Ministerio de Cultura y Educación **(Programa del año 2017)**
Universidad Nacional de San Luis
Facultad de Ciencias Físico Matemáticas y Naturales
Departamento: Informatica
Area: Area IV: Pr. y Met. de Des. del Soft.

I - Oferta Académica

Materia	Carrera	Plan	Año	Período
INGENIERIA DE SOFTWARE II	LIC.CS.COMP.	32/12	2017	2° cuatrimestre

II - Equipo Docente

Docente	Función	Cargo	Dedicación
RIESCO, DANIEL EDGARDO	Prof. Responsable	P.Asoc Exc	40 Hs
ABDELAHAD, CORINA NATALIA	Prof. Colaborador	P.Adj Exc	40 Hs
BERNARDIS, HERNAN	Auxiliar de Práctico	A.1ra Simp	10 Hs

III - Características del Curso

Credito Horario Semanal				
Teórico/Práctico	**Teóricas**	**Prácticas de Aula**	**Práct. de lab/ camp/ Resid/ PIP, etc.**	**Total**
Hs	3 Hs	2 Hs	4 Hs	9 Hs

Tipificación	Periodo
B - Teoria con prácticas de aula y laboratorio	2º Cuatrimestre

Duración			
Desde	**Hasta**	**Cantidad de Semanas**	**Cantidad de Horas**
07/08/2017	17/11/2017	15	135

IV - Fundamentación

Dar las bases teóricas y prácticas que permiten al Ingeniero de Software aplicar métodos de desarrollo utilizando herramientas, para construir distintos tipos de sistemas, capaces de automatizar las actividades que se realizan durante el proceso de desarrollo del software.

V - Objetivos / Resultados de Aprendizaje

Introducir al alumno en el desarrollo de sistemas aplicando métodos de desarrollo que permiten producir software de manera fiable, de calidad y que funcione en máquinas reales cubriendo las distintas etapas del proceso de desarrollo, comprendiendo y aplicando, además, conceptos de Arquitecturas Orientadas a Servicios, Sistemas de Tiempo Real y de Ingeniería Basada en Componentes. Introducir al alumno conceptos de Reingeniería de Sistemas de Software.

VI - Contenidos

Unidad 1: Requerimientos

Introducción. Requerimientos de Software. Tipos. Procesos de la Ingeniería de Requerimientos. Estudio de Factibilidad. Obtención y Análisis. Especificación. Modelado. Validación de Requerimientos. Gestión de los Requerimientos.

Unidad 2: Modelos Avanzados Orientados a Objetos en UML

Introducción. Modelos. Importancia de los modelos. Modelos estáticos. Modelos dinámicos. Persistencia. Concurrencia. Estado. Comportamiento. Mecanismos comunes. Estereotipos. Valores etiquetados. Restricciones. Máquinas de Estado. Modelo Arquitectónico. Componentes. Despliegue.

Unidad 3: Proceso Unificado: Framework.
Introducción. Dirigido por Casos de Usos. Centrado en la Arquitectura. Iterativo e Incremental. Modelo de Casos de Usos. Captura de requisitos. Contexto del Sistema. Modelo del Dominio. Distintas Instanciaciones del Proceso. Análisis de la arquitectura. Relación con el Diseño. Componentes.

Unidad 4: Patrones de Diseño
Introducción. Conceptos. Descripción. Selección de un patrón de Diseño. Utilización. Problema. Solución. Consecuencia. Catálogo de Patrones de Diseño: patrones creacionales, patrones estructurales y patrones de comportamiento.

Unidad 5: Proceso Unificado: Análisis y Diseño.
Introducción. Propósito. Diferencias. Artefactos. Modelo del Análisis. Arquitectura. Flujo de Trabajo. Rol del diseño. Artefactos. Modelo del Diseño. Subsistemas. Interfaz. Modelo de Desarrollo. Aplicación de Patrones en el Diseño.

Unidad 6: Ingeniería de la Información y Basada en Componentes.
Ingeniería de la Información. Clasificación. Modelado del área de Negocio. Modelo de Negocio. Notación de Modelado de Procesos. BPMN. Tecnología. Ingeniería de software basada en componentes. Reuso. Modelo de componentes. Desarrollo Basado en Componentes. Composición de Componentes. Arquitectura de Componentes.

Unidad 7: XML.
Introducción. Estructura. Elementos. Atributos. Subelementos. Documento bien formado. Espacio de Nombres. Esquema. Declaración de Tipos de Datos. XPath. XQuery.

Unidad 8: Conceptos de Arquitecturas Basadas en Servicios (SOA).
Introducción. SOA y BPM (Gestión Orientada hacia los Procesos). Nociones de Sistemas Colaborativos. Importancia de la adopción de una estrategia SOA en la organización. La evaluación contextual de la tecnología y el reconocimiento de un cambio de mentalidad como claves para el éxito de la adopción de SOA. Beneficios. Introducción al Protocolo de Acceso a Objetos Simple. Modelo de Procesamiento SOAP. Arquitectura Web y SOAP. WSDL (Web Service Definition Language). Servicios Web. Arquitectura SOA.

Unidad 9: Reingeniería de Software
Concepto de Reingeniería de Software. Reingeniería de Procesos de Negocio. Reingeniería de Software. Ingeniería Inversa. Reestructuración. Ingeniería Directa. Aspectos económicos.

Unidad 10: Sistemas de Tiempo Real
Introducción a los sistemas de Tiempo Real. Tipos de Requisitos Temporales. Aplicaciones. Características de los Sistemas de Tiempo Real. Diseño. Métodos y Herramientas. UML. Redes de Petri. Propiedades. Análisis. Árbol de Cobertura. Matriz de Incidencia. Redes de Petri Temporizadas. Lenguajes de Programación.

VII - Plan de Trabajos Prácticos

Laboratorio 1: Modelado Estático y Dinámico con UML. Ingeniería Directa e Ingeniería Inversa con Java.
Laboratorio 2: Uso de Herramientas CASE en Ingeniería Reversa de Modelos de Datos / Objetos
Laboratorio 3: Herramienta CASE para la construcción de Modelos de Negocio.
Laboratorio 4: XML. XQuery.
Laboratorio 5: Herramientas CASE para la Ingeniería del Software Basada en Componentes.
Laboratorio 6: Construcción de Software bajo SOA
Laboratorio 7: Uso de Herramientas basadas en Redes de Petri para el modelado de Sistemas de Tiempo Real Distribuidos.
Laboratorio 8: Patrones de Diseño.

Práctico 1: Modelos Estáticos en UML. Ingeniería Directa e Ingeniería Inversa con Java.
Práctico 2: Modelos de Dominio.
Práctico 3: Modelos Dinámicos en UML. Ingeniería Directa e Ingeniería Inversa con Java.

Laboratorio Integrador: Construcción de un software orientado a objetos usando herramientas que automatizan el proceso de desarrollo generando los distintos artefactos desde los requerimientos hasta su implementación con un caso de estudio real. Deberán aplicar los distintos conceptos aprendidos y utilizados en teoría, en laboratorios anteriores y en las prácticas.

VIII - Regimen de Aprobación

La materia se desarrolla con la modalidad de promoción sin examen final. Existen dos niveles:

a) Regularización solamente: Para regularizar la materia se deberá:
1.- Tener como mínimo un 80% de asistencia a clases teóricas y prácticas.
2.- Tener los prácticos, solicitados por la cátedra, aprobados, como método aplicado por la cátedra para la evaluación continua del alumno.
3.- Presentación y aprobación del proyecto integrador de laboratorio con nota mayor o igual a 7 (siete).
4.- Aprobar un parcial, o cualquiera de sus dos recuperaciones, con nota mayor o igual a 6 (seis).

b) Promoción sin examen final: Para regularizar y aprobar la materia se deberá:
1.- Cumplir con los requisitos a.1, a.2 y a.3.
2.- Aprobar el parcial, o cualquiera de sus dos recuperaciones, con una nota mayor o igual a 7 (siete).
3.- Aprobar una prueba final integradora con una nota mayor o igual a 7 (siete).

La nota final será la que surja de aplicar la siguiente fórmula:
NotaFinal = (max(p, r1, r2) + pfi + pil) / 3
max : función maximo
p : parcial,
r1 : primer recuperatorio del parcial
r2 : segundo recuperatorio del parcial
pfi: prueba final integradora
pil : proyecto integrador de laboratorio.

Aquellos alumnos que sólo regularicen la materia deberán rendir un examen final, en los turnos establecidos.

Alumnos Libres: Por las características propias del proyecto de laboratorio a desarrollarse durante todo el cuatrimestre, no se aceptan alumnos libres.

IX - Bibliografía Básica

[1] Roger S Pressman. Software Engineering: A Practitioner's Approach, 7/e. R. S. Pressman & Associates Inc. 2010.
[2] Booch, Rumbaugh, Jacobson. El Proceso de Desarrollo de Software Unificado. Addison-Wesley. 1999.
[3] Booch, Rumbaugh, Jacobson. The Unified Modeling Language User Guide, 2nd Edition. Addison-Wesley. 2005.
[4] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley. 1995.
[5] Ian Sommerville. Ingeniería de Software, 8 edición. Addison Wesley. 2005.
[6] Jeff Davis. SOA: Open Source (Spanish Edition). Anaya Multimedia. 2009
[7] Alan Burns, Andy Wellings. Sistemas de Tiempo Real y Lenguajes de Programacion - 3ra Edición. Pearson Educacion. 2005
[8] Priscilla Walmsley. XQuery. O'Reilly. 2007.
[9] Tadao Murata. Petri nets: Properties, Analysis and Applications. Proc. of the IEEE, 77(4), 1989.
[10] OpenUP/Basic. http://epf.eclipse.org/wikis/openupsp/. Último acceso Agosto de 2016.
[11] Apuntes de Cátedra.

X - Bibliografia Complementaria

[1] Mark Grand. Patterns in Java. Volume 1. A Catalog of Reusable Design Patterns Illustred with UML. John Wiley & Sons Inc. 1998.

[2] UML Semantics. http://www.omg.org - UML Notation Guide. http://www.omg.org. Último acceso Agosto de 2017.
[3] Craig Larman. UML y Patrones: Introducción al análisis y diseño orientado a objetos. Prentice Hall. 1999.
[4] Booch, Grady. Object-Oriented analysis and design with applications. The Benjamin/Cummings Publishing Company Inc. 1994.
[5] George T. Heineman, William T. Councill. Component-Based Software Engineering: Putting the Pieces Together. Addison-Wesley Professional. 2001.
[6] Dirk Krafziq, Karl Banke, Dirk Slama. Enterprise SOA. The Coad Series, Prentice Hall. 2004.
[7] SOAP - Messaging Framework, http://www.w3.org/TR/soap12-part1/. Último acceso Agosto de 2013.
[8] SOAP - Adjuncts, http://www.w3.org/TR/soap12-part2/. Último acceso Agosto de 2017.
[9] Web Services Description Language, http://www.w3.org/TR/wsdl.html. Último acceso Agosto de 2017.
[10] Doug Tidwell, James Snell, Pavel Kulchenko. Building Application Distribuited Programming Web Services with SOAP. O'Reilly, First Edition. Diciembre de 2001
[11] Booch, Rumbaugh, Jacobson. The Unified Modeling Language Reference Manual, 2nd Edition. Addison-Wesley. 2005.
[12] Dan Pilone, Neil Pitman. UML 2.0 in a Nutshell. O'Reilly. 2005.
[13] Bernard Kolman. Álgebra Lineal con Aplicaciones en Matlab. Sexta edición. Prentice Hall. 2007.

XI - Resumen de Objetivos

Introducir al alumno en el desarrollo de sistemas aplicando métodos de desarrollo que permiten producir software de manera fiable, de calidad y que funcione en máquinas reales, cubriendo desde la especificación de requisitos hasta la obtención del producto, construyendo artefactos tanto formales como semi-formales. Introducir conceptos relacionados con Arquitecturas Orientadas a Servicios, Sistemas de Tiempo Real y Reingeniería de Sistemas de Software.

XII - Resumen del Programa

Requerimientos
Modelos Avanzados Orientados a Objetos en UML
Proceso Unificado: Framework
Patrones de Diseño
Proceso Unificado: Análisis y Diseño.
Ingeniería de la Información y Basada en Componentes.
XML. Conceptos de Arquitecturas Basadas en Servicios (SOA).
Reingeniería de Software
Sistemas de Tiempo Real

XIII - Imprevistos

.

XIV - Otros