

Introducción a la Ingeniería del Software

Contenido

1. El producto software

Naturaleza del software

Definición de software

Características del software

Tipos y dominios de aplicación del software

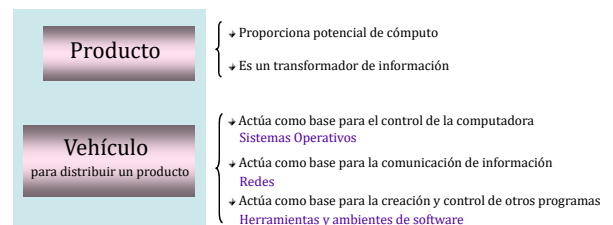
Proceso de producción

1. El producto software

2. El concepto de Ingeniería del Software

3. El proceso de desarrollo del software

Naturaleza del software



El software distribuye el producto más importante de nuestro tiempo

INFORMACIÓN

Definición de software

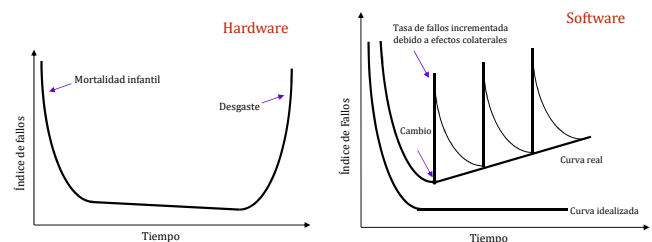
¿Software = Programa de computadora? *Definición incompleta*

El software es:

1. Instrucciones (programas) que cuando se ejecutan proporcionan las funciones y características buscadas
2. Estructuras de datos que permiten a los programas manipular la información adecuadamente
3. Información en papel o en forma virtual (documentación) que describe la operación y uso de los programas

Características del software

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



- 2) El software **crea modelos de la realidad**: modelo de funcionamiento, modelo de la información ...
- 3) El software está formado por **múltiples piezas** que deben **encajar perfectamente**

Tipos y dominios de aplicación del software

Dominios de aplicación

- ✦ **Software de sistemas**
Conjunto de programas que proporcionan servicio a otros programas
- ✦ **Software de aplicación**
Programas que resuelven una necesidad específica de negocios
- ✦ **Software de ingeniería y ciencias**
Implementa algoritmos "devoradores" de números
- ✦ **Software empotrado**
Reside dentro de un producto o sistema e implementa y controla características y funciones para el usuario final y para el sistema en sí
- ✦ **Software de gestión**
Proporciona una capacidad específica para uso de muchos consumidores diferentes
- ✦ **Aplicaciones Web**
Software centrado en redes que agrupa una amplia gama de aplicaciones
- ✦ **Software de inteligencia artificial**
Implementa algoritmos no numéricos para resolver problemas complejos difíciles de tratar computacionalmente o con análisis directo

Tipos y dominios de aplicación del software

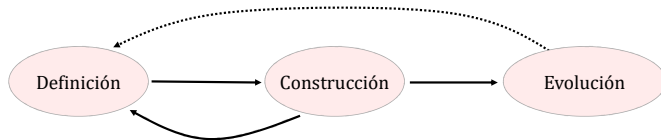
Según destinatario

- ✦ **Para distribución (software genérico)**
Sistema autónomo producido por una organización de desarrollo y vendido en el mercado abierto a cualquier cliente que pueda comprarlo
- ✦ **Usuario final (software hecho a medida)**
Sistema desarrollado por una empresa especialmente para un cliente particular

Según derechos de autor

- ✦ **Software de código abierto**
Su código fuente está disponible para que cualquiera pueda usarlo, examinarlo, modificarlo
- ✦ **Software de código cerrado**
Su código fuente no se encuentra disponible para cualquier usuario
- ✦ **Software de dominio público**
No tiene derechos de autor. Si el código fuente es de dominio público, se trata de un caso especial de software libre sin copyleft

Proceso de producción



¿Qué se desarrolla?

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

¿Cómo se desarrolla?

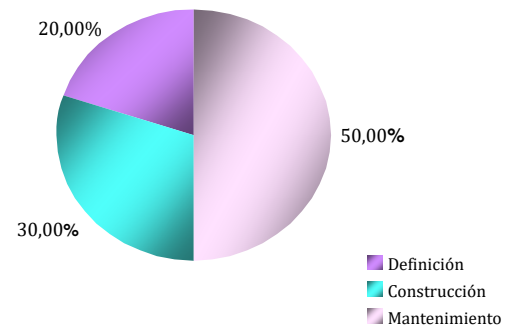
Tareas a realizar
Diseño del software
Generación del código
Prueba del software

¿Qué va a cambiar?

Tareas a realizar
Corrección
Adaptación
Mejora
Prevención

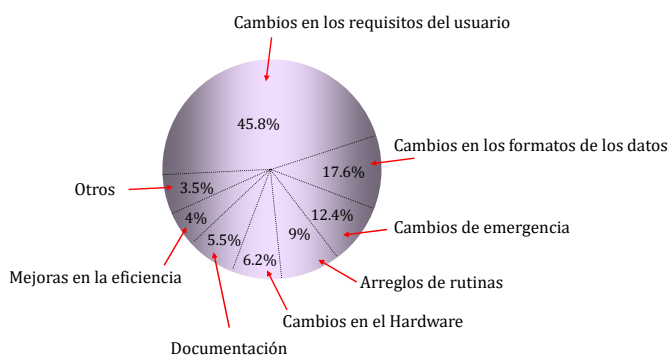
Proceso de producción

Esfuerzo requerido por etapas



Proceso de producción

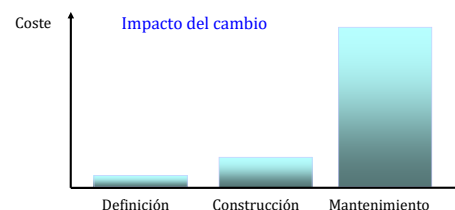
Mantenimiento



Proceso de producción

Problemas

- ✦ **Comunicación entre personas**
CLIENTES ↔ DESARROLLADORES
- ✦ **Incumplimiento de la planificación**
¿SOLUCIÓN: HORDA MONGOLIANA?
- ✦ **Incorporar cambios en etapas avanzadas del proceso**



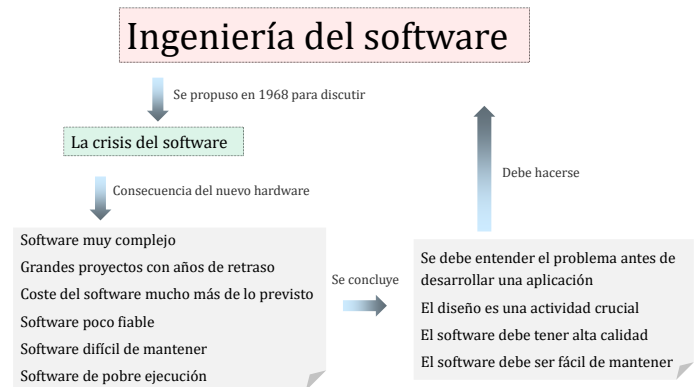
2. El concepto de ingeniería del software

Historia y necesidad de la ingeniería del software

Definición de ingeniería del software

Terminología usada en ingeniería del software

Principios generales de la ingeniería del software

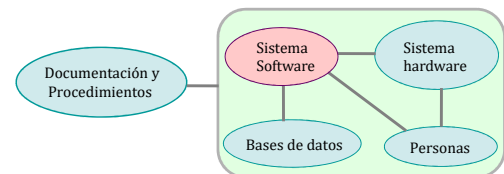


Definición de ingeniería del software

- ✦ "Establecimiento y uso de **principios fundamentales de la ingeniería** con objeto de desarrollar en forma económica software que sea **fiable** y que trabaje con eficiencia 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)
- ✦ "Aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento del software; es decir, aplicación de la ingeniería al software (estándar - IEEE, 1993)

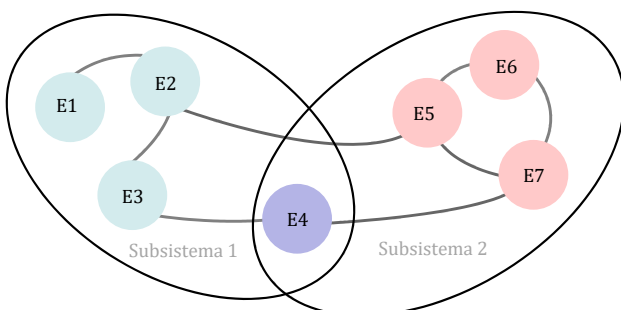
Terminología usada en ingeniería del software

- ✦ **Sistema**
Conjunto de elementos relacionados entre sí 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 ingeniería del software

- ✦ **Sistema software**
Conjunto de piezas o elementos software relacionados entre si y organizados en subsistemas



Terminología usada en ingeniería del software

- ✦ **Modelo**
Representación de un sistema en un determinado lenguaje: De un mismo sistema se pueden construir muchos modelos
- ✦ **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 ingeniería del software

Heurísticas

Conjunto de reglas empíricas, que al ser aplicadas producen modelos que se adecuan a los principios

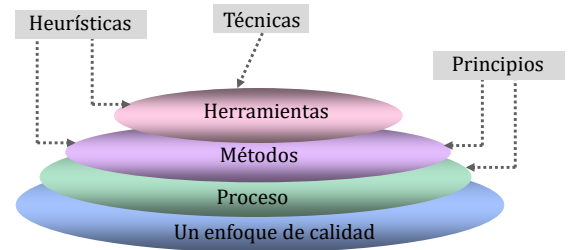
Proceso

Estructura que debe establecerse para la obtención eficaz de un producto de ingeniería

Métodos

Secuencia de actividades para la obtención de un producto (modelo), que describen cómo usar las herramientas y las heurísticas

Resumen



Definición de ingeniería del software

Estudio de principios, metodologías y herramientas que forman parte de un proceso para facilitar el desarrollo y mantenimiento de sistemas software de calidad

Principios generales

1. La razón de que exista todo (The reason it all exists)
Un software solo existe para aportar valor a sus usuarios
"Si no aporta valor, no lo hagas"
2. KISS (Keep it Simple, Stupid)
Diseña con simplicidad, pero sin sacrificar calidad
"La elegancia está en la simplicidad"
3. Mantener la visión (Maintain the vision)
Conserva la integridad conceptual durante todo el proyecto
"Un diseño consistente evita problemas futuros"
4. Lo que produzcas, otros lo consumirán
(What you produce, others will consume)
Lo que tu haces, alguien más tendrá que entenderlo
"Facilita el trabajo a los que vengan después"

Principios generales

5. Estar abierto al futuro (Be open to the future)
Diseña sistemas adaptables y listos para cambios
"Prepárate para lo inesperado"
6. Planificar pensando en la reutilización (Plan ahead for reuse)
Diseña pensando en componentes reutilizables
"Reutilizar ahorra tiempo y esfuerzo"
7. Piensa (Think)
Reflexiona para lograr mejores resultados y aprender de los errores
"El pensamiento claro produce valor"

Contenido

3. El proceso de desarrollo del software

Concepto de proceso de desarrollo

Modelo de proceso

Modelo genérico

Modelos prescriptivos:

Modelo en cascada

Modelo de prototipos

Modelos evolutivos

Proceso unificado

Desarrollo ágil

Concepto de proceso de desarrollo

Proceso de desarrollo del software

Conjunto de actividades, acciones y tareas que se realizan cuando va a crearse un producto o sistema software

Actividad

Busca alcanzar un objetivo amplio y se aplica sin importar el dominio de aplicación, tamaño del proyecto, o complejidad (p. e., comunicarse con los interesados)

Acción

Conjunto de tareas que generan un producto de trabajo (p. e., un modelo arquitectónico)

Tarea

Se centra en un objetivo pequeño, pero bien definido que produce un resultado tangible (p. e., realizar una prueba de unidad)

Concepto de proceso de desarrollo

Tipo de actividades

Estructurales: Dedicadas a obtener el producto

Comunicación: Colaboración con el cliente para entender objetivos y requisitos del proyecto

Planificación: Definir el plan del proyecto en el que se describen los riesgos probables, los recursos que se requieren, los productos que se obtienen y se programan las actividades, acciones y tareas

Modelado: Representación mediante modelos del sistema propuesto junto con la solución o soluciones apropiadas

Construcción: Generación de código y su prueba

Implementación: Entrega al cliente que lo evalúa y proporciona retroalimentación con base en dicha evaluación

Concepto de proceso de desarrollo

Tipo de actividades (continuación)

Sombrilla (continuación)

Gestión de la configuración: Gestiona los efectos del cambio a lo largo del proceso

Gestión de la reutilización: Define los criterios para la reutilización del producto de trabajo y establece los mecanismos para obtener componentes reutilizables

Preparación y producción del producto de trabajo: Actividades requeridas para crear productos de trabajo (modelos, documentos, ...)

Concepto de proceso de desarrollo

Tipo de actividades (continuación)

Sombrilla: Se aplican a lo largo de un proyecto software

Seguimiento y control del proyecto: El equipo evalúa el progreso y lo compara con el plan del proyecto

Gestión de riesgos: Se evalúan los riesgos que pueden afectar al resultado del proyecto o a la calidad del producto

Aseguramiento de la calidad: Actividades requeridas para garantizar la calidad del software

Revisiones técnicas: Se evalúan los productos para descubrir y eliminar errores

Medición: Define mediciones del proceso y del producto para entregar software que cumpla con las necesidades del cliente

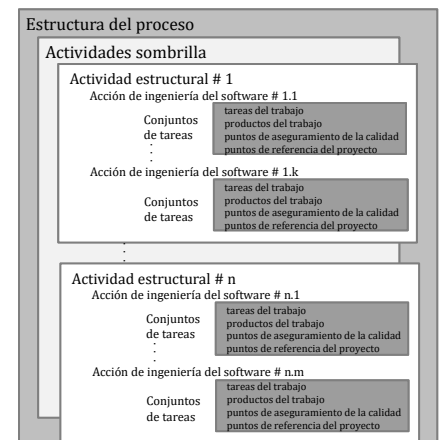
Modelo de proceso: Modelo genérico

Estructura del proceso

Cada una de las **actividades, acciones y tareas** que forman parte de un proceso, reside dentro de un marco de trabajo que define su relación con el proceso y entre sí

Cada **actividad**, del marco de trabajo está formada por un conjunto de **acciones** de ingeniería del software

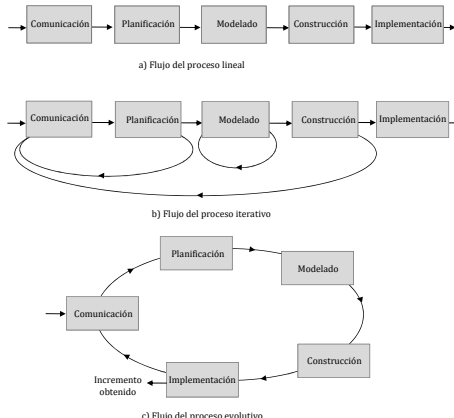
Cada **acción** de ingeniería del software se define por un conjunto de **tareas**



Modelo de proceso: Modelo genérico

Flujo del proceso

Describe la forma en que se organizan las **actividades** estructurales, además de las **tareas** y **acciones** que ocurren dentro de cada actividad estructural con respecto a la secuencia de tiempo



Modelo de proceso: Modelo genérico

Acciones y tareas de las actividades estructurales

Obtención de requisitos: Indagación para obtener información sobre qué es lo que debe realizar el software

Estimación y planificación del proyecto: Estimar el tiempo y los costes de desarrollo del software

Análisis 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

Prueba del software: Revisión y validación de todo el código

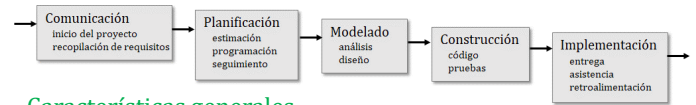
Evaluación y aceptación: Evaluación del producto y aceptación en su caso por parte de los interesados en el sistema software

Entrega y asistencia: Sistema operando y asistencia para su funcionamiento correcto

Modelos prescriptivos

- Definen un **conjunto predefinido** de elementos del proceso y un flujo de trabajo predecible (modelos de proceso tradicionales)
- Buscan la **estructura** y el **orden** en el desarrollo de software
- Las actividades y tareas ocurren de manera **secuencial** con lineamientos definidos para el progreso
- ¿Son apropiados para un mundo de software que se nutre del cambio?
- Si se sustituyen con algo menos estructurado ¿sería posible conseguir la coordinación y coherencia en el trabajo software?

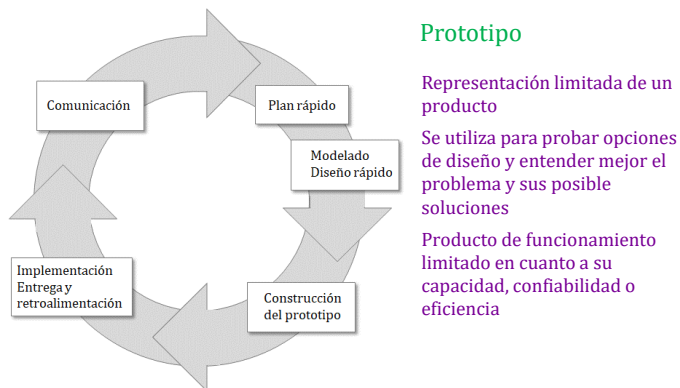
Modelos prescriptivos: Modelo en cascada



Características generales

- Estructura **secuencial** y flujo de proceso **lineal**
- Problemas que presenta
 - Los proyectos reales raras veces se adecuan al flujo de trabajo secuencial que propone el modelo
 - Es difícil indicar todos los requisitos de forma explícita al principio de un proyecto
 - No hay una versión funcional de los programas hasta etapas avanzadas del proyecto
 - Los errores graves no se detectan hasta que se revise el programa funcional

Modelos prescriptivos: Modelo de prototipos



Modelos prescriptivos: Modelo de prototipos

Se usa para:

- Facilitar la obtención y validación de requisitos
- Estudios de viabilidad
- Propuestas de soluciones (diseños) alternativas
- En casos muy concretos como producto final

Inconvenientes:

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

Su uso vendrá determinado por:

- Tipo y complejidad de la aplicación
- Características del cliente
- Disponibilidad de herramientas para su construcción

Modelos prescriptivos: Modelos evolutivos

Son **iterativos** y surgen por:

- La exigencia de tiempo de entrega muy limitado
- La necesidad de facilitar la incorporación de cambios
- La necesidad de satisfacer al usuario/cliente

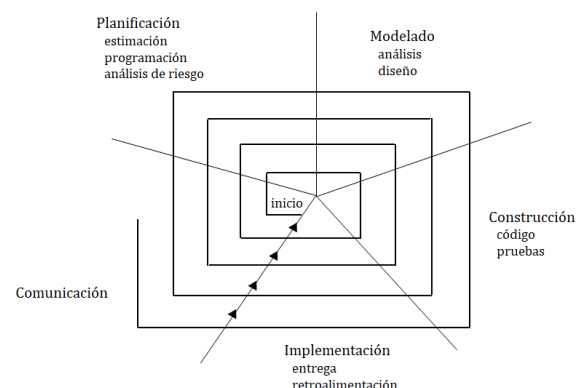
En cada iteración se obtiene un **producto terminado y operativo**

Características generales

- Afrontan los riesgos altos tan pronto como sea posible
- Retroalimentación temprana por parte del usuario
- Manejo de la complejidad (pasos cortos y sencillos)
- El conocimiento adquirido durante una iteración de la evolución se puede usar en el resto de iteraciones
- Involucra continuamente al usuario (evaluación, retroalimentación y obtención y refinamiento de requisitos)

Modelos prescriptivos: Modelos evolutivos

Modelo en espiral de Boehm



Modelos prescriptivos: Modelos evolutivos

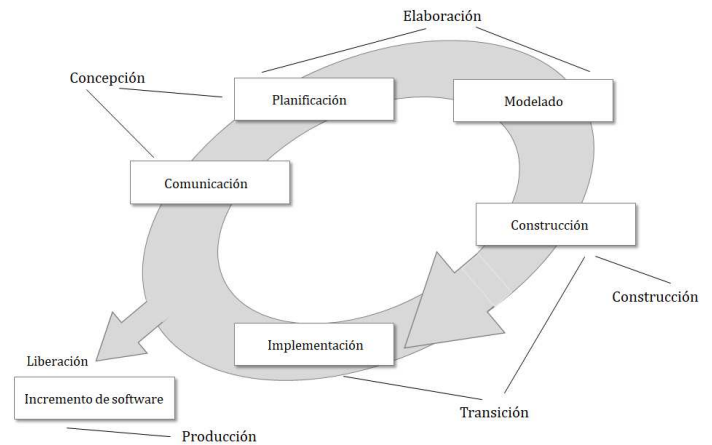
Características específicas:

- Centrado en el análisis de riesgo, haciendo uso de construcción de prototipos para su estudio
- La espiral puede continuar una vez finalizado todo el proceso y entregado el producto
- Es un enfoque adecuado para el desarrollo de sistemas a gran escala

Inconvenientes:

- Modelo no adaptable a la complejidad ni al tipo de sistema
- Requiere un equipo de desarrollo con gran experiencia en análisis de riesgos

Proceso unificado



Proceso unificado

Fases del proceso unificado

✦ Concepción

Se lleva a cabo la comunicación y planificación con el cliente
Los requisitos fundamentales se describen a través de **casos de uso** que describen las características y funciones de cada clase principal de usuarios
La planificación identifica recursos, evalúa riesgos importantes y define un calendario preliminar para los incrementos de software

✦ Elaboración

Incorpora las actividades de planificación y modelado del modelo genérico
Se refina y expande los casos de uso
Incluye cinco perspectivas del software: el **modelo de casos de uso**, el de **análisis**, el de **diseño**, el de **implementación** y el de **despliegue**
Las modificaciones al plan se realizan en este momento

Proceso unificado

Fases del proceso unificado (continuación)

✦ Construcción

Incorpora la actividad de construcción definida para el modelo genérico
Las características y funciones requeridas para el incremento de software se **implementan** en código fuente
Se diseñan y ejecutan **pruebas unitarias** para cada componente y se llevan a cabo actividades de **integración**
Se emplean casos de uso para derivar **pruebas de aceptación**

✦ Transición

Incorpora el final de la actividad de construcción genérica e inicio de la actividad de despliegue genérica
Se proporciona el **software** y la **documentación** a los usuarios finales para la prueba beta
La retroalimentación del usuario reporta los **defectos** y **cambios** necesarios
El **incremento** de software se convierte en una **versión** de software utilizable

Proceso unificado

Fases del proceso unificado (continuación)

✦ Producción

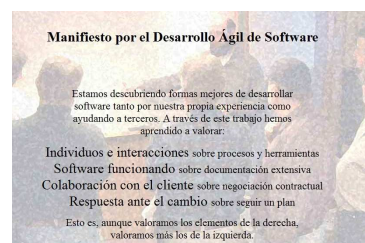
Coincide con la actividad de implementación del modelo genérico
Se **supervisa** el uso continuo del software
Se proporciona **soporte** para el entorno operativo (infraestructura)
Se envían y evalúan informes de **defectos** y solicitudes de **cambios**

Desarrollo ágil

¿Cómo surge?

SNOWBIRD, UTAH (USA), FEBRERO 2001

¿Por qué tantos proyectos de desarrollo de software no se terminan a tiempo, cuestan más de lo presupuestado originalmente, tienen problemas serios de calidad y generan menor valor del esperado?



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 Steven Mellor
Ken Schwaber Jeff Sutherland
Dave Thomas

¿Qué es la agilidad?

- El principal impulso es la preponderancia del **cambio**
- Fomenta estructuras y actitudes de equipo que faciliten la **comunicación**
- Hace hincapié en la entrega **rápida** de software **operacional**
- Resta importancia a lo productos de trabajo intermedios (**documentación**)
- Adopta al **cliente** como parte del equipo de desarrollo
- Un plan de proyecto debe ser **flexible**

¿Qué es un proceso ágil?

- Proceso que debe ser **adaptable** para gestionar la **imprevisibilidad**
- La adaptabilidad debe ser **incremental**
- Requiere **retroalimentación** del cliente
- Los incrementos de software deben entregarse en periodos **cortos**
- El enfoque iterativo permite al cliente **evaluar** el incremento de software

Modelos ágiles

- Scrum
- XP (Extreme Programing)
- Kanban
- DevOps