



## Universidad Nacional de Ucayali (UNU)

Programa Profesional de  
Ciencia de la Computación  
Sílabo 2023-I

### 1. CURSO

CS292. Ingeniería de Software II (Obligatorio)

### 2. INFORMACIÓN GENERAL

2.1 Créditos	:	4
2.2 Horas de teoría	:	2 (Semanal)
2.3 Horas de práctica	:	2 (Semanal)
2.4 Duración del periodo	:	16 semanas
2.5 Condición	:	Obligatorio
2.6 Modalidad	:	Híbrido
2.7 Prerrequisitos	:	CS291. Ingeniería de Software I. (5 <sup>to</sup> Sem)

### 3. PROFESORES

Atención previa coordinación con el profesor

### 4. INTRODUCCIÓN AL CURSO

Los tópicos de este curso extienden las ideas del diseño y desarrollo de software desde la secuencia de introducción a la programación para abarcar los problemas encontrados en proyectos de gran escala. Es una visión más amplia y completa de la Ingeniería de Software apreciada desde un punto de vista de Proyectos.

### 5. OBJETIVOS

- Capacitar a los alumnos para formar parte y definir equipos de desarrollo de software que afronten problemas de envergadura real.
- Familiarizar a los alumnos con el proceso de administración de un proyecto de software de tal manera que sea capaz de crear, mejorar y utilizar herramientas y métricas que le permitan realizar la estimación y seguimiento de un proyecto de software.
- Crear, evaluar e implementar un plan de prueba para segmentos de código de tamaño medio , Distinguir entre los diferentes tipos de pruebas , sentar las bases para crear, mejorar los procedimientos de prueba y las herramientas utilizadas con ese propósito.
- Seleccionar con justificación un apropiado conjunto de herramientas para soportar el desarrollo de un rango de productos de software.
- Crear, mejorar y utilizar los patrones existentes para el mantenimiento de software . Dar a conocer las características y patrones de diseño para la reutilización de software.
- Identificar y discutir diferentes sistemas especializados , crear , mejorar y utilizar los patrones especializados para el diseño , implementación , mantenimiento y prueba de sistemas especializados

### 6. COMPETENCIAS

- b) Analizar problemas e identificar y definir los requerimientos computacionales apropiados para su solución. (**Usar**)
- c) Diseñar, implementar y evaluar un sistema, proceso, componente o programa computacional para alcanzar las necesidades deseadas. (**Usar**)
- f) Comunicarse efectivamente con audiencias diversas. (**Usar**)
- i) Utilizar técnicas y herramientas actuales necesarias para la práctica de la computación. (**Evaluar**)
- k) Aplicar los principios de desarrollo y diseño en la construcción de sistemas de software de complejidad variable. (**Usar**)

## 7. TEMAS

Unidad 1: Herramientas y Entornos (12)	
Competencias esperadas: c,f,i	
Temas	Objetivos de Aprendizaje
<ul style="list-style-type: none"> <li>• Administración de configuración de software y control de versiones.</li> <li>• Administración de despliegues.</li> <li>• Análisis de requerimientos y herramientas para modelado del diseño.</li> <li>• Herramientas de <i>testing</i> incluyendo herramientas de análisis estático y dinámico.</li> <li>• Entornos de programación que automatizan el proceso de construcción de partes de programa (ejem., construcciones automatizadas) <ul style="list-style-type: none"> <li>– Integración continua.</li> </ul> </li> <li>• Mecanismos y conceptos de herramientas de integración.</li> </ul>	<ul style="list-style-type: none"> <li>• Administración de configuración de software y control de versiones. [Usar]</li> <li>• Administración de despliegues. [Usar]</li> <li>• Análisis de requerimientos y herramientas para modelado del diseño. [Usar]</li> <li>• Herramientas de <i>testing</i> incluyendo herramientas de análisis estático y dinámico. [Usar]</li> <li>• Entornos de programación que automatizan el proceso de construcción de partes de programa (ejem., construcciones automatizadas) <ul style="list-style-type: none"> <li>– Integración continua.</li> </ul> </li> <li>[Usar]</li> <li>• Mecanismos y conceptos de herramientas de integración. [Usar]</li> </ul>
<b>Lecturas :</b> [Pre04], [Blu92], [Sch04], [WK00], [Key04], [WA02], [PS01], [Sch04], [Mon96], [Amb01], [Con00], [Oqu03]	

Unidad 2: Verificación y Validación de Software (12)	
Competencias esperadas: c,f,i	
Temas	Objetivos de Aprendizaje
<ul style="list-style-type: none"> <li>• Verificación y validación de conceptos.</li> <li>• Inspecciones, revisiones, auditorias.</li> <li>• Tipos de pruebas, incluyendo la interfaz humano computador, usabilidad, confiabilidad, seguridad, desempeño para la especificación.</li> <li>• Fundamentos de testeo: <ul style="list-style-type: none"> <li>– Pruebas de Unit, integración, validación y de Sistema</li> <li>– Creación de plan de pruebas y generación de casos de test</li> <li>– Técnicas de test de caja negra y caja blanca</li> <li>– Test de regresión y automatización de pruebas</li> </ul> </li> <li>• Seguimiento de defectos.</li> <li>• Limitaciones de testeo en dominios particulares, tales como sistemas paralelos o críticos en cuanto a seguridad.</li> <li>• Enfoques estáticos y enfoques dinámicos para la verificación.</li> <li>• Desarrollo basado en pruebas.</li> <li>• Plan de Validación, documentación para validación.</li> <li>• Pruebas Orientadas a Objetos, Sistema de Pruebas.</li> <li>• Verificación y validación de artefactos no codificados (documentación, archivos de ayuda, materiales de entrenamiento)</li> <li>• Logeo fallido, error crítico y apoyo técnico para dichas actividades.</li> <li>• Estimación fallida y terminación de las pruebas que incluye la envios por defecto.</li> </ul>	<ul style="list-style-type: none"> <li>• Distinguir entre la validación y verificación del programa [Usar]</li> <li>• Describir el papel que las herramientas pueden desempeñar en la validación de software [Usar]</li> <li>• Realizar, como parte de una actividad de equipo, una inspección de un segmento de código de tamaño medio [Usar]</li> <li>• Describir y distinguir entre diferentes tipos y niveles de pruebas (unitaria, integracion, sistemas y aceptacion) [Usar]</li> <li>• Describir tecnicas para identificar casos de prueba representativos para integracion, regresion y pruebas del sistema [Usar]</li> <li>• Crear y documentar un conjunto de pruebas para un segmento de código de mediano tamaño [Usar]</li> <li>• Describir cómo seleccionar buenas pruebas de regresión y automatizarlas [Usar]</li> <li>• Utilizar una herramienta de seguimiento de defectos para manejar defectos de software en un pequeño proyecto de software [Usar]</li> <li>• Discutir las limitaciones de las pruebas en un dominio particular [Usar]</li> <li>• Evaluar un banco de pruebas (<i>a test suite</i>) para un segmento de código de tamaño medio [Usar]</li> <li>• Comparar los enfoques estáticos y dinámicos para la verificación [Usar]</li> <li>• Identificar los principios fundamentales de los métodos de desarrollo basado en pruebas y explicar el papel de las pruebas automatizadas en estos métodos [Usar]</li> <li>• Discutir los temas relacionados con las pruebas de software orientado a objetos [Usar]</li> <li>• Describir las técnicas para la verificación y validación de los artefactos de no código [Usar]</li> <li>• Describir los enfoques para la estimación de fallos [Usar]</li> <li>• Estimar el número de fallos en una pequeña aplicación de software basada en la densidad de defectos y siembra de errores [Usar]</li> <li>• Realizar una inspección o revisión del de código fuente de un software para un proyecto de software de tamaño pequeño o mediano [Usar]</li> </ul>
<b>Lecturas :</b> [Pre04], [Blu92], [Sch04], [WK00], [Key04], [WA02], [PS01], [Sch04], [Mon96], [Amb01], [Con00], [Oqu03]	

Unidad 3: Evolución de Software (12)	
Competencias esperadas: c,f,i	
Temas	Objetivos de Aprendizaje
<ul style="list-style-type: none"> <li>• Desarrollo de Software en el contexto de código grande pre existente <ul style="list-style-type: none"> <li>– Cambios de software</li> <li>– Preocupaciones y ubicación de preocupaciones</li> <li>– <i>Refactoring</i></li> </ul> </li> <li>• Evolución de Software.</li> <li>• Características de Software mantenible.</li> <li>• Sistemas de Reingeniería.</li> <li>• Reuso de Software. <ul style="list-style-type: none"> <li>– Segmentos de código</li> <li>– Bibliotecas y <i>frameworks</i></li> <li>– Componentes</li> <li>– Líneas de Producto</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>• Identificar los problemas principales asociados con la evolución del software y explicar su impacto en el ciclo de vida del software [Usar]</li> <li>• Estimar el impacto del cambio de requerimientos en productos existentes de tamaño medio [Usar]</li> <li>• Usar refactorización en el proceso de modificación de un componente de software [Usar]</li> <li>• Estudiar los desafíos de mejorar sistemas en un entorno cambiante [Usar]</li> <li>• Perfilar los procesos de pruebas de regresión y su rol en el manejo de versiones [Usar]</li> <li>• Estudiar las ventajas y desventajas de diferentes tipos de niveles de confiabilidad [Usar]</li> </ul>
<b>Lecturas :</b> [Pre04], [Blu92], [Sch04], [WK00], [Key04], [WA02], [PS01], [Sch04], [Mon96], [Amb01], [Con00], [Oqu03]	

Unidad 4: Gestión de Proyectos de Software (12)	
Competencias esperadas: c,f,i	
Temas	Objetivos de Aprendizaje
<ul style="list-style-type: none"> <li>• La participación del equipo: <ul style="list-style-type: none"> <li>– Procesos elemento del equipo, incluyendo responsabilidades de tarea, la estructura de reuniones y horario de trabajo</li> <li>– Roles y responsabilidades en un equipo de software</li> <li>– Equipo de resolución de conflictos</li> <li>– Los riesgos asociados con los equipos virtuales (comunicación, la percepción, la estructura)</li> </ul> </li> <li>• Estimación de esfuerzo (a nivel personal)</li> <li>• Riesgo. <ul style="list-style-type: none"> <li>– El papel del riesgo en el ciclo de vida</li> <li>– Categorías elemento de riesgo, incluyendo la seguridad, la seguridad, mercado, finanzas, tecnología, las personas, la calidad, la estructura y el proceso de</li> </ul> </li> <li>• Gestión de equipos: <ul style="list-style-type: none"> <li>– Organización de equipo y la toma de decisiones</li> <li>– Roles de identificación y asignación</li> <li>– Individual y el desempeño del equipo de evaluación</li> </ul> </li> <li>• Gestión de proyectos: <ul style="list-style-type: none"> <li>– Programación y seguimiento de elementos</li> <li>– Herramientas de gestión de proyectos</li> <li>– Análisis de Costo/Beneficio</li> </ul> </li> <li>• Software de medición y técnicas de estimación.</li> <li>• Aseguramiento de la calidad del software y el rol de las mediciones.</li> <li>• Riesgo. <ul style="list-style-type: none"> <li>– Identificación de riesgos y gestión.</li> <li>– Análisis riesgo y evaluación.</li> <li>– La tolerancia al riesgo (por ejemplo, riesgo adverso, riesgo neutral, la búsqueda de riesgo)</li> <li>– Planificación de Riesgo</li> </ul> </li> <li>• En todo el sistema de aproximación al riesgo, incluyendo riesgos asociados con herramientas.</li> </ul>	<ul style="list-style-type: none"> <li>• Discutir los comportamientos comunes que contribuyen al buen funcionamiento de un equipo [Usar]</li> <li>• Crear y seguir un programa para una reunión del equipo [Usar]</li> <li>• Identificar y justificar las funciones necesarias en un equipo de desarrollo de software [Usar]</li> <li>• Entender las fuentes, obstáculos y beneficios potenciales de un conflicto de equipo [Usar]</li> <li>• Aplicar una estrategia de resolución de conflictos en un ambiente de equipo [Usar]</li> <li>• Utilizar un método ad hoc para estimar el esfuerzo de desarrollo del software (ejemplo, tiempo) y comparar con el esfuerzo actual requerido [Usar]</li> <li>• Listar varios ejemplos de los riesgos del software [Usar]</li> <li>• Describir el impacto del riesgo en el ciclo de vida de desarrollo de software [Usar]</li> <li>• Describir las diferentes categorías de riesgo en los sistemas de software [Usar]</li> <li>• Demostrar a través de la colaboración de proyectos de equipo los elementos centrales de la construcción de equipos y gestión de equipos [Usar]</li> </ul>
<b>Lecturas :</b> [Pre04], [Blu92], [Sch04], [WK00], [Key04], [WA02], [PS01], [Sch04], [Mon96], [Amb01], [Con00], [Oqu03]	

## 8. PLAN DE TRABAJO

### 8.1 Metodología

Se fomenta la participación individual y en equipo para exponer sus ideas, motivándolos con puntos adicionales en las diferentes etapas de la evaluación del curso.

### 8.2 Sesiones Teóricas

Las sesiones de teoría se llevan a cabo en clases magistrales donde se realizarán actividades que propicien un aprendizaje activo, con dinámicas que permitan a los estudiantes interiorizar los conceptos.

### 8.3 Sesiones Prácticas

Las sesiones prácticas se llevan en clase donde se desarrollan una serie de ejercicios y/o conceptos prácticos mediante planteamiento de problemas, la resolución de problemas, ejercicios puntuales y/o en contextos aplicativos.

## 9. SISTEMA DE EVALUACIÓN

\*\*\*\*\* EVALUATION MISSING \*\*\*\*\*

## 10. BIBLIOGRAFÍA BÁSICA

- [Amb01] Vincenzo Ambriola. *Software Process Technology*. Springer, July 2001.
- [Blu92] Bruce I. Blum. *Software Engineering: A Holistic View*. 7th. Oxford University Press US, May 1992.
- [Con00] R Conradi. *Software Process Technology*. Springer, Mar. 2000.
- [Key04] Jessica Keyes. *Software Configuration Management*. CRC Press, Feb. 2004.
- [Mon96] Carlo Montangero. *Software Process Technology*. Springer, Sept. 1996.
- [Oqu03] Flavio Oquendo. *Software Process Technology*. Springer, Sept. 2003.
- [Pre04] Roger S. Pressman. *Software Engineering: A Practitioner's Approach*. 6th. McGraw-Hill, Mar. 2004.
- [PS01] John W. Priest and Jose M. Sanchez. *Product Development and Design for Manufacturing*. Marcel Dekker, Jan. 2001.
- [Sch04] Stephen R Schach. *Object-Oriented and Classical Software Engineering*. McGraw-Hill, Jan. 2004.
- [WA02] Daniel R. Windle and L. Rene Abreo. *Software Requirements Using the Unified Process*. Prentice Hall, Aug. 2002.
- [WK00] Yingxu Wang and Graham King. *Software Engineering Processes: Principles and Applications*. CRC Press, Apr. 2000.