PROGRAMACIÓN ORIENTADA A OBJETO AVANZADA

Código: MIC 107	***Obligatoria***
Horas Módulo: 40 (cuarenta)	***Créditos: 2.6***
Área: Especializada (S-D)	***Correlativa:***
Horas teóricas: 50%	***Horas prácticas:50%***

1. FUNDAMENTACIÓN

La programación orientada a objetos es una forma de organizar el código de un programa agrupándolo en objetos, que son elementos individuales que contienen información (valores de datos) y funcionalidad. La utilización de un enfoque orientado a objetos para organizar un programa permite agrupar partes específicas de la información (por ejemplo, información de una canción como el título de álbum, el título de la pista o el nombre del artista) junto con funcionalidad o acciones comunes asociadas con dicha información (como "añadir pista a la lista de reproducción" o "reproducir todas las canciones de este artista"). Estos elementos se combinan en un solo elemento, denominado objeto (por ejemplo, un objeto "Album" o "MusicTrack").

Poder agrupar estos valores y funciones proporciona varias ventajas, como la capacidad de hacer un seguimiento de una sola variable en lugar de tener que controlar varias variables, agrupar funcionalidad relacionada y poder estructurar programas de maneras que reflejen mejor el mundo real.

2. CAPACIDADES

- Comprender y aplicar los conceptos avanzados de la Programación Orientada a Objetos desarrollando aplicaciones relativamente complejas utilizando este paradigma de programación, y particularmente los patrones de diseño como un ayuda para crear aplicaciones más robustas, escalables, fiables y fáciles de mantener.

- Desarrollar un producto software de calidad, reconociendo y aplicando los elementos fundamentales de un programa orientado a objetos (clase, objeto, atributos, métodos y mensajes), y las diferentes relaciones que se establecen entre ellos (composición, uso, asociación), utilizando los mecanismos de abstracción más apropiados y estableciendo responsabilidades y colaboraciones.

- Valorar la importancia de la reutilización, pilar fundamental de la POO, y aplicar utilizando los diferentes mecanismos que ofrece la POO, tales como polimorfismo, herencia, clases abstractas, y particularmente con el uso de patrones, para construir software robusto, escalable, fiable y fácil de mantener.

3. CONTENIDOS CURRICULARES

Análisis de Contexto. Crisis e Ingeniería del software. Calidad del Software. El modelo de objetos. Encapsulamiento y doble encapsulamiento. Mecanismos de abstracción. Conocimiento entre objetos. Polimorfismo. Clases Abstractas. Responsabilidades y Colaboraciones. Definición de los servicios de un objeto. El proceso de desarrollo. Modelar por contrato. Definición y refinamiento de contratos. Lenguaje de modelado. Principios del Diseño Orientado a Objetos. Reutilización de código: Delegación y Colaboración. Herencia y Composición. División de responsabilidades. Reutilización de diseño: Patrones. Clasificación. Criterios. Comparación. Ventajas y Desventajas del uso de Patrones.

.

4. ESTRATEGIAS DE ENSEÑANZA-APRENDIZAJE

Se realizarán clases teóricas expositivas del docente con el apoyo de diapositivas presentadas mediante proyector multimedia.
Los alumnos resolverán ejercicios prácticos orientados a captar y evaluar la comprensión de los conceptos relevantes de la exposición.
Para consolidar los conceptos y estimular la participación de los alumnos se propondrán actividades de desarrollo grupal que consisten en el análisis de artículos científicos breves relacionados con los temas del curso, propuestos por el docente, y la elaboración y exposición de una síntesis del trabajo realizado por parte de los alumnos.

5. CRITERIOS DE EVALUACIÓN Y APROBACIÓN

Considerando los distintos perfiles de los asistentes, se ofrecen diferentes modalidades para la aprobación: a) elaboración de un producto software que integre los conceptos impartidos en el curso, con énfasis en el diseño, y la justificación de las decisiones de diseño adoptadas, b) análisis y elaboración de informe síntesis de artículo científico sobre temas del curso seleccionado por el profesor, c) elaboración de una RSL (Revisión Sistemática de la Literatura) temas del curso.

7. BIBLIOGRAFÍA

Básica

- WEITZENFELD, Alfredo. 2005. Ingeniería de Software Orientada a Objetos con UML, Java e Internet. Editorial Thomson.
- LARMAN, Craig - HALL, Prentice.2003. UML y Patrones: introducción al análisis y diseño orientado a objetos – 2º Edición.
- GAMMA, Erich – WESLEY, Adison. 2003.Patrones de Diseño.
- STELTING, Stephen, MAASSEN, Olav – HALL, Prentice.2003. Patrones de Diseño aplicados a Java.

Complementaria

- BRAUDE, Eric. 2003Ingeniería de Software. Una Perspectiva Orientada a Objetos.. RA-MA,