

## INGENIERÍA DE SOFTWARE

<b>CLAVE:</b>		<b>SECTOR :</b>	OPTATIVO
<b>SEMESTRE:</b>	6 - 8	<b>ÁREA:</b>	INFORMÁTICA
<b>CRÉDITOS:</b>	10	<b>SERIACIÓN:</b>	
		ASIGNATURA PRECEDENTE INDICATIVA: Materias del sector básico del Área de Informática	
		ASIGNATURA SUBSECUENTE INDICATIVA: Ninguna	
<b>HORAS POR CLASE</b>		<b>TEÓRICA:</b>	1
<b>CLASES POR SEMANA</b>		<b>TEÓRICA:</b>	4
<b>HORAS POR SEMESTRE</b>		<b>TEÓRICA:</b>	64
		<b>PRÁCTICAS:</b>	2
		<b>PRÁCTICAS:</b>	1
		<b>PRÁCTICAS:</b>	32

**Objetivos generales:** Al finalizar el curso el alumno:

- Este curso presenta un estudio profundo de muchos de los temas de ingeniería de software, incluyendo especificación de requisitos, diseño, verificación y validación entre otros.

### **Tema 1. Especificación de requisitos de software teóricas**

**12 horas**

**6 horas prácticas**

Conocerá el desarrollo de especificaciones formales e informales dirigidas a definir las necesidades y requisitos del software.

- 1.1 El uso de las especificaciones.
- 1.2 Cualidades de las especificaciones.
- 1.3 Clasificación de estilos de especificaciones.
- 1.4 Especificaciones y verificación.
- 1.5 Especificaciones operativas.
- 1.6 Especificaciones descriptivas.
- 1.7 Construcción y uso de especificaciones en la práctica.

### **Tema 2. Diseño de software teóricas**

**12 horas**

**6 horas prácticas**

Identificará las distintas metodologías que existen para el diseño de software, evaluando sus ventajas y desventajas.

- 2.1 La actividad de diseño y sus objetivos.
- 2.2 Técnicas para la modularización.
- 2.3 Diseño funcional orientado al proceso.
- 2.4 Diseño desde abajo y apoyo para la reutilización.
- 2.5 Estrategias de implementación (desde arriba, desde abajo, desarrollo de equipos).
- 2.6 Tópicos de implementación; mejoría en el desempeño, depuración y prevención de problemas.
- 2.7 Diseño orientado a objetos.
- 2.8 Manejo de anomalías.



**Tema 3. Programación  
teóricas**

**18 horas**

**9 horas prácticas**

Adquirirá los conceptos y las habilidades de programación para elaborar software.

- 3.1 Construcción de software confiable.
- 3.2 Desarrollo de software reutilizable.
- 3.3 Ingeniería de software asistida por computadora (CASE).
- 4.4 Ambientes de desarrollo de software.

**Tema 4. Verificación y validación  
teóricas**

**14 horas**

**7 horas prácticas**

Comprenderá los métodos y técnicas para verificación y validación de sistemas de software.

- 4.1 Metas y requisitos para la verificación.
- 4.2 Enfoques para la verificación.
- 4.3 Pruebas
  - Generación de un plan de pruebas.
  - Aceptación de las pruebas.
  - Pruebas por unidad.
  - Pruebas por integración.
  - Pruebas de regresión.
- 4.4 Análisis. Técnicas informales vs pruebas de correctez.
- 4.5 Ejecución simbólica.
- 4.6 Depuración.
- 4.7 Otros aspectos: desempeño, robustez, métricas del software.

**Tema 5. Administración del desarrollo de software  
teóricas**

**8 horas**

**4 horas prácticas**

Distinguirá los principios fundamentales necesarios para la administración del desarrollo de software.

- 5.1 Planeación de proyectos y calendarización.
- 5.2 Estimación de costos.
- 5.3 Mantenimiento de software.
- 5.4 Control de cambios.
- 5.5 Documentación.
- 5.6 Aseguramiento de la calidad.

**Bibliografía básica:**

- Ghezzi, C. *et al. Fundamentals of Software Engineering.* (s. l.). Prentice-Hall. 1991.
- Ledgard, H. and J. Tauer. *Professional Software. Software Engineering Concepts*, vol I. (s. l.) Addison Wesley Publishing Company. 1987.

**Bibliografía complementaria:**

- Gilb, T. *Principles of Software Engineering Management.* (s. l.). Addison Wesley Publishing Company. 1988.

- Glass, R. L. *Software Conflict. Essays on the Art and Science of Software Engineering.* (s. l.) Yourdon Press Computing Series. 1991.
- Hrieta, J. and K. Jackson. *Computer Security Solutions.* (s. l.) Blackwell Scientific Publications. 1990.
- Naur, P. *Computing: A Human Activity.* (s. l.) ACM Press, Addison Wesley Publishing Company. 1992.
- Schulmeyer, G. G. *Zero Defect Software.* (s. l.) McGraw-Hill Inc. 1990. (Software Engineering Serier).
- Thimbleby, H. *User Interface Design.* (s. l.) ACM Press, Addison Wesley Publishing Company. 1990.
- Nutt, G. J. *Centralized and Distributed Operating Systems.* (s. l.) Prentice Hall. 1991.
- Klerer, M. *Design of Very High Level Computer Languages. A user-oriented approach.* (s. l.) 2<sup>nd</sup> edition. McGraw-Hill Inc. 1991.
- Tracz, W. (de.) *Software Reuse-Emerging Technology.* (s. l.) IEEE Computer Society Press. 1988.

#### **Sugerencias didácticas:**

Se recomiendan tareas regulares en las cuales el alumno aplique el material visto en clase y esté obligado a revisar diversas fuentes bibliográficas para que amplíe sus conocimientos con diferentes enfoques.

Asimismo se sugieren prácticas de cómputo para la experimentación con los algoritmos vistos en clase y el análisis de casos prácticos.

#### **Forma de evaluación:**

Se recomiendan de 3 a 4 exámenes parciales y un examen final, así como la realización de tareas sobre los temas vistos en clase para reforzar los conocimientos teóricos adquiridos.

#### **Perfil profesiográfico:**

El profesor que imparta el curso deberá ser egresado de las carreras de Actuaría, Matemáticas o alguna afín, de preferencia tener un postgrado, y deberá tener experiencia docente en el área o en las aplicaciones de la ingeniería de software, incluyendo especificación de requisitos, diseño, verificación y validación, entre otros.