



**Programa de
TALLER INTRODUCTORIO DE INGENIERÍA DE SOFTWARE**

1. NOMBRE DE LA UNIDAD CURRICULAR

Taller introductorio de ingeniería de software

2. CRÉDITOS

10 créditos

3. OBJETIVOS DE LA UNIDAD CURRICULAR

La ingeniería de software es la disciplina relacionada a la aplicación de teoría, conocimiento y práctica para construir de forma efectiva y eficiente sistemas de software confiables que satisfagan los requisitos de los clientes y usuarios. Aplica a sistemas pequeños, medianos y grandes y acompaña todas las fases del ciclo de vida del software, desde su concepción hasta su operación por parte de los usuarios.

El objetivo de esta unidad curricular es brindar un panorama general de la disciplina a los estudiantes profundizando en los aspectos más relevantes. Busca introducir al estudiante en el uso de algunas técnicas y herramientas particulares para que pueda participar en distintos roles en un proyecto de ingeniería de software.

4. METODOLOGÍA DE ENSEÑANZA

Esta unidad curricular tiene una fuerte componente de taller. La parte de taller será llevada adelante por grupos de estudiantes. Cada grupo tendrá un tutor con el que se reunirá al menos cada dos semanas. En el taller se realizarán prácticas y ejercicios que buscan que los estudiantes incorporen técnicas y habilidades básicas de la ingeniería de software.

El contenido teórico deberá ser estudiado de forma autónoma por los estudiantes. El material disponible será libro de curso, videos especialmente realizados para este curso, notas de teórico y artículos, entre otros.

Se dictarán clases de presentación del curso, presentación del taller, introducción a la ingeniería de software, y además, al menos cada dos semanas, cada grupo tendrá clase con su tutor asignado. En esas clases, con un grupo reducido de estudiantes, se trabajará



en temas del taller así como también se podrán plantear dudas del teórico estudiado.

Cada semana los estudiantes deberán avanzar en los temas teóricos pautados para esa semana y realizarán trabajos prácticos y de taller en grupos. Los grupos de estudiantes también deberán realizar presentaciones finales de las entregas del taller.

Todo el curso, y en particular el taller, buscan que el estudiante incorpore elementos prácticos de las actividades más fundamentales de la disciplina (a modo de ejemplo, llevar a la práctica, con alguna dinámica, la forma de recolectar requisitos, aprender a escribir un caso de uso o historia de usuario, entender un diagrama de componentes de la arquitectura de software, realizar un plan de pruebas de un sistema).

Fuera de clase los estudiantes deben: estudiar el material teórico, realizar ejercicios en grupos que serán discutidos en las clases de taller, rendir las evaluaciones.

Carga horaria:

- Estudio teórico – 4 horas semanales.
- Trabajo en el taller, clases de taller, repaso del teórico, ejercicios prácticos, otros – 6 horas semanales.

5. TEMARIO

1. **Introducción a la ingeniería de software:**

Subtemas: Descripción del desarrollo profesional de software, descripción de la ingeniería de software, ética en la ingeniería de software y algunos casos de estudio que permitan introducir los problemas particulares a tratar en el resto del curso.

2. **Procesos de software:** Un proceso de software es un conjunto coherente de actividades para la producción de software.

Subtemas: Descripción de los modelos de procesos de software: modelo en cascada, desarrollo incremental e integración y configuración (reuso de software). Descripción de la actividades del proceso: especificación del software, diseño e implementación, validación y evolución. Cómo hacer frente al cambio: prototipado, entrega incremental. Mejora del proceso. Desarrollo ágil de software: descripción de los métodos ágiles, técnicas ágiles de desarrollo, gestión ágil de proyectos, escalando los métodos ágiles.

3. **Ingeniería de requisitos:** Descripción de las funciones o servicios que el sistema proveerá y de las posibles restricciones.

Subtemas: Requerimientos funcionales y no funcionales. Proceso de ingeniería de requisitos. Obtención de requisitos: técnicas, historias y escenarios.



Especificación de requisitos. Validación de requisitos. Cambio en los requisitos. Modelado del sistema: modelos de contexto, modelos de interacción, modelos estructurales, modelos de comportamiento, arquitectura guiada por modelos.

4. **Diseño de software:** Descripción de la organización general del sistema y diseño de sus componentes e interacciones.
Subtemas: Diseño de la arquitectura: decisiones, vistas, patrones y arquitecturas de aplicaciones. Diseño orientado a objetos usando UML. Patrones de diseño.
5. **Construcción de software:** Construcción de la solución de software.
Subtemas: Código robusto mediante manejo de excepciones, prácticas de programación defensiva, estrategias de integración, implementación de cambios.
6. **Verificación y validación de software:** Su objetivo es mostrar que el software cumple con su especificación, encontrar comportamiento no esperado del software y apoyar en la validación de su adecuación al uso.
Subtemas: Calidad del software. Revisiones e inspecciones. Descripción de pruebas durante el desarrollo, desarrollo guiado por las pruebas, pruebas de liberación y pruebas de usuario.
7. **Evolución del software:** Descripción de los cambios que sufre el software luego de su liberación.
Subtemas: Proceso de evolución. Sistemas legados. Mantenimiento de software.
8. **Gestión de proyectos de software:** Descripción de las principales actividades en la gestión de proyectos de software y aplicación de algunas técnicas específicas.
Subtemas: Gestión de riesgos del proyecto: identificación, análisis, planificación y monitoreo. Planificación y seguimiento del proyecto: precios del software, desarrollo basado en planes, cronograma del proyecto, planificación ágil, técnicas de estimación.
9. **Herramientas y ambientes:** Herramientas que se utilizan y su contribución a la ingeniería de software.
Subtemas: Gestión de la configuración del software y control de versiones. Gestión de liberaciones. Herramientas de modelado de análisis de requisitos y diseño. Herramientas para pruebas (dinámicas y estáticas). Herramientas para armado de versiones.

6. BIBLIOGRAFÍA



| Tema | Básica | Complementaria |
|--|-------------------------|----------------|
| Introducción a la ingeniería de software | (1) – Capítulo 1 | |
| Procesos de software | (1) – Capítulos 2 y 3 | |
| Ingeniería de requisitos | (1) – Capítulos 4 y 5 | |
| Diseño de software | (1) – Capítulos 6 y 7 | |
| Construcción de software | (1) – Capítulo 7 | |
| Verificación y validación de software | (1) – Capítulos 8 y 24 | |
| Evolución del software | (1) – Capítulo 9 | |
| Gestión de proyectos de software | (1) – Capítulos 22 y 23 | |
| Herramientas y ambientes | (1) – Capítulo 25 | |

6.1 Básica

1. Sommerville, Ian (2015). Software Engineering (10th Edition). USA: Pearson.

6.2 Complementaria

N/A

7. CONOCIMIENTOS PREVIOS EXIGIDOS Y RECOMENDADOS

7.1 Conocimientos Previos Exigidos: Conocimientos teóricos y prácticos de la programación imperativa, orientación a objetos, y de las bases de datos.

7.2 Conocimientos Previos Recomendados: No tiene



ANEXO A

Para todas las Carreras

A1) INSTITUTO

Instituto de Computación

A2) CRONOGRAMA TENTATIVO

| | |
|-----------|---|
| Semana 1 | Tema Introducción a ingeniería de software (4hs de clase). |
| Semana 2 | Tema Procesos de software (4hs de clase). |
| Semana 3 | Tema Procesos de software (2hs de clase). Tema Ingeniería de requisitos (2hs de clase). |
| Semana 4 | Tema Ingeniería de requisitos (4hs de clase). |
| Semana 5 | Tema Ingeniería de requisitos (2hs de clase). Tema Diseño de software (2hs de clase). |
| Semana 6 | Tema Diseño de software (4hs de clase). |
| Semana 7 | Tema Construcción de software (4hs de clase). |
| Semana 8 | Tema Verificación y validación (4hs de clase). |
| Semana 9 | Tema Verificación y validación (4hs de clase). |
| Semana 10 | Tema Verificación y validación (2hs de clase). Tema Evolución de software (2hs de clase). |
| Semana 11 | Tema Evolución de software (2hs de clase). Tema Gestión de proyectos de software (2hs de clase). |
| Semana 12 | Tema Gestión de proyectos de software (4hs de clase). |
| Semana 13 | Tema Gestión de proyectos de software (2hs de clase). Tema Herramientas y ambientes (2hs de clase). |
| Semana 14 | Tema Herramientas y ambientes (2hs de clase). |
| Semana 15 | Repaso |

A3) MODALIDAD DEL CURSO Y PROCEDIMIENTO DE EVALUACIÓN

De forma de lograr un aprendizaje continuo, acompañado de una evaluación continua de los conocimientos y habilidades adquiridas, el curso se basa en actividades de taller periódicas, en cuatro instancias de evaluación individuales distribuidas coherentemente a lo largo del curso (la última de estas instancias es al final del curso y es una evaluación integradora de conocimientos). Las instancias de evaluación individual buscan evaluar conceptos teóricos y prácticos mediante preguntas teóricas y pequeños ejercicios prácticos.



Distribución de puntajes según instancias de evaluación:

| | |
|-----------------------------------|-------------|
| Evaluación individual 1 | : 10 puntos |
| Evaluación individual 2 | : 10 puntos |
| Evaluación individual 3 | : 10 puntos |
| Taller | : 50 puntos |
| Evaluación individual integradora | : 20 puntos |

El taller se evalúa mediante entregas de ejercicios prácticos y presentaciones de los estudiantes. Para aprobarlo, se deberá tener un mínimo del 60% del puntaje del taller.

Escala de evaluación:

Exoneración: Para exonerar el curso se deben cumplir las siguientes condiciones:

- Obtener al menos 60 puntos entre todas las evaluaciones.
- Obtener al menos 18 puntos en la suma de las cuatro evaluaciones individuales (evaluación individual 1-3 y evaluación individual integradora).
- Aprobar el Taller, es decir, obtener al menos 30 puntos en el Taller.

Reprobación: Si no se cumple alguna de las condiciones anteriores se reprueba el curso.

Este curso no tiene examen.

A4) CALIDAD DE LIBRE

Los estudiantes no podrán acceder a la Calidad de Libre.

A5) CUPOS DE LA UNIDAD CURRICULAR

Sin cupo.