Universidad Tecnológica Nacional
Facultad Regional Buenos Aires

INGENIERÍA EN SISTEMAS DE INFORMACIÓN

ASIGNATURA: **TÉCNICAS AVANZADAS DE PROGRAMACIÓN - ELECTIVA**
DEPARTAMENTO: **SISTEMAS**
ÁREA: **CIENTIFICO-TECNICA**

Código: **0820xx**
Clase: **cuatrimestral**
Horas sem.: **4**
Horas/año: **64**
Nivel: **3°**
Año de dictado: **2010**

Objetivos:

Que el alumno logre:

- Seleccionar herramientas conceptuales y estrategias de programación orientadas a objetos, a partir de la identificación de sus ventajas y desventajas en el contexto específico de utilización para resolver problemas de desarrollo de software de complejidad creciente
- Acercar a los alumnos a la complejidad real de los sistemas, donde el software construido debe adaptarse a las interacciones con un entorno que presente restricciones o que no maneje las mismas abstracciones conceptuales del software en construcción.
- Combinar, extender y/o modificar las herramientas conocidas para su adaptación a problemas donde la implementación “de libro” no resulte adecuada.
- Comprender cómo las tecnologías de implementación afectan al proceso de construcción e incorporar herramientas que permitan mantener una estrategia conceptual en ese contexto.
- Incorporar buenas prácticas de desarrollo que simplifiquen la construcción de software comercial de alta complejidad.
- Adquirir una visión técnica de más alto nivel para, desde un rol de arquitecto o gerente comprender las prácticas en las que se basan los proyectos desarrollados y/o sistemas utilizados en el área de su incumbencia.
- Vincular la programación con las demás áreas de incumbencia del profesional de sistemas para poder basar sus decisiones gerenciales en las cuestiones técnicas subyacentes.

1

Universidad Tecnológica Nacional
Facultad Regional Buenos Aires

Programa sintético:

- Conceptos avanzados de programación orientada a objetos.
- Técnicas de desarrollo iterativo. Introducción a metodologías ágiles.
- Relación Tecnología/Diseño. Herramientas de testing y manejo de errores.
- Herramientas y buenas prácticas para el modelado de problemas complejos. Criterios para la toma de decisiones entre varias soluciones posibles.
- Componentes lógicos de una aplicación. Modelos arquitecturales para la capa de presentación. Introducción a la problemática de persistencia. Configuración de una aplicación

Programa analítico:

UNIDAD 1: Revisión de conceptos OO
Polimorfismo tipado. Comparación con polimorfismo no tipado. Contratos fuertes y débiles. Interface vs. Clase abstracta. Binding estático y dinámico. Modelado con clases/instancias. Separación de los momentos de instanciación y uso de objetos.

UNIDAD 2: Herramientas tecnológicas
Implementando un diseño. Vinculación diseño-código. Codificación de casos de prueba. Herramientas de testeo unitario. Introducción a TDD. Manejo de errores. Inversion of Control. Domain Driven Design.

UNIDAD 3: Patrones y buenas prácticas
Implementación y adaptación de patrones de diseño. Patrones GoF. Double dispatch. Type object. Wrappers. Framework vs. librería. Best practices: “Once and only once”, “Tell, don’t ask”, “Program to an interface not to an implementation”, “Favor object composition over class inheritance”, “Make it work, make it right, make it fast”.

UNIDAD 4: Aspectos metodológicos
Elección e implementación de un modelo de ciclo de vida el de desarrollo de software. Prácticas de diseño y programación iterativas. Técnicas de refactorización. Cualidades del software. Software configuration management.

UNIDAD 5: Componentes lógicos de una aplicación
Definición e implementación de interfaces entre componentes funcionales. Separación de las distintas características no funcionales en un sistema de información. Modelos arquitecturales de las aplicaciones para la problemática de presentación: pedido/respuesta vs. orientado a componentes/eventos. Introducción básica a la persistencia de un modelo de objetos: características, soluciones existentes, problemática del mapeo entre el modelo de objetos y el relacional, transacciones, manejo de la identidad. Introducción al desarrollo en capas.

Universidad Tecnológica Nacional
Facultad Regional Buenos Aires

UNIDAD 6: Extensiones al paradigma: declaratividad y metaprogramación
Elementos declarativos en la construcción de software. Metadatos. XML. Annotations. Introducción a la programación reflexiva. Modelo y metamodelo.

Bibliografía:

1. Design Patterns: Elements of Reusable Object-Oriented Software – Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides – Addison-Wesley.
2. The Design Patterns Smalltalk Companion – Sherman Alpert, Kyle Brown, Bobby Woolf – Software Pattern Series.
3. Design Patterns in Java - Steven John Metsker, William C.Wake – Software Pattern Series.
4. Object Design: Roles, Responsibilities, and Collaborations - Rebecca Wirfs-Brock, Alan McKean - Addison-Wesley Object Technology Series.
5. Domain-Driven Design: Tackling Complexity in the Heart of Software - Eric Evans – Hardcover.
6. Refactoring: Improving the Design of Existing Code – Martin Fowler, Kent Beck, John Brant, William Opdyke, Don Roberts – Addison-Wesley Prentice Hall
7. Test Driven Development: By Example – Kent Beck - The Addison-Wesley Signature Series.
8. Extreme Programming Explained: Embrace Change – Kent Beck - Paperback.
9. The pragmatic programmer : from journeyman to master – Andrew Hunt, David Thomas – Paperback
10. Working Effectively with Legacy Code - Michael C. Feathers – Robert C.Martin Series.

Correlativas:

Para cursar: Cursadas:
- Gestión de Datos.
- Análisis de Sistemas.

Aprobada:
- Paradigmas de Programación.

Para rendir: Aprobadas:
- Gestión de Datos.
- Análisis de Sistemas.