INGENIERÍA DE SOFTWARE
INGENIERÍA EN MULTIMEDIA

Marcela Iregui Guerrero

INGENIERÍA DE SOFTWARE
UNIDAD 1
INTRODUCCIÓN A LA INGENIERÍA DE SOFTWARE

Marcela Iregui Guerrero

Análisis

- ¿Qué es software?
- ¿Cómo se construye un Software?
- ¿Qué es la Ingeniería de Software?
- ¿Porqué es importante la Ingeniería de Software?

Qué es Software

- “El software no solamente son los programas sino los documentos y la configuración de los datos que se necesitan para que los programas funcionen”.
- “Los sistemas de software con creaciones complejas:
  - Realizan muchas funciones
  - Comprenden muchos componentes
  - Sistemas difíciles de comprender
  - Están sujetos a cambios constantes”
- El software entrega el producto más importante: información
- Hay software genérico y software hecho a la medida.
- El software no se desgasta, se deteriora debido a los cambios.

Qué es la Ingeniería de Software

- IEEE:
  - “Aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento del software”.
- Pressman:
  - “El marco que incluye un proceso, un conjunto de métodos y una serie de herramientas”.
- Sommerville:
  - “Es una disciplina que comprende todos los aspectos de la producción de software desde las etapas iniciales de la especificación del sistema, hasta el mantenimiento de este después de que se utiliza”.

Qué es la Ingeniería de Software

- Bruegge:
  - Es una actividad de modelado: se crea una representación del sistema que se va a desarrollar
  - Es una actividad para la solución de problemas: se debe manejar la complejidad, la limitación en recursos, los tiempos de entrega
  - Es una actividad para la adquisición de conocimientos: Recopila, organiza, analiza la información y la formaliza en conocimiento
  - Es una actividad dirigida por una fundamentación: captar el contexto y la razón que hay tras las mismas
- Rubby Casallas
  - Hablar de ingeniería de Software es expresar el deseo de contar con prácticas más disciplinadas que el ciclo programar-corregir, que reemplazarán la manera artesanal como se construían las aplicaciones en el pasado.

Atributos del Software

- Mantenibilidad:
  - Que pueda evolucionar, adaptarse fácil y rápidamente al cambio del entorno de los negocios.
- Confiabilidad:
  - Fiabilidad, seguridad, producción.
  - Evitar daños físicos.
  - Evitar daños económicos.
- Eficiencia
  - Tiempos de respuesta.
  - Utilización de memoria, etc.
- Usabilidad
  - Fácil de usar.
  - Interfaz apropiada.

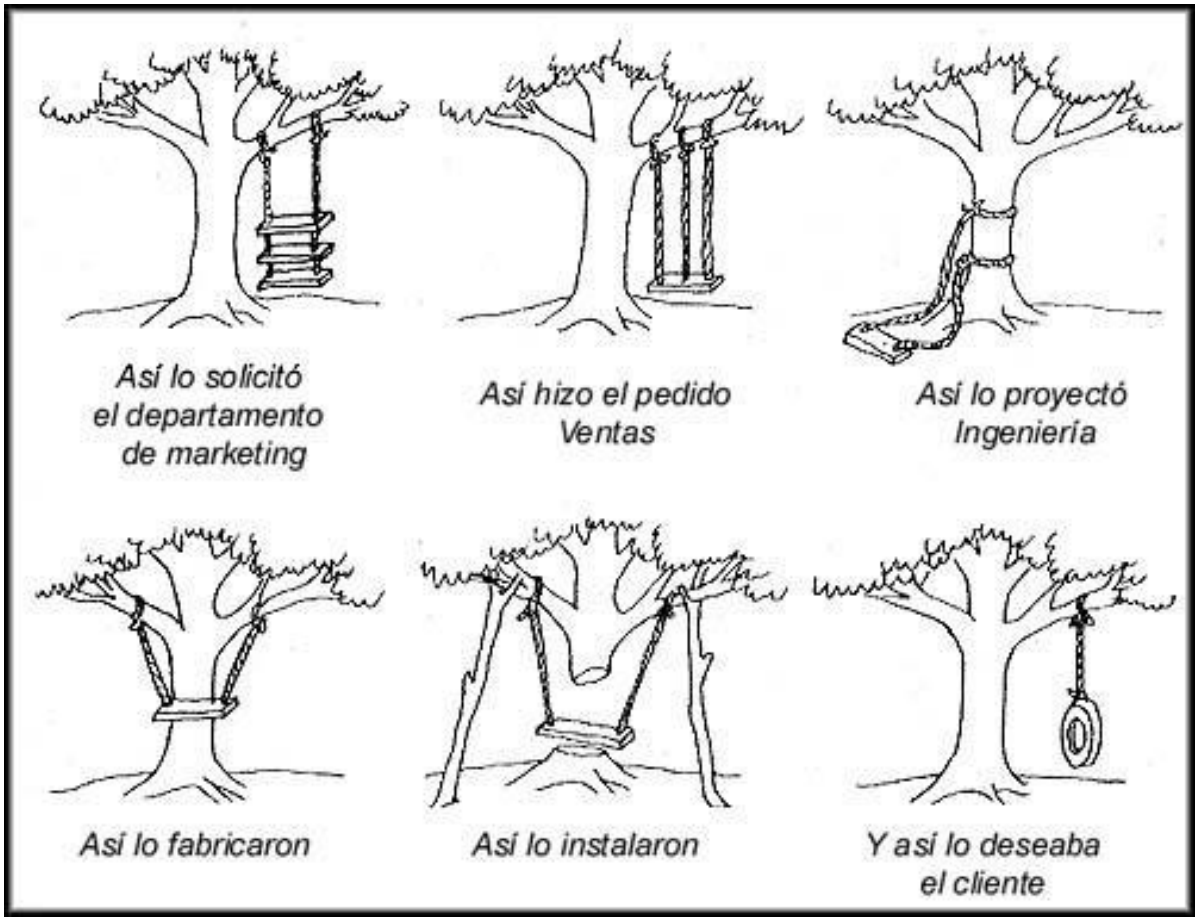
Los fracasos del software

- Retrasos en las entregas del software.
- Abandono de los proyectos de desarrollo.
- Proyectos superan hasta dos veces el costo inicial.
- Los productos no satisfacen los requisitos del cliente.
- Programas con muchos defectos.
- Programas difíciles y costosos de mantener.

Por qué de los fracasos del software

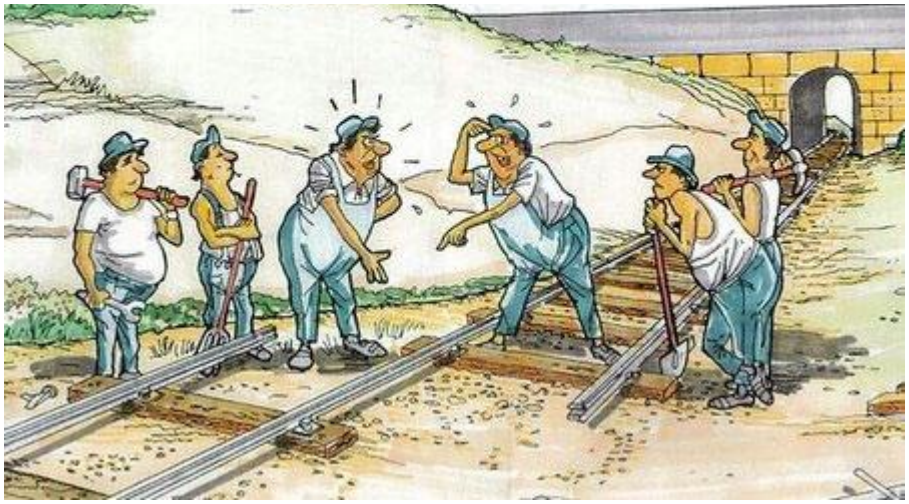
- Falta de análisis y planeación del proyecto
- Por la complejidad de las aplicaciones y el cambio constante
- Se menosprecia la opinión del cliente
- No se tienen en cuenta situaciones excepcionales
- No se anticipa el mal uso del sistema por parte del usuario
- Existen fallas en la administración de los proyectos

Por qué de los fracasos del software



Por qué de los fracasos del software

- No hay liderazgo dentro del proyecto
- Falta una buena comunicación entre el grupo de trabajo
- No están claros los roles y las responsabilidades de cada cual



Los mitos del Software

- Requerimientos
  - MITO: “Se deben tener todos los requerimientos del sistema claramente definidos antes de empezar el proyecto para que este sea exitoso”
    - REALIDAD:
      - El cliente y los usuarios no saben con claridad los detalles de lo que quieren
      - Los requerimientos son cambiantes porque las condiciones cambian
      - Debe existir comunicación efectiva entre el cliente y el desarrollador.
- Diseño
  - MITO: “ Diseñar completamente antes de empezar a programar para que sea más efectiva la labor de programación y más independiente”
    - REALIDAD: Ante los cambios el diseño se tiende a deteriorar y se cae en el ciclo programar-corregir

Los mitos del Software

- Planeación y Seguimiento
  - MITO: Para administrar en forma adecuada el proyecto, se debe tener un plan detallado de todo el proyecto desde el principio
    - REALIDAD: Si los requerimientos no son claros es imposible planear el proyecto desde el comienzo.

Los mitos del Software

- La Entrega
  - MITO: Una vez el programa se ha puesto a funcionar, el trabajo está terminado.
    - REALIDAD: Mientras el programa no se esté ejecutando, no existe forma de evaluar totalmente su calidad.
  - MITO: El único entregable del proyecto es el software funcionando.
    - REALIDAD: Dentro de un proyecto se deben establecer entregables que permitan el seguimiento del desarrollo del proyecto.
  - MITO: La documentación es innecesaria y tornará más lento y costoso el proyecto.
    - REALIDAD: La documentación ayuda a articular las comunicaciones entre los actores, a atesorar el conocimiento generado, a hacer más sencillo es proceso de mantenimiento de las aplicaciones.

Los mitos del Software

- La Gente
  - MITO: Personas entrenadas y especializadas lograrán el éxito del proyecto.
    - REALIDAD: El mito es verdadero pero es difícil encontrar las personas suficientemente cualificadas. Los inexpertos manejan las nuevas tecnologías, los experimentados manejan mejor los procesos.
  - MITO: Contratar más programadores puede ayudar a terminar a tiempo
    - REALIDAD: Agregar gente a un proyecto atrasado lo puede atrasar más. La gente se debe agregar en forma planeada y ordenada.

La Ingeniería de Software

La Ingeniería de Software: Bruegge

- Es una actividad de modelado: se crea una representación del sistema que se va a desarrollar
- Es una actividad para la solución de problemas: se debe manejar la complejidad, la limitación en recursos, los tiempos de entrega
- Es una actividad para la adquisición de conocimientos: Recopila, organiza, analiza la información y la formaliza en conocimiento
- Es una actividad dirigida por una fundamentación: captar el contexto y la razón que hay tras las mismas

Modelado

- Comprender el sistema donde va a operar el Software
  - Los Ingenieros de software necesitan aprender los conceptos del dominio del problema que son relevantes para el sistema
  - Necesitan construir un **modelo de dominio**.
- Comprender los sistemas que se podrían construir para evaluar posibles soluciones y compromisos
  - Los ingenieros de software describen los aspectos importantes de los sistemas alternativos que investigan
  - Necesitan construir un **modelo de solución**.

Solución de problemas

MÉTODO DE INGENIERÍA		INGENIERÍA DE SOFTWARE	
Formular el problema	→	Requerimientos	Modelo del dominio
Analizar el problema	→	Análisis	
Buscar Soluciones	→	Diseño del sistema	Modelo de la solución
Decidir cuál es la solución adecuada	→	Diseño OO	
Especificar la solución	→	Implementación	Traslado del modelo del dominio de solución hacia una representación ejecutable.

Evaluación de los modelos

La evaluación mide si los modelos son adecuados

Modelo de dominio **VS** La realidad del cliente

Modelo de la solución **VS** Objetivos del proyecto

Modelo de proceso
de desarrollo **VS** Realidad

Adquisición del conocimiento

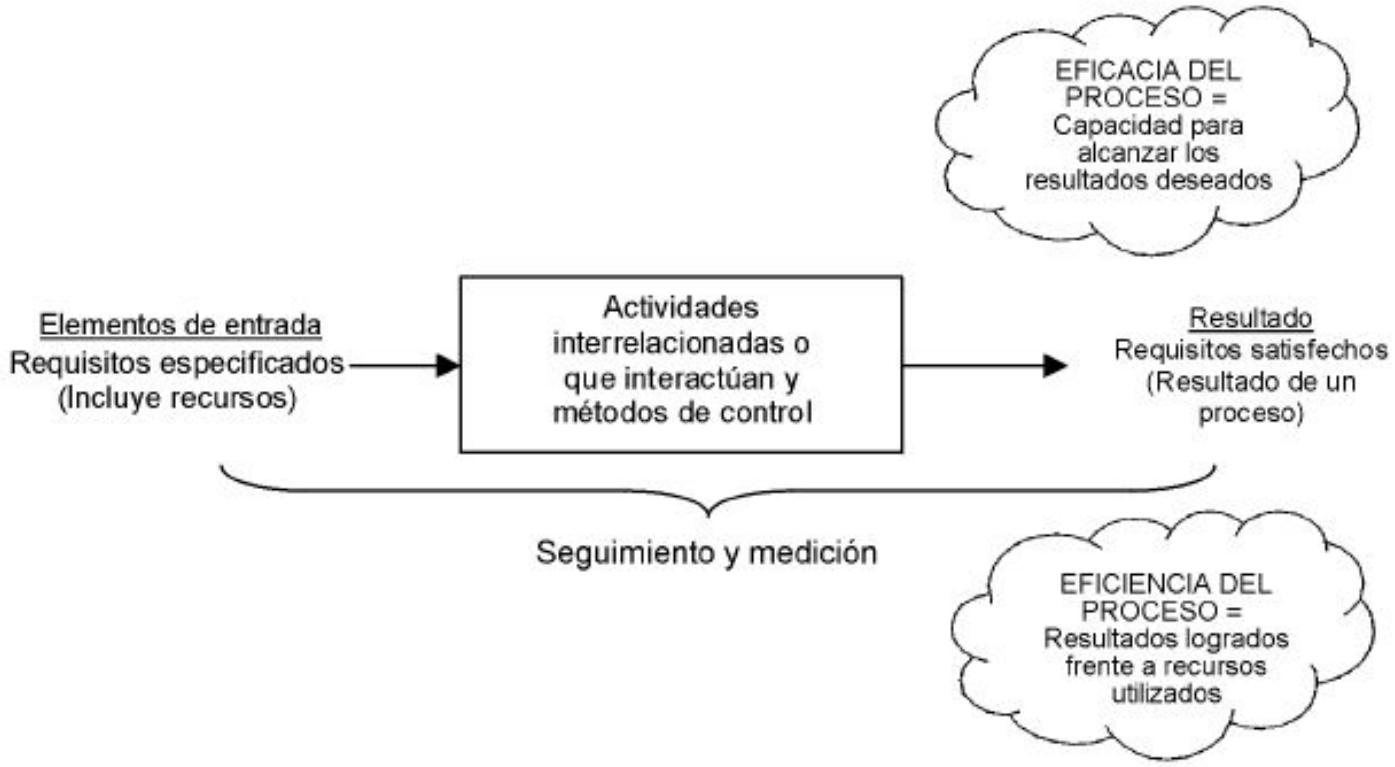
- No es lineal: La adición de un nuevo conocimiento puede invalidar todo el conocimiento que hemos adquirido
- Tiene implicaciones serias sobre los proyectos
- Desarrollo basado en riesgos
- Desarrollo basado en problemas
- Los modelos no lineales son difíciles de manejar

Administración de la fundamentación

- Las suposiciones acerca del sistema cambian constantemente
- Los errores encontrados producen cambios en los modelos de solución
- Cambios tecnológicos conducen a formulación de nuevos requerimientos.
- Cambios en la ley o las normativas formulan nuevos requerimientos
- Para cambiar un sistema no basta con comprender sus componentes y comportamientos
- Para cambiar un sistema se requiere conocer el **CONTEXTO** en que se realizó el sistema.

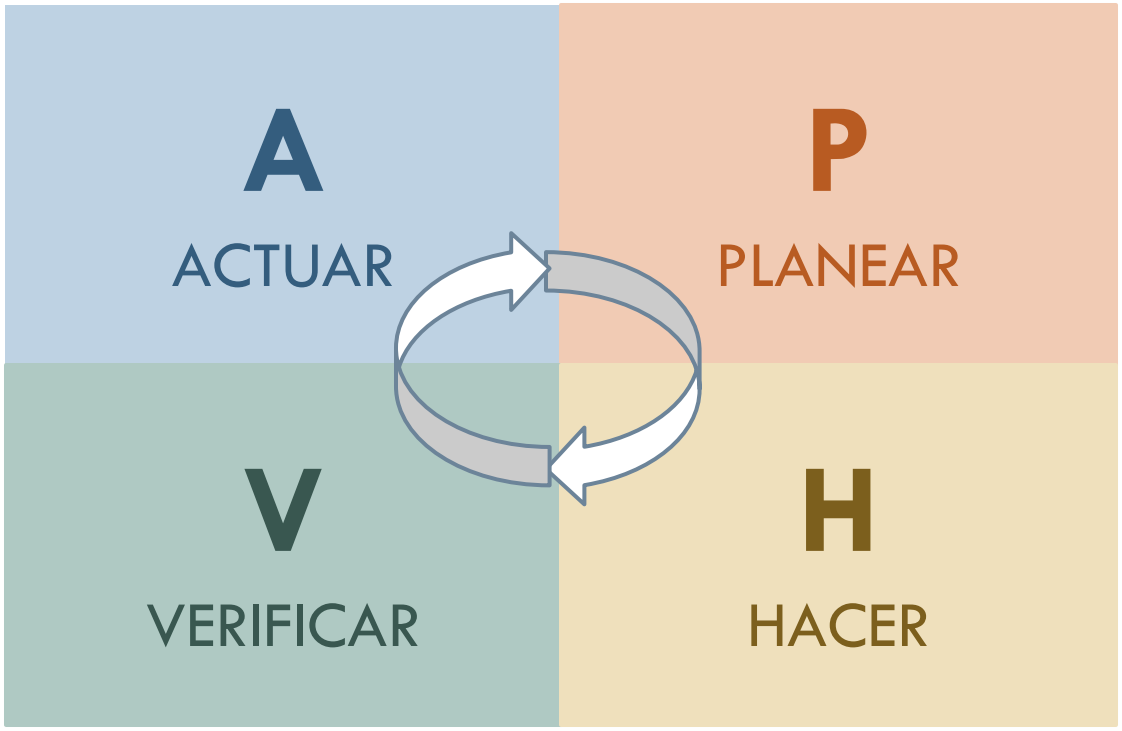
Procesos

Proceso: Es un marco de trabajo para las tareas que se requieren en la construcción de software de alta calidad



Calidad

En el desarrollo de Software el compromiso con la calidad es muy importante.



Proceso de Software

- Marco de trabajo genérico del proceso:
  - Actividades técnicas
    - Obtención de requerimientos
    - Análisis
    - Diseño del sistema
    - Diseño de objetos
    - Implementación
    - Despliegue: Entrega y evaluación
  - Administración del desarrollo
    - Administración del proyecto
    - Administración de la configuración
    - Medición
    - Actividades de modelado del ciclo de vida
    - Administración de la fundamentación

Obtención de requerimientos

- Definir el propósito del sistema: requerimientos funcionales y no funcionales

Actores: usuarios finales, otros sistemas con los que interactúa, medio ambiente.
Casos de uso: Secuencias de eventos generales.
Acciones posibles entre los actores y el sistema

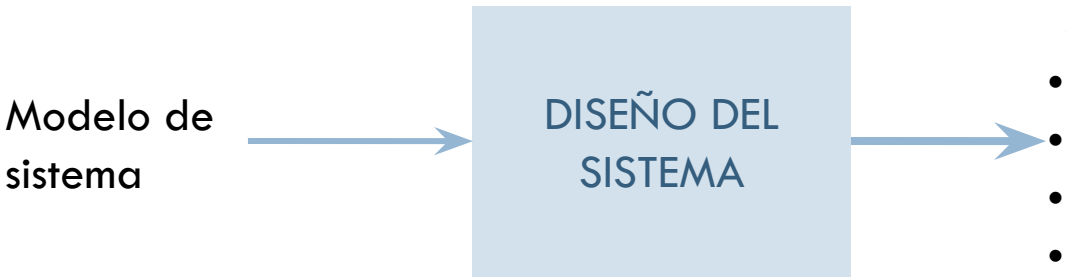
Análisis

- Producir un modelo del sistema correcto, completo, consistente, claro, realista, verificable

UML: Notación para la representación de modelos

Diseño del sistema

- Dividir el sistema en subsistemas que puedan realizar los equipos individuales



División en subsistemas
Estrategias para la construcción del sistema

- Plataforma de HW y SW
- Almacenamiento de datos
- Flujo de control global
- Política de control de acceso
- Manejo de condiciones de frontera

Diseño de objetos

- Producir un modelo del sistema correcto, completo, consistente, claro, realista, verificable

Implementación y despliegue

- Traducir el modelo de objetos en código fuente, Integración de los programas, instalación y validación por el usuario final

Proceso de Software

- Marco de trabajo genérico del proceso:
  - Actividades técnicas
    - Obtención de requerimientos
    - Análisis
    - Diseño del sistema
    - Diseño de objetos
    - Implementación
    - Despliegue: Entrega y evaluación
  - Administración del desarrollo
    - Administración del proyecto
    - Administración de la configuración
    - Medición
    - Actividades de modelado del ciclo de vida
    - Administración de la fundamentación

Proceso de Software

- Marco de trabajo genérico del proceso:
  - Actividades técnicas
    - Obtención de requerimientos
    - Análisis
    - Diseño del sistema
    - Diseño de objetos
    - Implementación
    - Despliegue: Entrega y evaluación
  - **Administración del desarrollo**
    - Administración del proyecto
    - Administración de la configuración
    - Medición
    - Actividades de modelado del ciclo de vida
    - Administración de la fundamentación

Administración del desarrollo

- Administración del proyecto
  - Alcance
  - Tiempo
  - Riesgos
  - Calidad
  - Recursos: Humano y Financiero
  - Comunicaciones
  - Compras
- Administración de la configuración (SCM)
  - Gestión de cambios de los productos de trabajo
- Actividades de modelado del ciclo de vida
- Administración de la fundamentación

Roles y Perfiles en IT

- **Gerente del proyecto:** planea y controla la ejecución del proyecto. Atiende las necesidades de todos los involucrados. Responsable del cumplimiento
- **UI Diseñador:** Creación del concepto y prototipo de interfaz para el funcionamiento y cumplimiento de los requisitos del cliente
- **Analista:** Análisis de requisitos, especificaciones funcionales del sistema y de las bases de datos requeridas.
- **Ingeniero de software y desarrollador:** diseñan la arquitectura del sistema y desarrollar los programas necesarios para asegurar el cumplimiento de los requisitos.
- **Otros: Administrador de la configuración, control de calidad**

Características de un buen líder

Es visionario

competente

entusiasta

buen comunicador

tiene empatía

sabe delegar

trabaja bajo presión

Características de un buen UI diseñador

Entender la tecnología

buen comunicador

creativo

metódico

habilidades en la creación de contenidos

Características de un buen Analista IT

Habilidades para resolver problemas

analítico

relaciones interpersonales

conocimiento de las nuevas tecnologías

Características de un buen desarrollador IT

Pensamiento lógico y solución de problemas

Programación y codificación

Programación OO

Comunicación verbal y escrita

Trabajo en equipo