Ministerio de Cultura y Educación **(Programa del año 2024)**
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
INGENIERÍA DE SOFTWARE II	LIC.CS.COMP.	RD-3 -1/20 23	2024	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	Responsable de Práctico	JTP Simp	10 Hs
THOMPSON, HORACIO JESUS	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	1 Hs	3 Hs	7 Hs

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

Duración			
Desde	**Hasta**	**Cantidad de Semanas**	**Cantidad de Horas**
05/08/2024	15/11/2024	15	105

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 estudiante 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 y Seguridad
Para cubrir dichos objetivos se integrarán conceptos, modelos y métodos en un proyecto integrador.
Durante el dictado de la asignatura se abordan los siguientes ejes transversales:
- Identificación, formulación y resolución de problemas de informática
- Concepción, diseño y desarrollo de proyectos de informática
- Utilización de técnicas y herramientas de aplicación en la informática
- Fundamentos para el desempeño en equipos de trabajo

- Fundamentos para la comunicación efectiva
- Fundamentos para evaluar y actuar en relación con el impacto social de su actividad en el contexto global y local
- Fundamentos para el aprendizaje continuo

VI - Contenidos

Ingeniería de Requerimientos. Proceso de Desarrollo de Software enmarcado en el Paradigma de la Orientación a Objetos. Arquitectura. Modelo de Análisis. Modelo de Diseño. Patrones de Diseño. Modelo de Implementación. Sistemas de Información. Seguridad.

Estos contenidos mínimos se desglosan en las siguientes unidades:

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: Modelo de Negocio

Modelado del área de Negocio. Modelo de Negocio. Notación de Modelado de Procesos. BPMN.

Unidad 7: Conceptos de Arquitecturas Basadas en Servicios (SOA).

Introducción. XML. XQuery. 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 8: Seguridad para Ingenieros de Software

Definiciones. Dimensiones de Seguridad. Capas. Diferencias en seguridad de aplicaciones/infraestructura. Gestión. Seguridad y confiabilidad. Tipos de amenazas. Garantías de seguridad. Seguridad y organizaciones. Políticas de seguridad. Evaluación y gestión de riesgos. Evaluación preliminar, de diseño, operacional. Requisitos de seguridad. Casos de uso de seguridad. Diseño de sistemas seguros. Compromisos de Diseño. Evaluación de Riesgos de Diseño. Diseño Arquitectónico. Pautas de Diseño para Ingeniería de Seguridad. Pautas para una Programación de Sistemas Seguros. Autenticación. Basada en el objeto. Basada en el conocimiento. Cracking. Biometría. Endurecimiento del código. Mitigación completa, fuerte y débil. Pruebas y aseguramiento de la seguridad.

VII - Plan de Trabajos Prácticos

Para cada unidad se deja disponible el material correspondiente a los contenidos de la unidad, las diapositivas de clase, el

apunte teórico y su correspondiente trabajo práctico en el repositorio digital.
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. Construcción de Software bajo SOA
Laboratorio 5: 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.

A continuación, se describe cómo se abordan y cómo se evalúan los ejes transversales trabajados en la asignatura:
Eje: Identificación, formulación y resolución de problemas de informática
Cómo se aborda: Se aborda a partir de la unidad 1 mediante el desarrollo de trabajos prácticos, guiadas por clases teóricas, diapositivas de clase, apuntes teóricos, prácticos de aula y consultas grupales e individuales.
Cómo se evalúa: Mediante un seguimiento continuo de parte de los docentes. Control de ejercicios en el pizarrón. Además, mediante una evaluación parcial en las fechas predeterminadas en el cronograma de la materia, cada evaluación posee su respectiva recuperación.

Eje: Concepción, diseño y desarrollo de proyectos de informática
Cómo se aborda: El estudiante debe trabajar en la 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. Para ello se trabajará de manera incremental, primero el desarrollo de los modelos correspondientes, para luego llegar al código y posteriormente las pruebas.
Cómo se evalúa: A través de la presentación de un informe con los artefactos construidos y problemas surgidos. Además, deben realizar una presentación oral explicando el caso de estudio, mostrando los modelos construidos y posteriormente la ejecución de dicho sistema.

Eje: Utilización de técnicas y herramientas de aplicación en la informática
Cómo se aborda: El estudiante debe modelar utilizando herramientas de modelados específicas, y la programación se desarrolla en el lenguaje Java.
Cómo se evalúa: A través de la presentación de los modelos, luego en la presentación oral del mismo se observa el código fuente del sistema desarrollado.

Eje: Fundamentos para el desempeño en equipos de trabajo
Cómo se aborda: El proyecto Integrador se realiza conformando equipos de trabajo de 2 o 3 personas.
Cómo se evalúa: A lo largo de las entregas parciales del proyecto integrador se verifica que cada integrante del grupo pueda explicar la tarea realizada.

Eje: Fundamentos para la comunicación efectiva
Cómo se aborda:
Expresión oral: Se realizan exposiciones de entregas parciales del Proyecto Integrador y se socializa con compañeros y docentes.
Expresión escrita: Se realiza un informe del Proyecto Integrador.
Cómo se evalúa:
Expresión oral: Mediante una rúbrica que se le entrega al estudiante con el enunciado del proyecto integrador. Se busca que el estudiante siga adquiriendo la capacidad de expresarse utilizando un vocabulario acorde a los contenidos vistos en la asignatura.
Expresión escrita: En las correcciones informadas se hace hincapié no sólo en lo disciplinar sino también en cuestiones de redacción.

Eje: Fundamentos para evaluar y actuar en relación con el impacto social de su actividad en el contexto global y local
Cómo se aborda: Desde el inicio se fomentará una actitud crítica que les permita evaluar el impacto social que tendrá el sistema a desarrollar.
Como se evalúa: Cada entrega del proyecto integrador posee un análisis sobre el impacto social que tendrá el sistema.

Eje: Fundamentos para el aprendizaje continuo
Cómo se aborda: Todas las unidades tienen actividades prácticas para que los estudiantes respondan participando de las clases teóricas y clases prácticas. En cada práctico se hace un seguimiento de los ejercicios realizados por el estudiante.
Cómo se evalúa: Cada entrega/consulta tiene una corrección informada.

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 estudiante.
3.- Presentación y aprobación del proyecto integrador de laboratorio con nota mayor o igual a 7 (siete).
4.- Aprobar una evaluación 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 la evaluación parcial, o cualquiera de sus dos recuperaciones, con una nota mayor o igual a 7 (siete).
3.- Aprobar un coloquio de carácter integrador oral o escrito con una nota mayor o igual a 7 (siete).

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

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

IX - Bibliografía Básica

[1] [1] Roger S Pressman. Software Engineering: A Practitioner's Approach, 7/e. R. S. Pressman & Associates Inc. 2010.
[2] [2] Booch, Rumbaugh, Jacobson. El Proceso de Desarrollo de Software Unificado. Addison-Wesley. 1999.
[3] [3] Booch, Rumbaugh, Jacobson. The Unified Modeling Language User Guide, 2nd Edition. Addison-Wesley. 2005.
[4] [4] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley. 1995.
[5] [5] Freeman, Eric, and Elisabeth Robson. Head First Design Patterns. O'Reilly Media, 2020.
[6] [6] Ian Sommerville. Ingeniería de Software, 8 edición. Addison Wesley. 2005.
[7] [7] Jeff Davis. SOA: Open Source (Spanish Edition). Anaya Multimedia. 2009
[8] [8] OpenUP/Basic. http://epf.eclipse.org/wikis/openupsp/.
[9] [9] Security for Software Engineers.James Helfrich. CRC Press. 2019.
[10] [10] Software engineering. Sommerville, Ian. Pearson. 2016.
[11] [11] Fred Long, Dhruv Mohindra, Robert C. Seacord, Dean F. Sutherland, David Svoboda, Java™ Coding Guidelines: 75 Recommendations for Reliable and Secure Programs, Addison Wesley, 2014
[12] [12] Derek Fisher, Application Security Program Handbook: A guide for software engineers and team leaders, Version 2, Manning Publications, 2021
[13] [13] Apuntes de Cátedra.

X - Bibliografia Complementaria

[1] [1] Mark Grand. Patterns in Java. Volume 1. A Catalog of Reusable Design Patterns Illustred with UML. John Wiley & Sons Inc. 1998.
[2] [2] UML Semantics. http://www.omg.org - UML Notation Guide. http://www.omg.org. Último acceso Agosto de 2017.
[3] [3] Sufyan Bin Uzayr. Software design patterns: the ultimate guide. CRC Press, 2023.
[4] [4] Craig Larman. UML y Patrones: Introducción al análisis y diseño orientado a objetos. Prentice Hall. 1999.
[5] [5] Booch, Grady. Object-Oriented analysis and design with applications. The Benjamin/Cummings Publishing Company Inc. 1994.
[6] [6] Dirk Krafziq, Karl Banke, Dirk Slama. Enterprise SOA. The Coad Series, Prentice Hall. 2004.
[7] [7] SOAP - Messaging Framework, http://www.w3.org/TR/soap12-part1/. Último acceso Agosto de 2013.
[8] [8] SOAP - Adjuncts, http://www.w3.org/TR/soap12-part2/. Último acceso Agosto de 2017.
[9] [9] Web Services Description Language, http://www.w3.org/TR/wsdl.html. Último acceso Agosto de 2017.
[10] [10] Doug Tidwell, James Snell, Pavel Kulchenko. Building Application Distribuited Programming Web Services with SOAP. O'Reilly, First Edition. Diciembre de 2001
[11] [11] Booch, Rumbaugh, Jacobson. The Unified Modeling Language Reference Manual, 2nd Edition. Addison-Wesley. 2005.
[12] [12] Dan Pilone, Neil Pitman. UML 2.0 in a Nutshell. O'Reilly. 2005.

XI - Resumen de Objetivos

Introducir al estudiante 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. Introducir conceptos relacionados a la Seguridad.

XII - Resumen del Programa

Requerimientos
Modelos Avanzados Orientados a Objetos en UML
Proceso Unificado: Framework
Patrones de Diseño
Proceso Unificado: Análisis y Diseño.
Seguridad para Ingenieros de Software

XIII - Imprevistos

Contacto con la cátedra: cabdelah@email.unsl.edu.ar

XIV - Otros