

Universidad Católica San Pablo
Facultad de Ingeniería y Computación
Escuela Profesional de
Ciencia de la Computación
SILABO



CS390. Ingeniería de Software II (Obligatorio)

2010-1

1. DATOS GENERALES

1.1 CARRERA PROFESIONAL	:	Ciencia de la Computación
1.2 ASIGNATURA	:	CS390. Ingeniería de Software II
1.3 SEMESTRE ACADÉMICO	:	6 ^{to} Semestre.
1.4 PREREQUISITO(S)	:	CS290T. Ingeniería de Software I. (5 ^{to} Sem)
1.5 CARÁCTER	:	Obligatorio
1.6 HORAS	:	2 HT; 2 HP; 2 HL;
1.7 CRÉDITOS	:	4

2. DOCENTE

3. FUNDAMENTACIÓN DEL CURSO

Los tópicos de este curso extienden las ideas del diseño y desarrollo de software desde la secuencia de introducción a la programación para abarcar los problemas encontrados en proyectos de gran escala. Es una visión más amplia y completa de la Ingeniería de Software apreciada desde un punto de vista de Proyectos.

4. SUMILLA

1. SE/Desarrollo de Sistemas Especializados.2. SE/Herramientas y Entornos de Software.3. SE/Validación y verificación de software.4. SE/Evolución del Software.5. SE/Administración de Proyectos de Software.6. SE/Evaluación de riesgos.

5. OBJETIVO GENERAL

- Capacitar a los alumnos para formar parte y definir equipos de desarrollo de software que afronten problemas de envergadura real.
- Familiarizar a los alumnos con el proceso de administración de un proyecto de software de tal manera que sea capaz de crear, mejorar y utilizar herramientas y métricas que le permitan realizar la estimación y seguimiento de un proyecto de software.
- Crear , evaluar e implementar un plan de prueba para segmentos de código de tamaño medio , Distinguir entre los diferentes tipos de pruebas , sentar las bases para crear, mejorar los procedimientos de prueba y las herramientas utilizadas con ese propósito.
- Seleccionar con justificación un apropiado conjunto de herramientas para soportar el desarrollo de un rango de productos de software.
- Crear , mejorar y utilizar los patrones existentes para el mantenimiento de software . Dar a conocer las características y patrones de diseño para la reutilización de software.
- Identificar y discutir diferentes sistemas especializados , crear , mejorar y utilizar los patrones especializados para el diseño , implementación , mantenimiento y prueba de sistemas especializados

6. CONTRIBUCIÓN A LA FORMACIÓN PROFESIONAL Y FORMACIÓN GENERAL

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- b) Analizar problemas e identificar y definir los requerimientos computacionales apropiados para su solución. [Nivel Bloom: 4]
- c) Diseñar, implementar y evaluar un sistema, proceso, componente o programa computacional para alcanzar las necesidades deseadas. [Nivel Bloom: 4]
- d) Trabajar efectivamente en equipos para cumplir con un objetivo común. [Nivel Bloom: 3]
- f) Comunicarse efectivamente con audiencias diversas. [Nivel Bloom: 3]
- i) Utilizar técnicas y herramientas actuales necesarias para la práctica de la computación. [Nivel Bloom: 3]
- j) Aplicar la base matemática, principios de algoritmos y la teoría de la Ciencia de la Computación en el modelamiento y diseño de sistemas computacionales de tal manera que demuestre comprensión de los puntos de equilibrio involucrados en la opción escogida. [Nivel Bloom: 3]
- k) Aplicar los principios de desarrollo y diseño en la construcción de sistemas de software de complejidad variable. [Nivel Bloom: 3]

7. CONTENIDOS

UNIDAD 1: SE/Desarrollo de Sistemas Especializados.(12 horas)

Nivel Bloom: 4

OBJETIVO GENERAL

CONTENIDO

- Identificar y discutir diferentes sistemas especializados.
- Discutir el ciclo de vida y tópicos sobre el proceso de software en el ámbito de sistemas diseñados para un contexto especializado incluyendo sistemas que podrían tener que operar en un modo de operación degradado.
- Seleccionar, con la justificación apropiada, métodos que darán como resultado el desarrollo eficiente y efectivo y el mantenimiento de sistemas de software especializado.
- Dado un contexto específico y un conjunto de tópicos profesionales relacionados, discutir como, un ingeniero de software envuelto en el desarrollo de sistemas especializados, debe de responder a estos tópicos.
- Sintetizar los temas técnicos centrales asociados con la implementación del crecimiento de sistemas especializados..

- Sistemas en tiempo real.
- Sistemas cliente-servidor.
- Sistemas distribuidos.
- Sistemas paralelos.
- Sistemas basados en web.
- Sistemas de alta integridad.

Lecturas: [Pressman, 2004], [Blum, 1992], [Schach, 2004], [Wang and King, 2000], [Keyes, 2004], [Windle and Abreo, 2002], [Priest and Sanchez, 2001], [Schach, 2004], [Montangero, 1996], [Ambriola, 2001], [Conradi, 2000], [Oquendo, 2003]

UNIDAD 2: SE/Herramientas y Entornos de Software.(12 horas)	
Nivel Bloom: 3	
OBJETIVO GENERAL	CONTENIDO
<ul style="list-style-type: none"> ▪ Seleccionar con justificación un apropiado conjunto de herramientas para soportar el desarrollo de un rango de productos de software. ▪ Analizar y evaluar un conjunto de herramientas en una área dada del desarrollo de software (ej: administración, modelamiento o pruebas). ▪ Demostrar la capacidad para usar un rango de herramientas de software en soporte del desarrollo de un producto de software de tamaño medio. 	<ul style="list-style-type: none"> ▪ Entornos de programación. ▪ Análisis de requerimientos y herramientas de modelamiento de diseño. ▪ Herramientas de pruebas incluyendo herramientas de análisis estático y dinámico. ▪ Herramientas de administración de configuración. ▪ Manejo de la configuración y herramientas de control de versión. ▪ Mecanismos de integración de herramientas.
Lecturas: [Pressman, 2004], [Blum, 1992], [Schach, 2004], [Wang and King, 2000], [Keyes, 2004], [Windle and Abreo, 2002], [Priest and Sanchez, 2001], [Schach, 2004], [Montangero, 1996], [Ambriola, 2001], [Conradi, 2000], [Oquendo, 2003]	

UNIDAD 3: SE/Validación y verificación de software.(12 horas)	
Nivel Bloom: 4	
OBJETIVO GENERAL	CONTENIDO
<ul style="list-style-type: none"> ▪ Distinguir entre validación de programas y verificación. ▪ Describir el rol que las herramientas pueden jugar en la validación de software. ▪ Distinguir entre los diferentes tipos y niveles de pruebas (unidad, integración, sistemas y aceptación) para productos de software de tamaño medio y el material relacionado. ▪ Crear, evaluar e implementar un plan de prueba para segmentos de código de tamaño medio. ▪ Encargarse, como parte de una actividad de equipo, de una inspección de un segmento de código de tamaño medio. ▪ Discutir los temas concernientes a la prueba de software orientado a objetos.. 	<ul style="list-style-type: none"> ▪ Distinción entre verificación y validación. ▪ Abordajes estáticos y dinámicos. ▪ Planeamiento de la validación y documentación para la validación. ▪ Diferentes tipos de tests, interfase humano-computador, usabilidad, confiabilidad, seguridad, conformidad con la especificación. ▪ Fundamentos del <i>Testing</i> incluyendo la creación de planes de prueba y la generación de casos de prueba. ▪ Técnicas de prueba de caja blanca y caja negra. ▪ Semilla por defecto. ▪ Unidad, integración, validación y sistemas de prueba. ▪ Prueba orientado a objetos, pruebas de sistema. ▪ Medidas de procesos, diseño, programa. ▪ Verificación y validación de partes que no son componentes (documentación, archivos de ayuda, material de entrenamiento). ▪ Defecto de historial (<i>fault logging</i>), defecto de rastreo y soporte técnico para esas actividades. ▪ Test de regresión. ▪ Inspecciones, revisiones, auditorías.
Lecturas: [Pressman, 2004], [Blum, 1992], [Schach, 2004], [Wang and King, 2000], [Keyes, 2004], [Windle and Abreo, 2002], [Priest and Sanchez, 2001], [Schach, 2004], [Montangero, 1996], [Ambriola, 2001], [Conradi, 2000], [Oquendo, 2003]	

UNIDAD 4: SE/Evolución del Software.(12 horas)	
Nivel Bloom: 3	
OBJETIVO GENERAL	CONTENIDO
<ul style="list-style-type: none"> ▪ Identificar los temas principales asociados con la evolución del software y explicar su impacto sobre el ciclo de vida del software. ▪ Discutir los desafíos de mantener sistemas heredados y la necesidad de la ingeniería reversa. ▪ Delinear el proceso de pruebas de regresión y su rol en la administración del lanzamiento. ▪ Estimar el impacto de un cambio de requerimiento para un producto existente de tamaño medio. ▪ Desarrollar un plan para hacer reingeniería a un producto de tamaño medio como respuesta a un cambio de requerimientos. ▪ Discutir las ventajas y desventajas del reuso de software. ▪ Explotar las oportunidades para reusar software en un contexto dado. ▪ Identificar debilidades en un simple diseño dado y resaltar como las mismas pueden ser removidas a través de la reconstrucción (<i>refactoring</i>). 	<ul style="list-style-type: none"> ▪ Mantenimiento de software. ▪ Características del software mantenible. ▪ Reingeniería. ▪ Sistemas heredados. ▪ Reuso de software.
Lecturas: [Pressman, 2004], [Blum, 1992], [Schach, 2004], [Wang and King, 2000], [Keyes, 2004], [Windle and Abreo, 2002], [Priest and Sanchez, 2001], [Schach, 2004], [Montangero, 1996], [Ambriola, 2001], [Conradi, 2000], [Oquendo, 2003]	

UNIDAD 5: SE/Administración de Proyectos de Software.(12 horas)	
Nivel Bloom: 3	
OBJETIVO GENERAL	CONTENIDO
<ul style="list-style-type: none"> ▪ Demostrar, involucrándose en un equipo de proyecto, los elementos centrales de la construcción y administración de un equipo. ▪ Preparar un plan para un proyecto de software que incluye estimación de tamaño y esfuerzo, asignación de tiempos y tareas, asignación de recursos, control de configuración, administración de cambios, identificación y administración de los riesgos del proyecto. ▪ Indicar un abordaje para tratar riesgos que ayudará a entregar el software a tiempo. ▪ Comparar y contrastar los diferentes métodos y técnicas usados para asegurar la calidad de un producto de software. 	<ul style="list-style-type: none"> ▪ Administración de equipos. a) Procesos de equipo. b) Organización de equipos y toma de decisiones. c) Roles y responsabilidades en un equipo de software. d) Identificación y asignación de roles. e) Seguimiento del proyecto. f) Resolución de problemas de equipo. ▪ Asignación de tiempos y tareas al proyecto. ▪ Medición de software y técnicas de estimación. ▪ Análisis de riesgos. a) El asunto de seguridad. b) Sistemas de alta integridad, sistemas de seguridad críticos. c) El rol del riesgo en el ciclo de vida. ▪ Aseguramiento de la calidad de software. a) El rol de las mediciones. ▪ Administración de la configuración y versiones de software. Manejo de la versión final (<i>release</i>). ▪ Herramientas de administración de proyectos. ▪ Modelos de proceso de software y medidas de proceso.
Lecturas: [Pressman, 2004], [Blum, 1992], [Schach, 2004], [Wang and King, 2000], [Keyes, 2004], [Windle and Abreo, 2002], [Priest and Sanchez, 2001], [Schach, 2004], [Montangero, 1996], [Ambriola, 2001], [Conradi, 2000], [Oquendo, 2003]	

UNIDAD 6: SE/Evaluación de riesgos.(6 horas)	
Nivel Bloom: 3	
OBJETIVO GENERAL	CONTENIDO
<ul style="list-style-type: none"> ▪ Definir los conceptos de peligros y riesgos. ▪ Reconocer riesgos comunes de seguridad en al menos dos sistemas operativos. ▪ Describir las categorías de amenazas a sistemas de redes de computadores. ▪ Mostrar un abordaje sistemático para la tarea de identificar peligros y riesgos en una situación particular. ▪ Aplicar los principios básicos de manejo de riesgos en una variedad de escenarios incluyendo alguna situación relacionada con seguridad. 	<ul style="list-style-type: none"> ▪ Definición de términos: en seguridad, vulnerabilidad, amenazas, brechas de seguridad, peligros. ▪ El concepto de riesgo, identificación de peligros y riesgos. ▪ Análisis de riesgo incluyendo evaluación. ▪ Necesidad de un abordaje completo de sistema que incluya peligros asociados con herramientas. ▪ Riesgo y las tecnologías inmaduras. ▪ Análisis de costo beneficio. ▪ Principios del manejo de riesgos.
Lecturas: [Pressman, 2004], [Blum, 1992], [Schach, 2004], [Wang and King, 2000], [Keyes, 2004], [Windle and Abreo, 2002], [Priest and Sanchez, 2001], [Schach, 2004], [Montangero, 1996], [Ambriola, 2001], [Conradi, 2000], [Oquendo, 2003]	

8. METODOLOGÍA

El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.

El profesor del curso presentará demostraciones para fundamentar clases teóricas.

El profesor y los alumnos realizarán prácticas

Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.

9. EVALUACIONES

Evaluación Permanente 1 : 20 %

Examen Parcial : 30 %

Evaluación Permanente 2 : 20 %

Examen Final : 30 %

Referencias

[Ambriola, 2001] Ambriola, V. (2001). *Software Process Technology*. Springer.

[Blum, 1992] Blum, B. I. (1992). *Software Engineering: A Holistic View*. Oxford University Press US, 7th edition.

[Conradi, 2000] Conradi, R. (2000). *Software Process Technology*. Springer.

[Keyes, 2004] Keyes, J. (2004). *Software Configuration Management*. CRC Press.

[Montangero, 1996] Montangero, C. (1996). *Software Process Technology*. Springer.

[Oquendo, 2003] Oquendo, F. (2003). *Software Process Technology*. Springer.

[Pressman, 2004] Pressman, R. S. (2004). *Software Engineering: A Practitioner's Approach*. McGraw-Hill, 6th edition.

[Priest and Sanchez, 2001] Priest, J. W. and Sanchez, J. M. (2001). *Product Development and Design for Manufacturing*. Marcel Dekker.

[Schach, 2004] Schach, S. R. (2004). *Object-Oriented and Classical Software Engineering*. McGraw-Hill.

[Wang and King, 2000] Wang, Y. and King, G. (2000). *Software Engineering Processes: Principles and Applications*. CRC Press.

[Windle and Abreo, 2002] Windle, D. R. and Abreo, L. R. (2002). *Software Requirements Using the Unified Process*. Prentice Hall.