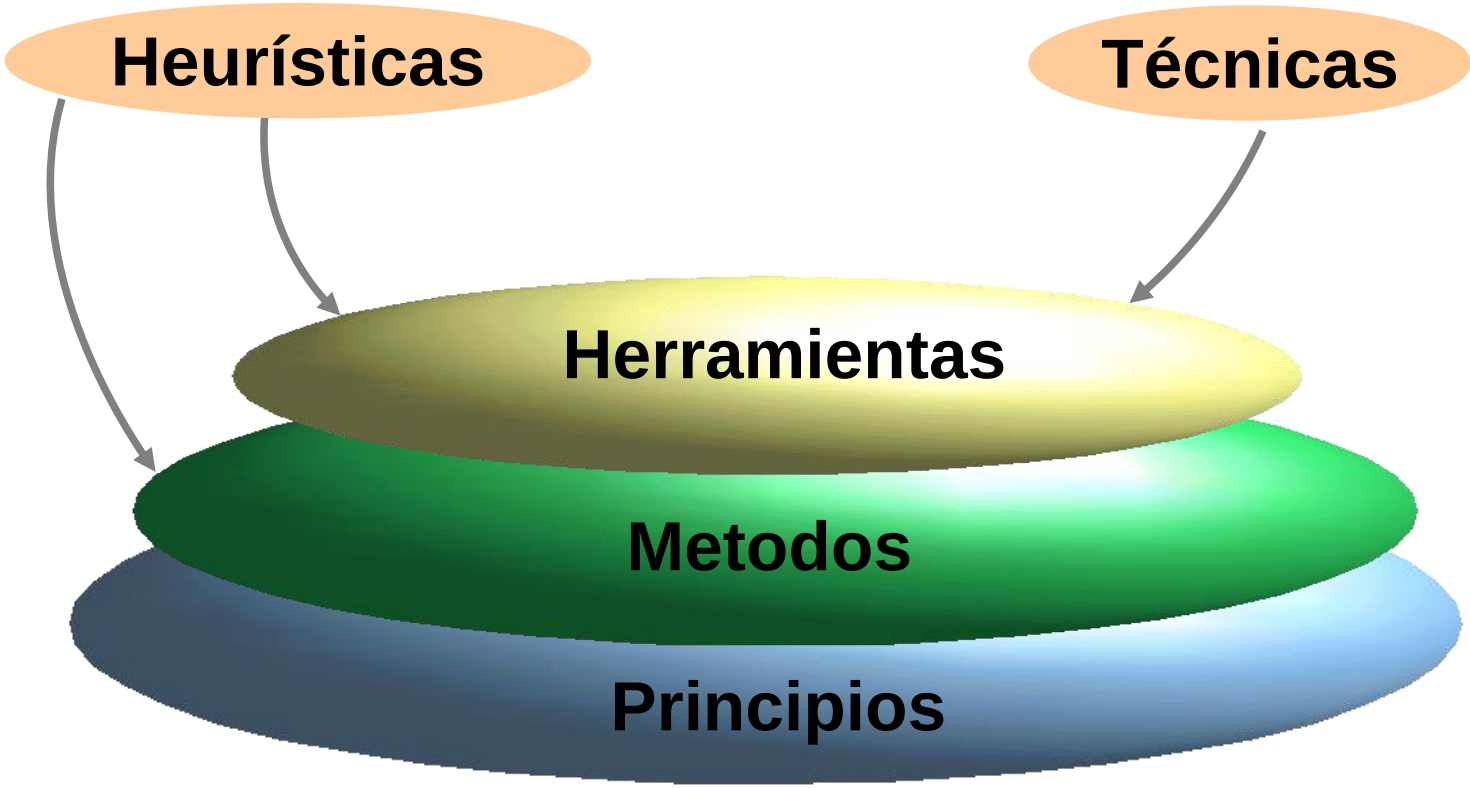


Terminología usada en IS

- Heurísticas: Conjunto de reglas empíricas, que al ser aplicadas producen modelos que se adecuan a los principios.
  - "no usar materiales flexibles para representar la maqueta de un edificio"
- Método: Secuencia de actividades, para la obtención de un producto (modelo), que describen como usar las herramientas y heurísticas.



Resumen



Definición de IS: Estudio de los principios, metodologías y herramientas que permiten o facilitan el desarrollo y mantenimiento de sistemas software

Contenido

3 El proceso de desarrollo del software

3.1. El proceso de desarrollo

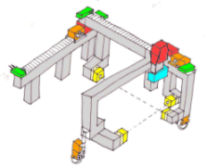
3.2. Modelo secuencial: Ciclo de vida clásico

3.2. Modelos Iterativos

- Modelo incremental parcial
- Prototipado
- Modelo en espiral

3.3. Proceso unificado

3.4. Métodos ágiles



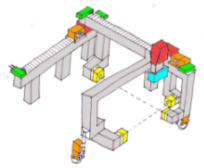
El proceso de desarrollo del software

Proceso de desarrollo o ciclo de vida del software: Estrategia que nos define la división y ubicación temporal de las etapas realizadas durante el desarrollo del software.

Una **etapa** se caracteriza por las tareas que se realizan en ella y por el producto o modelo (documento, artefacto...) que se obtiene.

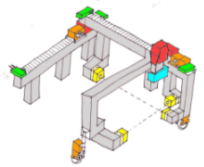
Existen diversos **Modelos** de proceso o **de ciclo de vida**, según las etapas consideradas, posición relativa de las mismas y las tareas a realizar en ellas. **Clasificación** general de estos modelos:

- Secuenciales: hasta que no se acaba totalmente una etapa no comienza la siguiente
- No secuenciales o iterativos: el desarrollo se considera un proceso evolutivo en el que se parte de un software con funcionalidad mínima y se desarrolla progresivamente



Etapas principales

- Especificación de requisitos: Análisis del problema a resolver. Documento en el que se dice qué debe hacer el sistema software.
- Diseño: Búsqueda de la solución. Descripción de los componentes, sus relaciones y funciones que dan solución al problema.
- Implementación. Traducción del diseño a un lenguaje de programación entendible por una máquina. El código
- Control de la calidad: debe realizarse durante todo el proceso de desarrollo. Revisiones de todo lo que se va obteniendo junto con la prueba del código.
- Mantenimiento o evolución - reparar fallos en el sistema cuando sea descubiertos o adaptar el software a los nuevos entornos.
- Planificación: Estimar el tiempo y los costes de desarrollo del software



Ciclo de vida clásico

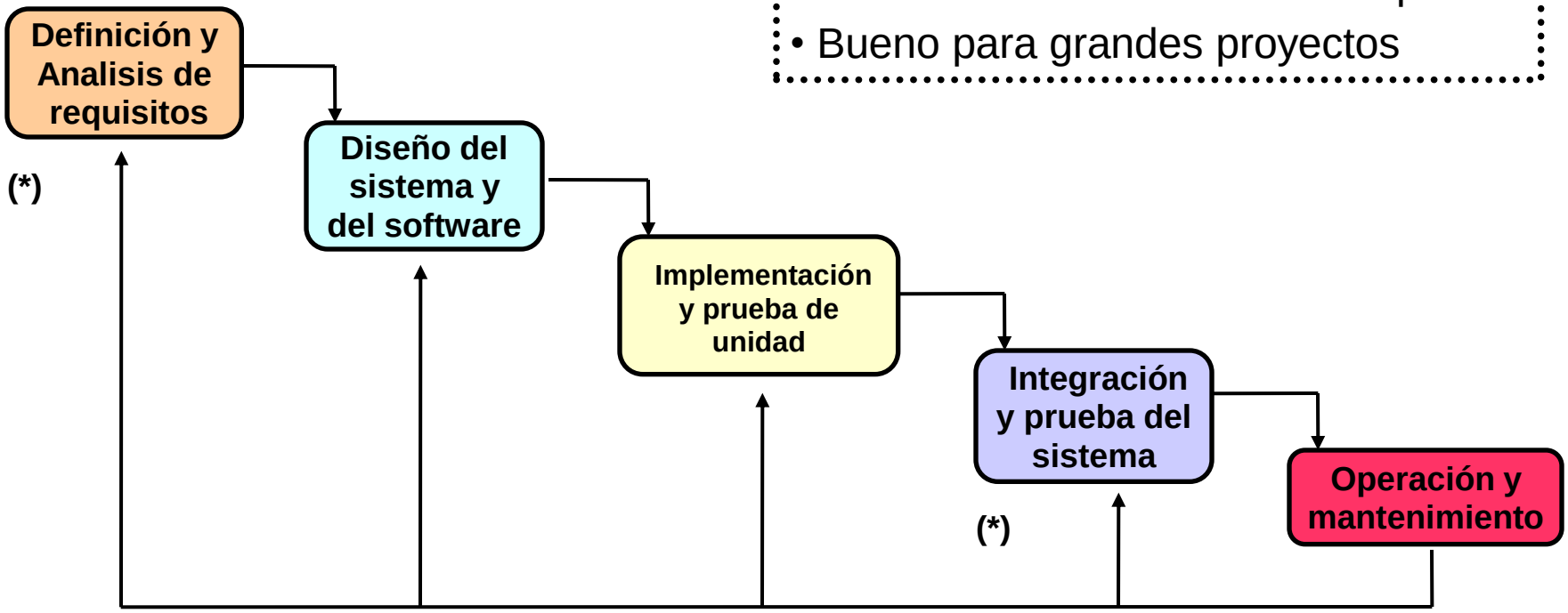
Características

- El modelo clásico es un pulcro, conciso y lógico encadenamiento de etapas que deben ocurrir una tras otra para obtener el producto final.
- Cada fase se caracteriza por una actividad fundamental (que da nombre a esa fase) y otras actividades parciales.
- Cada una de estas fases posee unos objetivos concretos, y una serie de productos que se tendrán que obtener al final de la misma.

Ciclo de vida clásico

- Ejemplo: Modelo en cascada de Boehm

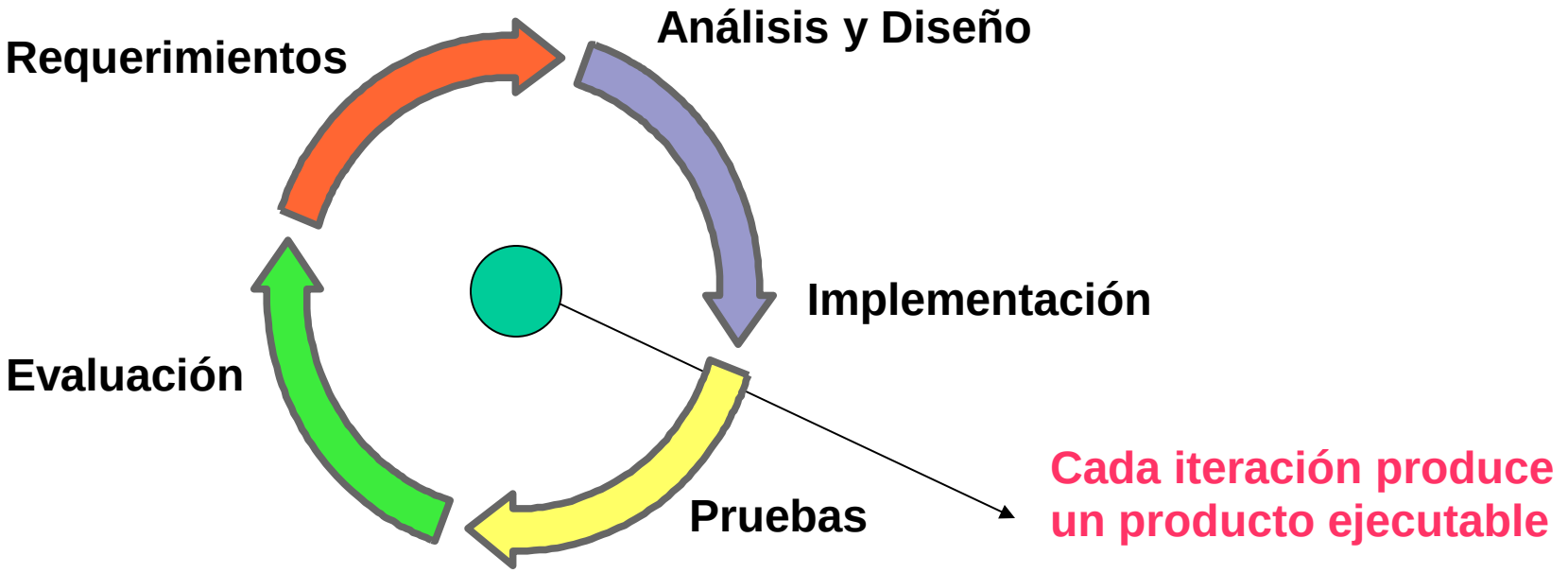
Características:
- Secuencial estricto.
- Poca comunicación con el cliente **(*)**.
- Control de calidad en cada etapa.
- Bueno para grandes proyectos

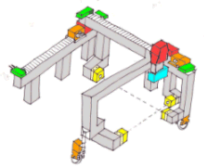


Modelos iterativos

Características:

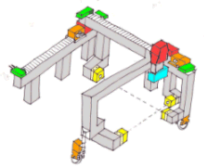
- Permite un entendimiento incremental del problema (refinamientos sucesivos)
- Habilita una fácil retroalimentación de usuario
- Objetivos parciales y Metas específicas
- El progreso es medido conforme avanzan las implementaciones



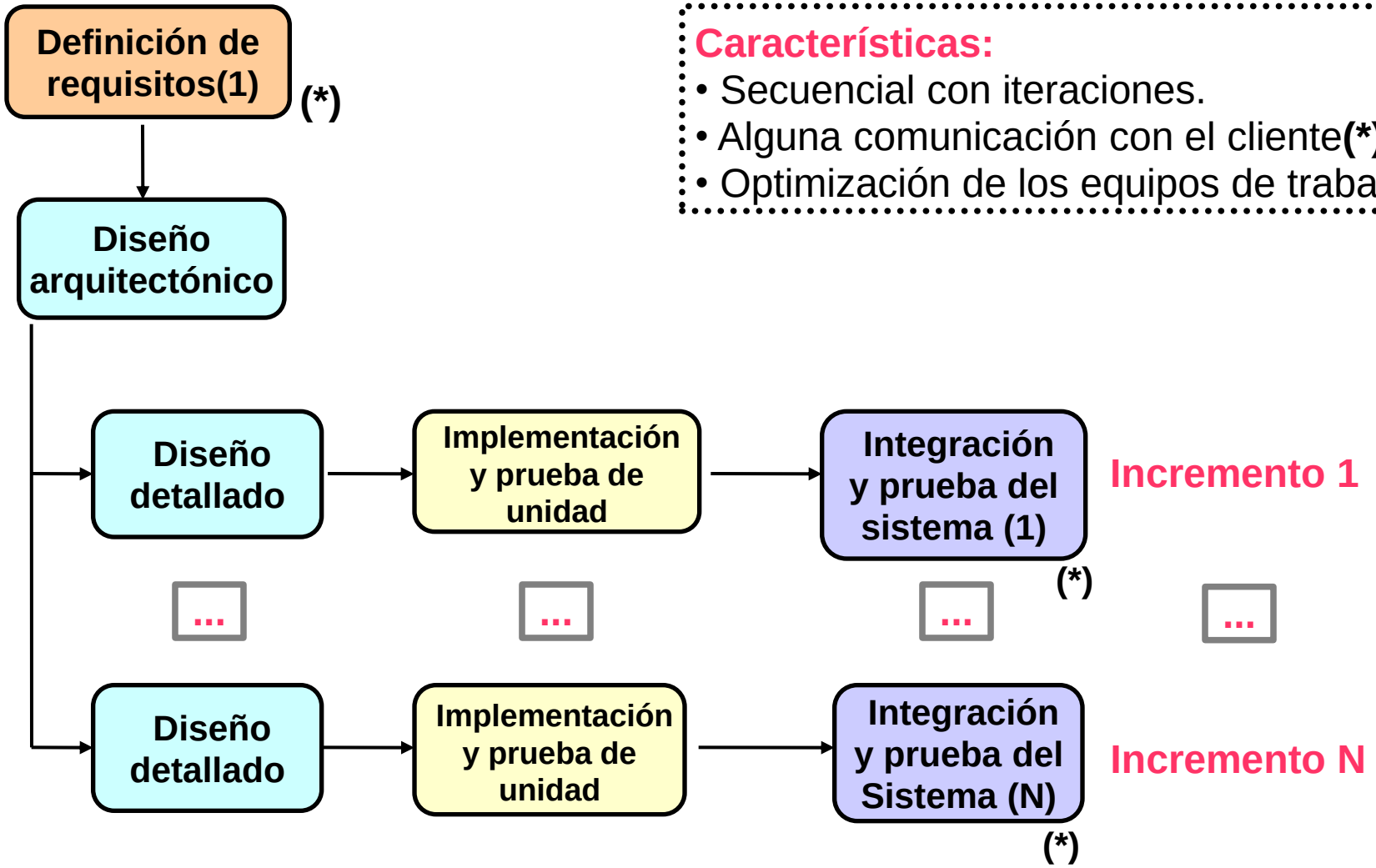


Beneficios del proceso iterativo

- Se reducen, tan pronto como sea posible, los riesgos altos (técnicos, de requisitos, de usabilidad ...)
- Progreso visible en las primeras etapas.
- Se produce una temprana retroalimentación. El producto se refina y terminara mas cerca de las necesidades reales del usuario.
- Manejo de la complejidad (pasos cortos y sencillos).
- El conocimiento que se adquiere en una iteración puede ser usado en el resto.
- Se involucra continuamente al usuario (evaluación, retroalimentación y obtención y refinamiento de requisitos).

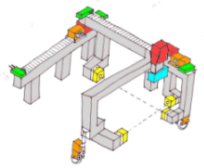


Modelo incremental parcial



Características:

- Secuencial con iteraciones.
- Alguna comunicación con el cliente**(*)**.
- Optimización de los equipos de trabajo.



Prototipado

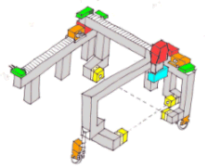
Prototipo: Versión del software que se utiliza para demostrar los conceptos, probar las opciones de diseño y de forma general, comprender mejor el problema y sus posibles soluciones. Producto de funcionamiento limitado en cuanto a la capacidad, confiabilidad o eficiencia (¿Calidad del producto?).

Tipos de prototipos:

- **Prototipo evolutivo**: El prototipo se transforma en etapas sucesivas hasta convertirse en el sistema final a entregar.
- **Prototipo desechable**: El prototipo se construye y evalúa con el usuario y posteriormente se desecha.

Se usa para:

- Facilitar la obtención y validación de requisitos (desechable).
- Estudios de viabilidad (desechable).
- Propuestas de soluciones alternativas a un problema (desechable).
- En casos muy concretos como producto final (evolutivo).



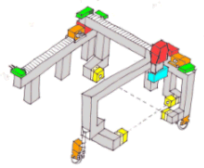
Prototipado

Ventajas:

- Se identifican las malas interpretaciones entre usuarios y desarrolladores
- Ayuda durante el proceso de validación (completitud)
- Demostramos rápidamente la viabilidad del proyecto
- Se reducen los costos de desarrollo
- El cliente ve una versión inicial del software al comienzo del proyecto

Inconvenientes:

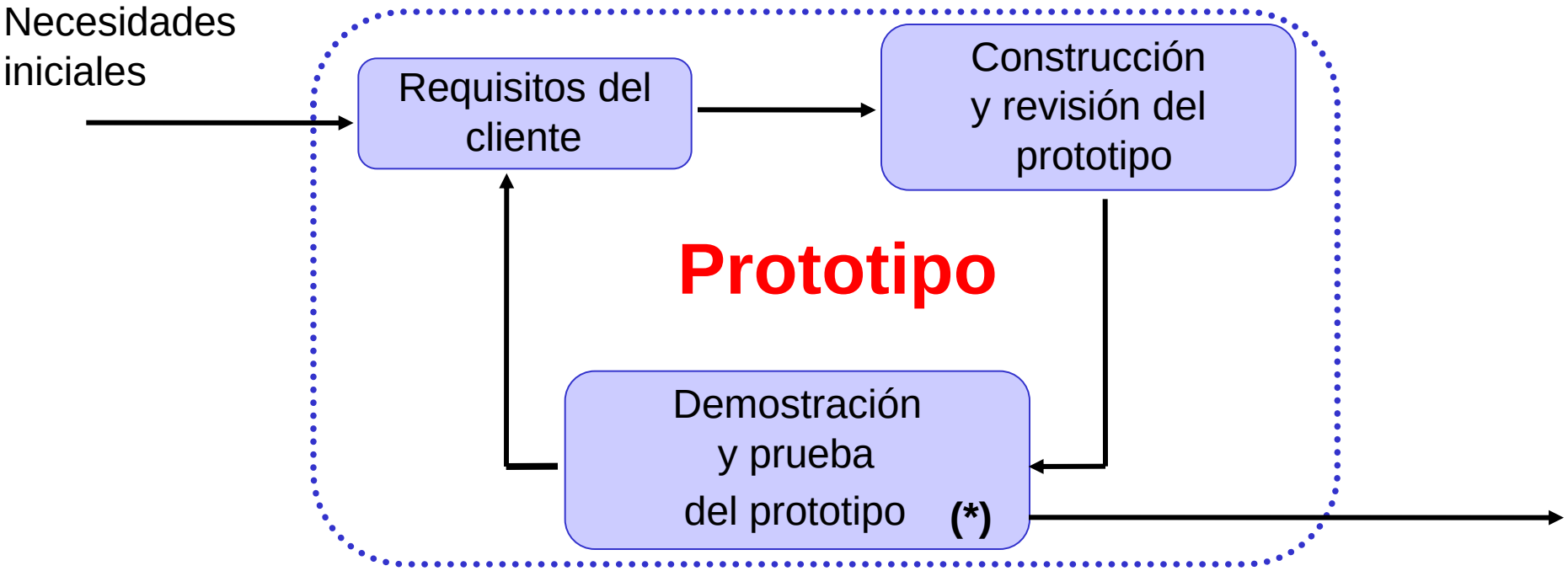
- Crea falsas expectativas por parte del cliente/usuario.
- Decisiones de diseño del prototipo que pasen a formar parte del producto final.

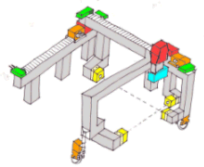


Modelos iterativos. Prototipado

Carácterísticas

- Buena comunicación con el cliente **(*)**
- Bueno para identificar requisitos poco claros
- El software satisface las necesidades del cliente ya que el prototipo las satisface

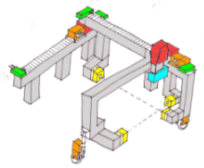




Prototipado

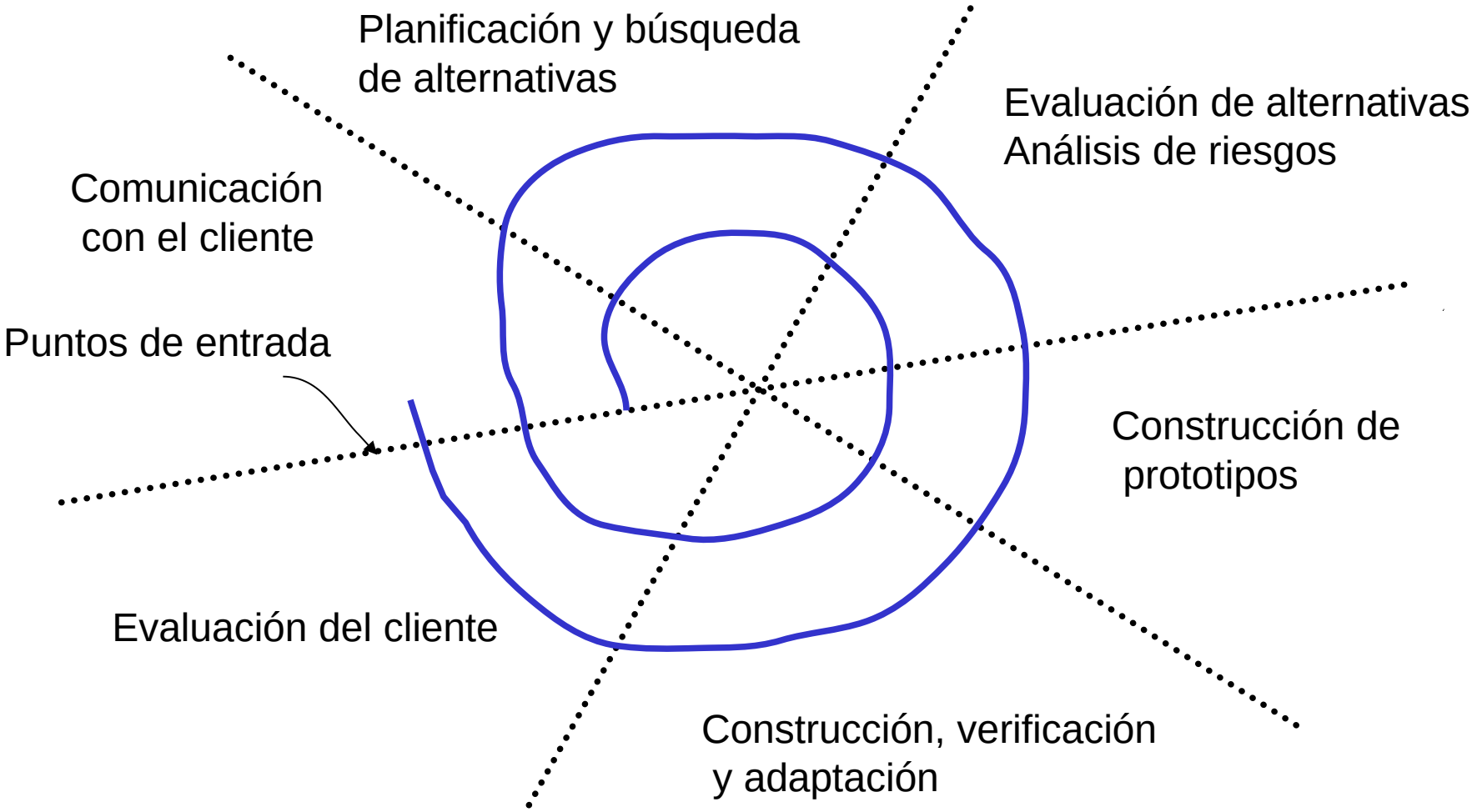
En la **evaluación del sistema** se decide:

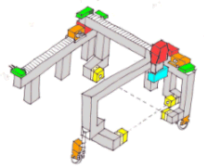
- El uso de prototipos en base a los siguientes criterios:
  - Tipo de aplicación.
  - Complejidad.
  - Características del cliente.
  - Características del proyecto.
  - Disponibilidad de herramientas para su construcción
    - Técnicas y lenguajes apropiados.
    - Componentes reutilizables.
- El tipo de prototipo a desarrollar
  - Evolutivo.
  - Desechable, en este caso a su vez pueden ser:
  - Horizontal: centrado en los niveles arquitectónicos y de datos de
  - alto nivel
  - Vertical: Centrado en una función determinada.



Modelo en espiral

Modelo en espiral de boehm.





Proceso unificado

Actividades

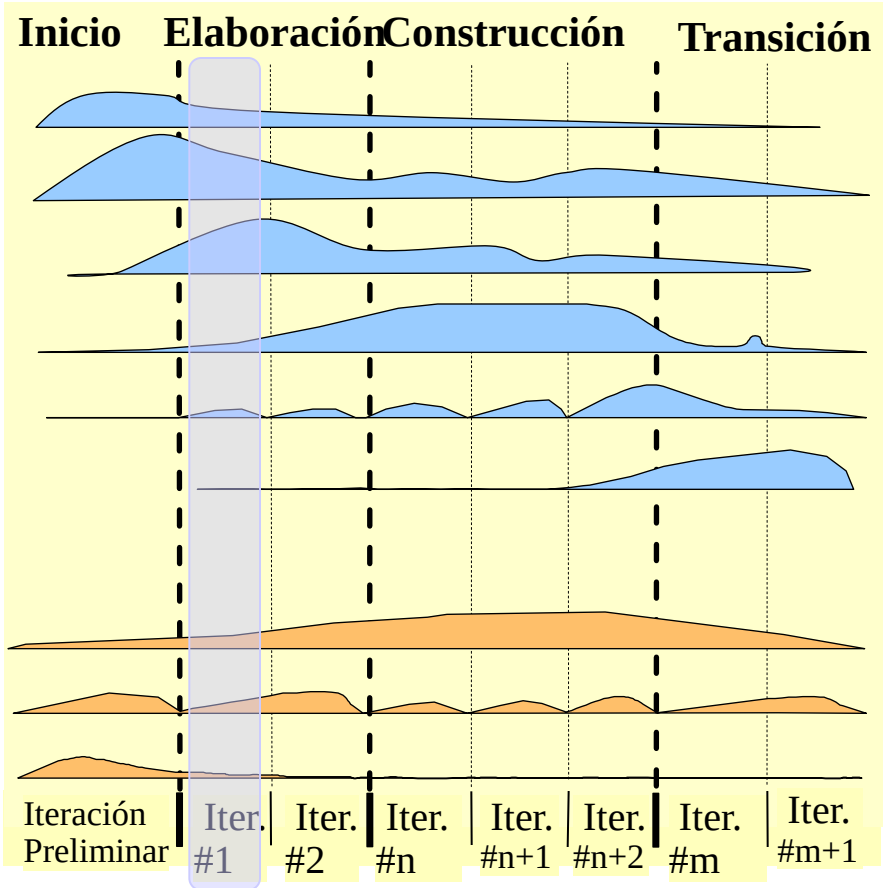
Flujos de Trabajo de Procesos

Modelado del sistema
Requerimientos
Análisis y Diseño
Implementación
Prueba
Implantación

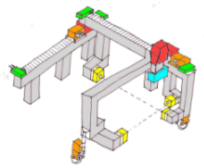
Flujos de Trabajo de Soporte

Admin. Configuración
Gestión del proyecto
Ambiente

Fases



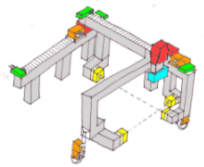
Iteraciones



Proceso unificado

Características:

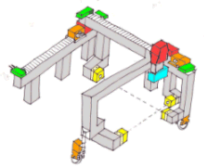
- Soporta técnicas y modelado Orientado a objetos.
- Adaptable a cualquier tipo de sistema, desde los más simples a los más complejos.
- Enfocado en los riesgos, afrontando en las etapas iniciales los elementos de mayor riesgo y complejidad.
- Dirigido por casos de uso, desarrollándose en cada iteración determinados casos de uso.
- Centrado en la arquitectura, mostrando y decidiendo los distintos aspectos arquitectónicos que presenta un sistema software en etapas tempranas, para que sirvan de base a las demás.



Proceso unificado

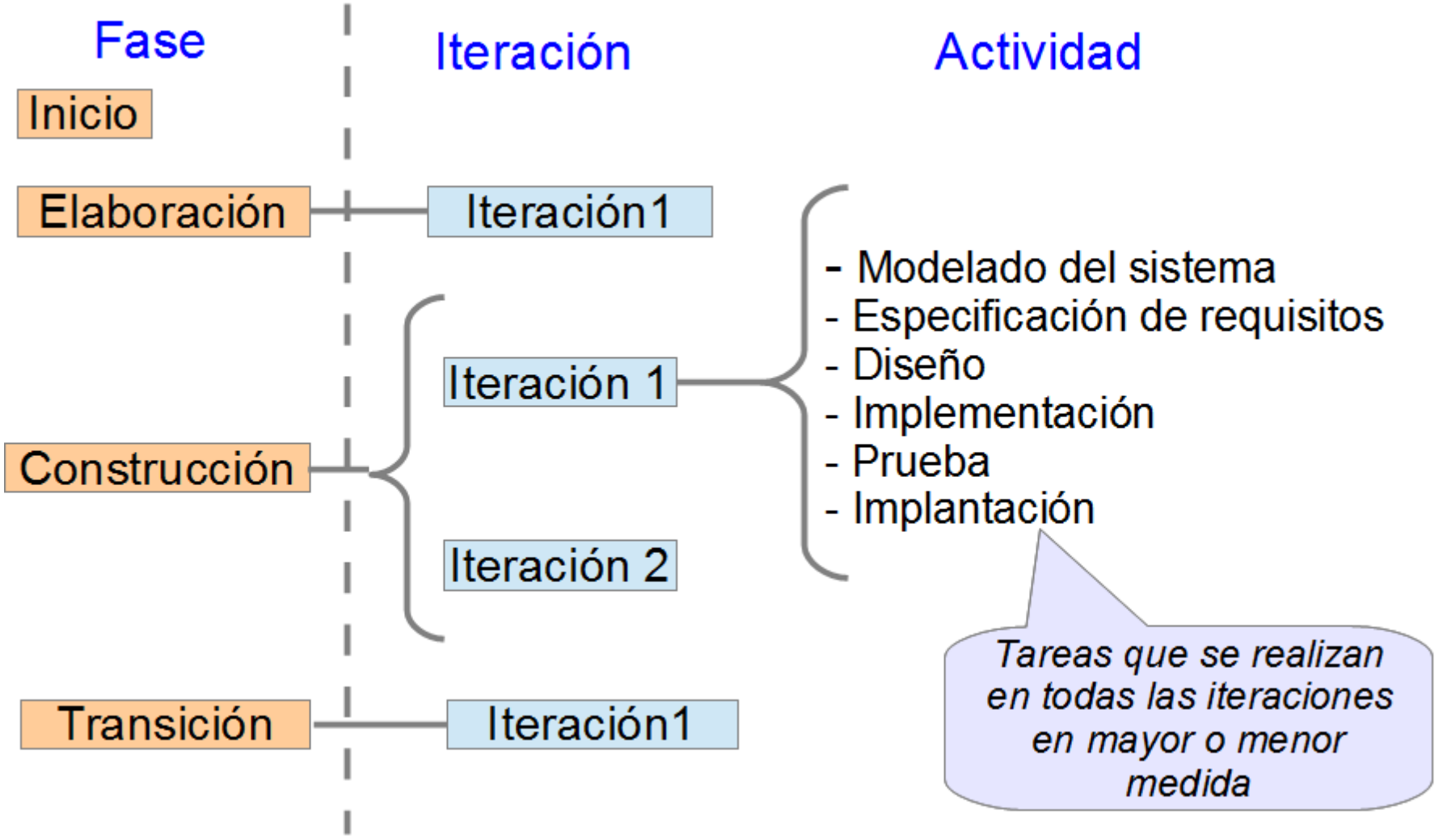
Fases:

- **Inicio:** Estudio de viabilidad, alcance, objetivos y planificación del proyecto.
- **Elaboración:** Desarrollo de la arquitectura básica sobre la que se asentara la fase de construcción.
- **Construccion:** Desarrollo por iteraciones de los demás elementos que componen el sistema hasta su terminación.
- **Transición:** Asegurarse que el sistema cumple con los requisitos especificados y que está disponible para los usuarios finales.



Proceso unificado

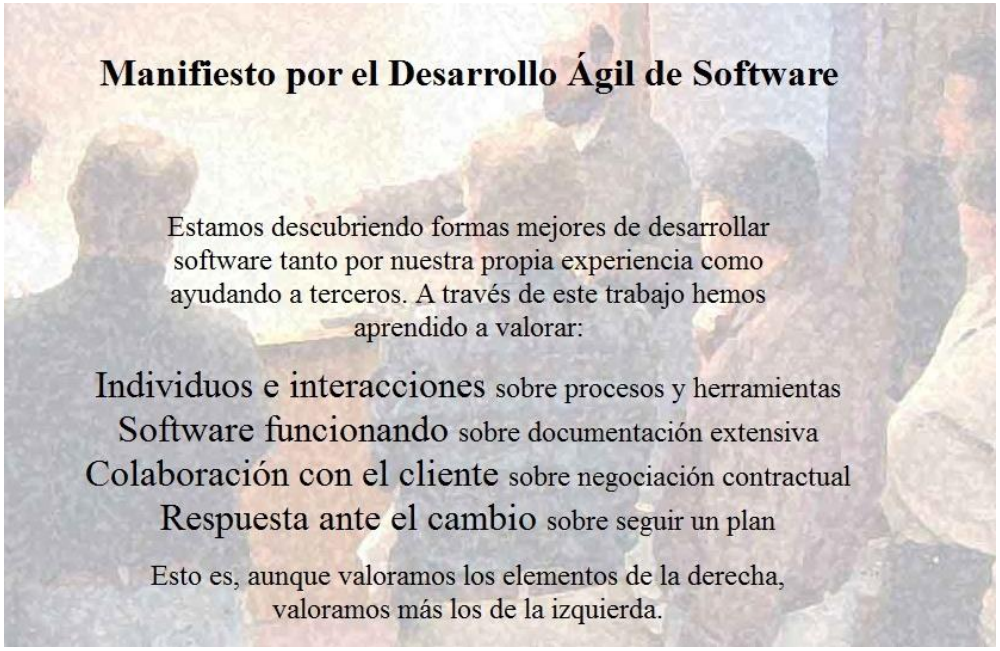
Ejemplo concreto de PU para un sistema simple



Métodos Agiles

Muchos proyectos generan menos valor del esperado…

Snowbird, Utah (USA) feb 2001. ¿Por qué tantos proyectos de desarrollo de software no se terminan a tiempo, cuestan más que lo presupuestado originalmente, tienen problemas de calidad serios y generan menor valor que el esperado?
Se preguntaron 17 expertos, que después elaboraron el Manifiesto Ágil



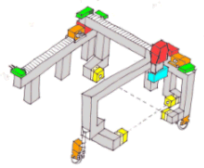
Kent Beck Mike Beedle
Arie van Bennekum Alistair Cockburn Ward Cunningham Martin Fowler
James Grenning Jim Highsmith Andrew Hunt
Ron Jeffries Jon Kern
Brian Marick Robert C. Martin Steve Mellor Ken Schwaber Jeff Sutherland
Dave Thomas

Métodos Agiles

Características:

- Proceso iterativo e incremental
- Entregas frecuentes
- Retroalimentación frecuente:
- Trabajo en equipo
- Autonomía del equipo de desarrollo
- Revisiones y reuniones retrospectivas frecuentes

(Ver Artículo introductorio acerca de metodologías ágiles (por Ricardo Colusso y Juan Gabardini) UBA)



Métodos Agiles

Beneficios:

- Desarrollo guiado por valor
- Mejor manejo de riesgos e incertidumbre
- Mejora de productividad

Métodos y técnicas:

- Scrum,
- XP (Extreme Programming),
- Kanban
- Programación en Parejas
- TDD - Test-Driven Development