Departamento de Informática | Universidad de Valladolid

Programación II (I.T.I. Gestión)

Introducción

Félix Prieto
Esperanza Manso

Curso 2009/10

Evolución del software

Primeros años (?-1965)
- Programas por lotes
- Software a medida
- Distribución limitada

Segunda era (1965-1975)
- Multiusuario
- Tiempo real
- Bases de datos
- Producto software
- Mercado de software

Tercera era(1975-1990)
- Sistemas distribuidos
- Inteligencia artificial
- Software de bajo coste
- Incremento del consumo
- Redes

Cuarta era (1990-?)
- Potencia en los pc's
- Orientación a Objetos
- Sistemas expertos, redes neuronales
- Computación en paralelo

Crisis del software: Los problemas

- Demanda superior a la oferta
- Problemas de mantenimiento
- Retrasos en las entregas
- Costes elevados
- Productos plagados de errores
- Problemas con el proceso de desarrollo del software
  - Modelos
  - Planificación
  - Asignación de recursos

Crisis del software: Los problemas (y II)

1962 Pérdida de la sonda *Mariner 1* camino de Venus. La sentencia `DO 17 I = 1, 10` fue interpretada como `DO 17 I = 1.10`

1985-87 Varios pacientes de cáncer fueron radiados de forma incorrecta mediante un aparato *TERAC-25* controlado por un software con errores. No se ha determinado el número total de fallecidos.

1991 Un error de software provocó un desajuste de un tercio de segundo en un sistema de misiles *Patriot* que debiera haber interceptado un misil *Scud*. La explosión del *Scud* provocó 28 muertos.

Crisis del software: Los problemas (y III)

1992 Error de software en el sistema de emergencias de Londres. Un número indeterminado de pacientes recibieron asistencia con retraso y/o con medios inadecuados.

1996 Pérdida del cohete Arian 5 en su primer vuelo. Un módulo reutilizado del software del Arian 4 recibió un argumento fuera de rango.

1999 Pérdida de la sonda *Mars Climate Orbiter*. La conversión entre las unidades de medida de dos módulos fue incorrecta o no se realizó.

Crisis del software: Los problemas (y IV)

1964-67 El desarrollo de OS/360 de IBM llegó a contar con 500 programadores simultáneos, incrementó su coste de 40 a 500 millones de dólares y se entregó con dos años de retraso y plagado de errores

1982-93 El sistema informático para Allstate Insurance se retrasó 6 años y su precio pasó de 8 a 100 millones de dólares.

1983-98 El proyecto Taurus para la bolsa de Londres presupuestado en 6 millones de libras fue abandonado cuando su precio alcanzaba los 800 millones de libras

1985 El sistema de defensa aérea del bombardero B1 requirió una inversión adicional de un billón de dólares, pero no alcanzó sus objetivos originales

Crisis del software: Los problemas (y V)

1988 Bank of America gastó 23 millones de dólares en un programa a 5 años para MasterNet, un sistema de gestión de cuentas. El sistema fue abandonado tras requerir una inversión adicional de 60 Millones de dólares

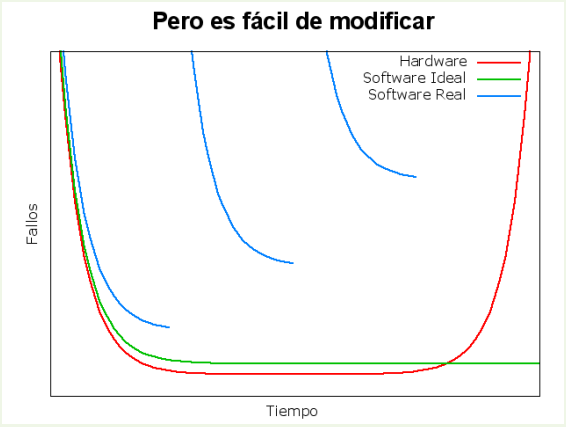
1989 El sistema electrónico de defensa ASJP, instalado en unos 2000 aviones de la marina, requirió una inversión extra de 1 billón de dólares y 4 años extra de desarrollo, pero no alcanzó la especificación inicial

2000 ;-)

Ingeniería del Software

- Disciplina que permite el establecimiento y uso de *principios robustos* de ingeniería, orientados a obtener con el *menor coste* posible sistemas de *calidad* que sean *fiables* y que funcionen *eficientemente* sobre máquinas reales (Naur, P. y Randell B.)
- Definición heredada de las ingenierías tradicionales
- ¿Es el software equiparable con el hardware?

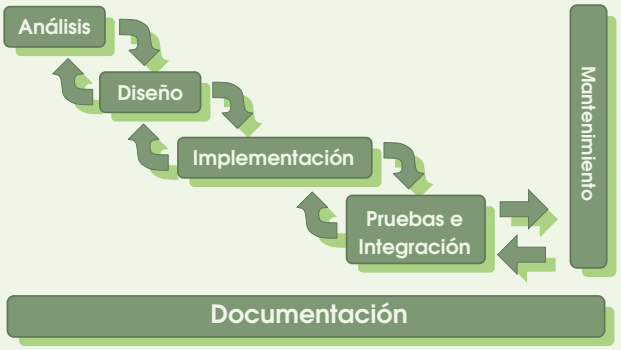
Software vs. Hardware



Ciclo de vida

- *Ingeniería del Software:* Aproximación sistemática al desarrollo, instalación, mantenimiento y retirada del software. IEEE 1993
- *Ciclo de vida:* Dinámica que describe un gran sistema de software en cuanto a su comportamiento y evolución.

Modelo en cascada



Modelo en cascada (y II)

- Rígido, con fases estancas que utilizan un lenguaje incompatible
- No refleja el proceso de desarrollo real del software
- Dificulta la revisión de los proyectos
- Dificulta la comunicación con el usuario
- Retarda el proceso de desarrollo

Otros modelos de desarrollo

- Basados en prototipos, Modelo en espiral,...
- Persiguen una interacción más rápida con los usuarios
- Intentan manejar la incertidumbre inherente al desarrollo de software
- Pretenden adaptarse al carácter evolutivo del software
- Utilizan prototipos y/o refinamientos sucesivos
- En su interior suelen contener versiones de la cascada

Desarrollo orientado a objetos

- Compatible con diversos modelos del ciclo de vida
- Utiliza una metáfora unificada para todas las fases del desarrollo
- Utiliza lenguajes y sistemas de representación unificados para todas las fases del desarrollo
- Facilita la creación de componentes reutilizables

Calidad del software

- Calidad es le grado en que un producto satisface las necesidades del cliente
- Es una propiedad multidimensional
- Depende de los requisitos
- Es subjetiva, medida como la diferencia entre lo esperado y lo obtenido
  - Se puede aumentar la calidad de un producto cambiando las expectativas de sus usuarios

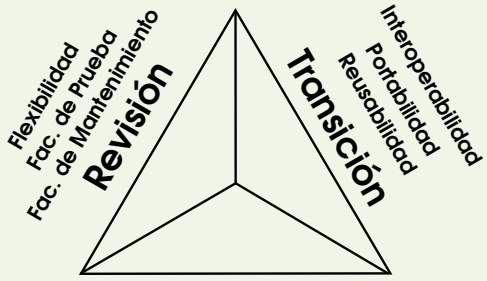
Calidad del software (y II)

- Calidad es la suma de todos los aspectos o características de un producto o servicio que influyen en su capacidad para satisfacer las necesidades explícitas o implícitas. *(ISO8402)*
- Capacidad del producto software para satisfacer los requisitos establecidos. *(DoD 2168)*

Factores de calidad

Factores Externos	Factores Internos
⋮	⋮
Fiabilidad	Tamaño
Robustez	Complejidad
Facilidad de Mant.	Cohesión
Transportabilidad	Acoplamiento
Reusabilidad	Corrección
⋮	⋮
De los Usuarios	De los Programadores

Modelo de McCall



Corrección Fac. de Uso
Fiabilidad Integridad
Eficiencia

Fiabilidad del Software

A partir del inicio de la Crisis del Software comenzamos a diferenciar los siguientes conceptos relativos a la *especificación del programa*:

- *Fiable* Con alta probabilidad de funcionar conforme a su especificación
- *Correcto* Que funciona conforme a su especificación
- *Robusto* Que funciona razonablemente en situaciones no contempladas en su especificación

Reutilización

- Ingeniería del software es el estudio de métodos y herramientas que se pueden utilizar para producir software práctico de calidad (*Bertrand Meyer*)
  - Requiere el desarrollo de componentes fiables (o mejor correctas)
  - Requiere que la documentación sea adecuada y esté integrada en la componente
  - Requiere que las componentes sean reutilizables

Documentación

- Elementos del software que aportan información sobre:
  - Qué hace
  - Cómo se construyó
  - Cómo se utiliza
  - Cómo funciona
  - Cómo ha sido mantenido
- Algo más que los comentarios en el código
- Tan *integrada* con el resto del software como sea posible
- Mantenida a lo largo de *todo* el ciclo de vida del software

Documentación (y II)

- Tan *necesaria* como el resto de los elementos del software
- Debe estar *bien escrita*
- Debe ser *adecuada* para su lector
- Debe ser elaborada por el equipo de desarrollo
- Clasificaciones:
  - Interna vs. Externa
  - Usuario vs. Programador
  - Útil vs. Inútil

Documentación externa del usuario

1. Documentos de descripción funcional
   - Especificación, ejemplos sencillos, visión general
2. Documentos detallados de instalación
   - Medio y forma de instalación, configuración de hardware requerida, elementos que se instalarán.
   - También información sobre desinstalación
3. Manual de introducción
   - Visión poco formal sobre un uso normal del sistema
4. Manual de referencia
   - Visión formal y completa de todos sus elementos
5. Guía del operador
   - En los sistemas en que sea necesaria