UNIVERSIDAD NACIONAL DE RÍO CUARTO
FACULTAD DE CIENCIAS EXACTAS, FÍSICO-QUÍMICAS Y NATURALES
DEPARTAMENTO DE COMPUTACIÓN

CARRERA: LICENCIATURA EN CIENCIAS DE LA COMPUTACIÓN. PLAN DE ESTUDIOS: 1999

ASIGNATURA: Ingeniería de Software **CÓDIGO:** 3304

DOCENTE RESPONSABLE: Mg. Marcela Daniele

EQUIPO DOCENTE: Lic. Marcelo Uva, Lic. Ariel Arsaute, Prof. Daniela Solivellas, An. Mariana Frutos

AÑO ACADÉMICO: 2019

REGIMEN DE LA ASIGNATURA: Cuatrimestral (2do cuatrimestre)
RÉGIMEN DE CORRELATIVIDADES: (para cursado)

Aprobada	*Regular*
	Análisis y Diseño de Sistemas - 3303

CARGA HORARIA TOTAL: 170 **TEÓRICO:** 60 hs **PRÁCTICAS y LABORATORIO:** 110 hs

CARÁCTER DE LA ASIGNATURA: Obligatorio

A. CONTEXTUALIZACIÓN DE LA ASIGNATURA

La asignatura se dicta en 3er año de las carreras de Ciencias de la Computación.

B. OBJETIVOS

- Introducir al estudiante en conceptos fundamentales de Ingeniería de Software y transversales al ciclo de vida de desarrollo de un sistema, como: planificación y gestión de proyectos, arquitectura de software, calidad y modelos de mejora de procesos de desarrollo, tecnologías de software distribuidas, gestión de la configuración de software.
- Modelar soluciones a problemas computacionales utilizando lenguajes de especificación, como: Redes de Petri. Object Constraint Language (OCL).
- Promover el uso de herramientas de sistemas colaborativos, y fortalecer la vinculación entre estudiantes y el trabajo en equipo para el desarrollo de software.
- Abordar conceptos sobre aspectos legales y auditoría de sistemas. Brindar fundamentos sobre propiedad Intelectual, licenciamiento de software y contratos informáticos.

C. CONTENIDOS BÁSICOS DEL PROGRAMA A DESARROLLAR

Gestión de Proyectos. Sistemas Colaborativos. Métricas del Proceso, del Proyecto y del Producto. Métricas Orientadas a Objetos. Planificación de Proyectos. Estimación. Técnicas de Descomposición. Métodos empíricos de estimación. Puntos de Función. Puntos de Casos de Uso. Planificación Temporal. Gestión de Configuración de Software. Elementos de Configuración de Software. Control de Versiones y Cambios. Auditoria de la Configuración. Herramientas. Aseguramiento de la Calidad del Software. Revisiones Técnicas Formales. Estándares de Calidad ISO 9000. Especificación de Software. Redes de Petri. Especificaciones usando Object Constraint Language (OCL). Arquitectura de Software y Patrones Arquitecturales. Introducción a Sistemas Distribuidos. Introducción a Tecnologías Web. Propiedad Intelectual, licenciamiento de software y contratos informáticos. Aspectos legales. Auditoría y Peritaje. Los contenidos se detallan en el programa analítico.

D. FUNDAMENTACIÓN DE LOS CONTENIDOS

Con el advenimiento de las computadoras de la 3era. generación, comenzaron a producirse muchos problemas en el desarrollo de software, dado que no se contaba con técnicas y métodos que permitieran

organizar y estructurar el desarrollo de sistemas de mayor tamaño y complejidad que los que se construían hasta ese momento.

Hacia fines de los años 60 se produce la llamada "crisis del software", caracterizada por carencia de fiabilidad, necesidad de mantenimiento permanente, retrasos en las entregas y costes superiores a los presupuestados, imposible de mantener, sin trasparencia y con pocas posibilidades reales de modificarse o mejorarse, entre otras.

A consecuencia de esta crisis surge la denominada "Ingeniería de Software". Se acordó en que el desarrollo de software debe verse como el desarrollo de un producto de ingeniería que requiere planeamiento, análisis, diseño, implementación, prueba y mantenimiento. La idea básica consistió en observar el sistema de software a construir como un producto complejo, y a su proceso de construcción como un trabajo ingenieril, basado en metodologías, técnicas, teorías y herramientas.

Es por ello que para desarrollar un proyecto de software es necesario seguir las bases de un proceso o metodología de software, que básicamente indica el conjunto y secuencia de actividades que se deben seguir para completar el ciclo de vida de desarrollo de un software. Y, como en cualquier otro tipo de proyecto, es necesario que exista una buena gestión del proyecto para que el mismo resulte exitoso. La Gestión de Proyectos de Software se basa en planificar, organizar, supervisar y controlar la evolución de los proyectos durante todo su ciclo de vida.

E. ACTIVIDADES A DESARROLLAR

Se dictan las clases teóricas a todos los alumnos que cursan la materia, con un total de entre 3 hs y 5 hs semanales. Además, los estudiantes asisten a las clases prácticas dos veces por semana de 2 hs de duración cada una. Y como actividad integradora, se desarrolla un proyecto/taller basado en la solución de un problema real con un cliente real.

CLASES TEÓRICAS: Presencial en aula, 60 Hs totales
CLASES PRÁCTICAS: Presencial en laboratorios, 50 Hs totales
CLASES TALLER: Presencial en laboratorios, 60 Hs totales

F. NÓMINA DE TRABAJOS PRÁCTICOS

1) Planificación Temporal.
2) Patrones Arquitecturales.
3) Estimación de Proyectos de Software.
4) Métricas Orientadas a Objetos.
5) Gestión de Configuración de Software
6) Técnicas de Modelado: Redes de Petri.
7) Especificación Formal: OCL.

G. HORARIOS DE CLASES:

HORARIO DE CLASES TEORICAS
Lunes 13 a 16 hs (y martes 11 a 13 hs se completan clases teóricas)
HORARIO DE CLASES PRACTICAS
Comisión 1: Martes 9 a 11 hs y Jueves 10 a 12 hs
Comisión 2: Martes 14 a 16 hs y Jueves 16 a 18 hs
HORARIO DE TALLER: Martes de 11 a 13 hs
HORARIO DE CLASES DE CONSULTAS: a coordinar con los estudiantes

H. MODALIDAD DE EVALUACIÓN:

- **EVALUACIONES PARCIALES:** 2 exámenes parciales escritos sobre el práctico-teórico de la materia, con sus respectivos recuperatorios.
- **EVALUACIÓN FINAL:** Los exámenes finales se desarrollan sobre la teoría y práctica de la asignatura, pudiendo ser oral o escrito. El estudiante que aprueba los exámenes parciales y el proyecto/taller, logrando en cada uno el 70% o más, podrá rendir un examen final incluyendo solo la parte teórica, compuesta por los temas incluidos en el programa de la asignatura.
- **CONDICIONES DE REGULARIDAD:** Aprobar los dos exámenes prácticos y el proyecto/taller integrador.

I. PROGRAMA ANALÍTICO

Unidad 1: Gestión de Proyectos

Personas. Organización. Coordinación y Comunicación. Producto. Proceso. Proyecto. Herramientas. El Proyecto construye el Producto. Artefactos de un Producto. El Proyecto dirige Proyectos. Herramientas en el Proceso. Prácticas Críticas. Conceptos fundamentales de Sistemas Colaborativos.

Unidad 2: Arquitectura de Software y Patrones Arquitecturales

Arquitectura. Conceptos. Descripción. Un proceso de desarrollo centrado en la Arquitectura. Vistas de la arquitectura de los diferentes modelos del proceso. Patrones. Categoría de Patrones. Patrones Arquitecturales: Layers. Pipes y Filter. Backboard. Broker. Model-View-Controller. Mikrokernel.

Unidad 3: Planificación de Proyectos.

Objetivos. Recursos. Humanos. Software. Técnicas de Descomposición. Estimación Basada en el Problema. Estimación basada en puntos de función. Estimación basada en el proceso. Modelos Empíricos de Estimación. Modelo COCOMO y COCOMO II. Ecuación del Software. Planificación Temporal. PERT. Diagrama de Gantt. Herramientas.

Unidad 4: Proceso de Software y Métricas de Proyecto.

Medidas. Métricas. Indicadores. Métricas en el Proceso. Métricas del Proyecto. Medición del Software. Métricas orientadas al Tamaño. Métricas orientadas a la Función. Reconciliación de diferentes aproximaciones de Métricas. Métricas para la Calidad del Software. Integración de Métricas con el Proceso de Software. Control Estadístico del Proceso. Estableciendo un Programa de Métricas. Herramientas. Métricas para Sistemas Orientados a Objetos: Objetivo. Métricas orientadas a Clases. Métricas CK. Métricas de Lorenz y Kidd. Métricas orientadas a Operaciones. Métricas para Proyectos orientados a Objetos.

Unidad 5: Especificación de Software

Uso. Calidad. Verificación. Declaración de tipos, valores, axiomas. Tipos base. Redes de Petri. Aplicación y Propiedades. Grafo de Alcanzabilidad. Especificaciones usando Object Constraint Language (OCL). Especificación Formal.

Unidad 6: Gestión de la Configuración del Software

Introducción. Línea Base. Elementos de Configuración de Software. El Proceso de Gestión de la Configuración del Software. Control de Versiones. Control de Cambios. Auditoria de la Configuración. Herramientas.

Unidad 7: Aseguramiento de la Calidad del Software

Conceptos. Calidad de Software. Factores de Calidad de McCall. Costos. Actividades SQA. Revisiones de Software. Revisiones Técnicas Formales. Calidad Estadística. Medidas. Fiabilidad. Disponibilidad. Seguridad. Plan de SQA. Estándares de Calidad ISO 9000.

Unidad 8: Propiedad Intelectual, licenciamiento de software y contratos informáticos. Aspectos legales. Auditoría y Peritaje.

Introducción. Conceptos Básicos. Fundamentos. Derecho de autor. Patentes y Marcas. Licencias de software libre. Software Libre y Open Source. Códigos de Ética. Leyes de la Propiedad Intelectual. Definición y Objetivos de la Auditoría Informática. Tipos y clases de Auditoría.

J. CRONOGRAMA DE CLASES TEORICAS, PRACTICAS Y TALLER

TEÓRICOS	TRABAJOS PRÁCTICOS	TALLER
1. Introducción. Gestión de Proyectos de Software.	1. Gestión de proyectos: Producto, Persona, Proceso, Proyecto (1 clase)	Inicio del Taller Revisión y reorganización de equipos
2. Arquitectura de Software. Patrones Arquitecturales	2.. Patrones Arquitecturales (4 clases)	Herramientas Refactoring
3. Planificación Temporal. PERT. GANTT	3. PERT. GANTT (2 clases)	Ing inversa Arquitectura/patrones arq
4. Planificación de Proyectos. Estimación. PF. PCU. COCOMO. COCOMO II	4. Puntos de Función, Puntos de Casos de Uso (1 clase) 5. COCOMO. COCOMO II (2 clases)	Planificación y Estimación ágil
5. Métricas para Sistemas Orientados a Objetos	6. MOO (1 clase)	Medición y análisis de SW
6. Gestión de Configuración del Software	7 GCS (2 clases)	Mostrar proceso de GCS (pt, git, github)
7. Especificación de Software. Redes de Petri.	8. PN (4 clases)	Presentación al cliente GCS implementación
8. Especificación de software. OCL	9. OCL (4 clases)	Ajustes y preparación de la presentación
9. Calidad de Software. SQA		Presentación TALLER
10. Auditoría. Propiedad Intelectual.		

BIBLIOGRAFÍA

- Software Engineering: A Practitioner's Approach. Roger S Pressman. 8th Edition. McGraw-Hill Education, 2014.
- Ingeniería de Software: Un enfoque práctco. Roger S. Pressman. 7ma Edición. McGraw Hill, 2007.
- An Integrated Approach to Software Engineering. Springer. Pankaj Jalote. 2005.
- Advanced Systems Design with Java, UML and MDA. Kevin Lano, 2005
- Fundamentals of Software Engineering. Carlo Ghezzi, Mehdi Jazayeri, Dino Mandrioli - Prentice Hall, 1991.
- Software Engineering. Ian Sommerville - Addison-Wesley, 5ta Edición. 1996. O cualquier version superior.
- Object Oriented Software Construction. Bertrand Meyer. Prentice Hall.
- El Proceso Unificado de Desarrollo de Software. Ivar Jacobson, Grady Booch, James Rumbaugh - Addison-Wesley, 2000.
- Software Testing: A Craftsman´s Approach. Paul C. Jorgensen - CRC Press, 1995.
- Design Patterns, Elements of Reusable Object/Oriented Software. Erich Gamma, Richard Helm, Ralph Johnson, John Vlissedes. - Addison-Wesley, 1995.
- Pattern-Oriented Software Architecture, Volume 1: A System of Patterns. Regine Meunier , Hans Rohnert , Peter Sommerlad , Michael Stal .
- Handbook of Software Quality Assurance- G. Gordon Schulmeyer, James I. McManus - Prentice Hall, 3ra Edición. 1999.
- Web Services. Concepts, Architectures and Applications. Gustavo Alonso, Fabio Casati, Harumi Kuno, Vijay Machiraju. Springer Verlag 2004. ISBN 3-540-44008-9
- Manual de Derecho Informático, Ed. Nova Tesis, Santa Fe. Luz Clara, Bibiana, 2001.
- Auditoria en Informática. 2da Edición. José Antonio Echenique García. Mc Graw Hill.
- Auditoría Informática: Un Enfoque Práctico. RAMA, Piattini, M., 2001

Profesora responsable: Mg. Marcela Daniele. _____