UNIVERSIDAD DEL BÍO-BÍO
La Libertad del Conocimiento

UNIVERSIDAD DEL BÍO-BÍO
VICERRECTORÍA ACADÉMICA – DIRECCION DE DOCENCIA

PROGRAMA DE ASIGNATURA

I. IDENTIFICACIÓN

Nombre asignatura: Ingeniería de Software	**Período de Vigencia**: 2018-2019
Código: 634187	
Tipo de Curso: Obligatorio/ Formación de especialidad	

Carrera: **Ingeniería Civil en Informática**	**Departamento**: Sistemas de Información, Ciencias de la Computación y Tecnologías de Información	**Facultad**: Ciencias Empresariales
Nº Créditos SCT: 5	**Total horas: 13** **Cronológicas: 162** **Pedagógicas: 234**	**Año/ semestre** 4to año, 2do semestre

Horas presenciales: **HT: 04** **HP: 02** **HL: 00**	**Horas trabajo autónomo:** **HT: 04** **HP: 03** **HL: 00**
Prerrequisitos: Asignatura: Base de Datos Código: 634180	**Correquisitos:** Asignatura: Código:

II.- DESCRIPCIÓN

II.1 Presentación: Relación de la Asignatura con las Competencias del Perfil de Egreso

Asignatura teórico-práctica que, al finalizar, permitirá al estudiante construir software usando diversos elementos propios de la Ingeniería de Software. Se ubica en el cuarto año, segundo semestre, de la carrera.

Contribuye a las competencias específicas de:

- CE2.1: Desarrollar software efectivo y eficiente, para diversos dominios, siguiendo un enfoque de ingeniería.

Además, la asignatura contribuye al desarrollo de las competencias del Perfil Genérico de la Universidad del Bío-Bío:

- CG3: Responsabilidad Social: Establecer relaciones dialogantes para el intercambio de aportes constructivos con otras disciplinas y actúa éticamente en su profesión, trabajando de manera asociativa en la consecución de objetivos.

II.2 Descriptor de competencias (metas de la asignatura)

Construir software utilizando un conjunto de métodos, técnicas, herramientas y estándares que permitan obtener un producto que cumpla con los requerimientos técnicos, funcionales y no funcionales.

Resultados de Aprendizaje:

- Analiza la aplicabilidad de los métodos de desarrollo de software para un proyecto considerando el contexto y riesgos de éste (aspectos relacionados con el personal, producto y proceso) y las restricciones en las que cada método es más o menos apropiado de aplicar para reducir los riesgos asociados a la ejecución del proyecto.
- Aplica de manera sistemática métodos, técnicas, herramientas y estándares de Requerimientos, Análisis y Diseño en el desarrollo de un software que satisfaga los requerimientos técnicos, funcionales y no–funcionales, para generar software de calidad.
- Aplica técnicas de prueba de software para comprobar que un software funciona correctamente y satisface los requerimientos funcionales.

II.3 Aprendizajes Previos

- Aplica técnicas de modelamiento de datos y funcionalidad.
- Construye software utilizando lenguajes de programación.
- Construye bases de datos que respondan a un conjunto establecido de necesidades.
- Reconoce el contexto de negocios donde se insertan las aplicaciones de software.

III. RESULTADOS DE APRENDIZAJE

Resultados de Aprendizaje	Metodología	Criterios de Evaluación	Contenidos conceptuales, procedimentales y actitudinales	Tiempo estimado
Analiza la aplicabilidad de los métodos de desarrollo de software para un proyecto considerando el contexto y riesgos de este (aspectos relacionados con el personal, producto y proceso) y las restricciones en las que cada método es más o menos apropiado de aplicar para reducir los riesgos asociados a la ejecución del proyecto.	-Clase expositiva con discusión socializada. -Trabajo colaborativo.	1.1 Contrasta métodos de desarrollo de software tradicionales y modernos, identificando fortalezas y debilidades según el tipo de proyecto. 1.2 Selecciona el método de desarrollo de software más apropiado para un proyecto basado en las fortalezas y debilidades del método y el contexto y riesgos de un proyecto. 1.3 Explica con argumentos claros las razones que justifican el método seleccionado para un proyecto dado.	Conceptual -Conceptos básicos -Proceso de desarrollo de software, producto software y modelo de calidad. -Métodos de desarrollo de software tradicionales y modernos. -Ingeniería de software ágil. Procedimental -Establecimiento de criterios de comparación. Actitudinal -Objetividad y rigurosidad en la comparación y selección de métodos. -Pensamiento crítico en la argumentación para justificar elecciones realizadas.	Horas presenciales: HT: 24 HP: 12 HL: 00 Horas de trabajo autónomo: HT: 24 HP: 18 HL: 00
Aplica de manera sistemática métodos, técnicas, herramientas y estándares de Requerimientos, Análisis y Diseño en el desarrollo de un software que satisfaga los requerimientos técnicos, funcionales y no funcionales.	-Clase expositiva con discusión socializada -Trabajo colaborativo -Aprendizaje orientado a proyecto	2.1 Emplea algunos métodos y técnicas de ingeniería de requerimientos, tales como casos de uso, entrevistas, cuestionarios, entre otros. 2.2Utiliza un estándar de documentación de producto software en cada etapa del desarrollo. 2.3Utiliza un método y las técnicas asociadas de análisis y diseño en el desarrollo de	Conceptual -Ingeniería de requisitos. -Análisis y diseño de software. -Arquitectura de software (microservicios). -Ingeniería de software en la nube. -Estándares de calidad de Software -Seguridad e Ing.sw. -Confiabilidad e Ing.sw Procedimental -Aplicación de un método de desarrollo de software.	Horas presenciales: HT: 28 HP: 14 HL: 00 Horas de trabajo autónomo: HT: 28 HP: 21 HL: 00

		un software. 2.4 Utiliza un estándar de calidad para el desarrollo de producto de software.	-Uso de técnicas de requerimientos, análisis y diseño. -Uso de herramientas CASE. -Uso de estándares. Actitudinal -Rigurosidad en el uso de métodos, técnicas, herramientas y estándares de Requerimientos, Análisis y Diseño en el desarrollo de un software.	
Aplica técnicas de prueba de software para comprobar que un software funciona correctamente y satisface los requerimientos funcionales.	-Clase expositiva con discusión socializada -Trabajo individual y colaborativo -Aprendizaje basado en proyecto	3.1 Describe el proceso y niveles de prueba de software, así como diferentes enfoques y técnicas existentes tales como partición equivalente, valor límite, caminos de prueba, entre otros. 3.2 Utiliza diferentes técnicas para definir datos y casos de prueba a nivel de unidad, sistema y aceptación. 3.3 Ejecuta los casos de pruebas en el software para comprobar que funciona correctamente y satisface los requerimientos funcionales.	Conceptual -Proceso y niveles de prueba de software. -Enfoques y técnicas de prueba de software. -DevOps y gestión de versiones. Procedimental -Utilización de técnicas de prueba en la definición de datos y casos de prueba. -Ejecución de los casos de pruebas en el software. Actitudinal -Pensamiento crítico para determinar los enfoques y técnicas de prueba a aplicar. -Rigurosidad en la definición y ejecución de casos de prueba.	Horas presenciales: HT: 20 HP: 10 HL: 00 Horas de trabajo autónomo: HT: 20 HP: 15 HL: 00

IV. SISTEMA DE EVALUACIÓN

RESULTADOS DE APRENDIZAJE	EVIDENCIAS DE APRENDIZAJE (proceso y producto)
1. Analiza la aplicabilidad de los métodos de desarrollo de software para un proyecto considerando el contexto y riegos de éste (aspectos relacionados con el personal, producto y proceso) y las restricciones en las que cada método es más o menos apropiado de aplicar para reducir los riesgos asociados a la ejecución del proyecto.	• Tests/Informes de Trabajos
2. Aplica de manera sistemática métodos, técnicas, herramientas y estándares de Requerimientos, Análisis y Diseño en el desarrollo de un software que satisfaga los requerimientos técnicos, funcionales y no funcionales.	• Portafolio virtual del proyecto de desarrollo de software. • Tests/Informes de Trabajos • Certamen (RA1, RA2)
3. Aplica técnicas de prueba de software para comprobar que un software funciona correctamente y satisface los requerimientos funcionales.	• Portafolio virtual del proyecto de desarrollo de software. • Tests/Informes de Trabajos • Certamen (RA1, RA2 y RA3)

La evaluación de la asignatura considera: **(%)**

Certámenes	40%
Test/Trabajos individuales y/o colaborativos	20%
Proyecto de desarrollo de software:	40%
- Portafolio virtual (evidencia de avance, evaluación continua)	
- Producto de software	
- Informe	
- Exposición	

En los siguientes casos el estudiante obtendrá la condición NCR:

- **Menos de 80% de asistencia a las clases prácticas.**
- **No cumplir con las entregas del proyecto en las fechas establecidas.**

V. BIBLIOGRAFÍA

Fundamental

- Sommerville, R. (2019). *Engineering Software Products: An Introduction to Modern Software Engineering*: Pearson.
- Sommerville, I. (2016). Software Engineering (10ª ed.): Addison Wesley

Complementaria

- Bohem, B. (2006). *A View of 20th and 21st Century Software Engineering*. The International Conference on Software Engineering (ICSE), Shanghai, China.
- Larman, C. (2003). UML y Patrones. Una Introducción al Análisis y Diseño Orientado a Objetos y al Proceso Unificado (2ª ed.): Pearson.
- Rumbaugh, J., Booch, G. y Jacobson, I. (2007). Lenguaje Unificado de Modelado: Pearson.
- Software Engineering Institute (SEI). Carnegie Mellon University. Sitio web http://www.sei.cmu.edu/.
- The Object Management Group (OMG). Sitio web http://www.omg.org/.
- Alaimo, M. "Proyectos ágiles con Scrum." Argentina: Kleer (2013).