INSTITUTO DE DESARROLLO ECONÓMICO E INNOVACIÓN
Año: 2020

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur.

PROGRAMA DE LA ASIGNATURA:
Ingeniería de Software I (IF031)

CÓDIGO: IF031
AÑO DE UBICACIÓN EN EL PLAN DE ESTUDIOS: 2 año
FECHA ULTIMA REVISIÓN DE LA ASIGNATURA: 2019-05-08
CARRERA/S: Analista Universitario de Sistemas 050/2017, Licenciatura en Sistemas 049/2017,

CARÁCTER: CUATRIMESTRAL (2do)
TIPO: OBLIGATORIA
NIVEL: GRADO
MODALIDAD DEL DICTADO: PRESENCIAL (EN LÍNEA)
MODALIDAD PROMOCION DIRECTA: SI
CARGA HORARIA SEMANAL: 10 HS
CARGA HORARIA TOTAL: 150 HS

EQUIPO DOCENTE

Nombre y Apellido	Cargo	e-mail
Adriana Urciuolo	Profesora Titular	aurciuolo@untdf.edu.ar
Ezequiel Moyano	Profesor Adjunto	jmoyano@untdf.edu.ar
Matías Moncho	Asistente de 1a	mmoncho@untdf.edu.ar

1. FUNDAMENTACION

Esta asignatura resulta fundamental para el futuro ejercicio de la profesión de Licenciado en Sistemas y/o Analista Universitario de Sistemas, dado que constituye el primer contacto del alumno con los conceptos vinculados al software como producto, al proceso de desarrollo de software y a métodos y herramientas de la ingeniería de software. En la asignatura se realiza un especial énfasis en las etapas de requerimientos, análisis y diseño del mencionado proceso, teniendo en cuenta que el alumno ya tiene incorporados conceptos de sistemas de información y de programación.
Se resalta además la importancia del modelado de sistemas a través de apropiadas metodologías, así como del desarrollo de software utilizando métodos ágiles . La enseñanza del desarrollo de software se propone mediante intensa práctica de la construcción de modelos en todas las etapas del mismo, en sucesivas iteraciones y centrado en la arquitectura.
- Con respecto a la ubicación de la materia dentro del Plan de estudios, cabe destacar que se trata de una asignatura del segundo año de la carrera, precedida por “Sistemas y Organizaciones” y Algorítmica y Programación I y II, donde ya han adquirido conceptos básicos de sistemas y programación fundamentales para comprender los procesos del desarrollo de software. Por ello es factible partir de dichos conocimientos para avanzar en conceptos de Ingeniería de Software
- Por otra parte, en el mismo cuatrimestre se dictan Bases de Datos I y Programación y diseño Orientado a Objetos, ambas con contenidos complementarios con esta asignatura, razón por la cual se debe coordinar el avance del dictado.
- Dado que la asignatura Ingeniería de Software II se ubica en el 3er año de la carrera, en esta asignatura se realiza una introducción a la Ingeniería de software sin profundizar en los conceptos vinculados a la administración de proyectos de software relacionados al siguiente curso, por cuanto el énfasis se pone en el proceso de desarrollo de software, desde la etapa de requerimientos hasta su verificación y validación.

- Se considera fundamental el trabajo en contacto permanente con los integrantes de otras cátedras relacionadas, tales como Programación y Diseño orientado a Objetos e Ingeniería de Software II, a los fines de brindar la mayor coherencia posible en el tratamiento de los temas y evitar superposiciones. En el caso de materias de programación y Bases de Datos, la relación debe ser continua, con el fin de compatibilizar criterios, conocer la bibliografía utilizada por cada asignatura y ajustar planes de trabajo.
Durante el año 2020, dada la situación originada por el COVID-19, la asignatura se dictará con modalidad presencial-en línea.

2. OBJETIVOS

a) OBJETIVOS GENERALES

Lograr que el alumno adquiera capacidad para la modelización y el desarrollo de software de calidad desde las primeras etapas de su concepción (con énfasis en las de requerimientos, análisis y diseño) aplicando métodos, herramientas y procedimientos adecuados a cada momento del proceso de desarrollo

b) OBJETIVOS ESPECIFICOS

Mediante el dictado de la materia se pretende que el alumno:
- Adquiera una base sólida en modelos de procesos de desarrollo de software, metodologías de análisis de requerimientos y de diseño de sistemas que le permita desarrollar software de calidad.
- Desarrolle habilidades para la modelización de sistemas
- Conozca las fases del proceso de desarrollo de software, valorando la importancia de llevar adelante adecuadamente las etapas de requerimientos, análisis y diseño en el marco de dicho proceso.
- Comprenda la importancia de la utilización de métodos, herramientas y procedimientos de la ingeniería de software en el desarrollo de sistemas informáticos.
- Sea capaz de realizar todos los pasos que comprende el análisis de un sistema y de diseñar los procesos y las estructuras de datos correspondientes.
- Esté preparado para realizar la especificación y la documentación de un sistema.
- Sea capaz de diseñar las interfaces del sistema.
- Comprenda la conveniencia de utilizar herramientas de automatización en la aplicación de las metodologías estudiadas para el desarrollo de sistemas informáticos.
- Valore la importancia de desarrollar apropiadas pruebas del software.

3. CONDICIONES DE REGULARIDAD Y APROBACION DE LA ASIGNATURA

Para la aprobación del cursado de la asignatura se requiere:
a) Asistencia al 70% de las clases prácticas.
b) Aprobación de los trabajos prácticos obligatorios.
b) Aprobación de dos parciales prácticos con un mínimo de cuatro (4) puntos o sus respectivas instancias recuperatorias.
d) Presentación de un trabajo integrador sobre: Desarrollo del Análisis y diseño de un sistema.
La calificación final de la cursada es el promedio de las notas obtenidas en los exámenes parciales, el trabajo integrador y los trabajos prácticos. En ningún caso la nota obtenida podrá ser inferior a 4..
Para la aprobación de la asignatura se deberá rendiir examen final con Nota igual a cuatro (4) o superior.

Para la Promoción de la asignatura, además de las condiciones básicas de cursada, se requiere:
a) Aprobación de los dos parciales prácticos en la primera instancia, con Nota igual o superior a

siete (7)
b) Aprobación de dos parciales teóricos con Nota igual o superior a siete (7)
c) Aprobación del trabajo integrador con Nota igual o superior a siete (7)
No se rinde examen final

4. CONTENIDOS DE LA ASIGNATURA

MODULO I: INTRODUCCIÓN A LA INGENIERÍA DE SOFTWARE
Ingeniería de Software. Importancia del software. Evolución. Características. Ingeniería de software e Ingeniería de Sistemas. Problemas en el desarrollo de software. Necesidad de la Ingeniería de Software. El Producto Software. Características. Aplicaciones. Visión general de la Ingeniería de software. Fases. El Proceso de desarrollo de software. Actividades del proceso. Métodos, herramientas y procedimientos. Paradigmas de la IS. Concepto de ciclo de vida del Software. Modelos de ciclo de vida del SW. Ciclos de desarrollo y ciclos de sistemas. Modelo lineal o secuencial, en cascada. Modelo DRA. Modelos de ciclo de vida de evolución. Modelo incremental y evolutivo. Modelo en espiral. Ensamblaje de componentes. Prototipos. Modelos formales: Modelo transformacional. Combinaciones de ciclos de vida.

MÓDULO II: REQUERIMIENTOS DEL SOFTWARE
Conceptos de requerimiento e Ingeniería de requerimientos. Tipos de requerimientos. Requerimientos funcionales y no funcionales. Procesos de la Ingeniería de requerimientos Productos entregables. Framework para los procesos de la IR. Procesos de: Elicitación, especificación, validación. Elicitación de requerimientos. Conceptos. Técnicas de Elicitación: Especificación de requerimientos del software (ERS). Qué incluye. Criterios a cumplir por la ERS. Métodos formales de especificación. Documentación de requerimientos. Esquema del Documento de requerimientos. Estándar IEEE/ANSI 830/1998. Características. Secciones del Estándar. Validación de requerimientos. Administración de requerimientos.

MÓDULO III: MODELADO DEL SISTEMA
Concepto de Modelo. Ventajas y funciones de los modelos. Herramientas. Principios del análisis. Modelado del análisis. Visión Esencial y de implementación. Modelos del contexto. Modelos de comportamiento. Modelo de datos. Modelos de objetos. Metodologías para el modelado de sistemas. Metodologías estructuradas orientadas a procesos. Análisis Orientado a Objetos, DSBC (Desarrollo basado en componentes), MDD (Desarrollo dirigido por modelos). Análisis Estructurado Moderno: Concepto de Análisis Esencial. Actividades Esenciales. Memoria Esencial. Especificación del sistema en Análisis estructurado. Análisis Orientado a Objetos. Conceptos de AOO: . Especificación del sistema en OO. Metodología CRC: Clases, Responsabilidades, Colaboraciones. Especificación formal de sistemas. Modelado del sistema con UML. Modelado de requerimientos del sistema con UML: Diagrama de Casos de uso. Modelado de la estática del sistema con UML: Diagrama de clases, Diagrama de Packages, diagrama de componentes. Modelado de la dinámica del sistema con UML: Diagrama de interacciones, diagrama de actividades. Modelado del comportamiento del sistema: Diagrama de máquinas de estado

MÓDULO IV: DISEÑO DEL SOFTWARE.
Concepto y Objetivos. Principios, claves y reglas del diseño. Diseño de alto nivel y diseño detallado. Actividades que comprende el diseño. Conceptos: Abstracción, refinamiento, modularidad división estructural. Mecanismos de abstracción. Modularización. Componentes. Independencia funcional. Cohesión y Acoplamiento. Arquitectura del Software. Componentes, conectores. Vistas de la Arquitectura. Decisiones de diseño arquitectónico. Organización del sistema. Estilos de descomposición modular. Descomposición en subsistemas, en capas, en particiones. Estilos de control. Patrones arquitecturales. Frameworks. Diseño de objetos. Relación ente arquitectura y diseño detallado. Transformación de modelos del análisis al diseño. Del diseño

a la implementación. Patrones de diseño. Diseño de interfaz. Modelos y procesos de diseño de interfaces. Principios fundamentales. Usabilidad.

MÓDULO V: METODOLOGÍAS PARA EL DESARROLLO ÁGIL DE SOFTWARE
Etapas del desarrollo de software. Actividades del proceso de desarrollo de SW. Importancia de los modelos de procesos de desarrollo de software. Requerimientos cambiantes. Desarrollo ágil de aplicaciones. Principios y prácticas. Importancia del desarrollo iterativo. Métodos ágiles. Programación extrema. Scrum. RAD. Prototipado del software. Reutilización de software. RUP Ágil: Características. Fases y Dimensiones. Disciplinas. Desarrollo iterativo en RUP.

MÓDULO VI: DESARROLLO DE SOFTWARE CON RUP AGIL
Modelando el contexto del sistema: Modelo del Dominio y Modelo del Negocio. Diagramas utilizados para su construcción. Modelando los Requerimientos del Sistema. Identificación de casos de uso. Descripción de casos de uso. Construcción de Diagramas de Casos de Uso. Modelado de requerimientos no funcionales. Modelo del Análisis estático del Sistema: Construcción del Diagrama de Clases de nivel conceptual partiendo del modelo de Dominio o del Negocio. Modelo de Análisis dinámico del sistema: Realización de casos de uso. Modelación de escenarios con diagramas de interacciones. Modelo de diseño: Completando el Diagrama de clases. Diseño arquitectural. Definición de subsistemas y dependencias. Diseño detallado. Mapeo del Diagrama conceptual al de diseño. Construcción del Diagrama de clases del Diseño. Del diseño a la implementación.
MÓDULO VII. DESARROLLO DE SOFTWARE PARA EL REUSO
Diseño para el reuso. Uso de patrones de diseño y arquitecturales. Refinamiento del modelo del software en iteraciones. Refinamiento del modelo de requerimientos y del dominio. Refinamiento del modelo de diseño. Aplicación de patrones GRASP en nuevas iteraciones. Realización de casos de uso con patrones GoF. Diseño de la arquitectura lógica con patrones arquitecturales. Análisis arquitectural y SAD.
MÓDULO VIII: VERIFICACIÓN Y VALIDACIÓN DE SOFTWARE
El Proceso de verificación y validación. Técnicas de verificación y Validación. Depuración. Inspecciones del software. Prueba del software. Técnicas de prueba de software. Tipos de pruebas: estáticas y dinámicas. Pruebas de caja blanca. Pruebas de caja negra. Diseño de casos de prueba. Prueba de unidad. Prueba de integración. Tipos de pruebas de integración. Trazabilidad de requerimientos. Metodologías de desarrollo y pruebas. TDD (Desarrollo dirigido por pruebas). Estrategias de prueba de software. Enfoque estratégico. Estructura del plan de pruebas. Entrega del sistema. Evolución del software. Los problemas del mantenimiento. Técnicas y herramientas para el mantenimiento

5. RECURSOS NECESARIOS

- Proyector
- Pc
- Laboratorio Informatica
- Los Recursos Especificados No Corresponden Al Dictado 2020, Dado Que La Modalidad De Dictado Será Presencial-en Línea.

6. PROGRAMACIÓN SEMANAL

Semana	Unidad / Módulo	Descripción	Bibliografía
1	I	Ing. de Software Conceptos de Ing. de Software. El Producto. Proceso de desarrollo y Ciclos de vida	Sommerville, Pressman, Ghezzi, Schach

2	II	Ing. de requerimientos. Conceptos de Ing. de requerimientos. Elicitación y especificación de requerimientos. Técnicas.	Sommerville (99),Pressman, Braude
3	III	Modelos y Herramientas del Análisis. Metodologías para modelar el sistema.	Kendall & Kendall, Sommerville, Lano, Weilkens
4	III	Modelos del contexto y de interacciones UML Modelos de estructuras y comportamiento UML	Fowler, Booch (2007), Sommerville
5	IV	Diseño de Software Arquitectura de Software	Martin, Bass, Fowler (2002), Booch (2007)
6	IV	Diseño del sistema con UML Parcial práctico Nº 1. Parcial teórico Nº 1.	Fowler (2004),, Booch (2006)
7	V	Desarrollo de software. El Proceso de Desarrollo de Software RUP Agil. Características, Modelos y Fases.	Larman, Jacobson, Sommerville
8	VI	Fase de Inicio. Modelado de los requerimientos Fase de Elaboración, it. 1: Modelado del contexto	Larman, Jacobson
9	VI	Modelado de estructura y dinámica. Asignación de responsabilidades con patrones GRASP	Larman, Jacobson
10	VI	Modelado de diseño y su implementación Modelado arquitectural del sistema	Larman, Jacobson
11	VII	Diseño para el reuso Patrones GRASP y GoF	Larman, Jacobson
12	VII	Fase Elaboración, it.2: Aplicación de patrones para refinar el diseño Parcial práctico Nº 2. Parcial teórico Nº 2	Larman, Jacobson
13	VII	Refinamiento del Modelo de requerimientos y del dominio	Larman, Jacobson
14	VII	Refinamiento de la arquitectura lógica	Larman, Jacobson
15	VII	Análisis arquitectural	Larman, Jacobson
16	VIII	Verificación y validación del software Presentación trabajo integrador	Sommerville

7. BIBLIOGRAFIA DE LA ASIGNATURA

Autor	Año	Título	Capítulo/s	Lugar de la Edición	Editor / Sitio Web
Roger S. Pressman	2005	Ingeniería de Software - Un enfoque práctico. 6a. Ed.	1-2		Mc Graw Hill
Carlo Ghezzi, et al	2000	Fundamentals of Software Engineering	1-2		Prentice Hall
Sommerville I., 9a ed	2011	Ingeniería del software	1-2-3-4-5		Addison Wesley
Schach S.	2006	Ingeniería de software clásica y orientada a objetos. 6a. ed.	1-2		Mc Graw Hill

Braude	2007	Ingeniería del software. Una perspectiva OO	1-2-3-5		Alfaomega
Sommerville I, Sawyer	1999	Requirements Engineering	1-2-3-4-5		Wiley & Sons
Booch, Rumbaugh, Jacobson	2007	Object oriented analysis and design with applications: 3a ed			Pearson ed.
Kendall & Kendall	2005	Análisis y diseño de sistemas. 6a ed.	1-2-3-4-5		Pearson ed.
Fowler M., Scott K.	2004	UML Gota a gota			Prentice Hall
Larman C.	2003	UML y Patrones. 2a ed.	1-2-3-4-5-6-7-8-9-10		Prentice Hall
Jacobson, Booch, Rumbaugh	2000	El Proceso unificado de desarrollo de software			Addison Wesley
Booch, Rumbaugh, Jacobson	2006	El Lenguaje UML. 2a ed.			Pearson ed.
Lano K.	2005	Advanced Systems Design with Java, UML and MDA			Elsevier ltd.
Weilkiens T.	2008	Systems Engineering with SysML/UML			The MK/OMG Press
Martin R y otros	2018	Clean Architecture			Prentice Hall
Bass, Clemens, Kazman	2013	Software Architecture in practice 3a ed.			Addison Wesley
Fowler M.	2002	Patterns of Enterprese Application Architecture			

Firma del docente-investigador responsable

VISADO		
COORDINADOR DE LA CARRERA	**DIRECTOR DEL INSTITUTO**	**SECRETARIO ACADEMICO UNTDF**
Fecha :	Fecha :	

Este programa de estudio tiene una validez de hasta tres años o hasta que otro programa lo reemplace en ese periodo