UNIVERSIDAD JORGE TADEO LOZANO

Facultad de Ciencias Naturales e Ingeniería

SÍLABUS DE LA ASIGNATURA

ARQUITECTURA DE SOFTWARE

Código 702302 · Ingeniería de Sistemas · Semestre IV

Información general del curso

Nombre de la asignatura	Arquitectura de Software
Código de asignatura	702302
Grupo	3
Programa	Ingeniería de Sistemas
Ubicación en el plan	Semestre IV, posterior a Ingeniería de Software (III) · 3 créditos
Periodo académico	2026-2S
Facultad	Ciencias Naturales e Ingeniería
Área académica responsable	Industrias y Tecnologías Digitales
Clase teórico-práctica	Sí

Información del profesor

Nombre	Elkin Aldemar Rodríguez Escamilla
Correo electrónico	elkina.rodrigueze@utadeo.edu.co
Atención a estudiantes	A convenir con los estudiantes

Resumen de trayectoria académica y profesional

Elkin Aldemar Rodríguez Escamilla es Ingeniero Industrial egresado de la Universidad Jorge Tadeo Lozano (2009-2015) y Especialista en Gerencia de Marketing de la Universidad Sergio Arboleda. Ha liderado la estrategia de analítica y transformación digital como BI Specialist en Stanley Black & Decker (multinacional), construyendo reportes y pipelines de datos para toda la operación. Actualmente es consultor independiente en Data World, donde diseña soluciones de datos, automatización de procesos e inteligencia artificial aplicada para pymes y medianas empresas. Previamente se desempeñó como BI Specialist en BIA Energy y como Controller en el CESA, liderando la creación de centros y repositorios de datos en tiempo real, y como Jefe de Administración de Ventas en Soldaduras West Arco, donde diseñó el modelo de administración de datos que redujo en 60% los tiempos de entrega de pedidos. Domina Power BI, SQL, Python, Cloud Computing (Azure, AWS) e inteligencia artificial aplicada a negocios (incluyendo herramientas como n8n y Cursor). Docente cátedra en la Tadeo desde 2022, dictó el primer diplomado de SQL de la universidad con más del 95% de satisfacción académica y actualmente cursa su tercer semestre como docente de pregrado, combinando su experiencia en consultoría de datos y estrategia con una enseñanza dinámica, participativa y anclada en la práctica real de la industria.

Presentación de la asignatura

Esta asignatura brinda los conocimientos esenciales sobre arquitectura de software, permitiendo al estudiante comprender los conceptos más recientes de esta disciplina y aplicarlos con éxito en proyectos de la vida real. El curso se dicta con un enfoque dinámico y anclado en la práctica profesional: cada decisión arquitectónica se conecta con casos reales de la industria, incluyendo el impacto que hoy tiene la inteligencia artificial sobre el rol del arquitecto de software.

Objetivos de aprendizaje — Capa 1 (cobertura oficial UTadeo)

Con el apoyo del grupo de estudiantes, el profesor y la comprensión de los recursos y actividades disponibles, el estudiante logrará:

1. Conocer qué es un patrón de diseño.
2. Conocer qué es un estilo y un patrón de arquitectura, sus características y diferencias.
3. Conocer el rol de un Arquitecto de Software en una organización.
4. Implementar soluciones reales basadas en diversos estilos de arquitectura.

Resultado de aprendizaje: el estudiante será capaz de diseñar y construir soluciones de arquitectura basadas en estilos de arquitectura reales, siendo consciente de sus ventajas, desventajas y características.

CAPA 2 — VALOR AGREGADO DEL DOCENTE

Lo que quiero que se lleven: pensar en trade-offs, no en recetas — con la IA como acelerador y como nuevo actor del acoplamiento

Esta es mi huella personal en el curso, construida sobre el marco de Mark Richards y validada semestre a semestre en el aula:

- El eje del curso es el marco de Richards sobre acoplamiento y cohesión (capas vs. microservicios): cada decisión arquitectónica se discute como un trade-off explícito, nunca como una "mejor práctica" absoluta.
- Generación de código asistida por LLMs como caso de estudio del propio acoplamiento: ¿qué tan acoplado queda un sistema cuando gran parte de su código lo escribió una IA que no conoce el negocio?
- El taller "Detectives de la Arquitectura" usa NotebookLM como apoyo de comprensión de lecturas técnicas densas —nunca como generador de respuestas—, con roles de lotería, contradictor y puente entre capítulos para forzar la discusión, no la copia.
- Los atributos de calidad clásicos (disponibilidad, escalabilidad, seguridad) se llevan a arquitecturas con componentes de IA: latencia de inferencia, costo por token y explicabilidad como los nuevos "atributos de calidad" que un arquitecto debe negociar en 2026.
- El rol del arquitecto se redefine en vivo: de escribir código a cuestionar, negociar y asumir consecuencias — con demostraciones de Claude Code / Cursor aplicadas a decisiones reales, nunca a simple autocompletado.

Contenidos temáticos — Capa 1

- Qué es una Arquitectura de Software. Historia.
- Patrones de diseño en el desarrollo de software.
- Responsabilidades de un Arquitecto de Software.
- Estilos de Arquitecturas de Software.
- Patrones de Arquitecturas de Software.

- Arquitecturas basadas en Microservicios.
- Arquitecturas CQRS.
- Arquitecturas basadas en Eventos.

Sistema de evaluación (definición del docente)

Cada corte del curso se evalúa bajo el mismo esquema, definido por el docente para privilegiar la conexión real con el estudiante y el pensamiento crítico sobre la memorización:

50% Dinámicas — actividades activas en clase: casos, talleres prácticos, gamificación, debate, uso en vivo de herramientas de IA, aula invertida, juego de roles.

10% Participación — no se mide por la cantidad de intervenciones, sino por el criterio: aportes que agregan valor a la discusión, preguntas que hacen avanzar el análisis, disposición a sustentar una posición con argumentos.

40% Parcial — el formato lo define el docente según el corte y la naturaleza del tema: puede ser examen escrito, sustentación oral, proyecto aplicado o entregable integrador.

Lineamientos del curso

Clases presenciales; sesiones virtuales solo bajo acuerdo con los estudiantes y la facultad.

Factores de éxito para el curso: respeto, dedicación e integridad.

Bibliografía básica y complementaria

- Software Engineering Institute (SEI) — Software Architecture: Principles and Practices. http://www.sei.cmu.edu/products/courses/saf.html
- MSDN Architecture Center — Learn Architecture. http://msdn.microsoft.com/en-us/architecture/aa699358.aspx
- Richards, M. & Ford, N. (2020). Fundamentals of Software Architecture: An Engineering Approach. O'Reilly.
- Bass, L., Clements, P. & Kazman, R. (2021). Software Architecture in Practice (4ª ed.). Addison-Wesley.
- Fowler, M. (2002). Patterns of Enterprise Application Architecture. Addison-Wesley.
- Newman, S. (2021). Building Microservices (2ª ed.). O'Reilly.
- Evans, E. (2003). Domain-Driven Design. Addison-Wesley.
- Martin, R. C. (2017). Clean Architecture. Prentice Hall.
- Amazon Web Services. AWS Well-Architected Framework (edición vigente).
- Microsoft Azure Architecture Center — Azure Well-Architected Framework (edición vigente).