Datos Generales de la asignatura

Nombre de la asignatura:	**Proceso Personal para el Desarrollo de Software**
Clave de la asignatura:	**ISC-2601**
SATCA[1]:	**2-2-4**
Carrera:	**Ingeniería en Sistemas Computacionales**

2. Presentación

Caracterización de la asignatura
Esta asignatura está basada en una metodología proveniente del Instituto de Ingeniería del Software (SEI). PSP es una alternativa dirigida a los ingenieros en sistemas, que les permitirá mejorar la forma en la que construyen software. Considerando aspectos como la planeación, calidad, estimación de costos y productividad, se aplica para aumentar la calidad de los productos de software que se desarrollan. La metodología PSP se relaciona con las materias precedentes como Fundamentos de Ingeniería de Software e Ingeniería de Software, y posteriores como Modelo de Desarrollo Integral. Requiere de competencias previas como: Manejo del modelado orientado a objetos, un lenguaje para realizar el modelado, conocer y manejar los conceptos de estructuras de datos, dominio en el uso de herramientas CASE, dominio de algún lenguaje de programación orientada a objetos, identificación de las etapas del ciclo de desarrollo de sistemas.
Intención didáctica
La asignatura debe ser teórico- práctico para que desarrolle en el alumno la habilidad de prevenir errores al estimar los costos del software y obtener productos de calidad. En la unidad uno, identifica el proceso de TSP y PSP. En la unidad dos, implementa los métodos de estimación populares, estimación de costos y estimación de tamaño del software. En la unidad tres, aplica los principios básicos del método de estimación PROBE. En la unidad cuatro, implementa los principios básicos de planeación de tiempo y calendario. En la unidad cinco, integra el enfoque de calidad de PSP, los costos de la calidad, el proceso de comparación y estrategias de prevención y remoción. En la unidad seis, integra los principios de diseño de software, revisiones y verificación.

[1] Sistema de Asignación y Transferencia de Créditos Académicos

3. Participantes en el diseño y seguimiento curricular del programa

Lugar y fecha de elaboración o revisión	Participantes	Observaciones
Dirección General de Institutos Descentralizados, México D.F. Fecha: 12 al 14 de septiembre de 2012.	Representantes: Tecnológico de Estudios Superiores de Coacalco. Instituto Tecnológico Superior de Atlixco, Comalcalco, Fresnillo, Santiago Papasquiaro, Tepexi de Rodríguez, Zapopan.	Análisis y adecuación por competencias del módulo de la especialidad “Ingeniería de Software" de la carrera de Ingeniería en Sistemas Computacionales.
Tecnológico de Estudios Superiores de Coacalco Fecha: 10 al 14 de agosto de 2015.	Representantes: Tecnológico de Estudios Superiores de Coacalco.	Actualización del módulo de la especialidad “Ingeniería de Software" de la carrera de Ingeniería en Sistemas Computacionales
Tecnológico de Estudios Superiores de Coacalco Fecha: 21 al 22 de junio de 2019.	Representantes: Tecnológico de Estudios Superiores de Coacalco.	Actualización de asignaturas del módulo de la especialidad "Ingeniería de Software" de la carrera de Ingeniería en Sistemas Computacionales.
Tecnológico de Estudios Superiores de Coacalco Octubre del 2025.	Representantes: Docentes de la Academia de Ingeniería en Sistemas Computacionales del Tecnológico de Estudios Superiores de Coacalco.	Análisis, formulación de competencias, actualización de temario, formulación de actividades y prácticas de la especialidad de "Ingeniería de Software y Arquitectura de Sistemas” de la carrera de Ingeniería en Sistemas Computacionales.

4. Competencia(s) a desarrollar

Competencia(s) específica(s) de la asignatura
Implementa una metodología bajo estándares para generar una disciplina de trabajo personal.

5. Competencias previas

- Aplica técnicas de modelado para la solución de problemas.
- Conocimientos avanzados en un lenguaje de programación orientado a objetos para la solución de problemas
- Conocimiento en el desarrollo de software.
- Conocimiento del proceso de la ingeniería de requerimientos

6. Temario

No.	Temas	Subtemas
1	Introducción a TSP y PSP	1.1 Introducción a PSP y TSP. 1.2 Principios del PSP. 1.3 Objetivos del PSP 1.4 Flujo de Proceso PSP. 1.5 Proceso Personal de Software. 1.6 TSP. 1.7 CMM y PSP. 1.8 ventajas y desventajas del PSP 1.9 Costos
2	Planeación, cálculo y estimación de tamaño	2.1 Planeación de Proyectos. 2.2 Medición de tamaño. 2.2.1 Conteo de programas. 2.2.2 Estándar de conteo. 2.3 Estándar de Codificación. 2.4 Antecedentes de estimación. 2.5 Principios de estimación. 2.6 Métodos de estimación populares. 2.6.1 El Método FuzzyLogic. 2.6.2 Método de estimación por puntos de función. 2.6.3 Método del componente estándar. 2.6.4 Método Delphi. 2.6.5 Método de puntos de casos de uso. 2.6.6 Estimación basada en Proxies. 2.7 Método de estimación COCOMO
3	El método de estimación PROBE	3.1 El método de estimación PROBE. 3.2 Estimación del tamaño del programa. 3.3 Cálculo del intervalo de predicción. 3.3.1 Distribución normal con rangos de tamaño. 3.3.2 Distribución Log-Normal. 3.4 Método A. Regresión con LOC de objeto estimadas. 3.5 Método B. Regresión con LOC nuevas y cambiadas estimadas. 3.6 Método C. Método de promedios. 3.7 Método D: A juicio del Ingeniero. 3.8 Tutorial: Utilizando PROBE con el

		StudentWorkbook 3.9 Método disciplinado 3.10 Instalación de Dashboard 3.11 Uso básico del Process Dashboard
4	Planeación de tiempo y calendario	4.1 Necesidad de planes de tiempo y calendario. 4.2 El proceso de planeación de tiempo. 4.3 Estimando el calendario. 4.4 Valor ganado. 4.5 Seguimiento del proyecto. 4.6 Calculando la terminación del trabajo. 4.7 Cambios al plan.
5	Calidad y Diseño de software	5.1 El enfoque de calidad de PSP. 5.2 El costo de la calidad. 5.3 La estrategia de calidad. 5.4 Proceso de comparación. 5.5 Estrategias de remoción de defectos. 5.6 Estrategias de prevención de defectos. 5.7 El proceso de diseño. 5.8 Niveles de diseño. 5.9 Métodos y notaciones de diseño. 5.10 El proceso de diseño de PSP. 5.10.1 Plantilla de escenario operacional. 5.10.2 Plantilla de especificación funcional. 5.10.3 Plantilla de especificación de estados. 5.10.4 Plantilla de especificación lógica. 5.11 Jerarquía del diseño. 5.12 Usando UML.
6	Revisiones de diseño y código, y verificación de diseño	6.1 Inspecciones, guías y revisiones. 6.1.1 Estrategia de revisión de PSP. 6.1.2 Control de proceso. 6.1.3 Listas de verificación. 6.2 Revisiones de diseño y código. 6.3 Métodos para evaluar y mejorar la calidad de las revisiones. 6.4 Necesidades y beneficios de las revisiones de diseño. 6.5 Tópicos de verificación de diseño.

7. Actividades de aprendizaje de los temas

UNIDAD 1. Introducción al TSP y PSP	
Competencias	Actividades de aprendizaje
Específica(s): Identifica los principios de TSP y PSP así como su aplicación. Genéricas: • Capacidad de análisis y síntesis • Capacidad de organizar y planificar • Comunicación oral y escrita • Habilidad para buscar y analizar información proveniente de fuentes diversas • Solución de problemas	• Investiga en diferentes fuentes de información la definición y principios de los TSP y PSP. • Desarrolla un cuadro sinóptico de las características de TSP y PSP. • Discute en clase la importancia que tiene los PSP en una empresa
UNIDAD 2. Planeación, cálculo y estimación de tamaño.	
Competencias	Actividades de aprendizaje
Específica(s): Analiza y aplica los modelos de estimación en el proceso personal de desarrollo de software Genéricas: • Capacidad de análisis y síntesis • Capacidad de organizar y planificar • Habilidad para buscar y analizar información proveniente de fuentes diversas • Solución de problemas • Capacidad crítica y autocrítica • Trabajo en equipo • Habilidades interpersonales	• Desarrolla un cuadro comparativo sobre los modelos de estimación. • Discute, en equipo, las características que diferencian a los modelos de estimación en un desarrollo de software. • Aplica un modelo de estimación a un caso práctico.
UNIDAD 3. El método de estimación PROBE.	
Competencias	Actividades de aprendizaje
Específica(s):	• Expone las fases del método de estimación PROBE.

Aplicar las fases del método PROBE. Genéricas: • Solución de problemas • Capacidad crítica y autocrítica • Trabajo en equipo • Habilidades interpersonales	• Realiza el estudio del costo beneficio para evaluar la estimación PROBE desde el punto de vista económico. • Aplica modelo de estimación PROBE a un caso práctico.
UNIDAD 4. Planeación de tiempo y calendario.	
Competencias	Actividades de aprendizaje
Específica(s): Desarrolla plan de trabajo Genéricas: • Capacidad de aplicar los conocimientos en la práctica • Habilidades de investigación • Capacidad de aprender • Capacidad de generar nuevas ideas (creatividad).	• Investiga técnicas de planeación de proyectos. • Formula un plan de trabajo donde se estimen tiempos y recursos de un proyecto.
UNIDAD 5. Calidad y Diseño de Software.	
Competencias	Actividades de aprendizaje
Específica(s): Aplica las vistas de diseño de PSP Genéricas: • Capacidad de aplicar los conocimientos en la práctica • Habilidades de investigación • Capacidad de aprender • Capacidad de generar nuevas ideas (creatividad).	• Investiga y debate los escenarios de PSP. • Aplica métricas de calidad al proceso de diseño. • Investiga estrategias de prevención y remoción de defectos
UNIDAD 6. Revisiones de diseño y código, y verificación de diseño.	
Competencias	Actividades de aprendizaje
Específica(s):	Investigar las diferentes técnicas de revisión de código y diseño.

Aplica técnicas de revisión a código y diseño Genéricas: • Capacidad de aplicar los conocimientos en la práctica • Habilidades de investigación • Capacidad de aprender • Capacidad de generar nuevas ideas (creatividad).	• Implementar técnicas de revisión al código y diseño.

8. Práctica(s)

- Implementar la metodología PSP en un proyecto de desarrollo de software.
- Elaborar ejercicios de técnicas de estimación de software.
- Recabar métricas y aplicar estadísticas basadas en fórmulas de calidad de PSP.
- Recabar métricas y aplicar el método de estimación PROBE.

9. Proyecto de asignatura

El objetivo del proyecto que planteé el docente que imparta esta asignatura, es demostrar el desarrollo y alcance de la(s) competencia(s) de la asignatura, considerando las siguientes fases:

- **Fundamentación:** marco referencial (teórico, conceptual, contextual, legal) en el cual se fundamenta el proyecto de acuerdo con un diagnóstico realizado, mismo que permite a los estudiantes lograr la comprensión de la realidad o situación objeto de estudio para definir un proceso de intervención o hacer el diseño de un modelo.

- **Planeación:** con base en el diagnóstico en esta fase se realiza el diseño del proyecto por parte de los estudiantes con asesoría del docente; implica planificar un proceso: de intervención empresarial, social o comunitario, el diseño de un modelo, entre otros, según el tipo de proyecto, las actividades a realizar los recursos requeridos y el cronograma de trabajo.

- **Ejecución:** consiste en el desarrollo de la planeación del proyecto realizada por parte de los estudiantes con asesoría del docente, es decir en la intervención (social, empresarial), o construcción del modelo propuesto según el tipo de proyecto, es la fase de mayor duración que implica el desempeño de las competencias genéricas y especificas a desarrollar.

- **Evaluación:** es la fase final que aplica un juicio de valor en el contexto laboral-profesión, social e investigativo, ésta se debe realizar a través del reconocimiento de logros y aspectos a mejorar se estará promoviendo el concepto de “evaluación para la mejora continua”, la metacognición, el desarrollo del pensamiento crítico y reflexivo en los estudiantes.

10. Evaluación por competencias

Se recomienda evaluar los siguientes puntos:
• Evaluación Teórica.
• Prácticas de laboratorio.
• Desarrollo y evaluación de un proyecto de software.
La ponderación de la evaluación debe ser propuesta por la academia correspondiente de cada institución.

11. Fuentes de información

Humphrey, Watts S. (2005). A Discipline for Software Engineering, Addison-Wesley.
Humphrey, Watts S. (2013). Five Core Metrics, The intelligence behind successful software management, DH.
Humphrey, Watts S. (2005). Introduction to PSP, Addison-Wesley.
Castells, M. y otros. (2009). El desafío tecnológico. España y las nuevas tecnologías, Madrid, Alianza Editorial.
Jacobson, I., Booch, Gardy y Rumbaugh, J. (2008). El proceso unificado de Desarrollo de Software. Madrid: Addison Wesley.
Pomeroy-Huff, M., Cannon, R., Chick, T., Mullaney, J., Nichols, W. (2009). The Personal Software Process (PSP) Body of Knowledge, Version 2.0. Pittsburgh, Pennsylvania: Software Engineering Institute, Carnegie Mellon University. Consultado el 17 de noviembre de 2014. Freely downloadable Special Report CMU/SEI-2009-SR-018, 2009.
IEEE. (2017). *SWEBOK V3.0: Guide to the Software Engineering Body of Knowledge*. IEEE Computer Society.
ISO/IEC/IEEE. (2017). *ISO/IEC/IEEE 12207: Systems and software engineering — Software life cycle processes*. ISO.
Pressman, R. S., & Maxim, B. R. (2019). *Software Engineering: A Practitioner's Approach* (9th ed.). McGraw-Hill.