

Planificación de la Asignatura: Ingeniería de Software II

Fecha: 23/10/2024 13:02

Código: L1336

Carrera: Licenciatura en Bioinformática

Departamento Académico: Informática

Docente a cargo:

Correo del docente a cargo: victor.valotto@uner.edu.ar

Régimen de Dictado: Cuatrimestral 2º Cuatrimestre

Carga Horaria Semanal: 5 horas semanales

Carga Horaria Total: 70 horas

Contenidos Mínimos:

Arquitectura de Software. Diseño y Patrones. Modelado de Software, UML. Pruebas de Software. Tareas del proceso de Pruebas. Niveles y Técnicas. Aseguramiento y gestión de la calidad. Administración y control de Proyectos. Estimación de Proyectos de Software.

Correlativas Regulares para cursar:

Ingeniería de Software I

Correlativas Aprobadas para cursar:

No posee

Correlativas Aprobadas para promocionar o rendir el examen final:

Primer año

Bases de Datos

Ingeniería de Software I

Objetivo General:

Transmitir al alumno la importancia de la especificación en el proceso de desarrollo de software. Estudio de la problemática que plantea el desarrollo de grandes sistemas de software, profundizar en los aspectos de especificación. Introducir al alumno en los formalismos y herramientas de especificación y especialmente en los de especificación del paradigma orientación a objetos.

- Dotar al alumno de conocimientos para que sea capaz de realizar especificaciones en modelos.
- Dotar al alumno de conocimientos sobre procesos de ingeniería del software haciendo hincapié en los métodos iterativos e incrementales. Mostrar el papel de las especificaciones en dichos procesos.
- Que el alumno logre comprender la importancia de las buenas prácticas y que toda construcción y mantenimiento de un producto de software con calidad debe ser soportado por procesos de ingeniería.

Objetivos Particulares:

- Describir las actividades técnicas e ingenieriles que se llevan a cabo en el ciclo de vida de un producto software.
- Describir los problemas, principios, métodos y tecnologías asociadas con la Ingeniería del Software.
- Introducir a la visión de arquitectura de software como base del proceso de desarrollo de los grandes sistemas.
- Comprender los fundamentos del diseño de sistemas software.
- Aplicar de forma práctica los conceptos teóricos sobre el desarrollo estructurado y orientado a objetos.
- Comprender las técnicas y métodos de planificación de proyectos de software.
- Comprender la importancia de las técnicas de estimación de software.
- Comprender el alcance de las actividades de aseguramiento de la calidad y administración de la configuración.
- Realizar un proyecto en grupo, aplicando los principios introducidos en la parte teórica de la asignatura.

Programa Analítico:

- **Módulo 1 - Diseño Arquitectural:** ¿Qué es la arquitectura de software? Decisiones de diseño. Estilos y modelos arquitectónicos. Modelo de Referencia y patrones de arquitectura. Sistemas distribuidos desde el punto de vista de arquitectura. Arquitectura de capas. Concepto de Servicios. Atributos de calidad. Especificación de los escenarios de calidad. Vista de arquitectura.
- **Módulo 2 - Diseño Detallado:** ¿Qué es el modelado? Fundamentos de UML. Elementos y diagramas. Paquetes, diagramas de actividad, diagramas de interacción (secuencia y colaboración), diagramas de clases, diagramas de transición de estados, diagramas de componentes y diagrama de despliegue.
- **Módulo 3 - Mantenimiento del Software:** Concepto de Cambio, impacto del cambio en la organización. Características y problemas. Políticas y administración del cambio. Proceso de Administración de cambios, roles y responsabilidades. Análisis de Impacto. Trazabilidad. Administración de requerimientos, seguimiento y control de cambios.
- **Módulo 4 - Testing de Software:** Conceptos. ¿Qué es testear software? Axioma del Testing. Taxonomías de Testing (Tipos de Testing). El proceso de testing. Planificación de Testing. Diseño de Caso de Testing. Testing de caja blanca y testing de caja negra. Estrategias de Testing: Particiones de Equivalencias y Valores Límites. Diseño de casos de pruebas funcionales. Axiomas del proceso de Testing.
- **Módulo 5 - Gestión de Proyectos:** Definiciones de Proyecto. Procesos de Administración de Proyectos. Organización de proyectos. Roles y Responsabilidades. Alcance. Ciclo de Vida. WBS. Programación de actividades. Diagrama de Gantt: tareas, asignaciones, hitos, holguras, camino crítico. Administración de Riesgos: Identificación del riesgo, análisis del riesgo, exposición del riesgo, mitigación y contingencia. Planificación de la comunicación. Plan de comunicación. Tipos y herramientas de comunicación. Control y seguimiento: Analizar datos del proyecto. Analizar variaciones. Cierre. Indicadores de gestión. Acciones adaptativas. Cierre del proyecto. Lecciones aprendidas.
- **Módulo 6 - Estimación de Proyectos:** ¿Porque estimar?. Métricas de Tamaño: Líneas de Código, Costeo del Proyecto, Puntos de Función, Puntos de Caso de Uso, StoryPoints. Métricas de esfuerzo y costo: COCOMO.
- **Módulo 7 - Gestión de la calidad :** Concepto de Calidad. Los hacedores de la Calidad: Juran, Deming y

Crosby. Total Quality Management. Problemas asociados a la calidad. Planificación de la Calidad, Aseguramiento de la Calidad, Control de la Calidad. Atributos de calidad. Cuantificando los atributos de Calidad. Costos de la Calidad. Roles y Responsabilidades del área de QA. Revisiones. Tipos de Revisiones. Revisiones Formales. Proceso. Administración de la Configuración. Problemas: en el desarrollo de software. Identificación de los elementos de configuración. Conceptos. Elementos de Configuración, Línea Base, Hitos y Entregables. Almacenado de los elementos de configuración. Gestión de Versiones, conceptos. Desarrollo concurrente. Recomendaciones. Gestión de entregas. Procedimientos. Tipos y clasificaciones de las entregas. Gestión de Entornos. Tipos de entornos.

Listado de Actividades de Formación Práctica:

- Ejercicio 1: Definición de Arquitectura:
- Ejercicio 2: Diseño detallado – Colaboración de Objetos.
- Ejercicio 3: Diseño detallado – Clases y Componentes del Sistema.
- Ejercicio 4: Diseño de Casos de Prueba.
- Ejercicio 5: Dimensionamiento del Sistema.

Metodología de Evaluación Durante el cursado:

Se hará un seguimiento de los estudiantes a lo largo del dictado de la materia, a través los trabajos prácticos.

Para la evaluación de los conocimientos adquiridos los estudiantes deberán realizar una exposición sobre un contenido definido por la cátedra, seguido de preguntas relacionadas con dichos contenidos.

Metodología de Evaluación en Exámenes Finales:

Tanto alumnos libres como regulares serán evaluados de acuerdo a un examen escrito.

Condiciones de Regularidad :

Para la evaluación de los conocimientos adquiridos los estudiantes deberán realizar una exposición sobre un contenido definido por la cátedra, seguido de preguntas relacionadas con dichos contenidos.

Habrà una instancia de recuperación en caso que la evaluación de esta presentación sea inferior a 6(seis) aprobado. Si la evaluación es de 8(ocho) o superior la condición del estudiante es promocionado. En cualquier otro caso, es decir, entre 6 inclusive o menor a 8 la condición de regular.

Bibliografía Principal:

Ingeniería del Software, un enfoque práctico. Roger Pressman. MCGRAW-HILL, 2005, Edición Número 6.

Ingeniería del Software, Ian Sommerville. PEARSON ADDISON WESLEY, 2005, Séptima edición.

Bibliografía Complementaria:

The Project Manager's Guide to Software Engineering Best Practices, Mark J. Christensen, Richard H. Thayer, IEEE Press, 2001.

The Software Engineering Book of Knowledge, IEEE, 2004.

Software Engineering, Editado por Merlin Dorfman y Richard H. Thayer, IEEE Press, 1997..

Software Engineering Project Management, Editado por Richard H. Thayer, IEEE Press, 2000.
Segunda Edición.

Software Architecture in Practice, Ediciones 2, Len Bass, Paul Clements, Rick Kazman, Addison -Wesley, 2003.

Documenting Software Architecture, Paul Clemens, Felix Bachman, Addison –Wesley, 2002.

A Guide to Software Configuration Management. Alexis Leon, Artech House, 2000.

Configuration Management Principles and Practice. Jonassen Hass, Anne Mette , Addison -Wesley, 2002.

Software Configuration Management Patterns: Effective Teamwork, Practical Integration. Stephen Berzuck, Brad Appleton, , Addison -Wesley, 2002

Practical Guide to Software Quality Management, Second Edition - John W. Horch, 2003.

Metrics and Models in Software Quality Engineering, Second Edition, Stephen H.Kan, Addison Wesley , 2002

Patrones de Diseño. Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. Addison-Wesley, 2005.

Object Oriented Software Construction. Bertrand Meyer, Prentice Hall, 1997

UML y patrones.: 2da Edicion. Craig Larman. Prentice Hall, 2002.

El lenguaje unificado de modelado. Grady Booch, James Rumbaugh, Iva Jacobson, Prentice Hall , 1999.