

1.- DATOS DE LA ASIGNATURA

Nombre de la asignatura: Desarrollo de proyectos de software
Carrera: Ingeniería en Sistemas Computacionales
Clave de la asignatura: SCM - 0406
Horas teoría-horas práctica-créditos 3-2-8

2.- HISTORIA DEL PROGRAMA

Lugar y fecha de elaboración o revisión	Participantes	Observaciones (cambios y justificación)
Instituto Tecnológico de Toluca 18 – 22 agosto 2003.	Representantes de la academia de sistemas y computación de los Institutos Tecnológicos.	Reunión nacional de evaluación curricular de la carrera de Ingeniería en Sistemas Computacionales.
Institutos Tecnológicos de: Cd. Guzmán, Iguala 23 agosto al 7 noviembre del 2003	Academia de sistemas y computación.	Análisis y enriquecimiento de las propuestas de los programas diseñados en la reunión nacional de evaluación
Instituto Tecnológico de León 1 – 5 marzo 2004	Comité de consolidación de la carrera de Ingeniería en Sistemas Computacionales.	Definición de los programas de estudio de la carrera de Ingeniería en Sistemas Computacionales.

3.- UBICACIÓN DE LA ASIGNATURA

a). Relación con otras asignaturas del plan de estudio

Anteriores		Posteriores	
Asignaturas	Temas	Asignaturas	Temas
Planificación y Modelado.	Todos ya que es una materia integradora para el desarrollo de cualquier tipo de proyecto de software.	Residencia Profesional.	Base para el Desarrollo de su Proyecto de Residencia Profesional
Programación de Redes.			
Tópicos Selectos de programación Interfaces.			
Desarrollo sustentable.			
Ética			

b). Aportación de la asignatura al perfil del egresado

Desarrolla aplicaciones de software de cualquier dominio.

4.- OBJETIVO(S) GENERAL(ES) DEL CURSO

El estudiante diseñará y construirá un proyecto de software conforme a los requerimientos establecidos en el dominio del proyecto de software.

5.- TEMARIO

Unidad	Temas	Subtemas
1	Conceptos Introdutorios.	1.1 La arquitectura de 4+1 vistas. 1.2 Desarrollo orientado a objetos. 1.3 Diagramación.
2	Diseño orientado a objetos.	2.1 Diseño del sistema en base a procesos. 2.1.1 Actividades y casos de uso. 2.1.2 Interfaces de usuario. 2.2 Diseño de la lógica. 2.2.1 Clases y Objetos. 2.2.2 Interacción. 2.2.3 Estados y Transiciones.
3	Construcción.	3.1 Despliegue de componentes y arquitectónico. 3.2 Técnicas de desarrollo de las arquitecturas de referencia en diferentes dominios. 3.2.1 Los modelos de componentes. 3.2.2 Arquitectura de referencia para sistemas de tiempo real fuente de alimentación. 3.2.3 Arquitectura de referencia para sistemas móviles con conexión a Internet. 3.2.4 Arquitectura de referencia para sistemas de información. 3.2.5 Arquitectura de referencia para ambientes virtuales de aprendizaje. 3.2.6 Arquitecturas de referencia para líneas de productos.
4	Pruebas de software.	4.1 Definiciones. 4.1.1 Prueba, caso de prueba, defecto, falla, error, verificación, validación. 4.1.2 Relación entre defecto-falla-error. 4.1.3 Pruebas estructurales, funcionales y aleatorias.

5.- TEMARIO (Continuación)

5	Implantación y mantenimiento.	<ul style="list-style-type: none">4.1.4 Documentación del diseño de las pruebas.4.2 Proceso de pruebas.<ul style="list-style-type: none">4.2.1 Generar un plan de pruebas.4.2.2 Diseñar pruebas específicas.4.2.3 Tomar configuración del software a probar.4.2.4 Configurar las pruebas.4.2.5 Evaluar resultados.<ul style="list-style-type: none">4.2.5.1 Depuración.4.2.5.2 Análisis de errores.4.3 Técnicas de diseño de casos de pruebas.4.4 Enfoque práctico recomendado para el diseño de casos.4.5 Estrategias de aplicación de las pruebas.<ul style="list-style-type: none">4.5.1 De unidad.4.5.2 De integración.4.5.3 Del sistema.4.5.4 De aceptación.5.1 Implantación e Integración de casos de uso y componentes de software.5.2 Mantenimiento del software.
---	-------------------------------	---

6.- APRENDIZAJES REQUERIDOS

- Aplicar una técnica de adquisición de información (entrevistas, cuestionarios, etc.).
- Integrar equipos de desarrollo.
- Discriminar los requerimientos de proyectos de software.
- Aplicar los requerimientos de usuario para diseñar casos de uso e interfaces correspondientes de un proyecto de software.

7.- SUGERENCIAS DIDÁCTICAS

- Contar con un sitio Web.
- Buscar oportunidades para el diseño, implantación y pruebas de un sistema computacional en las diferentes organizaciones de la localidad.
- Aplicar una técnica de adquisición de información (entrevistas, cuestionarios, sondeo, entre otros).
- Exponer en el aula el proyecto realizado.
- Integrar equipos de desarrollo motivando el aprendizaje en equipo.
- Dar un uso didáctico a medios audiovisuales, emplear dinámicas grupales (lluvia de ideas, mesa redonda, paneles, foros, conferencias, debates, entre otros), realizar prácticas.
- Desarrollo de un proyecto dosificado durante el semestre.

8.- SUGERENCIAS DE EVALUACIÓN

- Evaluación diagnóstica (valoración de conocimientos previos).
- Dar seguimiento al desempeño en el desarrollo del proyecto. (dominio de los conceptos, capacidad de comunicación interpersonal, aplicación de los conocimientos en problemas reales, transferencia del conocimiento).
- Evaluación de la presentación del proyecto. (Informe, presentación y defensa de la congruencia del proyecto).
- Dar valor a la participación (mesas redondas y de debate).
- Actividades de auto evaluación.
- Exámenes departamentales.

9.- UNIDADES DE APRENDIZAJE

UNIDAD 1.- Conceptos introductorios.

Objetivo Educativo	Actividades de Aprendizaje	Fuentes de Información
El estudiante comprenderá el enfoque de los diferentes actores involucrados en un proyecto de software y aplicación de cada uno de los diagramas del paradigma orientado a objetos.	<p>1.1 Plantear preguntas relativas a los roles de:</p> <ul style="list-style-type: none">• Usuario.• Analistas.• Diseñadores.• Desarrolladores.• Probador.• Integradores. <p>1.2 Realizar un ejercicio que muestre la aplicación del concepto de desarrollo orientado a objetos.</p> <p>1.3 Realizar un ejercicio que muestre la aplicación del concepto de cada uno de los diagramas.</p>	7,6,1,2,4,12

UNIDAD 2.- Diseño orientado a objetos.

Objetivo Educativo	Actividades de Aprendizaje	Fuentes de Información
Construirá los diagramas que muestren el comportamiento del sistema acorde a los requerimientos del usuario.	<p>2.1 Aplicar el Lenguaje Unificado Modelado (UML) específicamente Diagramas de Secuencia, Colaboración y Estado, Clases y Objetos para realizar el Diseño del proyecto de software. Discutir y exponer por equipo, los Diagramas resultantes.</p> <p>2.2 Igual a 2.1</p>	1,2,5,6,7,8,12

UNIDAD 3.- Construcción.

Objetivo Educativo	Actividades de Aprendizaje	Fuentes de Información
Construirá los diagramas correspondientes a la implementación y desarrollará la programación del sistema acorde a la arquitectura de referencia del dominio de su proyecto.	3.1 Aplicar el Lenguaje Unificado Modelado (UML) específicamente Diagramas de componentes y despliegue para mostrar la implementación del proyecto de software. Discutir y exponer por equipo, los Diagramas resultantes. 3.2 Realizar la programación del proyecto de software mediante el paradigma Orientado a Objetos.	4, 5, 6, 7

UNIDAD 4.- Pruebas de software.

Objetivo Educativo	Actividades de Aprendizaje	Fuentes de Información
Implementará una estrategia para realizar pruebas a su proyecto de software.	4.1 Discutir y exponer por equipo, las Técnicas de Prueba usadas y sus resultados. 4.5 Discutir y exponer por equipo, las estrategias de aplicación de las pruebas.	4,5,6,8,12

UNIDAD 5.- Implantación y mantenimiento.

Objetivo Educativo	Actividades de Aprendizaje	Fuentes de Información
Conocerá técnicas para la implantación y mantenimiento del software.	5.1 Realizar una síntesis sobre las técnicas para la implantación y el mantenimiento del software. Discutir, en equipo, las características que diferencian a cada uno de las técnicas. 5.2 Realizar una síntesis sobre el mantenimiento del software, discutir, en equipo, las características del mantenimiento de software.	4,10,11,12

10. FUENTES DE INFORMACIÓN

1. Fowler, Martin, (1999).
UML Gota a Gota
Ed. Addison Wesley.
2. Larman, Craig (1999).
UML y patrones.
Ed. Pearson.
3. Bruegge Bernd (2001).
Ingeniería de Software Orientada a Objetos.
Ed. Prentice Hall.
4. Braude, Eric (2003).
Ingeniería de Software Una perspectiva Orientada a Objetos.
Ed. Alfaomega.
5. Meyer, Bertrand (1999).
Construcción de Software Orientada a Objetos.
Ed. Prentice Hall.
6. Oestereich Bernd (1999).
Developing Software with UML, Object-Oriented Analysis and Desing in
Practice.
Ed. Addison Wesley.
7. Reed R.Paul (2001).
Developing Applications with Visual Basic and UML.
Ed. Addison Wesley.
8. Jacobson,Ivar. (2000).
El Proceso unificado de desarrollo de Software.
Ed. Addison Wesley.
9. Humphrey, Watts S. (2000).
Introducción al Proceso Software Personal.
Ed. Addison Wesley.
10. Sommerville, Ian (2001).
Ingeniería de Software.
Ed. Prentice Hall.
11. Pressman Roger S (2001).
Ingeniería del Software, 5/E.
Ed. Mc.Gaw-Hill.

12. Laudon & Laudon 8/E (2003).
Management Information Systems.
Ed. Prentice-Hall.

11.- PRACTICAS

Unidad Práctica

- 1 Desarrollo de un proyecto dosificado durante el semestre, involucrando todas las unidades de aprendizaje, donde se aplique los diagramas del Lenguaje Unificado de Modelado