Ministerio de Cultura y Educación
Universidad Nacional de San Luis
Facultad de Ciencias Físico Matemáticas y Naturales
Departamento: Informatica
Area: Area IV: Pr. y Met. de Des. del Soft.

(Programa del año 2013)
(Programa en trámite de aprobación)
(Presentado el 03/12/2013 11:51:04)

I - Oferta Académica

Materia	Carrera	Plan	Año	Período
INGENIERIA DE SOFTWARE I	PROF.CS.COMPUT.	06/08	2013	1° cuatrimestre
INGENIERIA DEL SOFTWARE	PROF.CS.COMPUT.	06/09	2013	1° cuatrimestre
INGENIERIA DE SOFTWARE I	LIC.CS.COMP.	006/05	2013	1° cuatrimestre

II - Equipo Docente

Docente	Función	Cargo	Dedicación
FUNES, ANA MARIA	Prof. Responsable	P.Adj Exc	40 Hs
DASSO, ARISTIDES JUAN	Prof. Colaborador	P.Tit. Exc	40 Hs
ABDELAHAD, CORINA NATALIA	Auxiliar de Práctico	A.1ra Semi	20 Hs

III - Características del Curso

Credito Horario Semanal				
Teórico/Práctico	**Teóricas**	**Prácticas de Aula**	**Práct. de lab/ camp/ Resid/ PIP, etc.**	**Total**
3 Hs	Hs	3 Hs	2 Hs	8 Hs

Tipificación	Periodo
B - Teoria con prácticas de aula y laboratorio	1º Cuatrimestre

Duración			
Desde	**Hasta**	**Cantidad de Semanas**	**Cantidad de Horas**
14/03/2013	19/06/2013	15	120

IV - Fundamentación

Se introduce al alumno en el desarrollo de software como una actividad ingenieril y al trabajo en equipo en el contexto de la Teoría General de Sistemas. Se pretende que el alumno aprenda los fundamentos básicos de cada una de las etapas que comprende el proceso de desarrollo de software así como las distintas técnicas y metodologías aplicables, conocimientos de los principios de diseño así como de las principales técnicas de validación y verificación del software.

V - Objetivos / Resultados de Aprendizaje

Al finalizar esta asignatura se espera:

Preparar al alumno para su desempeño en la Industria del Software con un enfoque Sistémico.

Que el alumno asimile los conceptos de procesos de desarrollo de software, desde su elicitación, análisis, diseño hasta su verificación y validación.

Que el alumno tome conocimiento de conceptos básicos de ingeniería de software desde un punto de vista de la automatización de procesos de desarrollo de software. Para cubrir dichos objetivos se integrarán conceptos, modelos y

métodos en un proyecto integrador.

VI - Contenidos

Unidad I:
El agotamiento del enfoque Analítico-Reduccionista en diversas ramas de la Ciencia: La Teoría General de los Sistemas. Concepto de Sistema. Características de los Sistemas. Elementos constitutivos de un Sistema. Clasificación de los Sistemas. Sistemas de Información. Elementos constitutivos de un Sistema de Información. Criterios de clasificación de los Sistemas de Información. Ejemplos de distintos tipos de Sistemas de Información.

Unidad II:
Ingeniería de Software. Principios. Modelos. Software. Conceptos. Características del software. Procesos, metodologías y herramientas. Ciclos de vida. Modelos de procesos de producción de software. Evolución de las metodologías de desarrollo de sistemas de software. Ingeniería de Requerimientos. Ingeniería de la Información.

Unidad III:
Métodos Formales. Conceptos básicos. Deficiencias de los enfoques menos formales. Las matemáticas en el desarrollo de software. Notación matemática para la especificación formal. Lenguajes de especificación formal. El método RAISE y el lenguaje de especificación formal RSL.

Unidad IV:
Modelos en UML. Introducción. Modelos. Importancia de los modelos. Modelos estáticos. Clases: atributos, operaciones y responsabilidades. Relaciones: dependencia, generalización y asociación. Modelos dinámicos. Diagrama de Interacción: Secuencia y Comunicación. Modelo de Casos de Usos.

Unidad V:
Proceso de Desarrollo. Introducción. Análisis, diseño, implementación, pruebas y mantenimiento de software. Modelo de Casos de Usos. Ingeniería de Requerimientos. Análisis y Diseño. Implementación. Metodologías Ágiles. Conceptos. Principios. Manifiesto. Ventajas y desventajas de las metodologías ágiles. Estudio de utilización concreta de las metodologías ágiles en la industria del software: Extreme Programming.

Unidad VI:
Calidad del Software. Gestión de Configuración del Software. Conceptos. Control. Garantía. Costos. Aseguramiento de la calidad del software. Validación y Verificación. Pruebas de errores del software. Estrategias de prueba. Técnicas de prueba: prueba estructural y prueba funcional. Prueba Unitaria. Uso de JUnit. Medidas de fiabilidad y disponibilidad. Líneas base, elementos y objetos en la configuración del software. Versiones.

Unidad VII:
Privacidad, integridad y seguridad en Sistemas de Información. Privacidad en Internet. Integridad: Conceptos y herramientas básicas para preservarla. Integridad en el caso de aplicaciones distribuidas.

VII - Plan de Trabajos Prácticos

Practico 1: La Ingeniería de Software y los modelos de ciclo de vida del software.

Práctico 2: Especificaciones formales de software.

Práctico 3: Modelos en UML.

Práctico 4: Prueba estructural.

Práctico 5: Prueba funcional.

Práctico 6: Prueba unitaria con JUnit.

Laboratorio 1: Especificación formal de un caso de estudio haciendo uso de un lenguaje de especificación formal con herramientas automatizadas.

Laboratorio 2: Utilización de herramientas para la generación de pruebas de software.

Laboratorio integrador: Se solicita la construcción de un proyecto real con un cliente a ser seleccionado para llevar adelante un caso de estudio donde se cubra todos los aspectos de la Ingeniería de Software.

VIII - Regimen de Aprobación

Condiciones para regularizar la asignatura:
- Haber asistido al menos al 80% de las clases de la asignatura.
- Haber aprobado los prácticos de máquina solicitados con toda su documentación entregada en tiempo y forma, más dos exámenes parciales escritos o sus respectivas recuperaciones, con notas mayores o iguales a seis.
- En caso de solo regularizar, el alumno deberá rendir un examen final para aprobar la asignatura.

Condiciones para promocionar la asignatura:
- Haber asistido al menos al 80% de las clases de la asignatura.
- Haber aprobado los prácticos de máquina solictados con toda su documentación entregada en tiempo y forma, más dos exámenes parciales escritos o sus respectivas recuperaciones, con notas mayores o iguales a siete.
- Haber aprobado una evaluación final integradora escrita con nota mayor o igual a siete.
- En caso de promocionar la materia el alumno aprobará la misma con una nota que surgirá del promedio entre los parciales y la evaluación final integradora.

Exámenes libres: Por las características propias de los proyectos de prácticos de máquina a desarrollarse durante todo el cuatrimestre, no se aceptan alumnos libres.

Alumnos que trabajan: Se otorga una única recuperación extra para aquellos alumnos que hayan acreditado su condición en Sección Alumnos de la FCFMyN. Dicha recuperación extra puede ser usada en uno de los parciales.

IX - Bibliografía Básica

[1] James O´Brien, George Marakas, Introduction to Information Systems, Mc Graw Hill, 15th edition. 2007.
[2] Roger S. Pressman, Software Engineering: A Practitioner's Approach (Sixth Edition), McGraw-Hill, 2006.
[3] Ian Sommerville, Software Engineering (Fifth Edition), Addison-Wesley, 1996.
[4] Ghezzi, Carlo y otros "Fundamentals of Software Engineering", Prentice Hall, 1991.
[5] The Unified Modeling Language User Guide, 2nd Edition. Booch, Rumbaugh, Jacobson. Addison-Wesley, 2005.
[6] The Unified Modeling Language Reference Manual, 2nd Edition. Booch, Rumbaugh, Jacobson. Addison-Wesley, 2005.
[7] Martín Fowler, The New Methodology, http://www.martinfowler.com/articles/newMethodology.html
[8] Kent Beck, “Extreme Programming Explained”, 1ra edición, 1999.
[9] Scott Ambler, “Agile Modeling: Effective practices for Extreme Programming and the Unified Process”, John Wiley & Sons, 2002.
[10] The RAISE Specification Language, Chris George et al., Prentice Hall, 1992.
[11] Paul Jorgensen, "Software Testing- A Craftsman's approach", CRC Press, 1995.
[12] Bolaños, Javier. "Pruebas de Sofware y JUnit".Pearson Education, 2008.
[13] Apuntes de la cátedra.

X - Bibliografia Complementaria

[1] Stephen Schach. "Ingeniería de Sofware Clásica y Orientada a Objetos, 6ta ed.", Mc Graw Hill, 2005.
[2] The RAISE Development Method, Chris George et al., Prentice Hall, 1995.
[3] Beck, Kent. Test-driven development by example. Pearson Education, 2003.
[4] Manifiesto para el Desarrollo de Software Ágil, http://www.agilemanifesto.org
[5] Scott Ambler, “Agile Modeling and the Unified Process”, http://www.agilemodeling.com/essays/agileModelingRUP.htm, 2002.
[6] Pekka Abrahamsson, Outi Salo, Jussi Ronkainen & Juhani Warsta, “Agile Software Development Methods: Review and

Analysis”, VTT, 2002.
[7] "Extreme Programming: A gentle introduction", http://www.extremeprogramming.org/
[8] Página Web del Instituto de Ingeniería de Software (CMU). http://www.cmu.edu,
[9] Martin, James, "Information Engineering", Prentice Hall, 1991.

XI - Resumen de Objetivos

Asimilar los conceptos de procesos de desarrollo de software, desde su especificación, análisis, diseño hasta su verificación y validación., incorporando conceptos de privacidad y calidad del software.

XII - Resumen del Programa

Teoría General de los Sistemas. Sistemas de Información. El Producto de Software integrado en un Sistema de Información. El proceso de software: Ciclos de vida, herramientas. Ingeniería de requerimientos, introducción a los métodos formales. Análisis, diseño, implementación, verificación, validación y mantenimiento de software. Conceptos de calidad de software. Conceptos de privacidad, integridad y seguridad en Sistemas de Información.

XIII - Imprevistos

.

XIV - Otros

ELEVACIÓN y APROBACIÓN DE ESTE PROGRAMA	
	Profesor Responsable
Firma:	
Aclaración:	
Fecha:	