Universidad Tecnológica Nacional
Facultad Regional Buenos Aires

PROGRAMA ANALÍTICO DE ASIGNATURA

DEPARTAMENTO: Ingeniería en Sistemas de Información

CARRERA: Ingeniería en Sistemas de Información

NOMBRE DE LA ACTIVIDAD CURRICULAR: Técnicas Avanzadas de Programación

Año Académico: 2023
Área: Desarrollo de Software
Bloque: Electivas
Tipo: Electiva
Modalidad: Cuatrimestral

Cargas horarias totales:

Horas reloj	*Horas cátedra*	*Horas cátedra semanales*
72	96	6

FUNDAMENTACIÓN

Actualmente es necesario que el ingeniero conozca las técnicas de desarrollo de software más avanzadas para su correcta construcción. A su vez, debe reconocer sus posibilidades tecnológicas para que pueda implementar las soluciones más adecuadas según cada situación. En este contexto, las empresas y las consultoras valoran profesionales con estos conocimientos, por lo que es de capital importancia que pueda integrar de manera exitosa estas técnicas de programación a su conocimiento.

OBJETIVOS

- Seleccionar herramientas conceptuales y estrategias de programación orientadas a objetos y funcional para resolver problemas de desarrollo de software de complejidad creciente.
- Identificar ventajas y desventajas (alcances y límites) de cada escenario para resolver problemas de desarrollo de software de complejidad creciente
- Reconocer 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.
- Identificar cómo las tecnologías de implementación afectan al proceso de construcción.

- Reconocer las herramientas que permiten mantener una estrategia conceptual según cada contexto.
- Realizar prácticas de desarrollo que simplifiquen la construcción de software de alta complejidad, frameworks y extensiones a lenguajes de programación existentes.
- Desarrollar una visión técnica de más alto nivel para comprender las prácticas en las que se basan los proyectos desarrollados y/o sistemas utilizados.

CONTENIDOS

Contenidos analíticos

Unidad 1: Extensiones modernas al paradigma orientado a objetos
Mixins y traits; comparación, ventajas y desventajas de cada uno. Contraste con el modelo clásico de Herencia Simple. Herencia Múltiple. Linearización y Aplanado de definiciones. Method lookups avanzados; resolución de conflictos y álgebras combinatorias. Impacto de los diferentes esquemas de definición de comportamiento en el diseño con objetos. Patrones de diseño aplicables al nuevo esquema e impacto en la arquitectura del software.

Unidad 2: Metaprogramación
Metaprogramación, programación reflexiva, introspección, self-modification. Comparación entre definiciones estáticas declarativas y modelos auto-descriptivos de runtime; open classes. Construcción de herramientas y frameworks, manipulación de artefactos abstractos. Diferencia entre meta-modelo y modelo, arquitectura de lenguajes complejos. Bloques, expresiones lambda, y contextos de ejecución. Scope de evaluación y binding dinámico.

Unidad 3: Sistemas de tipos y esquemas de binding
Clasificación de los sistemas de tipos: tipos nominales y estructurales, tipado explícito e implícito, dinámico y estático. Chequeo e inferencia de tipos. Concepto de binding, implicancias del uso de late binding, polimorfismo. Varianza de tipos: Invarianza, Covarianza, Contravarianza. Smart-casting. Tipado contextual.

Unidad 4: Programación híbrida Objetos-Funcional
Inmutabilidad, efecto, transparencia referencial. Diferencia entre mutable y reasignable. Aplicación de diversas formas de polimorfismo en contextos con efecto: Polimorfismo paramétrico vs. Ad-hoc, Pattern Matching, Polimorfismo de objetos; aplicaciones prácticas de la rotura del polimorfismo y encapsulamiento. Desarrollo orientado al comportamiento vs. desarrollo orientado a la estructura. Aplicación de teoría de categorías al modelo jerárquico orientado a objetos; monoides, mónadas, functores. Manejo de conjuntos inmutables. Maybe vs. Null. Manejo de errores con valores.

BIBLIOGRAFÍA OBLIGATORIA

- Gamma, E., Helm, R., Johnson, R. Vlissides, J. (1995). Design Patterns: Elements of Reusable Object-Oriented Software.Ed. Addison-Wesley Professional.
- Odersky, M. Lex Spoon, and Bill Venners (2008). Programming in scala. Ed. Artima Inc.
- Perrotta, P. (2014) Metaprogramming Ruby 2. Program Like the Ruby Pros. Ed. Pragmatic Bookshelf.
- Chiusano, P., Runar Bjarnason (2014). Functional Programming in Scala. Ed. Manning.

PÁGINAS WEB DE INTERÉS

- Sitio de la materia: ***https://tadp-utn-frba.github.io/***

CORRELATIVAS

Para cursar y rendir

- Cursadas:
    - Análisis de Sistemas de Información
    - Sintaxis y Semántica de los Lenguajes
    - Paradigmas de Programación