Fecha: diciembre de 2021

PROGRAMA ACADÉMICO: Ingeniería De Sistemas Y Computación

SEMESTRE: Sexto

ASIGNATURA: Ingeniería de software I

CÓDIGO: 8105517

NÚMERO DE CRÉDITOS: 4

PRESENTACIÓN

El curso de ingeniería de software I está orientada a la formación del estudiante para la creación de proyectos de software, cumpliendo con lineamientos técnicos, administrativos y de calidad, para que sean aplicados al sector productivo. El curso está diseñado para brindar herramientas para la construcción de proyectos de software mediante diferentes procesos, metodologías y modelos.

El curso inicia con una introducción a la ingeniería de software, donde se profundizará sobre las características generales y la evolución de la disciplina en el desarrollo de software en los últimos años. En la segunda parte se presenta los diferentes tipos de modelos de proceso de desarrollo de software para analizar sus fortalezas, características y estrategias. Después se introduce al diseño de software mediante conceptos de diseño, diseño de arquitecturas de software, diseño de componentes, diseño de interacción hombre computador y patrones de diseño necesarias para la optimización de los procesos en la ingeniería de software. En el curso se realiza una introducción a la calidad de software mediante el estudio de algunas técnicas y métricas que son estratégicas para la revisión de pruebas en un producto software. Por último, para cada uno de los proyectos de software es necesario conocer la gestión de proyecto, en este capítulo se realizarán un análisis de conceptos de administración de proyectos de software, explorando metodologías y estimaciones necesarias para que un proyecto de software sea exitoso.

JUSTIFICACIÓN

La forma de hacer software no permanece estática como lo hacen otras disciplinas, el desarrollo de software ha cambiado en la última década a tal ritmo que es difícil llevar un control de dichos cambios, cada año se presentan nuevas prácticas, paradigmas, tecnologías y herramientas que pretenden simplificar este proceso y lograr una mayor calidad, pero que han hecho difícil consolidar un modelo unificado por ello los profesionales en el áreas de las ciencias de la computación deben estar al tanto de estos cambios para que puedan ser aprovechados en las organizaciones donde trabajan y así logran mejoras en los procesos y productos.

Es importante destacar que un profesional que haya estudiado el proceso de desarrollo de software hace unos años se llevara algunas sorpresas en los cambios que se han presentado en este entorno, aunque las etapas del ciclo de vida se mantienen la forma de desarrollar software ha tenido que adaptarse a las necesidades del mercado, el software en muchos campos ya no se ve como un producto sino como un servicio; un ejemplo de cómo ha cambiado el panorama de desarrollo de software está en que tan frecuentemente se actualiza un programa, hace varios años se acostumbraba a las actualizaciones anuales o en el mejor de los casos mensuales, ahora el código base de una aplicación usada por más de 800 millones de personas como es Instagram (aplicación para compartir fotos) se despliega de 30 a 50 veces por día y muy pocas veces falla, esto debería generar una pregunta ¿Cómo lo hacen?, la conclusión es que un profesional en TI competitivo debería ser capaz de contestar esta pregunta así sea de manera general.

Por lo anterior, se ha evidenciado que en la actualidad la ingeniería de software es una disciplina importante que ha evolucionado de manera significativa para la solución de problemas en el área de las ciencias computacionales o de la informática. Este curso ayudará al estudiante a analizar y aplicar conceptos de procesos de software, metodologías y diseño que son elementales para la aplicación de proyectos de software a nivel empresarial.

COMPETENCIAS

Las competencias del curso de ingeniería de software I se centran en:

- Apropiación de una cultura de actualización permanente en herramientas, métodos, metodologías y buenas prácticas en la aplicación de la ingeniería de software.
- Identificar y modelar adecuadamente procesos de software a partir de la ingeniería de requerimientos.
- Apropiación y aplicación de patrones y traducción en un diseño que proporciona una base para la verificación y construcción de aplicaciones.
- Adaptar modelos de diseño de software, herramientas y técnicas en función del contexto de trabajo a realizar y seleccionando adecuadamente los métodos y metodologías.
- Gestionar de manera adecuada proyectos de software, mediante una buena organización en la planificación, optimización de recursos, seguimiento y buenas prácticas.

RESULTADOS DE APRENDIZAJE

- Apropio una cultura de actualización permanente en herramientas, métodos, metodologías y buenas prácticas en la aplicación de la ingeniería de software, en la solución de problemas del entorno.
- Adopto modelos de diseño de software, herramientas y técnicas en función del contexto de trabajo a realizar y seleccionando adecuadamente los métodos y metodologías, aplicando patrones que proporcionen una base para la verificación y construcción de aplicaciones.
- Gestiono proyectos de software, Identificando y modelando adecuadamente procesos de software a partir de la ingeniería de requisitos, mediante una buena organización en la planificación, optimización de recursos, seguimiento y buenas prácticas.

METODOLOGÍA

Esta asignatura será guiada en los espacios presenciales por la complementación conceptual del docente al trabajo de preparación previo, que los estudiantes han realizado sobre la temática particular a tratar en la sesión; por lo tanto, un tema será abordado en cuatro momentos:

1. Preparación, consulta e investigación conceptual por cuenta del estudiante y su grupo de trabajo.
2. Tratamiento conceptual del tema en sesión del gran grupo junto con el docente.
3. Aplicación de talleres individuales y cooperativos a nivel tutorial.
4. Desarrollo de actividades de refuerzo en sesiones autónomas.
5. Se tendrá en cuenta el aprendizaje basado en proyectos

De lo anterior se verifica que en la actividad 1, el estudiante constituirá conflictos conceptuales de baja complejidad, a solucionar en el transcurso de la actividad 2, entre tanto, la actividad 4 generará conflictos cognitivos orientados a la aplicación, a subsanar con la actividad 3.

INVESTIGACIÓN

La asignatura por sus condiciones particulares promueve en el estudiante la investigación aplicativa en el área de la ingeniería de software, se recomienda iniciar con la participación en los semilleros de investigación pertenecientes a los grupos de investigación dinamizados al interior de la Escuela de Ingeniería de Sistemas y Computación.

MEDIOS AUDIOVISUALES

Para las clases asincrónicas es necesario contar con la plataforma virtual Moodle. Para las clases sincrónicas utilizará plataforma de videoconferencia Google Meet.

La asignatura es teórica-practica algunas clases se programarán en salas de informática.

EVALUACIÓN

EVALUACIÓN COLECTIVA

No aplica.

EVALUACIÓN INDIVIDUAL

La evaluación en la asignatura está orientada a determinar el nivel de desarrollo de los procesos lógicos en el estudiante, junto con su capacidad para abstraer problemas y generar soluciones informáticas.

RANGOS DE VALORACIÓN (%)

PRIMER 50%
60% Evaluaciones
40% Tareas, talleres, Quiz, Exposiciones, ente otros.

SEGUNDO 50%
35% Evaluaciones
15% Tareas, talleres, Quiz, Exposiciones, ente otros 50% Proyecto final

CONTENIDOS TEMÁTICOS CENTRALES

1. **Introducción a la ingeniería de software**
1.1. Historia y evolución de la ingeniería de software
1.2. Conceptos generales de la ingeniería de software
1.3. Tipos de Software

2. **Procesos del software**
2.1. Modelos de procesos de software
2.2. Modelos de procesos prescriptivo
2.3. Modelos de procesos especializados
2.4. Evaluación de procesos

2.5. Proceso unificado
2.6. Modelos de proceso de personal y del equipo

3. **Metodologías de desarrollo de software**
3.1. Metodologías tradicionales
3.2. Procesos agiles

4. **Diseño de software**
4.1. Conceptos de diseño
4.2. Arquitectura de software
4.3. Diseño arquitectónico
4.4. Diseño de componentes
4.5. Diseño de interfaz de usuario
4.6. Diseño basado en patrones

5. **Calidad de software**
5.1. Conceptos de calidad
5.2. Técnicas y estrategias de revisión
5.3. Pruebas de aplicaciones
5.4. Métricas

6. **Gestión de proyectos de software**
6.1. Conceptos de administración de proyectos
6.2. Metodologías para gestión de proyectos
6.3. Estimación de proyectos

LECTURAS MÍNIMAS

Artículos seleccionados de Science Direct, SCOPUS, IEEE, ACM y Proquest. Artículos donde se evidencia la aplicación de las diferentes áreas de la Ingeniería de software.

BIBLIOGRAFÍA

Amos Q. Havi, MEAN Web Development, 2016, Packt Publishing Ltd.

Beck Kent, Test Driven Development: By Example 1st Edition, 2012, Addison-Wesley Professional. Bentley Jon, More Programming Pearls: Confessions of a Coder, 1988, Addison-Wesley Professional.

Brandon, Daniel M, Software Engineering for Modern Web Applications: Methodologies and Technologies: Methodologies and Technologies, 2008, IGI Global.

Brooks Frederick P. Jr, The Mythical Man-Month: Essays on Software Engineering, Anniversary Edition (2nd Edition), 1995, Addison-Wesley Professional.

DeMarco Tom et al, Peopleware: Productive Projects and Teams, 1999, Dorset House.

Esposito D, Modern Web Development: Understanding domains, technologies, and user experience, 2016, Pearson Education.

Gasston Peter, The Modern Web: Multi-device Web Development with HTML5, CSS3, and JavaScript, 2013, No

Starch Press.

Humble Jez et al, Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation (Addison-Wesley Signature Series (Fowler)), 2010, Addison-Wesley Professional.

Hunt Andrew; Thomas David, The Pragmatic Programmer: From Journeyman to Master, 1999, Addison-Wesley Professional.

Krug Steve, Don't Make Me Think, Revisited: A Common-Sense Approach to Web Usability (3rd Edition) (Voices That Matter), 2014, New Riders.

Layka Vishal, Learn Java for Web Development: Modern Java Web Development, 2014, Apress. Martin Robert C., Agile Software Development, Principles, Patterns, and Practices, 2002, Pearson.
Martin Robert C., Clean Architecture: A Craftsman's Guide to Software Structure and Design (Robert C. Martin Series), 2017, Prentice Hall.

McConnell Steve, Code Complete: A Practical Handbook of Software Construction, Second Edition, 2004, Microsoft Press.

Nutter Mark, Modern Web Development, Hackernoon, https://hackernoon.com/modern-web-development-bf0b2ef0e22e.

Nygard Michael T., Release It! Design and Deploy Production-Ready Software (Pragmatic Programmers), 2007, Pragmatic Bookshelf.

Pressman Roger S., Software Engineering: A Practitioner's Approach, 2014, McGraw-Hill Education. Prusty Narayan, Modern JavaScript Applications, 2016, Packt Publishing Ltd.
Ries Eric, The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses, 2011, Currency.

Seibel Peter, Coders at Work: Reflections on the Craft of Programming, 2009, Apress.

Sommerville Ian, Software Engineering, 2015, ADDISON WESLEY Publishing Company Incorporated