UNIVERSIDAD NACIONAL DE LA PLATA
FACULTAD DE INFORMÁTICA
Secretaria de Postgrado

Tecnicas y Herramientas

Año 2010

Carrera:Master en Ingeniería de Software
Año: **2010**
Duración: 15 semanas
Profesor a Cargo***: Dr. Federico Balaguer***
Hs. semanales : 2.5 hs

OBJETIVOS GENERALES:

El objetivo de la materia es introducir a los alumnos en técnicas modernas de la programación, en particular en aspectos de la programación orientada a objetos, diseño orientado a objetos, técnicas complementarias como testing de unidad, y áreas de desarrollo actual como: desarrollo basado en Aspectos, Integración Contínua, Métodos Ágiles y Valuación de Innovación tecnológica.
Se enfatiza en la construcción de arquitecturas de software modulares, extensibles y reusables, que son conceptos claves para aplicaciones de gran porte.
Se introduce también al alumno en el uso de un lenguaje de modelado gráfico orientado a objetos (UML), que le permitirá construir diagramas especificando distintos aspectos de un sistema.
Los trabajos prácticos se realizan usando el lenguaje de modelado y diferentes lenguajes de implementación, tales como Smalltalk, Java, etc. que son los más apropiados de acuerdo a estos objetivos.

MODALIDAD DE EVALUACION

La evaluación se lleva adelante mediante un Trabajo Práctico obligatorio y un Trabajo Final obligatorio. La nota final de cada alumno es una combinacion de la nota del Trabajo Práctico y del Trabajo Final
El trabajo práctico se compone de ejercicios de conceptos presentados en clase sobre programación orientada a objetos y UML. El Trabajo Práctico se debe resolver en un plazo de tres o cuatro semanas. La nota de este trabajo acumula hasta un 15% de la nota final de la materia.
El Trabajo Final requiere la resolución de un problema en el cual los alumnos deben aplicar los conocimientos adquiridos en diseño, programación y testing. El

diseño debe documentarse utilizando UML. La implementación se realiza en el lenguaje que disponga la cátedra (Java o Smalltalk). Los test de unidad se realizan en el framework adecuado al lenguaje seleccionado. El Trabajo Final se realiza en un plazo de cuatro o cinco semanas. Su entrega es obligatoria y aquellos alumnos que hagan una entrega parcial acceden a una segunda entrega definitiva. El trabajo Final acumula hasta el 85% de la nota final de la materia.

Programa

1. La crisis del software. Problemas de las técnicas tradicionales (procedurales). Resolución de problemas complejos. El problema de la extensibilidad, el reuso y el mantenimiento.
2. Conceptos básicos: Tipos Abstractos de Datos. Encapsulamiento. Information hiding. Objetos y Programa Orientado a Objetos. Comportamiento de un Objeto. Mensaje y Método. Clasificación: Clases e Instancias. Instanciación. Jerarquías de Clases. Relación isA. Generalización / Especialización. Herencia, Herencia Simple. Clases Abstractas. Hacia mayor genericidad de código: polimorfismo y binding dinámico. Diseño de objetos complejos. Relaciones entre Objetos. Relación de conocimiento. Relación isPartOf.
3. Lenguajes orientados a objetos: variantes. Modelo de programación. Tipos de Mensajes. Variables de instancia. PseudoVariables: identidad (self/this) y super. Constructores. Biblioteca de clases y jerarquías pre-definidas. El lenguaje Smalltalk: bloques y estructuras de control como objetos.
4. Estructuras de datos como Objetos. Objetos contenedores. Colecciones de Objetos. Protocolo estándar. Iteración. Streams. Composición de Streams.
5. Lenguajes de modelado orientados a objetos: historia y variantes. El lenguaje de Modelado Unificado (Unified Modeling Language). Diagramas de Clases. Diagramas Dinámicos ó de Comportamiento: Diagramas de Interacción (Diagramas de Secuencia y Diagramas de Colaboración). Diagramas de Casos de Uso. Uso de Casos de Uso para estimar esfuerzo.
6. Testing de Unidad. Factores que favorecen la aparición de errores. Comparación entre diferentes alcances en el testing: Unidad, Funcional, Regresión. Frameworks para implementar Test de Unidad. Validación en tiempo de ejecución.

7. Integración Continua. El problema de la integración tradicional. Prototipación vs Modelo de cascada. Elementos fundamentales: Compilación Automática, Testing Automático, Manejo de Releases. Técnicas para mejorar el proceso de Integración Continua. Herramientas que soportan el proceso.
8. Métodos Ágiles. El módelo de cascada tradicional. Procesos de desarrollo basados en prototipos. Precepto fundamental sobre el cambio de código. Elementos fundamentales de los métodos Ágiles: Usuarios, Testing y Refactoring. Métodos Ágiles: Extreme Programming, Scrum.
9. Valuación de Tecnología. Costo de Proyecto vs Valor de Tecnología.. Impacto de la tecnología informática en una organización. Medida económica actual y futura de proyectos y tecnologías.

Bibliografía

- Timothy Budd. *"An Introduction to Object-Oriented Programming". Addisson-Wesley.*
- Rebecca Wirfs-Brock. *Object Design: Roles, Responsibilities, and Collaborations, Addisson-Wesley*
- Alex Sharp. *"Smalltalk By Example". McGraw Hill*
- Martin Fowler, Kendall Scott. *"UML Distilled". Addisson-Wesley*
- Christopher Gardner *."The Valuation of Information Technology" John Wiley & Sons*
- Brian Roulstone, Jack J. Phillips ."*ROI for Technology Projects: Measuring & Delivering Value*"