Universidad Nacional de Rio Cuarto
Facultad de Ciencias Exactas, Físico-Químicas y Naturales

FORMULARIO PARA LA PRESENTACIÓN DE PROGRAMAS DE ASIGNATURAS

Año Lectivo: 2025

UNIVERSIDAD NACIONAL DE RÍO CUARTO

FACULTAD DE CIENCIAS EXACTAS, FÍSICO-QUÍMICAS Y NATURALES

DEPARTAMENTO DE COMPUTACION

CARRERA/S: LICENCIATURA EN CIENCIAS DE LA COMPUTACIÓN, ANALISTA EN COMPUTACIÓN, PROFESORADO EN CIENCIAS DE LA COMPUTACIÓN

PLAN DE ESTUDIOS: 2024

ASIGNATURA: Ingeniería de Software I **CÓDIGO: 3385**

MODALIDAD DE CURSADO: Presencial

DOCENTE RESPONSABLE: **Mgter. Marcela Daniele, PAD simple interino**

EQUIPO DOCENTE: **Dr. Marcelo Uva, AY1 Exclusivo; Lic. Ariel Arsaute AY1 SemiExclusivo; Prof. Daniela Solivellas JTP SemiExclusivo**

RÉGIMEN DE LA ASIGNATURA: cuatrimestral

UBICACIÓN EN EL PLAN DE ESTUDIO: 2do año/2do cuatrimestre

RÉGIMEN DE CORRELATIVIDADES:

Asignaturas aprobadas: cod 3375

Asignaturas regulares: cod 3380

CARÁCTER DE LA ASIGNATURA: Obligatoria

CARGA HORARIA TOTAL: 112 horas

Teóricas	42 hs	Prácticas	42 hs	Teóricas -Prácticas	hs	Laboratorio	28 hs

CARGA HORARIA SEMANAL: 8 horas

Teóricas	3 hs	Prácticas	3 hs	Teóricas -Prácticas	hs	Laboratorio	2 hs

1. CONTEXTUALIZACIÓN DE LA ASIGNATURA

INCORPORE AQUÍ EL TEXTO

La asignatura se desarrolla en el segundo cuatrimestre de segundo año de las carreras: Analista en Computación, Profesorado en Ciencias de la Computación y Licenciatura en Ciencias de la Computación. Permite introducir al estudiante en los desafíos que propone la Ingeniería de Software para diseñar y construir productos de software sobre la base de un trabajo ingenieril, aplicando métodos, técnicas y herramientas a fin de producir un producto exitoso y de calidad.

2. OBJETIVOS PROPUESTOS

Con el desarrollo de esta asignatura se espera que el estudiante pueda apropiarse de los conocimientos, y desarrollar habilidades y aptitudes necesarias para:

● Comprender los desafíos de la ingeniería de software, los aspectos generales que hacen a la construcción de software, y los factores que influyen en su calidad.

● Conocer y aplicar las etapas del ciclo de vida del software, empleando modelos de procesos de desarrollo en la planificación y construcción de soluciones.

● Modelar e instanciar proyectos sobre diferentes métodos de desarrollo de software, valorando sus ventajas y limitaciones.

● Identificar y gestionar requerimientos de software, destacando el rol crítico de la ingeniería de requerimientos en el éxito de los proyectos.

● Conocer principios de análisis y diseño orientado a objetos, usando conceptos de encapsulamiento, abstracción, herencia y polimorfismo.

● Reconocer las características de calidad del diseño de software.

● Seleccionar los patrones de diseño adecuados para resolver problemas recurrentes en la construcción de sistemas de software.

● Aplicar técnicas y herramientas de pruebas de software para asegurar la calidad del producto.

3. EJES TEMÁTICOS ESTRUCTURANTES DE LA ASIGNATURA Y ESPECIFICACIÓN DE CONTENIDOS

3.1. Contenidos mínimos (según plan de estudio vigente)

Introducción a la Ingeniería del Software. Escalabilidad, productividad y calidad en el desarrollo de software. Procesos de software y modelos de procesos de desarrollo de software. Ciclos de vida del software. Ingeniería de Requerimientos. Enfoques para el análisis de problemas y especificación de requisitos. Arquitectura y Diseño de software. Modularidad y abstracción. Conceptos de cohesión y acoplamiento. Diseño orientado a objetos. Clases y objetos. Relaciones entre clases. Uso, agregación y herencia. Lenguajes para el modelado orientado a objetos.

Patrones de Diseño. Diseño detallado. Métodos de prueba del software.

3.2. Ejes temáticos o unidades

Unidad 1: **Ingeniería de software. Introducción.** Historia. Definición. El Ciclo de Vida del software. Atributos del software. Metodologías tradicionales. Metodologías ágiles. Modelos de desarrollo de Software: Construcción de Prototipos, Procesos Evolutivos, Incremental, Espiral, Ensamblaje de Componentes, Desarrollo Concurrente, Métodos Formales, Orientado a Objetos. Diseño por Contratos. Proceso Unificado. SCRUM. XP. Tipos de Sistemas.

Unidad 2: **Ingeniería de Requerimientos del Software**. Requerimientos funcionales y no funcionales. Requerimientos de usuario y de sistema. Documentación de requerimientos -SRS. Reingeniería del Software: Ingeniería Inversa e Ingeniería Directa. Definición y documentación de Requerimientos en diferente metodologías ágiles y tradicionales. Proceso Unificado: Características, Conceptos, Artefactos, Actividades, Trabajadores, Flujos de Trabajo y Fases. Captura de Requerimientos. Modelo de Negocio. Procesos de negocio. Glosario de términos. Reglas de negocio. Modelo de Casos de Uso. Modelo de Análisis. Clases de análisis. SCRUM: Características. Conceptos. Historias de usuario. Roles.

Unidad 3: **Modelado de sistemas.** Principios. Técnicas de Descripción de Requerimientos. Modelado estructural o estático y modelado dinámico. MOO. UML: Estereotipos. Valores etiquetados. Restricciones. Diagramas de clases: Clases: atributos, operaciones y responsabilidades. Relaciones. Interfaces. Paquetes. Diagrama de objetos: Objetos, Instancias, enlaces. Diagrama de Interacción: escenarios. Diagrama de casos de uso. Diagrama de actividades. Diagrama de estados. Eventos y señales. Modelado de nodos y componentes.

Unidad 4: **Diseño de Software.** Conceptos de diseño. Abstracción. Refinamiento. Modularidad. Arquitectura del software. Estructura de programa. Ocultamiento de información. Diseño modular. Cohesión y acoplamiento. Diseño detallado. Diseño OO. Metodología tradicional PU: Etapa Modelo de diseño: artefactos, actividades y trabajadores. Modelo de implementación. Nodos y componentes. **Patrones de diseño.** Introducción. Conceptos. Descripción. Utilización. Problema. Solución. Consecuencia. Catálogo de patrones de diseño. Categoría de patrones: creacionales, estructurales y de comportamiento.

Unidad 5: Prueba de software. Fundamentos teóricos. Principios de la prueba. Métodos de Prueba. Prueba funcional. Análisis del valor límite. Clases de equivalencia. Tablas de decisión. Prueba estructural. Cobertura de sentencia. Cobertura de arco. Cobertura de condición. Cobertura de camino. Complejidad ciclomática. Herramientas. PU: Modelo de prueba. Casos de prueba. Procedimiento de prueba. Plan de prueba.

4. ACTIVIDADES A DESARROLLAR

Se brindan clases teóricas a todo el estudiantado que cursa la asignatura, de manera presencial, y algunas clases podrán desarrollarse de forma híbrida sincrónica. Además, los estudiantes asisten a clases prácticas y de laboratorio, y como actividad integradora, se desarrolla un proyecto/taller que consiste en la aplicación de técnicas y herramientas de gestión de proyectos de software sobre un proyecto/producto de software preestablecido, para ello se conforman equipos de trabajo de entre 2 y 4 estudiantes.

CLASES TEÓRICAS: Presencial o híbrida sincrónica en aula/laboratorio, 3 hs semanales, 42 totales

CLASES PRÁCTICAS: Presencial o híbrida sincrónica en aula/laboratorio, 3 hs semanales, 42 totales

CLASES DE TRABAJOS PRÁCTICOS DE LABORATORIO: Presencial o híbrida sincrónica en laboratorio que se dictan conjuntamente en los horarios de clases prácticas de las comisiones o en las clases teóricas, 2 hs semanales, 28 totales.

OTRAS: tres parciales y un recuperatorio por cada uno. Además, el estudiante debe concluir y aprobar el proyecto taller integrador que se realiza en equipos de 2 a 4 estudiantes. Se toman evaluaciones periódicas de los contenidos teóricos para los estudiantes que aspiran a la promoción, a través de presentaciones orales, investigaciones sobre ciertas temáticas y espacios de debate organizados para tal fin.

5. PROGRAMAS Y/O PROYECTOS PEDAGÓGICOS INNOVADORES E INCLUSIVOS

6. CRONOGRAMA TENTATIVO DE CLASES E INSTANCIAS EVALUATIVAS

INCORPORE AQUÍ EL TEXTO

Que muestre coherencia y consistencia con el logro de los objetivos y las competencias definidas. Las fechas de parciales deberán ser consensuadas con los responsables de las demás asignaturas del cuatrimestre correspondiente, en acuerdo con la Res. C.S. 120/17).

Semana	Día/Horas	Actividad: tipo y descripción*
1	Lunes 11/08-3 hs Martes 12/8-2 hs Jueves 14/8-2 hs	Teórico. Presentación de la asignatura. Introducción al proceso de desarrollo de software, ciclo de vida, tipos de metodologías. Ingeniería de requerimientos y análisis del problema. Práctico/Lab. Presentación de la modalidad de trabajo de los prácticos. Armado de equipos. Presentación de herramientas.
2	Lunes 18/08-3hs Martes 19/8-2hs Jueves 21/8-2hs	Teórico. Modelados de software. Diagramas de Clases UML. Práctico/Lab. **P1 clase 1)** Trabajamos a partir de la narrativa 1 para analizar un problema, elaborar el glosario, e identificar posibles requerimientos funcionales y no funcionales. Incisos a, b y c. **P1 clase 2)** Analizamos DC, reconocemos elementos, completamos la identificación de funcionalidades y las expresamos como HU. Organizamos y priorizamos las HU. d, e, f y g
3	Lunes 25/08-3hs Martes 26/8-2hs Jueves 28/8-2hs	Teórico. Modelados de software. UML: Diag Clases, Diag Objetos. Ingeniería Directa e Inversa. Práctico/Lab. **P1 clase 3)** Trabajamos sobre narrativa 2, realizamos análisis del problema, glosario, DC, identificamos funcionalidades y las expresamos como HU. Organizamos y priorizamos las HU. Realizamos DO.
4	Lunes 1/09-3hs Martes 2/9-2hs Jueves 4/9-2hs	Teórico. Modelos de Negocio, Casos de Uso. UML DCU. Plantillas Genéricas p/descripción de CU, HU. Práctico/Lab. **P1 clase 5)** Trabajamos sobre la narrativa 4, realizamos análisis del problema, glosario, DC, identificamos funcionalidades y las expresamos como HU. Organizamos y priorizamos las HU. Realizamos DO, comprobar si el modelo soporta escenario. **P1 clase 6)**

		Trabajamos sobre la narrativa 5, realizamos análisis del problema, glosario, DC, identificamos funcionalidades y las expresamos como HU. Organizamos y priorizamos las HU. Realizamos DO, comprobar si el modelo soporta escenario.
5	Lunes 8/09-3hs Martes 9/9-2hs	Teórico. Arquitectura y Diseño de Software. Diseño Funcional. Modularidad y abstracción. Conceptos de cohesión y acoplamiento. Diseño Detallado. Análisis y Diseño c/PU. UML. Diagramas de Interacción: Diagrama de Secuencia. Práctico/Lab. Inicia Ejercicio transversal integrador "Sistema de Gestión Académica Saber Más". Se trabaja en equipos el análisis del problema, glosario y DC.
6	Lunes 15/9-3hs Martes 16/9-2hs Jueves 18/9-2hs	Teórico. Arquitectura y Diseño de Software. Patrones de Diseño. Los estudiantes estudian y se preparan para taller de PD del 29/9. Práctico/Lab. **P1 clase 8)** Ejercicio transversal integrador (ETI), identifican funcionalidades, las expresan como HU, las priorizan, organizan y construyen el product backlog del proyecto. sorteo de grupos para compartir ETI. clase 9) Puesta en común del ETI. Repaso y consulta primer parcial.
7	Lunes 22/9-3hs Martes 16/9-2hs Jueves 18/9-2hs	Instancia Evaluativa. Primer parcial (DC, RF y NoF, DO) Práctico/Lab. **P2 clase 1)** Análisis DC de diseño con PD, identificar patrón. Analizar el PD en/con DS, analizar código que implementa PD y construir DC identificando PD. **P2 clase 2)** aplicar Ingeniería inversa construir DC, identificar PD, modificar código.
8	Lunes 29/9-3hs Martes 30/9-2hs Jueves 2/10-2hs	Teórico. Continuación Arquitectura y Diseño de Software. Taller Patrones de Diseño. Práctico/Lab. **P2 clase 3)** Trabajo ETI: implementación de funcionalidad. **P2 clase 4)** Modelado con DC y PD.
9	Lunes 6/10-3hs Martes 7/10-2hs Jueves 9/10-2hs	Teórico. Eventos y señales. Modelado de nodos y componentes. UML Diag Estados y Actividades. Práctico/Lab. **P2 clase 5)** ETI: implementación de funcionalidad. **P2 clase 6)** Modelado con DC y PD.
10	Lunes 13/10-3hs Martes 14/10-2hs Jueves 16/10-2hs	Teórico. Prueba. Funcional y Estructural. PU: Modelo de Prueba. Prueba de CB**.** Práctico/Lab. **P2 clase 7) ETI** modelado diseño DC con PD, **P2 clase 8)** Cierre P2. Repaso y consulta para parcial
11	Lunes 20/10-3hs Martes 21/10-2hs Jueves 23/10-2hs	Instancia Evaluativa. Segundo parcial (Diseño PD) Práctico/Lab. **P3 clase 1y2)**práctica 3 Prueba CB
12	Lunes 27/10-3hs Martes 28/10-2hs Jueves 30/10-2hs	Teórico. Prueba. Funcional y Estructural. Prueba de CN**.** Práctico/Lab. **P3 clase 3) ETI** aplicar criterio de Prueba CB, definir CP, testing e informe. **P3 clase 4)** Prueba CN
13	Lunes 3/11-3hs Martes 4/11-2hs Jueves 6/11-2hs	Teórico. Modelos de Implementación. Conceptos. Nodo y Componente. UML: Diag Despliegue. Diag Artefactos**.** Práctico/Lab. **P3 clase 5)** ETI aplicar criterio de CN para definir el conj CP, testing, informe y entregarlo al equipo desarrollador. **P3 clase 6) ETI** a partir del informe entregado por el otro equipo revisar la funcionalidad, y aplicar criterio CN, definir conj CP, testing e informe. Cierre P3. Repaso y consulta para parcial
14	Lunes 10/11-3hs Jueves 13/11-2hs	Instancia Evaluativa. Tercer parcial (Testing) Consultas para recuperatorios

*Teóricos, teóricos-prácticos, trabajos de laboratorios, salidas a campo, seminarios, talleres, coloquios, instancias evaluativas, consultas grupales y/o individuales, otras.

7. BIBLIOGRAFÍA

7.1. Bibliografía obligatoria y de consulta

1. Pressman, Roger, Maxim Bruce. Software Engineering: A Practitioner's Approach. 9na edition. McGraw Hill. 2020.
2. Sommerville Ian. Ingeniería del Software, 7th edition, Addisson Wesley. Pearson Education, 2005.
3. Ghezzi Carlo, Jazayeri M., Mandrioli D.. Fundamentals of Software Engineering. Prentice Hall, 1991.
4. Kendall y Kendall. Análisis y Diseño de Sistemas. Pearson Education. 2005.
5. Pankaj Jalote. An Integrated Approach to Software Engineering. Springer. 2005.

6. Meyer Bertrand. Object Oriented Software Construction. Prentice Hall. 1997.
7. Booch Grady, Rumbaugh J., Jacobson Ivar. The Unified Modeling Language. Addison Wesley. 1999.
8. Booch Grady. Object-oriented analysis and design with applications. Benjamin Cummins, 1994. Versión español: Análisis y Diseño Orientado a Objetos: Con Aplicaciones. Addison Wesley. 1996.
9. Jacobson Ivar, Booch Grady, Rumbaugh James. The Unified Software Development Process. A. Wesley. 1999.
10. Gamma Erich, Helm Richard, Johnson Ralph, Vlissides John. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley. 1995.
11. Gerard O'Regan. Concise Guide to Software Engineering From Fundamentals to Application Methods. Springer 2017
12. Mike Cohn. *Agile Estimating and Planning*. Pearson Education, 2006.
13. Paul Ammann and Jeff Offutt. Introduction to Software Testing. Cambridge University Press. Second Edition. 2016
14. Paul C. Jorgensen. Software Testing: A Craftsman's Approach, Auerbach Publications; Fourth Edition. 2013
15. Grand Mark. Patterns in Java. Volume 1. A Catalog of Reusable Design Patterns Illustrated with UML. John Wiley & Sons Inc. 1998.
16. OMG. Object Management Group. Unified Modeling Language Specification. http://www.omg.org .
17. Rational Unified Process. http://www.rational.com/rup/
18. www.agilemanifesto.org http://www.agile-spain.com/
19. The Mythical Man-Month. Frederick Brooks. Addisson Wesley First Edition 1975.
20. Specification by Example: How Successful Teams Deliver the Right Software. Gojko Adzic, 2011

NOTA: Se utilizan materiales de lectura complementarios extraídos de publicaciones en revistas, sitios web, manuales de herramientas utilizadas, entre otros. En cada una de las actividades se indican los capítulos de lectura del material presentado en la bibliografía y se ponen a disposición otros materiales digitales utilizados.

7.2. Otros: materiales audiovisuales, enlaces, otros.

8. DÍA Y HORARIOS DE CLASES

+ **HORARIO DE CLASES TEÓRICAS: LUNES 13 a 16 hs.** tres horas, un día a la semana (y se completan con otros encuentros y actividades que se planifican en laboratorios, que suele implicar una hora semanal aprox)
+ **HORARIO DE CLASES PRÁCTICAS y PROYECTO/TALLER**
  Comisión 1: martes y jueves 10 a 12 hs.
  Comisión 2: martes y jueves 16 a 18 hs.
  Comisión 3: martes 16 a 18 hs. y jueves 18 a 20 hs.

9. DÍA Y HORARIO DE CLASES DE CONSULTAS

Lunes 16 hs, Lunes 9 hs, Jueves 10 hs y otros horarios se coordinar por mail o slack con los docentes.

10. REQUISITOS PARA OBTENER LA REGULARIDAD Y LA PROMOCIÓN

- **EVALUACIÓN PARCIAL:** tres exámenes parciales escritos sobre los contenidos de la materia. Un recuperatorio por cada parcial, que se desarrollan al final del cursado. Desarrollo y presentación de un proyecto–taller de manera grupal.
- **EVALUACIÓN FINAL:** escritos u orales sobre la teoría y práctica de la materia.
- **CONDICIONES DE REGULARIDAD:** Aprobar los exámenes parciales y el proyecto final. Además, evaluamos todo el proceso a través de otras instancias como exposiciones, investigación sobre algún tema previamente presentado y acordado, presentación y participación en talleres y debates.
- **CONDICIONES DE PROMOCIÓN:** Posee promoción. Los estudiantes que regularizan la asignatura, y cumplen y aprueban todas las instancias parciales evaluativas con el 70% o más, tienen la posibilidad de rendir un coloquio final teórico, el cual es opcional, y en caso de

aprobarlo, obtienen la promoción en la asignatura.

11. CARACTERÍSTICAS, MODALIDAD Y CRITERIOS DE LAS INSTANCIAS EVALUATIVAS

Los tres parciales incluyen ejercicios prácticos de los temas abordados en las clases teóricas y prácticas. Además, el ejercicio integrador permite profundizar en la construcción de habilidades de técnicas y herramientas de modelado y desarrollo de software. Los espacios de investigación, taller y debates permiten abordar algunos temas emergentes que se relacionan con los contenidos de la asignatura.

La asignatura puede rendirse en condición de libre para lo cual se le solicitará al estudiante que implemente un proyecto, luego un examen práctico y teórico, oral o escrito, que incluye los temas del programa.

Mg. Marcela Daniele.

Firma Profesor/a Responsable **Firma Secretario/a Académico/a**