Ministerio de Educación
Cultura Ciencia Tecnología
Universidad Tecnológica Nacional
Facultad Regional Rosario

Rosario, 11 de noviembre de 2019

VISTO el Expediente ID N° 8113176, relacionado con el programa analítico de la asignatura electiva Técnicas y Tecnologías Avanzadas de Desarrollo de Software, de la carrera Ingeniería en Sistemas de Información – Plan 2008, y

CONSIDERANDO

Que los objetivos y contenidos del mismo se ajustan a la reglamentación vigente.

Que dicho programa cuenta con el aval del respectivo Consejo Departamental.

Que la Comisión de Enseñanza evaluó la presentación y aconsejó su aprobación.

Por ello y atento a las atribuciones otorgadas por el artículo 85° del Estatuto Universitario.

EL CONSEJO DIRECTIVO DE LA FACULTAD REGIONAL ROSARIO
DE LA UNIVERSIDAD TECNOLÓGICA NACIONAL
RESUELVE:

ARTÍCULO 1°.- Aprobar el programa analítico de la asignatura electiva Técnicas y Tecnologías Avanzadas de Desarrollo de Software, que se agrega como Anexo I de la presente resolución, de la carrera Ingeniería en Sistemas de Información – Plan 2008.

ARTÍCULO 2°.- Establecer que la misma tendrá validez durante cuatro ciclos lectivos consecutivos, según la Ordenanza N° 1383 – Lineamientos para la implementación de asignaturas electivas para las carreras de grado en el ámbito de la Universidad.

ARTÍCULO 3°.- Regístrese. Comuníquese. Cumplido, archívese.

RESOLUCIÓN N° **491**

UTN
FAC. REG. ROS.

Ing. Oscar CHIOCCHINI
Vicedecano

Ing. Antonio Luis MUIÑOS
Secretario Académico

ANEXO N° 1

RESOLUCION N° 491

UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL ROSARIO
DEPARTAMENTO DE INGENIERÍA EN SISTEMAS

Programa analítico de asignatura electiva

Técnicas y Tecnologías Avanzadas de Desarrollo de Software[1]

Carrera[2]	Ingeniería en Sistemas de Información							
Departamento[3]	Ingeniería en Sistemas de Información							
Titulación[2]	X	Ingeniería en Sistemas de Información					Analista universitario de Sistemas	
Plan de Estudio	2008 – ordenanza 1150		Área[3]				Sistemas	
Dictado	X	Anual		Cuatrimestral	Nivel	4	Electiva	Si
Carga horaria Semanal	4		Carga horaria total de la asignatura				128	
Fecha de Confección[4]	30/08/19				Versión[5]		1.01	

Fundamentación de la asignatura:[6]	*Esta asignatura procura completar la formación que el alumno necesita para poder trabajar en equipos, utilizando prácticas y técnicas que son de amplia difusión en la industria de software permitiendo al alumno:* • *Desarrollar aplicaciones de medianas a grandes que cumplan con parámetros de calidad mínimos.* • *Insertarse en cualquier equipo de trabajo profesional, conociendo las prácticas y técnicas más utilizadas.* • *Conocer y dominar las herramientas de trabajo colaborativas y de productividad utilizadas normalmente en la industria, dándoles una ventaja competitiva en el mercado laboral.* • *Contar con herramientas tanto técnicas como metodológicas y aplicarlas experimentalmente para el desarrollo de aplicaciones medianas y grandes.* *Para ello el alumno debe ser capaz de:* • *Utilizar técnicas modernas para la solución de la problemática que representa el desarrollo de software de calidad.* • *Aplicar e integrar los conocimientos adquiridos en otras cátedras al desarrollo de aplicaciones que operen con requerimientos y restricciones que enfrentarán en una implementación real.* • *Construir e implementar una solución de software de acuerdo con el análisis y diseño realizados.* • *Desarrollar software en equipo, colaborando y cooperando como un grupo de desarrolladores, con las complejidades que esto implica.* • *Identificar y utilizar aquellas tecnologías que pueden facilitar y mejorar los procesos de desarrollo de software.*
Objetivos Generales[7]:	*Capacitar al alumno para:* • *Seleccionar herramientas y estrategias de desarrollo a partir de la identificación de sus ventajas y desventajas, en función del contexto donde serán utilizadas, para resolver problemas de desarrollo de software de complejidad creciente.* • *Conocer y comprender la complejidad real de los sistemas, donde el software construido debe adaptarse a situaciones que presentan restricciones y cambios constantes.*

1 *Reemplazar por el nombre de la asignatura*
2 *Indique los títulos de la carrera para los que se propone el programa analítico. Márquelos con una cruz.*
3 *Área a la que pertenece la asignatura*
4 *refiere a la fecha en que se confecciona o desarrolla la versión*
5 *Si el programa no es la primera vez que se entrega se produce un cambio en el número de versión cambio. Si el cambio es significativo cambia el entero sino los dígitos después del punto.*
6 *Importancia para la formación profesional en función del perfil del egresado*
7 Objetivos generales que justifican la inclusión de la asignatura.

	• *Comprender cómo las tecnologías de implementación afectan al proceso de desarrollo de software.* • *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.* • *Poner en Práctica los conceptos teóricos adquiridos en otras Materias de la Carrera, especialmente en lo referente a Diseño de Sistemas, Gestión de Datos, Administración de Recursos e Ingeniería de Software (en caso de cursada simultánea).* • *Utilizar y aplicar tecnologías del desarrollo de aplicaciones de uso actual y aquellas que están surgiendo.* • *Conocer las arquitecturas utilizadas en ambientes reales de desarrollo de software y aplicarlas a casos prácticos.*

Programa de contenido analítico

Unidad temática Nº: 1 – Conceptos básicos de desarrollo de software comercial

Eje Conceptual: Bases del desarrollo de aplicaciones comerciales

Objetivo/s Específico/s[8]: Que el alumno pueda comprender y aplicar:

- Los conceptos y restricciones de los lenguajes comerciales que integran orientación a objetos, eventos y prototipado y las diferencias entre las mismas.
- Los componentes básicos de un sistema web moderno.
- Las herramientas actuales de desarrollo que facilitan y agilizan el desarrollo de aplicaciones web.

Temas:

- Implementación y aplicación de los conceptos de orientación a objetos aplicados en un lenguaje.
- Programación prototipada: uso y aplicación.
- Lenguajes de programación multiparadigma
- Sistemas Web
  - Protocolo HTTP
  - Request y Response
  - Implementación y aplicación de Timeouts y Callbacks
  - HTML
  - CSS
    - Aplicación de Frameworks CSS : Bootstrap, Foundation y Skeleton.
    - Implementación y aplicación de preprocesadores en CSS en un sitio web: LESS, SASS y Stylus.
  - Implementación y aplicación de mecanismos de intercambio de información: JSON y AJAX
  - Routing: Single page aplications
  - Components based Frameworks

[8] Objetivos específicos que justifican la inclusión de la asignatura.

Unidad temática Nº: 2 – Metodologías y enfoques de desarrollo

Eje Conceptual: Planificación y Gestión de proyectos de software

Objetivo/s Específico/s: Que el alumno sea capaz de:

- Aplicar metodologías y técnicas de desarrollo aprendidos en otras cátedras a casos prácticos.
- Comprender la importancia y beneficios que la aplicación de estas metodologías y técnicas brindan al proceso de desarrollo de software.
- Organizar el trabajo en equipos, coordinar tareas, entregas.
- Aplicar las distintas herramientas para organizar el trabajo en equipo, coordinar tareas y entregas y seleccionar la herramienta más adecuada de acuerdo a las necesidades del proyecto y del equipo.
- Utilizar estas técnicas en un caso real para comprender la dificultad que implica su correcta implementación y obstáculos que deben sortearse para obtener los beneficios que brindan.
- Aplicarlas a la gestionar tiempos y tareas dentro de un proyecto

Temas:

- Aplicación de modelos de desarrollo de software a casos realistas:
  - Metodologías ágiles: Scrum, Lean y Kanban.
  - Gestión de pendientes: Backlog y Progress Boards.
  - Gestión de cambios.
  - Gestión de configuraciones.
- Aplicación e implementación de Prácticas de diseño y programación iterativas e incrementales.
- Aplicación de las siguientes técnicas para el desarrollo y mantenimiento del software
  - Estimación de esfuerzo
  - Gestión de cambios
  - Gestión de configuraciones

Unidad temática Nº: 3 – Calidad en el desarrollo de software

Eje Conceptual: Calidad en el desarrollo de software

Objetivo/s Específico/s: Que el alumno logre:

- Conocer y aplciar las directrices para desarrollar software de buena calidad.
- Comprender la importancia y conveniencia de aplicar estas directrices y hacerlo de la forma adecuada.
- Decidir cuáles directrices aplicar a cada situación en particular de acuerdo a lo que la situación requiera.

Temas:

- Aplicación de herramientas y técnicas de gestión de código al desarrollo de código
  - Source control, metodologías (branching, reverts, tagging, etc) y buenas prácticas.
  - Código y artefactos: diferencia y relaciones
  - Técncas de Versioning
  - Code review: concepto, técnica y herramientas
  - Aplicación de las mismas al desarrollo de software nuevo y preexistente.
- Documentación
  - De código: comentarios, documentación generada
- Aplicación de técnicas y metodologías de testing
  - Niveles y tipos de testing, aplicación con herramintas adecuadas.
  - Criterios de cobertura y aceptación, herramientas y aplicación.
  - Técnicas de desarrollo basadas en testing: TDD, BDD, DDD.
- Aplicación de buenas prácticas aplicadas al código fuente
  - Organización de código en unidades lógicas (packages o namespaces y projects)
  - Convención de nombres y formateo del código
  - 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
  - Refactoring

Unidad temática Nº: 4 – Tecnologías de desarrollo y técnicas de calidad asociadas

Eje Conceptual: Aplicación de tecnologías avanzadas de desarrollo de software

Objetivo/s Específico/s:Que el alumno:

- Conozca y apliqué las nuevas tecnologías de desarrollo software.
- Identificar y los pros y contras de cada una y decidir la utilización o no de las mismas según la situación lo requiera.
- Aplicar ls de diversas tecnologías para mejorar el proceso de desarrollo de software y por ende el producto final.
- Aplicar estas tecnologías al desarrollo de software en equipo.

Temas:

- IDE vs GUI de desarrollo. Configuración del entorno, uso de las distintas vistas y perspectivas
- Detectores de bugs de runtime y validadores de estándares de programación, uso y aplicación.
- Aplicar técnicas y herramientas de tracking software
  - Project tracking utilizando metodologías ágiles
  - Herramientas de Issues/Request tracking
  - Herramientas de code review
  - Change tracking
- Herramientas para testing automatizado
  - Aplicación de herrmaintas de testing Unitario, Headless, User Interface, Integration
  - Uso de herramientas con metodologías de TDD, BDD y DDD
- Deployment
  - Aplicación de herramientas y técnicas
  - Tecnicas de deploy para lenguajes Compilados vs Interpretados
  - Aplicación de herramientas de deploy de artefactos
  - Release Management
  - Continuous Integration y Continuous Delivery: Herramientas y su implementación
  - Builds automatizados
  - Aplicar Web hooks para deploys
  - Integración con testing y mediciones
  - Aplicar AB Development y Canary releases
  - Virtualización, para virtualización y conteiners

Unidad temática Nº: 5 – Componentes de desarrollos de software profesional

Eje Conceptual: Arquitectura de software

Objetivo/s Específico/s: Que el alumno logre:

- Aplicar los distintos elementos, técnicas y mecanismos que permiten desarrollar una aplicación multicapa de forma correcta y consistente.
- Definir y construir una arquitectura para la aplicación que le permita realizar un desarrollo iterativo incremental, gestionando la complejidad creciente del software y permitir la reutilización de componentes, la escalabilidad de la misma y extensión de sus funcionalidades más allá de las previstas inicialmente por los involucrados en el proyecto.

Temas:

- Patrones de diseño y programación (Inversión de control, Inyección de dependencias, DAO, GoF, singleton, decorator, Callback, Configuration objects, Returning functions, Partial application, Initialization patterns, Immediate functions, Memoization, Selfdefining functions, etc).
- Utilización de Frameworks.
- Desarrollo en capas.
    - Componentes e implementación de una arquitectura de N-Capas y comparativa con arquitecturas de 2-Capas y 3-Capas.
    - Aplicación de Técnicas y Herramientas para la Capa de datos
        - Mapeo de objetos. Implementar Object-Relational Mapping y Object-Document Mapping.
        - Pool de conexiones.
        - Cache de objetos (memcache, redis, etc)
        - Read Replicas
    - Capa de presentación
        - Aplicación e implementación de frameworks de frontend
        - Construcción Agnostic frontend: HTML+CSS+JS+AJAX+REST
    - Capas de servicios
        - Web Services
        - Servicios internos y externos
        - Arquitecturas de servicios
        - Interfaces de servicios
        - Rest vs GraphQL
        - Macro y micro servicios

Unidad temática Nº: 6 – Características no funcionales de un sistema de información

Eje Conceptual: Arquitectura de software

Objetivo/s Específico/s: Que el alumno logre:

- Conocer e identificar los distintos elementos que componen una aplicación de mediano/gran tamaño.
- Entender la necesidad de los mismos en los sistemas modernos, cuando implementarlos y las ventajas que se pueden obtener al utilizarlos.
- Comprender cuándo y cómo aplicar las herramientas disponibles para implementar estas estas características.

Temas:

- Cómo aplicar componentes estandar de seguridad:
  - Usuarios, roles, perfiles y permisos
  - Auditoría y Autorizaciones
  - Single Sign-On y sistemas externos de validación de usuarios
- Manejo de errores y excepciones
  - Jerarquía y niveles de excepciones
  - Log de errores de aplicación
- Manejo de Sesiones
  - Alternativas de manejo de sesiones
  - Sesiones en sistemas distribuidos
- Alta disponiblidad y escalabilidad.
  - Limitaciones de performance y arquitecturas de escalabilidad
  - Consideraciones de arquitecturas de alta disponibilidad
- Interacción entre sistemas
  - APIs: definición, objetivo componentes esenciales, especificación y documentación
  - Comunicación vía web services
  - Orquestación: colas, workflow engines, distributed processing, sistemas de mensajería

Bibliografía[9]

Obligatoria o básica:

Título	Autor/es	Editorial	Año de Edición
Patterns of Enterprise Application Architecture	Martin Fowler	Addison-Wesley	2003
Refactoring: Improving the Design of Existing Code(new edition)	Martin Fowler, Kent Beck, John Brant, William Opdyke, Don Roberts	Addison-Wesley Prentice Hall	2012
Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation (Addison-Wesley Signature Series (Fowler))	Jez Humble, David Farley	Addison-Wesley Professional	2011
Patterns, Principles, and Practices of Domain-Driven Design	Scott Millett, Nick Tune	Wrox	2015

Complementaria:

Título	Autor/es	Editorial	Año de

[9] *Para textos: citar autor, título, ciudad, editorial, año. Para revistas: citar autor, título del artículo, nombre de la revista, nº, lugar, edición, año, páginas., Para sitios web: dirección de la página.*

			Edició n
Design Patterns in Java	Steven John Metsker; William C. Wake	Addison-Wesley Professional	2006
Object Design: Roles, Responsibilities, and Collaborations.	Rebecca Wirfs-Brock, Alan McKean	Addison-Wesley Object Technology Series	2002
Agile Project Management QuickStart Guide: A Simplified Beginners Guide To Agile Project Management	Ed Stark	CreateSpace Independent Publishing Platform	2014
Cloud Native Java: Designing Resilient Systems with Spring Boot, Spring Cloud, and Cloud Foundry	Josh Long, Kenny Bastani	O'Reilly Media	2015

Propuesta Pedagógica

Fundamentación

Esta asignatura procura completar la formación que el alumno necesita para poder trabajar en equipos, utilizando prácticas y técnicas que son de amplia difusión en la industria de software permitiendo al alumno:

- Desarrollar aplicaciones de medianas a grandes que cumplan con parámetros de calidad mínimos.
- Insertarse en cualquier equipo de trabajo profesional, conociendo las prácticas y técnicas más utilizadas.
- Conocer y dominar las herramientas de trabajo colaborativas y de productividad utilizadas normalmente en la industria, dándoles una ventaja competitiva en el mercado laboral.
- Contar con herramientas tanto técnicas como metodológicas y aplicarlas experimentalmente para el desarrollo de aplicaciones medianas y grandes.

Si bien la asignatura tendrá un alto contenido técnico será de utilidad para aquellos alumnos que persigan un perfil más orientado al análisis y la gestión ya que les brindará una visión de las tareas que deben realizar los desarrolladores y la dificultades y obstáculos que éstos deberán enfrentar así como también prácticas para gestionar y coordinar equipos de trabajo y dar seguimiento de tareas.

Adicionalmente al utilizar técnicas y tecnologías que actualmente son utilizadas en el medio laboral o se encuentran en auge tanto en nuestra región como en el mundo le brindará al alumno grandes posibilidades de inserción laboral al finalizar la carrera, disminuyendo la brecha entre el ámbito académico y el profesional.

Los conocimientos y experiencias que les serán impartidos a los alumnos durante el cursado de la asignatura se encuentran en límite del estado actual del conocimiento en diversos aspectos del desarrollo de software, brindando al alumno y al docente en una posición inmejorable para la investigación y desarrollo de nuevas técnicas, metodologías y tecnologías aplicadas al desarrollo y arquitectura de software.

Perfiles ideales de los actores educativos

Docente: Dado que la asignatura plantea la aplicación e implementación de técnicas y herramientas de desarrollo de software de uso comercial y actual como así también plantea la formación de alumnos en metodologías ágiles y herramientas de gestión del ciclo de desarrollo, se recomeienda que el docente tenga:

- Experiencia en el desarrollo de aplicaciones web.
- Gestión de equipos utilizando metodologías ágiles.
- Experiencias en diversos lenguajes y frameworks de desarrollo.
- Experiencia en herramientas de testing automatizado, deployment automatizado, integración continua y gestión de versiones.

Idealmente se recomienda que el docente tenga experiencia en la utilización de tecnologías y lenguajes de programación open source. Debido a la diversidad de herramientas necesarias para aplicar las técnicas y metodologías propuestas en los contenidos de la asinatura la utlización de un lenguaje propietario restringiría las oportunidades de mantenerse actualizados en las mismas. Es por ello que el uso de tecnologías open source facilitaría la enseñanza de las mismas y mantener la materia actualizada sin necesidad de incurrir en costos adicionales de la materia.

Alumno: Se espera que el alumno sea proactivo y tenga constancia para la realización del trabajo y el análisis de los ejemplos presentados.
Se espera que presente interés por la programación y por la calidad en el desarrollo de software así como también en la utilización de herramientas y frameworks.
También está orientado a alumnos que pretendan desarrollar capacidades de trabajo en equipo y de gestión de proyectos.

Estrategia de enseñanza

Clases expositivas utilizando ejemplos, se utilizará un mismo ejemplo para exponer los diversos temas de la materia de forma incremental sobre el mismo ejemplo. También se utilizarán otros ejemplos más cortos para explicar alternativas y temas específicos.

Aplicación de los temas dictados a uno o más ejemplos para interiorizar los conceptos en formato de un trabajo práctico. El docente deberá acompañar al alumno en la ejecución del trabajo brindandole andamiaje[10] para el mismo, esto significa que lo dejará trabajar libremente pero en caso de ser necesario brindará soporte para la ejecución del proyecto de manera transitoria cuando lo considere necesario.

El código deberá estar disponible en un repositorio centralizado como github o gitlab y el matierial/documentación debería ser publicado en formato digital en una wiki, plataforma de e-learning o e-group.

Evaluación

La evaluación consistirá de 2 etapas:

Etapa 1:
Desarrollo de un sistema integral donde el alumno pueda aplicar los temas de la asignatura en grupos.
Defensa oral individual de los temas basado en la aplicación de los mismos en el sistema desarrollado.

Etapa 2:
Evaluación escrita con la siguiente metodología:
Se le brindará a los alumnos la especificación de uno o más casos (ya sea mediante ejemplos de código o especificación del negocio) y el alumno deberá aplicar uno o más conceptos de la asignatura para los casos (diseño de pruebas unitarias, refactorización de código, priorización de un backlog, gestión de los cambios propuestos, diseño de un modelo de versionado, diseño de un modelo de branching, etc).

Sugerencia del lugar de dictado

Debido a la naturaleza teórico-práctico de los temas de la electiva se sugiere el dictado de la misma en un laboratorio de informática.

Recursos necesarios

Los recursos mínimos necesarios serían:

- Utilización de PCs para que los alumnos realicen la práctica, las cuales deben al menos contar con 2 GB de RAM y 1 GB de espacio libre en disco, para instalar y ejecutar el software y herramientas de desarrollo a utilizar en la asignatura.
- Conexión a internet para hacer uso de herramientas de desarrollo gratuitas que se utilizarían en la asignatura.

Recursos deseables

Proyector para el dictado de clases para mostrar ejemplos y materias.
Servidor con Linux instalado para simular casos más reales, puede utilizarse un servidor virtual para esto.
Máquinas virtuales para el dictado de las clases.

Asignaturas Correlativas del plan[11]

Asignaturas regulares para el cursado:	Diseño de Sistemas Gestión de Datos Lenguaje de Programación Java **o** Soporte a la Gestión de Datos con Programación Visual **o** Tecnología de Desarrollo de Software IDE
Asignaturas aprobadas para el cursado:	Análisis de Sistemas Sintaxis y Semántica de Lenguajes Paradigmas de Programación
Asignaturas aprobadas para rendir:	Diseño de Sistemas Gestión de Datos Lenguaje de Programación Java **o** Soporte a la Gestión de Datos con Programación Visual **o** Tecnología de Desarrollo de Software

[10] Teoría del andamio de Bruner.

[11] *No está permitido indicar asignaturas electivas como correlativas. Además todos los cuadros deben estar completados.*

	IDE

Justificación de correlatividades

Diseño de Sistemas: Se requiere como correlativa ya que la materia propuesta es de aplicación de los contenidos dictados en Diseño de Sistemas, caso contrario sería repetir dichos contenidos. Adicionalmente se necesita la asimilación de los conceptos básicos de patrones y arquitectura para poder comenzar con la aplicación de los mismos.

Gestión de Datos: Se requiere que los alumnos hayan asimilado los temas impartidos en Gestión de Datos ya que esta materia parte sobre la base de utilizar herramientas como ORMs y ODMs, pooles de conexiones y otras estrategias que requieren una comprensión del funcionamiento básico de las bases de datos. Adicionalmente las condiciones prácticas de la asignatura requieren que el alumno sea capaz de construir, utilizar y manipular bases de datos sin necesidad de supervisión constante.

Analisis de Sistemas y Diseño de Sistemas: Dadas las condiciones prácticas de la materia se requiere en primer lugar la habilidad de detectar y especificar los requerimientos de las aplicaciones así como entender los artefactos que se utilizarán para especificar los ejercicios y las práctias. Estos artefactos son enseñados en su mayoría en Análisis de Sistemas y luego en Diseño de Sistemas.
También dadas las características de evaluación se requiere que el alumno cuente con la capacidad de relevar y documentar los requerimientos de una aplicación así como también los componentes del sistema y el desarrollo del mismo.

Sintaxis y Semántica de los Lenguajes: Se requiere una comprensión de los conceptos de programación y elementos de los lenguajes de programación y sus componentes para el desarrollo de los temas incluidos en la asignatura así como también para la comprensión de los ejemplos utilizados.

Paradigmas de Programación: Dado el foco de la asignatura en el desarrollo de aplicaciones y ejemplos con lenguajes orientados a objetos a la vez que intenta profundizar en la orientación a objetos aplicando diversos patrones y arquitecturas para la POO se requiere que el alumno tenga una comprensión del paradigma de orientación a objetos. A su vez se espera instruir en lenguajes que combinan multiples paradigmas lo que requiere que los alumnos comprendan al menos los principales paradigmas de programación es que se requiere a Paradigmas de Programación como correlativa.

Lenguaje de programación Java o Soporte a la Gestión de Datos con Programación Visual o Tecnología de desarrollo de software IDE: Esta materia pretende instruir y capacitar a los alumnos para aplicar conceptos avanzados de desarrollo con lenguajes de uso comercial, como tal se requiere que los alumnos tengan conocimientos de los componentes básicos (API, librerías base de los lenguajes, IDEs, sintáxis y técnicas básicas de desarrollo) para poder aplicar rápidamente los conceptos que se expongan y también poder comprender los ejemplos utilizados por la cátedra para instruir a los alumnos en los diversos temas.

Asignaturas Equivalentes respecto del plan anterior[12]

Asignatura/s equivalente respecto del plan anterior:	

[12] *Consignar asignaturas que se pueden otorgar como equivalentes para las posibles solicitudes de cambio de plan.*