FACULTAD DE INGENIERIA

Fundamentos de Ingeniería de Software

Introducción

¿Qué es la ingeniería de software?

Agenda

- Surgimiento
- Definiciones
- Relación con otras disciplinas
- El software en la sociedad actual
- Ética y responsabilidad
- Calidad del software
- Participantes en el desarrollo
- Ingeniería de software e ingeniería de sistemas
- Cuestiones fundamentales

Surgimiento

En los comienzos:

- El programador era el usuario.
- Los problemas a resolver eran bien conocidos y simples.

Desarrollo de las computadoras:

- Aparece la figura del programador especializado.
- La mayor potencia de los equipos permite atacar problemas más complejos.
- Se habla de la «crisis del software»:
  - Los proyectos de software llevaban más tiempo y esfuerzo y dan peores resultados que los previstos.

El término «ingeniería de software» aparece por 1.ª vez en 1968.

Definiciones

Software:

- Programas de computador, procedimientos, y la documentación y los datos posiblemente asociados relacionados con la operación de un sistema de computador. (IEEE 1990)

… (el ingeniero) aplica el método y enfoque científico a la solución de problemas ...

Ingeniería de software:

- (1) aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento de software, esto es, la aplicación de la ingeniería al software
- (2) El estudio de enfoques como en (1). (IEEE 1990)

Ingeniería de software

La ingeniería de software es una disciplina de ingeniería que se ocupa de todos los aspectos de la producción de software desde las primeras etapas de la especificación del sistema hasta el mantenimiento del sistema una vez que se ha puesto en uso. (Sommerville)

- Disciplina de ingeniería

  Usar teorías y métodos apropiados para resolver problemas teniendo en cuenta las limitaciones organizacionales y financieras.

- Todos los aspectos de la producción de software

  No solo el proceso técnico de desarrollo. También gestión de proyectos y desarrollo de herramientas, métodos, etc. para respaldar la producción de software.

Importancia de la ingeniería de software

- Cada vez más, las personas y la sociedad confían en los sistemas avanzados de software.
- Se debe producir sistemas confiables de forma económica y rápida.

Confianza

- A largo plazo, es más barato utilizar métodos y técnicas de ingeniería de software para sistemas de software en lugar de simplemente escribir los programas como si se tratara de un proyecto de programación personal.
- Para la mayoría de los tipos de sistema, la mayoría de los costos consisten en cambiar el software después de ponerlo en producción.

Costos

Relación con otras disciplinas

Ciencias de la computación aporta:

- Teorías
- Funciones de computadoras

Cliente trae:

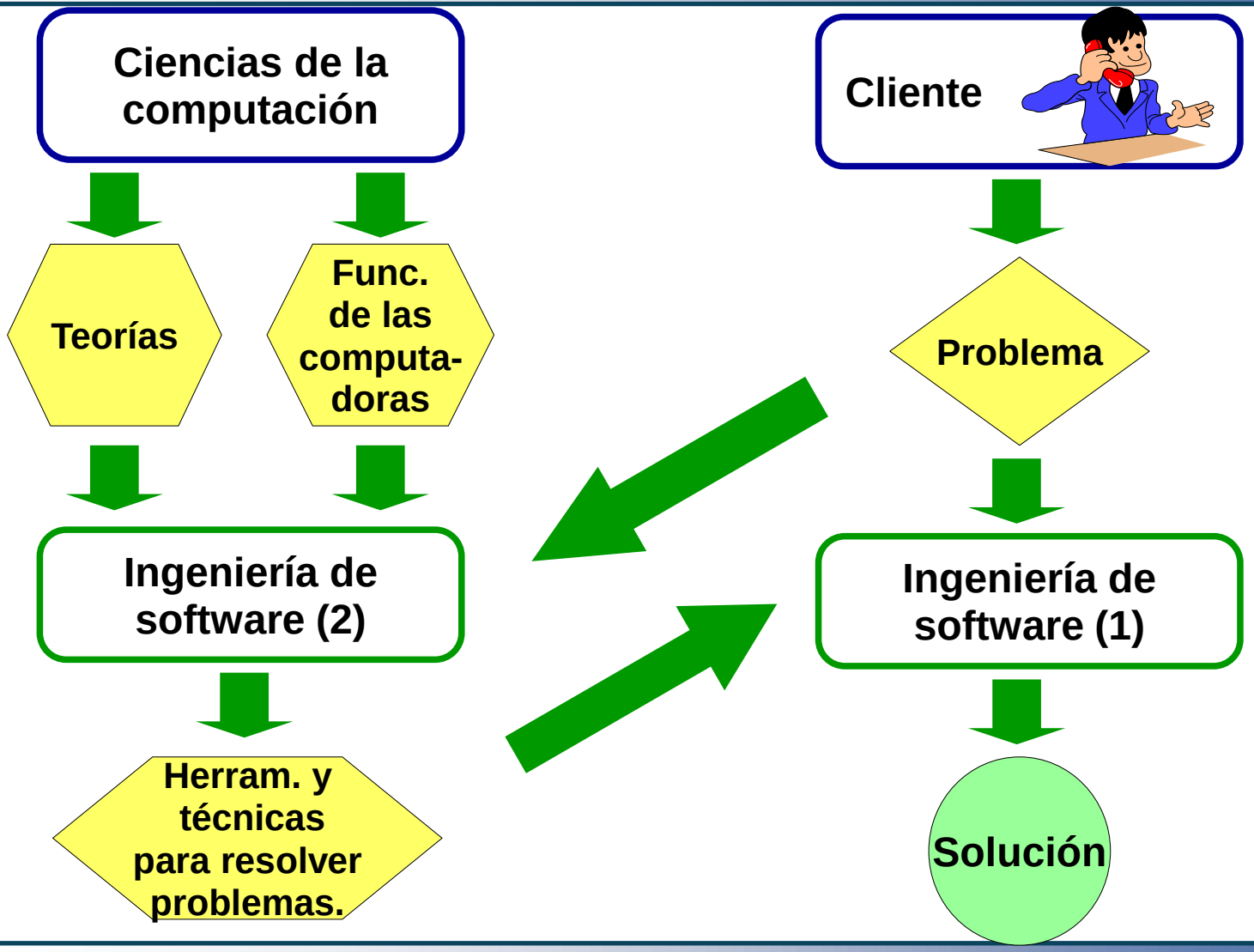
- Problema

Ingeniería de software (2) desarrolla:

- Métodos, herramientas, procedimientos, paradigmas para resolver problemas

Ingeniería de software (1) resuelve problemas

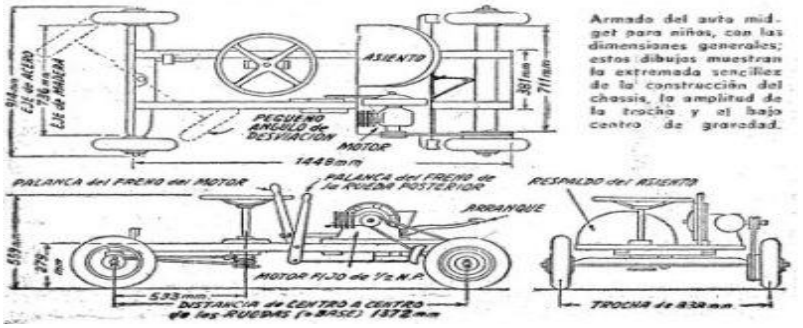
Relación con otras disciplinas



Relación con otras disciplinas (2)

Ingeniería: Construcción de artefactos cumpliendo con restricciones.

- Arte más que ciencia (hacer más que saber)
- Plazo, costo, calidad, otras.



Administración general:

- Gestión de proyectos
- Recursos humanos

Software en la sociedad actual

¿Dónde no está presente?

- Energía
- Comunicaciones
- Automóviles
- Electrodomésticos
- Equipos médicos
- …..

¿Crisis del software?

Software en la sociedad actual (2)

- El software es esencial para el funcionamiento de los gobiernos, la sociedad, los negocios nacionales e internacionales y las instituciones.
- Los sistemas de software son abstractos e intangibles (no obedecen leyes físicas).
- Hay muchos tipos de sistemas de software.
- Todavía hay muchos proyectos de software que fracasan, así como muchas fallas en el software que usamos.

Ejemplos recientes de fallos en SW

- British Airways se enfrentó a una falla mundial masiva que la llevo a cancelar todos los vuelos de Heathrow y Gatwick en mayo de 2017 (más de 1.000).
- En 2016, HSBC se convirtió en el primer banco en sufrir un corte de TI importante. Millones de clientes del banco no pudieron acceder a cuentas en línea durante dos días.
- En 2015, un problema provocó que más de 3.200 prisioneros estadounidenses fueran liberados temprano (2 meses aprox.). El software calcula utilizando el comportamiento.
- 911 no disponible durante seis horas en siete estados de EE. UU. en abril de 2017. 6.000 personas hicieron llamadas al 911 que no pudieron conectarse en Washington y partes de otros seis estados.
- Mayo de 2017, Fiat retiró más de un millón de camiones debido a un problema de software que estaba relacionado con al menos un accidente mortal. Un código erróneo que deshabilitó temporalmente las bolsas de aire y la funcionalidad del cinturón de seguridad.

Ejemplos recientes de fallos en SW

- Y si quedaron con ganas de saber más...
- https://ingenieriadesoftware.es/grandes-errores-historia-software-informatico/

Causas de proyectos fracasados

- Aumento de la complejidad del sistema

  Hacemos sistemas más grandes y complejos → las demandas cambian. Entregas más rápidas, sistemas más grandes y complejos.

- No utilizar métodos de ingeniería de software

  Es bastante fácil escribir programas de computadora sin utilizar métodos y técnicas de ingeniería de software

  → el software a menudo es más caro y menos confiable de lo que debería ser.

Ética y responsabilidad

Repercusiones de fallas en el software:

- Pérdidas financieras
- Riesgo a la seguridad
- …..

Más allá de las fallas.....

- Impacto social
- Calidad de vida
- Cuestiones legales

Ética en la ingeniería de software

- **Ética**: conjunto de reglas morales que rigen el comportamiento humano.
- La ingeniería de software implica responsabilidades más amplias que simplemente la aplicación de habilidades técnicas.
- Los ingenieros de software deben comportarse de una manera honesta y éticamente responsable si deben ser respetados como profesionales.
- El comportamiento ético es más que simplemente mantener la ley, pero implica seguir un conjunto de principios que son moralmente correctos.

Cuestiones de responsabilidad profesional

- Confidencialidad

  Respetar la confidencialidad de sus empleadores o clientes (con o sin acuerdo de confidencialidad).

- Competencia

  No aceptar trabajos que están fuera de su competencia.

- Derechos de propiedad intelectual

  Conocer las leyes locales que rigen el uso de la propiedad intelectual, como patentes, derechos de autor, etc.

- Mal uso de la computadora

  No usar sus habilidades técnicas para usar mal las computadoras de otras personas.

Ingeniería de software ¿una profesión?

- En la industria y academia se conoce como una profesión inmadura.
- Ha tenido avances en los últimos años.
- Modelo que permite caracterizar madurez de profesión [Ford y Gibbs]
  - Formación profesional inicial
  - Acreditación
  - Desarrollo de habilidades (competencias)
  - Certificación
  - Concesión de licencias
  - Desarrollo profesional
  - Sociedades profesionales
  - Código de ética

Ingeniería de software ¿una profesión? (2)

SWEBOK V4 – IEEE 2024

- Guide to the Software Engineering Body of Knowledge

Código de ética – ACM – IEEE-CS

- Versión corta: resume aspiraciones a alto nivel.
- Los ingenieros de software deberán comprometerse a convertir la ingeniería de software en una profesión benéfica y respetada.
- De acuerdo a su compromiso con la salud, seguridad y bienestar social, los ingenieros de software deberán sujetarse a los ocho principios siguientes:

Ética y responsabilidad

1. PÚBLICO – Los ingenieros de software deberán actuar consistentemente con el interés público.
2. CLIENTE Y EMPLEADOR – Los ingenieros de software actuarán de la mejor manera acorde a los intereses de los clientes o de sus empleadores siendo consistentes con el interés público.
3. PRODUCTO - Los ingenieros de software garantizarán que sus productos y sus modificaciones cumplen con los estándares profesionales más altos posibles.
4. JUICIO- Los ingenieros de software mantendrán su independencia e integridad en su juicio profesional.
5. GERENCIA – Los gerentes y lideres de ingeniería de software se suscribirán y promoverán un enfoque ético en la gestión del desarrollo del software y su mantenimiento.
6. PROFESIÓN – Los ingenieros de software incrementarán la integridad y la reputación de la profesión consistentemente con los intereses públicos.
7. COLEGAS – Los ingenieros de software deberán ser justos y comprensivos con sus colegas.
8. UNO MISMO – Los ingenieros de software participarán en la formación continua en relación con el ejercicio de su profesión y promoverán un enfoque ético en la práctica de la profesión.

Un caso real y reciente

¿Qué opinan de este caso?

https://www.elobservador.com.uy/nota/hackeo-a-google-y-ahora-accedio-a-las-cedulas-de-casi-todos-los-uruguayos-202021412275O

SWEBOK v4

- Describe el conocimiento generalmente aceptado de la ingeniería de software.
- Brinda una caracterización consensuada de los límites de la disciplina de IS.
- Provee acceso por tópicos al cuerpo de conocimiento que la soporta.
- La Guía no debe confundirse con el Cuerpo de Conocimiento en sí mismo.
- Lograr el consenso de todos los sectores de la comunidad en un cuerpo de conocimiento (según IEEE-CS).
  - Es un hito crucial para el desarrollo de la IS como una profesión
  - Establece lo que debe saber un profesional de esta disciplina

SWEBOK v4 (2)

- Caracteriza los contenidos de la disciplina de IS y para la práctica de esta.
- El material se organiza jerárquicamente en KA (áreas de conocimiento) y tópicos.
- SWEBOK 2004 (v3): hito importante en el establecimiento de la ingeniería de software como una disciplina reconocida en la Ingeniería.
- La guía continuó evolucionando para satisfacer las necesidades de la comunidad de IS.

SWEBOK v4 – Objetivos

1. Promover una visión consistente de la ingeniería de software en todo el mundo.
2. Especificar el alcance y aclarar el lugar de la ingeniería de software con respecto a otras disciplinas como ciencias de la computación, la gestión de proyectos, ingeniería en computación y matemáticas.
3. Caracterizar los contenidos de la disciplina de ingeniería de software.
4. Proporcionar un acceso por tópicos al Cuerpo de Conocimiento de Ingeniería de Software.
5. Proporcionar una base para el desarrollo de currículos y para la certificación individual y el material para el licenciamiento.

Calidad del software

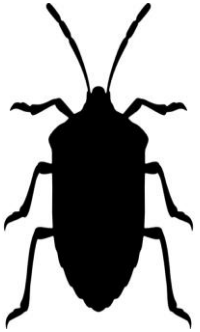
Problemas:

- Plazo
- Costo respecto a presupuesto
- Utilidad
- Requisitos oscuros o cambiantes
- Fallas
- Rigidez
- Alto costo de mantenimiento
- Riesgos asociados a su uso

Calidad (falta - falla)

Un error humano

puede generar



una falta
(interna)

que puede generar

una falla
(externa)

Calidad - Visiones

¿Qué es la calidad?

Visión (Kitchenham, Pfleeger-1996/Garvin) :

- trascendente - se reconoce, pero no se puede definir
- del usuario - adecuación al uso
- del productor - adecuación a las especificaciones
- del producto - características específicas
  - comportamiento externo (visible para todos)
  - características internas (normalmente sólo visibles al productor)
- basada en el valor - cuánto estaría dispuesto a pagar

Calidad – Visiones (2)

Del usuario:

- satisfacer necesidades/expectativas (utilidad, tiempo de respuesta)
- esfuerzo necesario (facilidades de aprendizaje y uso)
- sin inconvenientes (frecuencia e impacto de fallas)

Del implementador:

- cantidad y tipo de faltas
- facilidad de entender
- bajo impacto de las modificaciones

Calidad – Visiones (3)

Según la visibilidad:

- factores externos (visibles a todos)
- factores internos (visibles a los implementadores)

Atinentes al

- producto (una vez que el producto ya existe)
- proceso de producción (mientras se produce)

¿Qué relación habrá entre...

- ... factores internos y externos?
- ... factores del proceso y del producto?

Categorías de proyectos de software

- Desarrollo a medida
  - Cliente (solicita)
  - Desarrollador construye
  - Usuario
- COTS (Commercial Off The Shelf)
- Desarrollador subcontrata parte
- Producto «llave en mano»
- Personalización
- Integración

Desarrollo profesional de software

- ¿Por qué es necesaria la separación entre programación amateur y desarrollo profesional de software?
- Productos de software
  - Productos genéricos

    Sistemas autónomos que se comercializan y venden a cualquier cliente que desee comprarlos (software para PC, software para mercados específicos, como sistemas de citas para dentistas).
  - Productos personalizados

    Software que es encargado por un cliente específico para satisfacer sus propias necesidades (software de control de tráfico aéreo).

La distinción entre ambas radica en quién controla la especificación.

Desarrollo a medida – quiénes participan

CLIENTE

Patrocina el desarrollo del sistema

$$$, Necesidades

Obligación contractual

DESARROLLADOR

Construye el sistema

USUARIO

Necesidades

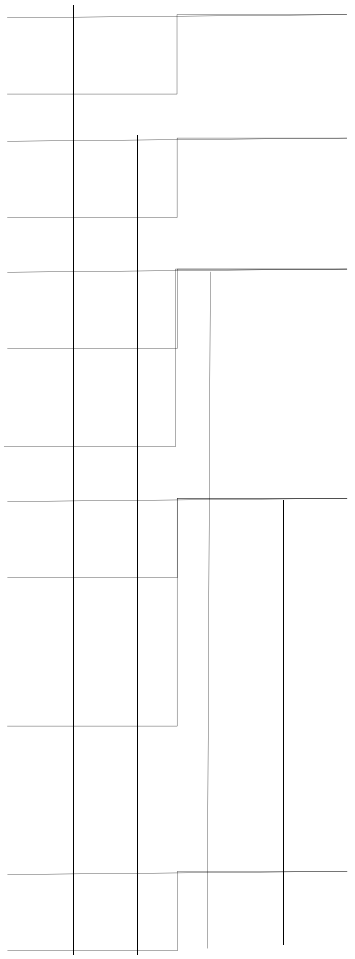
Sistema de software

Usa el sistema

Etapas y roles en el desarrollo

Etapas en el desarrollo de software

- Análisis y def. de requisitos
- Diseño del Sistema
- Diseño del programa
- Implementación del programa
- Prueba unitaria
- Prueba de integración
- Prueba del sistema
- Liberación del sistema
- Mantenimiento



Roles de los desarrolladores

Analista

Diseñador

Programador

Tester

Capacitador

Ingenierías de software y de sistemas

El software como componente de un sistema:

- Hardware
- Software
- Personas
- Firmware

Sistema

- Límite
- Interacción con el exterior
- Componentes y sus relaciones

Cuestiones fundamentales de la IS

Abstracción

- Es una descripción del problema a cierto nivel de generalización que permite concentrarnos en los aspectos esenciales del problema sin preocuparnos de detalles.
- Identificar clases de objetos que permiten agrupar elementos
- Formar jerarquías

Métodos de notación de análisis y diseño

- Construir modelos y verificar completitud y consistencia
- Usar notación estándar para ayudarnos a comunicar y documentar las decisiones

Cuestiones fundamentales de la IS (2)

Prototipación

- Construir una pequeña versión de un sistema para
  - ayudar al usuario y al cliente a identificar los requisitos clave
  - demostrar la factibilidad de un diseño o enfoque
  - evaluar riesgos (cosas que pueden ir mal)
    - ¿El usuario se siente cómodo con la apariencia y forma de interacción?
    - ¿Los tiempos de respuesta/consumo de recursos serán adecuados?

Cuestiones fundamentales de la IS (3)

Arquitectura

- La arquitectura de un sistema lo describe en términos de un conjunto de unidades arquitectónicas y de cómo esas unidades se relacionan entre sí.
- Existen distintos «estilos arquitectónicos» y distintos enfoques para identificar las unidades arquitectónicas

Proceso de desarrollo de software

- Organización y disciplina en las actividades
- Actividades, tareas, roles, responsabilidades
- Contribuir a la calidad del software y a la velocidad con la que se desarrolla
- Distintos tipos de software requieren características distintas del proceso

Cuestiones fundamentales de la IS (4)

Reuso

- Sacar partido de los elementos comunes entre aplicaciones reutilizando elementos de desarrollos previos.
- Componentes reusables como bienes de activo del negocio.

Mediciones

- Cuantificando dónde estamos y lo que podemos hacer, describimos nuestras acciones y sus resultados en un lenguaje matemático común que nos permite evaluar nuestro avance.

Próxima clase:

Procesos de desarrollo de software