Introducción a la Ingeniería de Software

Curso de Ingeniería de Software

Primer Semestre 2022

UNIVERSIDAD DE COSTA RICA

SR-CIE

Carrera de
Informática Empresarial
Sedes Regionales

Contenidos

- ¿Qué es la Ingeniería de Software?
- Cuatro p de la Ingeniería de Software
- Signos de que un proyecto de sistemas de información se encuentra en peligro

¿Qué es la Ingeniería de Software?

La Ingeniería de Software

- **Parnas -1978**
  La construcción de múltiples versiones de un software llevada a cabo por múltiples personas

- **Ghezzi - 1991**
  Construcción de software de una envergadura o complejidad tales que debe ser construido por equipo de ingenieros

- **Jackson - 1998**
  La ingeniería tradicional es altamente especializada y se basa en colecciones de diseños estándar o normalizados.¿Hay especialidades en la informática o cualquiera hace cualquier cosa? ¿Se basa la producción en diseños estándar? ¿puede?

- **Parnas -1978**
  Reemplazar las renuncias de responsabilidad por garantías

La Ingeniería de Software

- Aplicación de un enfoque sistemático, disciplinado y cuantificable para el desarrollo, operación y mantenimiento de software; es decir, la aplicación de la ingeniería de software (ISO/IEC/IEEE, 2010, p. 331)

La Ingeniería de Software

La Ingeniería de Software es o debería ser:

- Desarrollo de software de dimensión industrial
- Desarrollo sistemático, disciplinado y cuantificable
- Desarrollo de productos que tienen una vida muy larga
- Desarrollo en equipo
- Especialización
- Diseños estandar
- Producir software garantizado

La Ingeniería de Software

- Manshreck, Winters, y Wright (2020) enmarcan la Ingeniería de Software desde las siguientes ópticas:
  - Desde el proyecto: preocupación por el paso del tiempo y una eventual necesidad de cambio
  - Desde la organización: inquietud por la escala y la eficiencia, tanto del producto de software como de la organización que lo produce
  - Toma de decisiones más complejas con resultados de mayor riesgo, a menudo basados en estimaciones imprecisas de tiempo y crecimiento

La Ingeniería de Software

- El tiempo impacta a un programa de acuerdo con su vida útil esperada: minutos, años o décadas.
- Sostenibilidad para el software
  - Un software es sostenible si, para una vida útil esperada, existe capacidad para reaccionar a cualquier tipo de cambio, ya sea por razones técnicas o asociadas al negocio (Manshreck y cols., 2020).

- EJERCICIO
  - Mencione ejemplos de razones técnicas y asociados al negocio que pueden derivar en cambios

La Ingeniería de Software

- Otra forma de mirar la ingeniería de software es considerar la **escala**. ¿Cuánta gente está involucrada y qué rol desempeñan en el desarrollo?

- Una **tarea de programación** es una frecuentemente un acto de **creación individual**, pero una **tarea de ingeniería de software** es un **esfuerzo de equipo** (Manshreck y cols., 2020)

- La colaboración en equipo genera nuevos retos, pero potencia la creación de sistemas valiosos más allá de lo que se podría lograr individualmente

La Ingeniería de Software

Que hace el ingeniero de software:

- **Maquinas de software**
  - No construye el hardware, sino el comportamiento y las propiedades que lo harán útil para algo específico.

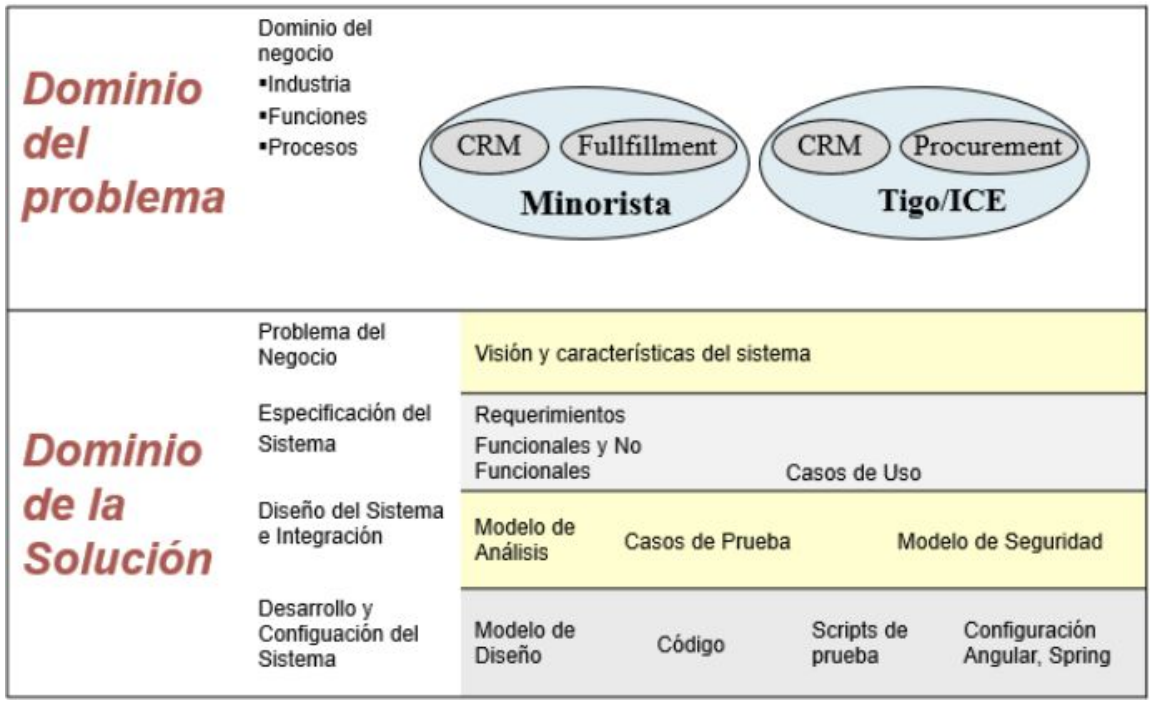
- **Escribe descripciones**
  - La actividad central del desarrollo de software es la descripción.
  - Cualquier desarrollo de software requiere muchas descripciones
  - Verificar las descripciones

La Ingeniería de Software

- Manassis (2003) provee una definición de Ingeniería de Software con un enfoque a la aplicación:
  - Refinamiento del conocimiento a través de sucesivos niveles de abstracción y de representación
  - Trazabilidad de cada ítem de información entre los niveles de abstracción

La Ingeniería de Software - Niveles de abstracción

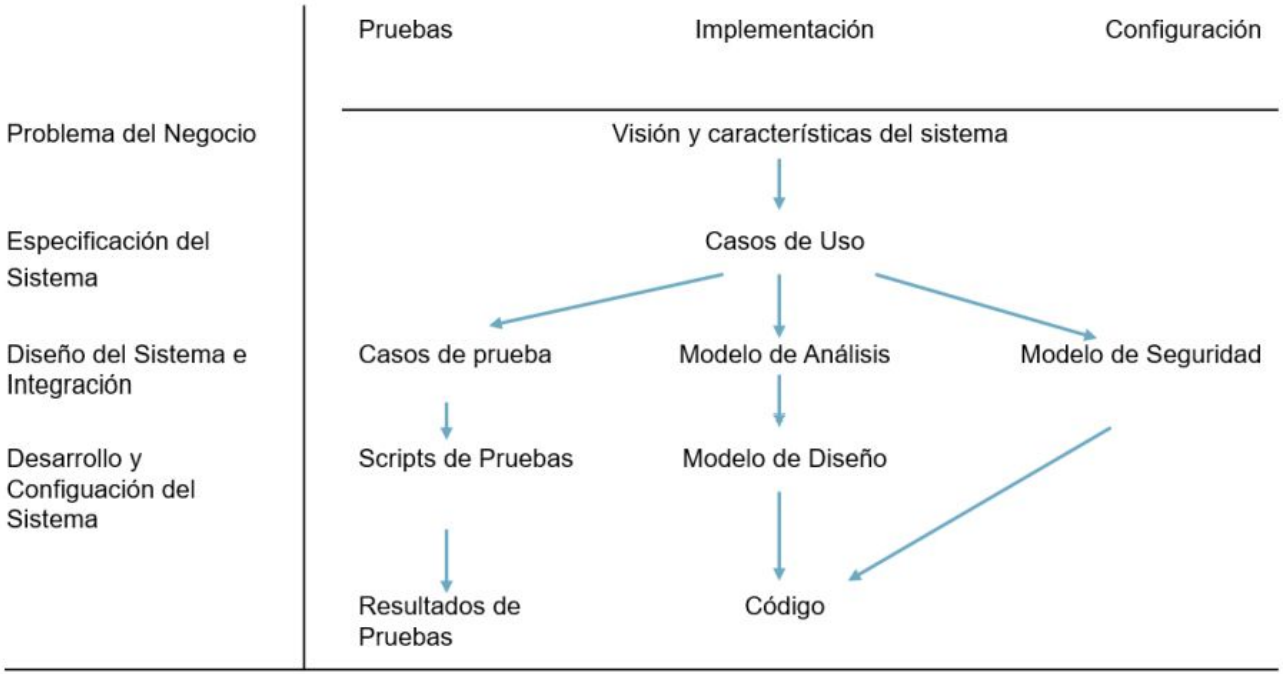
- La Ing.Sw. implica expresar el conocimiento de cierta forma: **REFINARLO** (ir de la visión al código) y **ABSTRAERLO** (cada nivel de abstracción toma diversas representaciones).



- **Figura 1:** Niveles de abstracción y representación de la Ing. Sw. Adaptado de: (Manassis, 2003)

La Ingeniería de Software - Trazabilidad

- La trazabilidad es método para rastrear un elemento del proyecto con otros elementos del proyecto, en especial requerimientos



- **Figura 2:** Trazabilidad en la Ing. Sw. Adaptado de: (Manassis, 2003)

La Ingeniería de Software - Matriz de trazabilidad

- Desde la disciplina del Testing se emplea la matriz de trazabilidad de requerimientos (RTM) para establecer la correlación entre lo que desea testear (los requerimentos) y los escenarios de prueba y casos de prueba.
- La RTM favorece la identificación de cualquier fuga en la calidad y asegura una cobertura completa de las pruebas.

#Id. requerimiento de negocio	#ID. caso de uso	Prioridad	#ID Caso de prueba
BR_1	UC_1	Alta	TC#001
	UC_2	Alta	TC#002 TC#005
BR_2	UC_3	Media	TC#003 TC#004

- **Cuadro 1:** Ejemplo de matriz de trazabilidad. Fuente: (Atar, 2019)

La Ingeniería de Software - Matriz de trazabilidad

- Entender la fuente de los requerimientos
- Administrar el alcance del proyecto
- Administrar los cambios en los requerimientos
- Estimar el impacto de cambio de req. en el proyecto
- Verificar que todos los req. son implementados
- Estimar el impacto de una falla de una prueba en los requerimientos

La Ingeniería de Software - ¿es ingeniería?

Sin garantía

Casi ningún programa se entrega con garantia. ¿Podemos decir que el software es el resultado de una ingeniería cuando todos los productos de otras ingenierías tienen garantía?

Sin diseños estándar

Excepto en pocas excepciones, los desarrolladores de software tienden a inventar todo en cada proyecto. ¿ Podemos decir que el software es el resultado de una ingeniería cuando casi todo se hace desde cero casi siempre

La Ingeniería de Software - ¿es ingeniería?

Aeropuerto de Denver en 1993 (EE.UU)

10 veces el tamaño de Heathrow (aeropuerto de londres)

Sistema subterráneo de traslado de equipaje: 4000 telecarros independientes

Software para el control del sistema de carros (21 meses)

La inauguración se debió postergar 3 veces

El presupuesto era de USD 193M

BAE Automated Systems reconoció que no podía predecir el momento en que lograría estabilizarlos

En 2005 United decidió abandonar el sistema y ahorrarse USD 1M por mes en mantenimiento

¿Por que la producción de software es así?

Posibles causas:

- No se atacan las dificultades esenciales de la producción de software
- Es una disciplina inmadura debido a su corta historia
- Es esencialmente diferente a las otras ingenierias

Dificultades esenciales de la producción de software

¿Donde estan las dificultades?

- La parte difícil de construir software es especificarlo, diseñarlo y verificarlo, no la tarea de representarlo.

- Es decir, lo difícil es saber que hay q hacer y cómo dividirlo en pequeñas partes, no implementar esas partes

Es inmadura.. ¿debería serlo?

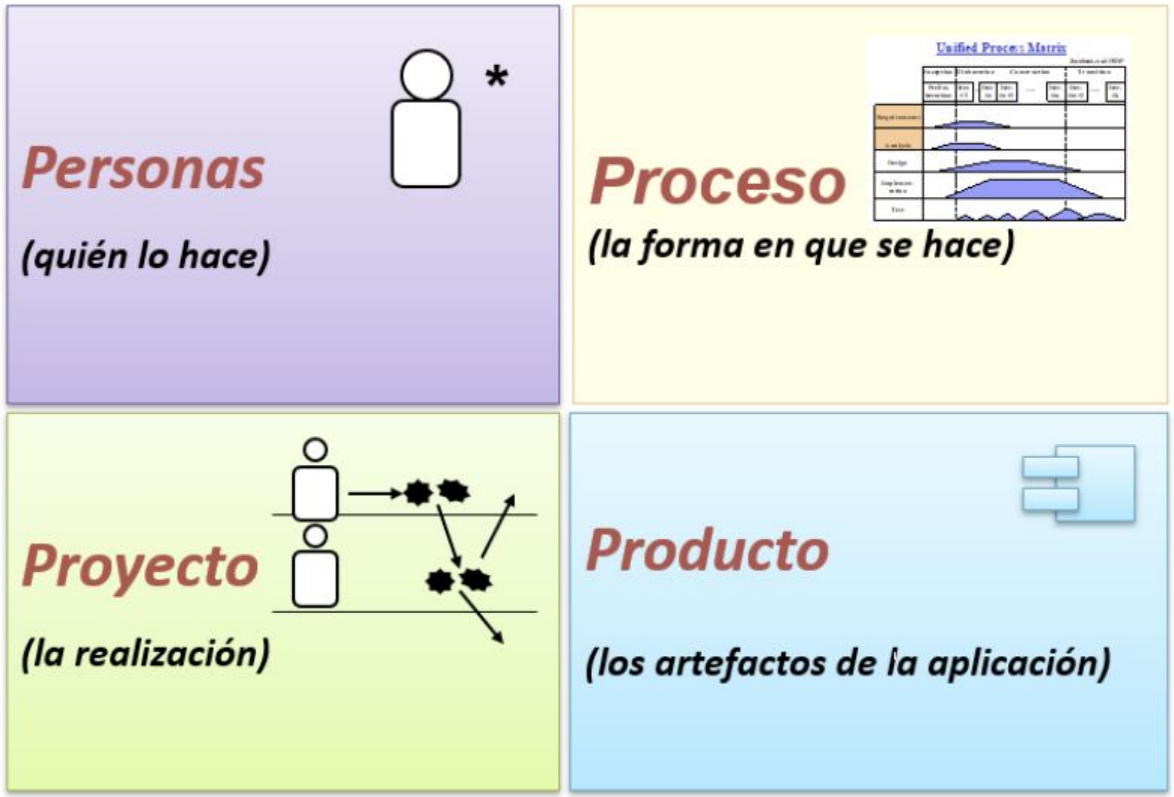
- ¿Electrónica? ¿Biotecnologia? ¿Industria aeroespacial?.
- Falta de especialización
- En la mayoría de los casos carece de diseños normalizados.
- Se habla siempre de construir sistemas y no de construir dispositivos

Es esencialmente diferente a las otras ingenierias

- La ciencia (formal) que subyace a la ingeniería de software es la lógica formal
- La ciencia (fáctica) que subyace a las ingenierías tradicionales es la física
- En la mayoría de las otras ingenierías el ingeniero se concentra en definir la solución
- Los ingenieros de software deben concentrarse también en definir el problema

Cuatro p de la Ingeniería de Software

Cuatro p de la Ingeniería de Software



- **Figura 4:** Cuatro p de la ingeniería de software. Adaptado de: (Braude, 2005)

Proceso

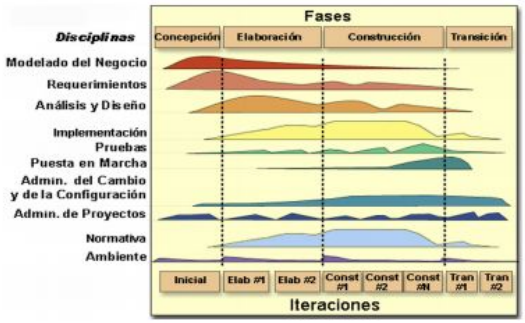
- Conjunto de actividades que se realizan para producir una aplicación

Modelo de proceso del software

- Cascada
- Desarrollo Evolutivo
- Desarrollo basado en componentes
- Iterativo-incremental
- Espiral
- Prototipado

Metodologías de desarrollo

- Proceso Unificado
- Programación Extrema
- OpenUP
- Scrum



- **Figura 5: Proceso unificado**

Proyecto

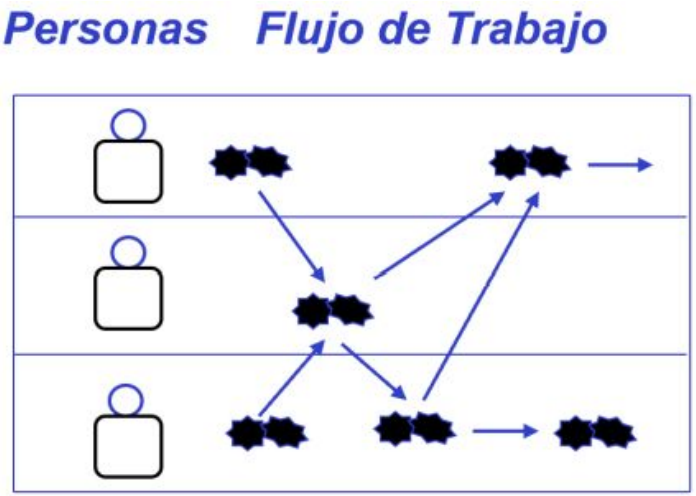
- Westland (2006, p.2) define un proyecto como un esfuerzo único para producir un conjunto de entregables en un período claramente establecido y con restricciones de costos y calidad

Características

• Es de naturaleza única: produce un resultado único (prod. o serv.)

• Tiene un tiempo de desarrollo definido: posee una fecha de inicio y de finalización

• Supone cambios benéficos



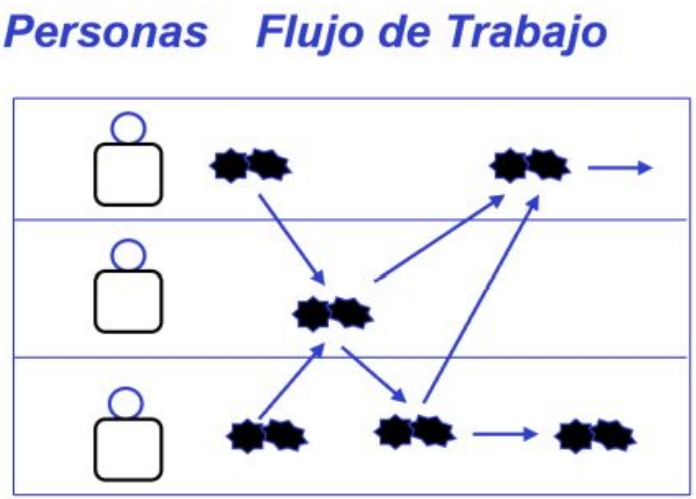
Proyecto

- **Características**

  • Posee recursos limitados: se le asigna una cantidad de trabajo, materiales y equipamiento

  • Involucra elementos de riesgo: genera un nivel de incertidumbre

  • Tiene un presupuesto aprobado: se le asigna un nivel de gastos en el cual debe producir los entregables



Personas

- El desarrollo de un sistema de software involucra la participación de personas
- El alcance de los objetivos y expectativas de un proyecto depende del factor humano

Stakeholders - personas interesadas

• personas que se verán afectadas de algún modo por el desarrollo del proyecto

• usualmente se involucran directamente con el trabajo del proyecto

Personas

- Es importante identificarlos con claridad
- Son claves para el éxito del proyecto
- Poseen conflictos de intereses

Tipos de stakeholders

• Patrocinador:ejecutivo/a de la organización con autoridad para asignar recursos y para influir en decisiones sobre el proyecto

• Clientes, usuarios finales, contratistas, proveedores

• Director/a de proyecto, equipo de desarrollo, gerentes de área

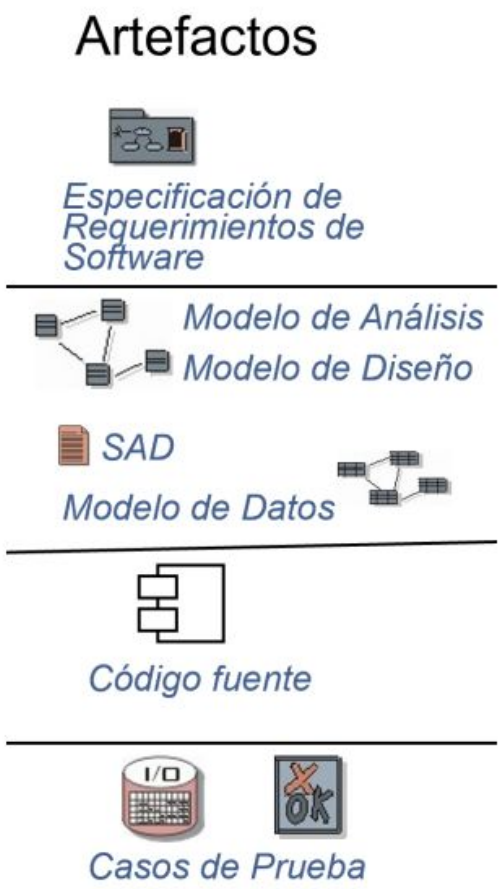
Producto

- Software: la aplicación y los artefactos asociados. Clasificación según el área:

Software de sistema

•Colección de programas escritos para servir a otros programas

• Ejemplos: compiladores, editores, utilitarios de gestión de archivos, drivers, componentes del sistema operativo



Producto

- Clasificación según el área (2):

Software de tiempo real

• Monitorea/analiza/controla eventos del mundo real cuando ocurren

Producto

- Clasificación según el área (3):

Software empotrado

• Asociado con los productos inteligentes. Reside en la memoria RAM

• Ejemplos: control del teclado de un microondas, funcionales digitales en un automóvil

Producto

- Clasificación según el área (4):

Software de negocios

• Facilitan las operaciones del negocio y la toma de decisiones. Manejan volúmenes grandes de información

Producto

- Clasificación según el área (5):

Otras clasificaciones

• Software para computadores personales: hojas electrónicas, procesador de palabras

• Aplicaciones web

• otros: software científico/ingenieril, de inteligencia artificial

Actividad: Principios, conceptos y ética en la Ingeniería de Software

Actividad

- Con base en los aportes de Tsui, Karam, y Bernal (2018) sobre la ética y la profesión de la ingeniería de software (capítulo 3):
  - Código de ética de Ingeniería de Software
  - Principios tempranos de la Ingeniería de Software de Davis
  - Principios más modernos de Royce
  - Conceptos fundamentales de la Ingeniería de software de Wasserman

Actividad

- Indicaciones - tiempo: 30 minutos
- 1) Formen cuatro grupos
- 2) En una hoja incluya todos los aspectos que componen el código de ética, principios o conceptos de la Ing. Sw. que se les asignó.
- 3) Seleccione y desarrolle cinco aspectos que más les llamó la atención
- 4) Realicen una presentación oral ante la clase

Signos de que un proyecto de sistemas de información se encuentra en peligro

Signos de peligro en un proyecto

- Pressman (2010, p. 660) plantea la necesidad de entender qué puede salir mal en un proyecto para tratar de evitarlo. Así, cita el artículo de Jonh Reel que aborda algunos signos que podrían indicarnos que un proyecto está mal encaminado:
  - El equipo de desarrollo no entiende las necesidades de sus clientes
  - El alcance del producto es pobremente definido
  - Los cambios son mal administrados
  - La tecnología escogida cambia
  - Las necesidades del negocio cambia o están mal definidas

Signos de peligro en un proyecto

- ...
  - Fechas de entrega irrealistas
  - Resistencia por parte de los usuarios
  - El patrocinio del proyecto se pierde
  - El equipo del proyecto carece de competencias apropiadas
  - Gestores y desarrolladores evitan las mejores prácticas y lecciones aprendidas

Referencias

Atar, A. (2019). Hands-on test management with jira: End-to-end test management with zephyr, synapsert, and jenkins in jira. Packt Publishing Ltd.

Braude, E. J. (2005). Ingeniería de software: una perspectiva orientada a objetos. Alfaomega.

ISO/IEC/IEEE. (2010, Dec.). International standard - systems and software engineering – vocabulary (Standard n.o ISO/IEC/IEEE 24765:2017(E)). ISO/IEC/IEEE. doi: https://doi-org.ezproxy.sibdi.ucr.ac.cr/10.1109/IEEESTD.2010.5733835

Referencias

Manassis, E. (2003). Practical software engineering: analysis and design for the. net platform. Addison-Wesley.

Manshreck, T., Winters, T., y Wright, H. (2020). Software engineering at google. O'Reilly Media, Inc.

Pressman, R. (2010). Software engineering:a practitioner's approach. Mc Graw Hill.

Tsui, F., Karam, O., y Bernal, B. (2018). Essentials of software engineering (4.a ed.). Jones & Bartlett Learning.

Westland, J. (2006). The project management life cycle: A complete step-by-step methodology for initiating planning executing and closing the project. Kogan Page.