

CC51H Programación Orientada al Objeto

10 UD

1 Vigencia

A partir del otoño 2008

2 Requisitos

CC31A, CC31B, CC41A. Es recomendable tener conocimiento de programación orientada a objetos, tipos de datos abstractos y nociones de ingeniería de software.

3 Objetivos

Al final del curso se espera que el alumno domine los conceptos y técnicas más importantes que existen para apoyar las etapas de análisis, diseño y programación orientada objetos, y sea capaz de aplicarlas correctamente en el desarrollo de software de mediana y gran complejidad. En particular, en este curso se dá énfasis a diseñar buenos objetos, a usar herencia solo cuando provee claras ventajas, a integrar objetos para resolver problemas complejos, a diseñar y resolver problemas usando patrones de diseño, a evaluar diseños usando métricas, a modelar un problema usando un lenguaje de modelamiento, y en general, a conocer en profundidad los aspectos de diseño y programación que permiten el desarrollo de componentes reusables, y a los que facilitan la extensión y mantención del software.

4 Programa

4.1 Introducción

Complejidad del software. Enfrentando la complejidad: descomposición, abstracción y jerarquías. Factores de calidad del software. Modularidad. Reutilización.

4.2 Conceptos básicos

Objetos: estado, comportamiento e identidad. Cómo definir buenos objetos? Diseño de buenos tipos de datos abstractos. Contratos. De tipos de datos abstractos a clases. Características de c++ y java que permiten una buena implementación de un TDA. Herencia. Polimorfismo y enlace dinámico. Contraste entre una solución con y sin herencia. Ejemplos.

4.3 Herencia y subtipos

Tipos y subtipos. Contraste entre herencia y subtipos. Qué formas de heredar existen? Contratos a través de la herencia. Herencia múltiple. Ejemplos.

4.4 Aspectos importantes en el desarrollo de software orientado a objetos

Relaciones entre clases: herencia, composición, agregación, asociación y de uso. Contraste entre composición y herencia. Cuándo es conveniente heredar? Cuándo es mejor usar composición o agregación? Patrones de diseño y aplicaciones. Evaluación de diseños: métricas.

4.5 Validación de software orientado a objetos

Métodos de validación de software. Testing y herencia. Testing el comportamiento de los objetos.

4.6 Análisis y diseño orientado a objetos

Etapas en el desarrollo de software orientado a objetos. Proceso de desarrollo recomendado (RUP). Lenguaje de modelamiento: unified modeling language (UML). Ejemplos.

5 Evaluación

El curso se evaluará a través de controles y examen (con apuntes), controles de lectura (con la lectura) y tareas computacionales.

- 2 controles y un examen
 - C1: Fecha: miércoles 29 de abril
 - C2: Fecha: miércoles 10 de junio
- 4 lecturas (no se elimina ninguna y su promedio vale como un tercer control)
- 3 tareas computacionales incrementales (no se elimina ninguna). (C++, java, rational, umbrello, BoUml, ArgoUml)

6 Lecturas

- Bjarne Stroustrup. What is object oriented programming? IEEE Software. 1988. Fecha: viernes 27 de marzo
- Bertrand Meyer. Applying design by contract. Computer. 1990. Fecha: viernes 17 de abril
- Daniel Halbert y Patrick O'Brien. Using types and inheritance in object oriented programming. 1987. Fecha: miércoles 13 de Mayo
- Por definir. Fecha: viernes 19 de junio.

References

- [Mey97] Bertrand Meyer. *Object-Oriented Software Construction*. Second Edition. Prentice Hall, 1997.
- [Eli95] A. Eliëns. *Principles of Object-Oriented Software Development*. Addison Wesley, 1995.
- [Gam95] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. *Design Patterns: Elements of Reusable Object Oriented Software*. Addison-Wesley, 1995.
- [Jac2000] Ivar Jacobson, Grady Booch, James Rumbaugh. *The unified software development process*. Addison Wesley, 2000.

- [Kho90] Setrag Khoshafian and Razmik Abnous. *Object Orientation - Concepts, Languages, Databases, User Interfaces*. John Wiley & Sons, Inc., 1990.
- [Cay2004] Cay Horstmann. *Object-Oriented Design & Patterns*. John Wiley & Sons, Inc., 2004.
- [Str2002] Bjarne Stroustrup. *The C++ Programming Language*. Addison-Wesley, 2002. Edición especial.
- [Cop92] J. Coplien. *Advanced C++. Programming styles and idioms*. Addison-Wesley, 1992.
- [Jac92] Ivar Jacobson. *Object Oriented Software Engineering: A use case driven approach*. Addison-Wesley, 1992.
- [Poh89] I. Pohl. *C++ for C Programmers*. Benjamin/Cummings, 1989.
- [Wie90] Richard Wiener and Lewis Pinson. *The C++ Workbook*. Addison-Wesley, 1990.
- [Cor97] G. Cornell, Cay S. Horstmann. *Java Core*. SunSoft Pres, 1997.

Todos estos libros están en biblioteca. Otras fuentes son: Journal of Object-Oriented Programming, ACM OOPS Messenger, Communications of ACM (Sep 90), actas de OOPSLA.