Fecha: diciembre de 2021

PROGRAMA ACADÉMICO: Ingeniería de Sistemas y Computación

SEMESTRE: Quinto

ASIGNATURA: Ingeniería de Requisitos

CÓDIGO: 8105574

NÚMERO DE CRÉDITOS: 4

PRESENTACIÓN

En la actualidad persisten problemas en el desarrollo de software, entre ellos, un inadecuado entendimiento de las necesidades de los usuarios, incapacidad de absorber cambios en los requisitos e insatisfacciones de los clientes por inaceptable o bajo desempeño del software.

Las principales causas son la administración insuficiente de requisitos; los problemas que afectan la comunicación; las inconsistencias no detectadas entre requisitos, diseño y programación; las validaciones tardías de requisitos; el enfrentamiento reactivo de riesgos y la propagación de cambios sin control. Los modelos de proceso de Ingeniería de Requisitos (IR), a pesar de su evolución, aún presentan carencias. Por tanto, para obtener un producto de calidad, se requiere una mejora en los procesos de IR.

JUSTIFICACIÓN

La parte más difícil de construir de un sistema software es decidir qué construir. Ninguna otra parte del trabajo afecta más negativamente al sistema final si se realiza de manera incorrecta. Ninguna otra parte es más difícil de rectificar después. Por lo tanto, el desarrollo del software sólo puede ser iniciado cuando se tiene bien establecido lo que se quiere producir.

El proceso de Ingeniería de Requisitos es un conjunto estructurado de actividades que sirven para derivar, validar y mantener los requisitos de un sistema (hardware, software o hardware+software).

Las distintas tareas que se ejecutan durante el proceso de requisitos suceden en paralelo y se solapan unas con otras. Por ejemplo, durante un proceso de educción de requisitos empleando prototipado, es inevitable realizar una pequeña validación de los requisitos que se van obteniendo, o incluso una pequeña negociación, si estamos tratando con varios usuarios a la vez.

Se pueden dar diferentes variaciones en el proceso, ya sea según la naturaleza del proyecto (dirigido a mercado, a medida), o según la naturaleza de la aplicación (riesgo, recursos, incertidumbre, sistemas empotrados).

COMPETENCIAS

Se pretende que al finalizar la presente asignatura el estudiante:

- Tenga una visión general del proceso de desarrollo de software y de sus principales problemas.
- Sea consciente de los aspectos socio-económicos de la ingeniería del software y de cómo se abordan en la fase de ingeniería de requisitos, sin duda la de mayor relevancia desde un punto de vista social.

- Conozca y aplique los conceptos básicos, estado actual y el futuro de la Ingeniería de Requisitos y del prototipado.
- Compare ventajas e inconvenientes de utilizar técnicas de especificación formal frente a técnicas clásicas en la especificación y análisis de requisitos.
- Explique los objetivos de las herramientas CARE (Computer-Aided Requirements Engineering) y sus funcionalidades esperadas. Se revisarán algunas de las herramientas actualmente existentes en el mercado.
- Produzca una documentación correcta y completa de especificación de requisitos de software de acuerdo con los estándares internacionales y prácticas más comunes.
- Maneje las principales técnicas de la notación estándar UML, que pueden utilizarse en el modelado de requisitos.
- Conozca los principales enfoques, estrategias y modelos de proceso en la aplicación de la Ingeniería de Requisitos.
- Realice trabajos prácticos que lo ayude a alcanzar una cierta capacitación para valorar, planificar y controlar las primeras etapas en el proceso de producción de software, como miembros de un grupo de desarrollo.

RESULTADOS DE APRENDIZAJE

- Conozco de manera general el proceso y principales problemas del desarrollo de software, y particularmente, los conceptos básicos, estado actual, el futuro, los enfoques, estrategias y modelos de proceso y su aplicación en la Ingeniería de Requisitos y prototipado, siendo consiente de los aspectos socio-económicos de la ingeniería del software y de la importancia de abordarlos en la fase de ingeniería de requisitos.
- Comparo ventajas, objetivos e inconvenientes de utilizar técnicas de especificación formal frente a técnicas clásicas en la especificación y análisis de requisitos, y las funcionalidades de herramientas CARE (Computer-Aided Requirements Engineering), para aplicarlas en trabajos prácticos, planificando y controlando cada una de las primeras etapas del desarrollo de software.
- Documento de manera correcta y completa, con las principales técnicas de la notación estándar UML y con base en la especificación de requisitos de software de acuerdo con los estándares internacionales y prácticas más comunes.

METODOLOGÍA

Se incentivará el deseo de la lectura proponiendo la consulta de cada uno de los temas para mayor enriquecimiento, los cuales serán desarrollados a través de exposiciones del profesor haciendo énfasis en la relación que existe entre estos y el mundo real, con la participación activa de los estudiantes, así mismo la realización de talleres con ejemplos y ejercicios. Es decir, el estudiante sea parte fundamental en su propio aprendizaje:

- La motivación será orientada al trabajo, la creación de hábitos de estudio, investigación y trabajo en equipo. Para poder desarrollar esta metodología, se integrará desde la lección magistral del Docente, hasta la facilitación de materiales de trabajo.
- Se tratará por tanto de fomentar en equilibrio la participación individual, en grupo y general de forma que ninguna predomine excesivamente sobre las otras.
- Se impartirán cinco horas semanales distribuidas en sesiones magistrales y talleres prácticos que permitan evaluar el conocimiento impartido al estudiante.

Se tendrá como objetivo fundamental el dotar al estudiante de los conocimientos teóricos y prácticos suficientes para afrontar el desarrollo de cada proyecto de ingeniera de software

INVESTIGACIÓN

Lecturas de cada uno de los temas propuestos en los contenidos temáticos.

MEDIOS AUDIOVISUALES

Ambiente Aula virtual moodle.
Salas de Informática.
Video Beam

EVALUACIÓN

EVALUACIÓN COLECTIVA

EVALUACIÓN INDIVIDUAL

El cálculo de la nota final se hará de la siguiente manera:

PRIMER y SEGUNDO 50%
- 40% Evaluaciones y/o sustentaciones de proyectos
- 60% Talleres, exposiciones o participaciones en clase

CONTENIDOS TEMÁTICOS CENTRALES

UNIDAD I: FUNDAMENTOS DE LOS REQUERIMIENTOS DE SOFTWARE
- Definición y conceptos básicos
- ¿Qué es la Ingeniería de Requisitos?
- ¿Qué es un Requisito?
- Importancia de la Ingeniería de Requisitos
- Esquema general de IR
- Métodos de aplicación de la IR
- Temas y problemas en la obtención de requisitos
- Clasificación de requisitos (Formales, No formales y de Dominio)
- Otros Requisitos (Usuario, Sistema)
- Stakeholders (interesados)

UNIDAD II: TÉCNICAS DE RECOLECCIÓN (ELICITACIÓN) DE REQUERIMIENTOS
- Técnicas de recogida de información
- La técnica de casos de uso y el documento de especificación
- Otras técnicas relacionadas con los casos de uso: historias de usuario y escenarios
- Técnicas formales
- Técnicas formales frente a intuitivas
- Otras técnicas intuitivas
- Listar las fuentes de requerimientos
- Categorías de los Stakeholders
- Perfil de los Stakeholders

- Plan de elicitación de Stakeholders
- Entrevistas
- Análisis de documentos
- Glosario
- Observación
- Encuesta/Cuestionario

UNIDAD III: ANÁLISIS DE REQUERIMIENTOS
- ¿Por qué se debería crear modelos de requisitos?
- El ciclo de análisis de requerimientos
- ¿Qué modelos se pueden crear?
- ¿Cómo elegir el modelo correcto?
- Herramientas y técnicas para analizar requerimientos
- El modelado del negocio
- Describir el alcance del proyecto
- Agregar detalle a los requerimientos de usuario
- Priorizar los requerimientos

UNIDAD IV: ESPECIFICACIÓN DE REQUERIMIENTOS
- Proceso de especificación de requerimientos
- Criterios de calidad
- Técnicas y herramientas para documentar requerimientos
- Beneficios de un buen documento SRS
- Un modelo de especificación de requisitos del software: el estándar IEEE 830-1998
- ¿Qué se incluye en el documento de especificación de requisitos del software (SRS)?
- Propiedades de un documento de especificación de requisitos
- Estructura del documento SRS del IEEE 830-1998
- Alternativas en la especificación de requisitos específicos, según IEEE 830-1998

UNIDAD V: VALIDACIÓN DE REQUERIMIENTOS
- Validación de requerimientos
- Técnicas de validación

UNIDAD VI: ADMINISTRACIÓN DE REQUERIMIENTOS
- Gestión de Requisitos
- Herramientas y técnicas que se utilizan en la gestión de requisitos
- Ventajas y desventajas del uso de herramientas software en la IR

UNIDAD VII. HERRAMIENTAS DE REQUERIMIENTOS DE SOFTWARE

LECTURAS MÍNIMAS

Artículos y apartados bibliográficos de los diferentes temas a tratar, proporcionados por el Docente. Consulta permanente a los foros de las comunidades de desarrolladores

BIBLIOGRAFÍA

1. Braude, Eric J. Ingeniería de Software: una perspectiva orientada a objetos, Editorial Alfaomega, 2003
2. Booch, G. Análisis y Diseño Orientado a Objetos con aplicaciones, 2ª edición, Addison-Wesley,

1996.
3. Booch, G. et al. Unified Modeling Language UML Notación guide and UML Semantics, version, 1.3.
4. Bruegge, B. Ingeniería de Software Orientada a Objetos, Prentice Hall, 2002.
5. Debrauwer, Laurent. et al. UML 2.5: iniciación, ejemplos y ejercicios corregidos. Ediciones ENI, cuarta edición. 2016.
6. Fontela, Carlos; Fernández, Damián. UML: modelado de software para profesionales. Alfaomega. 2011.
7. Jacobson I., et al., El Proceso Unificado de desarrollo de software, Addison-Wesley, 2000.
8. Jacobson, Ivar; Booch, Grady; Rumbaugh, James; Sánchez, Salvador. UML : el proceso unificado de desarrollo de software. Pearson. 2000.
9. Kneuper, R. Software Processes and Life Cycle Models. Springer Verlag A.G. 2018. Tomado de: https://biblio.uptc.edu.co:2625/content/pdf/10.1007%2F978-3-319-98845-0.pdf
10. Kotonya, G., y Sommerville, I. Requirements Engineering. Processes and Techniques, 1998
11. Robertson, S., y Robertson, J, Mastering the requirement process. Addison-Wesley. 1999
12. Larman, C. UML y Patrones: Una Introducción al Análisis y Diseño Orientado a Objetos y al Proceso Unificado, Segunda Edición, Prentice-Hall, 2002.
13. Meyer, B. Construcción de software orientado a objetos, 2ª Edición,Prentice-Hall, 1998.
14. Müller, P. Schaefer. I. Principled Software Development. Springer Verlag A.G. 2018. Tomado de: https://biblio.uptc.edu.co:2625/content/pdf/10.1007%2F978-3-319-98047-8.pdf
15. Pressman, Roger S. Ingeniería del Software: Un enfoque práctico, McGraw Hill, quinta edición.
16. Rumbaugh, J., et al., El Lenguaje Unificado de Modelado, Manual de Referencia, Addison-Wesley, 2000.
17. Sommerville, Ian. Ingeniería del Software. Prentice Education. 10ª Edición. 2016. Tomado de: https://biblio.uptc.edu.co:2566/?il=7500
18. Dick, Jeremy ; Hull, Elizabeth; Jackson, Kenneth. Requirements engineering. Springer international publishing, cuarta edición. 2017.
19. Gracia Burgués, Julián Esteban. Aprende a modelar aplicaciones con UML. IT Campus Academy. Segunda edición. 2016.
20. Vizcaino, A. Desarrollo global de Software. Editorial Ra-Ma. 2015. Tomado de: https://biblio.uptc.edu.co:2566/?il=7928