Ministerio de Cultura y Educación **(Programa del año 2026)**
Universidad Nacional de San Luis **(Programa en trámite de aprobación)**
Facultad de Ciencias Físico Matemáticas y Naturales **(Presentado el 13/05/2026 08:20:54)**
Departamento: Informatica
Area: Area IV: Pr. y Met. de Des. del Soft.

I - Oferta Académica

Materia	Carrera	Plan	Año	Período
ARQUITECTURA DE SOFTWARE	ING. INFORM.	026/12-08/15	2026	1° cuatrimestre

II - Equipo Docente

Docente	Función	Cargo	Dedicación
RIESCO, DANIEL EDGARDO	Prof. Responsable	P.Asoc Exc	40 Hs

III - Características del Curso

Credito Horario Semanal				
Teórico/Práctico	**Teóricas**	**Prácticas de Aula**	**Práct. de lab/ camp/ Resid/ PIP, etc.**	**Total**
Hs	2 Hs	Hs	3 Hs	5 Hs

Tipificación	Periodo
B - Teoria con prácticas de aula y laboratorio	1º Cuatrimestre

Duración			
Desde	**Hasta**	**Cantidad de Semanas**	**Cantidad de Horas**
11/03/2026	23/06/2026	15	75

IV - Fundamentación

La arquitectura de software se ha vuelto indispensable con el crecimiento en complejidad y tamaño de los sistemas de software que se desarrollan. Esta describe la organización fundamental del sistema, y esta descripción ilumina decisiones de diseño de alto nivel: su composición y partes interactuantes, sus interacciones y patrones de comunicación presentes, y las propiedades claves de sus partes en las que la totalidad del sistema descansa o impone al sistema. Se convierte entonces en un modelo relativamente pequeño con respecto al sistema real, que permite a los distintos interesados comunicarse, y tratar decisiones de diseño que tendrán un profundo impacto en el trabajo subsiguiente, y en el éxito del sistema.

V - Objetivos / Resultados de Aprendizaje

Al finalizar el curso se espera que el alumno sea capaz de:
* Describir las arquitecturas de software con precisión y de manera efectiva.
* Desarrollar las habilidades para reconocer patrones arquitectónicos en sistemas de software existentes.
* Proponer arquitecturas alternativas para encarar un determinado problema.
* Desarrollar módulos de software correspondientes a una definición de arquitectura determinada.
* Aplicar herramientas para definir arquitecturas.
* Utilizar el “dominio de conocimiento” para instanciar una arquitectura para una familia de aplicaciones en particular.
* Diseñar arquitecturas que cumplan con requisitos de seguridad establecidos asi como otros requisitos no funcionales.

Durante el dictado de la asignatura se abordan los siguientes ejes transversales:
- Identificación, formulación y resolución de problemas de ingeniería en sistemas de información/informática.
- Concepción, diseño y desarrollo de proyectos de de ingeniería en sistemas de información/informática.
- Utilización de técnicas y herramientas de aplicación en de ingeniería en sistemas de información/informática.
- Fundamentos para el desempeño en equipos de trabajo.
- Fundamentos para una comunicación efectiva.
- Fundamentos para el aprendizaje continuo.

VI - Contenidos

Contenidos Mínimos

Los proyectos de software de alta complejidad: El nivel de abstracción de la arquitectura. El diseño de la arquitectura de productos de software de alta complejidad. Modelos y métodos formales orientados a la arquitectura. Herramientas de generación de instancias específicas de la arquitectura. Evaluación de la arquitectura de sistemas complejos existentes. Estilos de arquitectura de sistemas de software de mayor difusión. El rol del dominio del espacio de problema en la especialización de una arquitectura general a una familia de aplicaciones en particular.
Laboratorio: Aplicar los conceptos aprendidos para instanciar una arquitectura de una familia de aplicaciones y construir software para la definición de dicha arquitectura.

Unidad 1 - Introducción

La Arquitectura del Software. Definición. Que es y que no es Arquitectura. El ciclo de negocio de la arquitectura. Procesos de software y el ciclo de negocio de la arquitectura. El papel de la arquitectura en proyectos de software de alta complejidad. Niveles de diseño de software y el nivel arquitectónico del diseño de software. Características de una buena arquitectura.

Unidad 2 - Lenguajes de Descripción Arquitectónica

Lenguajes para el modelado, la descripción y prueba de la arquitectura. Documentación. IEEE 42010. Distintas Vistas Arquitectónicas. Vista de Modelos y Métodos Formales. Vista de Módulo. Estilos. Vista de Componente y Conector. Vista de Asignación. Estilos. Despliegue. Instalación.

Unidad 3 – Análisis y Diseño Arquitectónico

Arquitectura lógica: conjunto de patrones y abstracciones coherentes que proporcionan el marco de referencia necesario para guiar la construcción del software para un sistema de información. La Arquitectura de Software como fundamento para que analistas, diseñadores, programadores trabajen coherentemente. Arquitecturas y atributos de Calidad. Disponibilidad. Interoperabilidad. Modificabilidad. Performance. Seguridad. Prueba. Usabilidad. Criterios de análisis y diseño. Evaluación de la Arquitectura.

Unidad 4 – Patrones Arquitectónicos

Caracterización de patrón arquitectónico. Patrones, Modelos de Referencia y Arquitecturas de Referencia. Catálogos de patrones arquitectónicos. Patrones de arquitecturas de sistemas de software de mayor difusión. Catálogos de Patrones Arquitectónicos. Relaciones entre patrones. Descripción del patrón. Capas. Multi-Tier. Flujos de Datos. Sistemas distribuidos. Broker. Sistemas interactivos. Modelo-Vista-Controlador. Sistemas Adaptables. Microkernel. Reflexivos. Web. Basados en Eventos. Basados en Recursos. Servicios. Computación en la Nube.

Unidad 5 – Orientado a Servicios

Evolución. Topología. Orientación a Objetos vs. Orientación a Servicios. Service Oriented Architecture (SOA). Componentes y su colaboración en SOA. Macro servicios o micro servicios. Stack de servicios Web. WSDL. SOAP. UDDI. Coreografía. Estilo Arquitectónico REpresntational State Transfer (RESTfull). Direccionalidad. Interfaz uniforme. Orientado a Recursos. Servicios vs Microservicios.

Unidad 6 – Arquitectura con Seguridad

Introducción. Atributos de Seguridad. Dimensiones de Seguridad. Seguridad con SOA. Protocolo SSL. Estandars de Seguridad de los Servicios Web. Amenazas. Mitigación. Seguridad.

Unidad 7 - Transacciones en SOA

Introducción. ACID. Protocolo COMMIT en dos fases. Patrón de Transacción compuesta. Patrón de Transacción de larga

duración. Patrón de Negociación.

Unidad 8 - Arquitectura para Aplicaciones Blockchain
Definiciones. Funcionamiento. Red de Pares. Ethereum. Características. Estructura. Dinámica. Protocolos de Consenso. Hashing. Firma digital. Actores en Blockchain. Libro Mayor. Cadena de bloques. Transacción. Mineros. Modelos de Gobernanza. Arquitectura como Almacenamiento. Arquitectura como Infraestructura Digital. Propiedades y requisitos no funcionales. Arquitectura: Análisis y Diseño. Diseño de Aplicaciones basadas en Blockchain. Criptomonedas. Contratos inteligentes. Casos de Uso con la Blokchain Federal Argentina.

VII - Plan de Trabajos Prácticos

Metodología de Enseñanza
Se promoverá la permanente vinculación de la teoría con la práctica, de forma de garantizar que los contenidos teóricos sean llevados a la práctica tanto a través de prácticas de aula y laboratorio para los temas teóricos dados como la integración en un proyecto de desarrollo de los conceptos adquiridos durante toda la materia.
Por lo tanto las clases serán:
* Clases teórico-prácticas
* Prácticas de aula
* Prácticas de laboratorio con herramientas de Ingeniería de Software asistida por computadora
* Proyecto integrador de máquina referente a un idea innovadora propuesta por el alumno

El proyecto integrador será un trabajo grupal ya que permitiría a los alumnos, en la integración de todos los conceptos teóricos de la materia, aplicar un proceso colectivo de reflexión y discusión; compartir el conocimiento personal, enriquecerlo y potenciar el conocimiento colectivo; mejorar la comunicación y cooperación entre los integrantes del grupo o de la clase; aprender a trabajar en forma grupal, tarea básica en la mayoría de los proyectos reales.

Las clases teóricas serán desarrolladas utilizando herramientas de presentación, en donde cada unidad es presentada a través de diapositivas, disponibles previamente a la clase, a los alumnos. A cada clase teórica se le indica al alumno la bibliografía básica que debe estudiar y su bibliografía complementaria (sirviendo las diapositivas como una guía).

Los criterios de realización que se llevan a cabo para lograr diferentes aspectos en la enseñanza del alumno son las siguientes:

El alumno integra conocimientos de diferentes áreas de formación para el abordaje de problemas de la disciplina.

Se busca que el alumno identifique y analice situaciones problemáticas y proyecte y evalúe posibles soluciones utilizando los conceptos y prácticas de la informática, en particular en las siguientes situaciones:

a. Analice la situación problemática a través de la información relevada del problema a solucionar. El criterio para llevar adelante esta característica, es que el alumno tenga a cargo un proyecto integrador resolviendo una problemática del mundo real.

b. Identifique adecuadamente el problema a resolver. El criterio de realización para ello, es que el alumno, en la búsqueda de la solución de una problemática para llevar adelante su proyecto integrador, debe identificar el problema y defenderlo de forma pública para definir si es adecuado a la materia.

c. Proyecte posibles soluciones al problema fijando los objetivos a alcanzar y delimitando los requerimientos a considerar. Para lograr esto, el alumno en el planteo del proyecto integrador define los objetivos que debe alcanzar, y junto con la cátedra, se delimitan los requerimientos según la cantidad de integrantes que participen en el proyecto (cuando no es posible que el numero de integrantes sea 3). Debe encontrar soluciones al problema planteado

d. Escojer patrones de solución, estructuras de datos, tipo de algoritmos y software de base a partir del análisis del problema considerado y de los recursos disponibles para su resolución. Para realizar esto, el alumno debe aplicar los patrones arquitectónicos para encontrar la mejor solución al problema planteado, y elegir el software de base adicional al dado por la

cátedra. Todo esto a partir de un análisis minucioso del problema planteado considerando los recursos disponibles para su solución

Se espera que el alumno diseñe e implemente sistemas informáticos. En particular:
a. Diseñe el sistema informático que cumpla con los requisitos para satisfacer la solución del problema planteado.

b. Defina las fases implicadas en el ciclo de vida de distintos modelos de desarrollo, donde la fase de definición de la arquitectura es la esencial en este caso. Esto se logra, cuando el alumno siga las fases definidas por el proceso de desarrollo que es dado en la cátedra para la construcción del sistema informático que deben desarrollar en el proyecto integrador

c. Implementa un sistema informático atendiendo normas de seguridad y de calidad acorde al diseño provisto.
El alumno debe llevar adelante requisitos de seguridad establecidos por la cátedra que deben ser soportados por la arquitectura diseñada y aplicarlos en la construcción del sistema informático que deben desarrollar en el proyecto integrador.

d. Documentar y fundamentar el diseño de la arquitectura y la implementación del prototipo propuesto. Para ello, el sistema informático que debe desarrollar el alumno con el proyecto integrador está documentado siguiendo los líneamientos del proceso de desarrollo indicado por la cátedra basado las vistas arquitectónicas de Clements.

Trabajo Práctico y de laboratorio 1. Arquitectura REST.
Objetivo: Análisis, diseño e implementación de distintos problemas usando REST.
Se pretende que el alumno entienda, comprenda y utilice la tecnología REST como medio para el desarrollo de arquitecturas distribuidas y en desarrollo Web mas utilizadas. Haga uso de esta tecnología para desarrollar cada uno de los ejercicios propuestos.

Trabajo Práctico y de laboratorio 2: MVC y REST
Objetivo: Analisis, diseño e implementación de distintos problemas usando Modelo-Vista-Controlador y Servicios Web.
Una de las arquitecturas mas utilizadas es el MVC. Se pretende que el alumno utilice herramientas para la solución de distintos problemas planteados en el trabajos prácticos donde el mismo es conveniente el uso las arquitecturas de MVC junto con REST.

Trabajo Práctico y de laboratorio 3: SOA y WSDL
Objetivo: Análisis, diseño e implementación de distintos problemas usando SOA con WSDL.
Una de las arquitectura que permite la implementación simple de arquitecturas distribuidas y utilizando distintas plataformas y diferentes lenguajes de desarrollo es SOA. En este trabajo practico y de laboratorio llevarán adelante ejercicios para analizar, diseñar e implementar utilizando en particular el lenguaje WSDL.

Trabajo teórico práctico 4: Calidad
Objetivo: Los atributos de calidad es un aspecto no funcional esencial en la definición de una arquitectura. Entender e interpretar los atributos de calidad.
Se dan problemas y el alumno debe asociar como afecta la calidad a una arquitectura en particular.

Trabajo Práctico y de laboratorio 5: Estilos Arquitectónicos
Objetivo: Análisis y Diseño Arquitectónico para distintos tipos de problemas. Definir Estilos Arquitectónicos.
En este práctico se plantea diferentes problemas de cierta magnitud importante y los alumnos deben buscar cual es el mejor estilo arquitectónico que resuelve dicho problemas. El alumno debe desarrollar el modelo BPMN que da estructura a la arquitectura definida. Deben construir los modelos arquitectónicos, la Vista de Modulo estilo Uso, la Vista Componente y Conectores; y la Vista de Asignación estilo Despliegue. Y para un ejercicio seleccionado mostrar el prototipo de funcionamiento de dicha arquitectura.

Trabajo Práctico y de laboratorio 6. Arquitectura Reflexiva.
Analisis, diseño e implementación de distintos problemas usando el patron arquitectónico Reflexivo.
En este trabajo práctico y de laboratorio se lleva adelante el análisis, diseño e implementación de problemas que requieren soluciones utilizando una arquitectura de meta modelos.

Proyecto Integrador
Objetivos: Aplicar los conceptos aprendidos para plantear un problema, analizarlo y buscar la mejor instanciación de una arquitectura de una familia de aplicaciones y construir el software para la definición de dicha arquitectura, combinando distintos patrones arquitectónicos. Este proyecto se realiza en grupos de 3 personas. El mismo debe ser presentado y defendido en forma oral y pública.

Durante el dictado de la asignatura se abordan los siguientes ejes transversales:

o Identificación, formulación y resolución de problemas de ingeniería en sistemas de información/informática.
¿Cómo se aborda?
Este eje se aborda en todos los trabajos prácticos y de laboratorio, donde el alumno debe analizar el problema que se plantea, formular una propuesta de resolución en base a la temática de cada práctico y realizar, en algunos casos, un prototipo esquemático donde se muestre dicha resolución.
¿Cómo se evalúa?
La forma de evaluar este eje es la exposición oral de la solución por parte del alumno y la revisión particular del docente de la catedra ante la solución y el prototipo de algunos de los ejercicios que es seleccionado por el docente y que el alumno debe justificar su solución.

o Concepción, diseño y desarrollo de proyectos de de ingeniería en sistemas de información/informática.
¿Cómo se aborda?
Este eje se aborda en el proyecto integrador formular una propuesta de resolución y realizar un prototipo donde se muestre dicha resolución.
¿Cómo se evalúa?
La forma de evaluar este eje es la exposición oral y defensa pública del proyecto al finalizar la materia donde participan todos los alumnos en dicha presentación.

o Utilización de técnicas y herramientas de aplicación en de ingeniería en sistemas de información/informática.
¿Cómo se aborda?
Este eje se aborda en todos los trabajos de laboratorio y en el proyecto integrador, donde el alumno debe utilizar técnicas y herramientas para la construcción de prototipos que den solución a la problemática planteada. El alumno debe utilizar diferentes técnicas y herramientas open source tanto dadas por la cátedra como alguna que fuera necesaria para la solución del problema.
¿Cómo se evalúa?
La forma de evaluar este eje es la exposición oral del proyecto integrador y de trabajos de laboratorio donde deben mostrar las herramientas y técnicas utilizadas para construir la solución final.

o Fundamentos para el desempeño en equipos de trabajo.
¿Cómo se aborda?
Este eje se aborda en el trabajo integrador con el proyecto de la materia, donde el alumno debe desarrollar el proyecto en equipos de 3 personas.
¿Cómo se evalúa?
La forma de evaluar este eje es la exposición oral y defensa pública del proyecto, donde se evalua el trabajo realizado y la forma en que trabajaron en equipo.

o Fundamentos para una comunicación efectiva.
¿Cómo se aborda?
Este eje se aborda en la presentación de los trabajos prácticos y de laboratorio donde el alumno debe explicar la solución al problema que resuelve el ejercicio seleccionado por el docente. Además en el proyecto de la materia, donde el alumno debe explicar y comunicar en una clase pública la defensa de su proyecto. En este proyecto no solo es la defensa oral sino también la defensa de los modelos que representan la solución al problema planteado.
¿Cómo se evalúa?
La forma de evaluar este eje es la exposición oral y defensa pública del proyecto, junto con el reporte con los modelos

arquitectónicos que soportan la solución al problema planteado.

o Fundamentos para el aprendizaje continuo.
¿Cómo se aborda?
Este eje se aborda durante toda la materia ya que los alumnos deben defender la solución de los trabajos prácticos y de laboratorio que el docente de la cátedra solicita que el alumno debe explicar.
¿Cómo se evalúa?
La forma de evaluar este eje es la comunicación del trabajo práctico y de laboratorio donde el docente analiza la resolución del mismo.

VIII - Regimen de Aprobación

Existen dos niveles para su aprobación:

a) Regularización solamente: Para regularizar la materia se deberá:
1.- Tener como mínimo un 80% de asistencia a clases prácticas.
2.- Tener los trabajos solicitados por la cátedra aprobados.
3.- Presentación y aprobación del proyecto integrador de laboratorio con nota mayor o igual a 7 (siete).
4.- Tener aprobado con el 60% todos los proyectos solicitados por la catedra, en las fechas solicitadas. En caso de no cumplir, tendrá recuperatorios con fechas diferenciadas.

b) Promoción sin examen final: Para regularizar y aprobar la materia se deberá:
1.- Cumplir con los requisitos a.1, a.2 y a.3.
2.- Haber aprobado los proyectos solicitados con el 70%.
3.- Aprobar la presentación integral final ante la presencia del resto de los alumnos, con una nota mayor o igual a 7 (siete).

Aquellos alumnos que sólo regularicen la materia deberán rendir un examen final, en los turnos establecidos.

Alumnos Libres: Se aceptan alumnos libres debiendo previamente el alumno entregar y aprobar el proyecto integrador de laboratorio.

IX - Bibliografía Básica

[1] Head First Software Architecture. Raju Gandhi, Mark Richards, Neal Ford, O'Reilly Media (2024).
[2] Software Architecture in Practice, 4th Edition. Len Bass, Paul Clements, Rick Kazman. Addison Wesley (2021).
[3] Documenting Software Architectures: Views and Beyond (2nd Edition), Paul Clements, Felix Bachmann, Len Bass, David Garlan, James Ivers, Reed Little, Paulo Merson, Robert Nord, Judith Stafford. Addison-Wesley (2011).
[4] SOA Patterns. Arnon Rotem, Gal, Oz. Manning Publications.
[5] ISO 25000 Calidad del Producto de Software. https://iso25000.com/index.php/normas-iso-25000/iso-25010 (ultimo acceso 2026)
[6] Software Modeling and Design: UML, Use Cases, Patterns, and Software Architectures, Hassan Gomaa, Cambridge University Press (2011).
[7] Cloud Computing, Nayan B. Ruparelia. The MIT Press (2016).
[8] NIST Special Publication 500-291 y NIST Special Publication 500-292. http://www.nist.gov/itl/cloud/index.cfm (ultimo acceso 2026).
[9] ISO/IEC/IEEE 42010: Systems and software engineering — Architecture description: http://www.iso-architecture.org/ (ultimo acceso 2026).
[10] Fundamentals of Software Architecture: An Engineering Approach. Mark Richards, Neal Ford. O'Reilly Media (2020).
[11] Software Architecture Patterns, Mark Richards, Mark Richards. O'Reilly (2015).
[12] Pattern-Oriented Software Architecture: A System of Patterns, Volume 1, Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sornmerlad, Michael Stal. JOHN WlLEY & SONS (2001).
[13] Software Architecture: Foundations, Theory, and Practice. R. N. Taylor, N. Medvidovic, E. M. Dashofy. Wiley (2009).
[14] Security for Software Engineers. James Helfrich. CRC Press (2019).

[15] Java™ Coding Guidelines: 75 Recommendations for Reliable and Secure Programs. Fred Long, Dhruv Mohindra, Robert C. Seacord, Dean F. Sutherland, David Svoboda. Addison Wesley (2014).
[16] Application Security Program Handbook: A guide for software engineers and team leaders, Version 2. Derek Fisher. Manning Publications (2021).
[17] Hands-on cybersecurity for architects plan and design robust security architectures. Aslaner, Milad_Rerup, Neil. Packt Publishing (2018).
[18] Blockchain for Dummies. Tiana Laurence. Wiley (2019).
[19] Ethereum Smart Contract Development in Solidity. Gavin Zheng, Longxiang Gao, Liqun Huang, Jian Guan. Springer (2021).
[20] Blockchain Federal Argentina, www.bfa.ar (ultimo acceso 2026).
[21] Architecture for Blockchain Applications. Xiwei Xu, Ingo Weber, Mark Staples. Springer (2019).

X - Bibliografia Complementaria

[1] Interacción con Inteligencia Artifical:www.chatpgt.com / gemini.google.com / www.claude.com (ultimo acceso 2026).
[2] Software Architecture Metrics: Case Studies to Improve the Quality of Your Architecture. Christian Ciceri, Dave Farley,
[3] Neal Ford, Andrew Harmel-Law, Michael Keeling, Carola Lilienthal, João Rosa, Alexander von Zitzewitz, Rene Weiss & Eoin Woods. O'Reilly Media (2022).
[4] Evaluating Software Architectures: Methods and Case Studies. Paul Clements, Rick Kazman, Mark Klein. Addison-Wesley (2005).
[5] Software Architecture: Perspectives on an Emerging Discipline. Mary Shaw, David Garland. Prentice Hall (1996).
[6] Models for Evaluating and Improving Architecture Competence, Len Bass, Paul Clements, Rick Kazman, Mark Klein. http://resources.sei.cmu.edu/asset_files/TechnicalReport/2008_005_001_14972.pdf (ultimo acceso 2026).
[7] Pattern-Oriented Software Architecture Volume 1: A System of Patterns. Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal. Wiley (1996).
[8] Patterns of Enterprise Application Architecture. Martin Fowler (2002). Addison-Wesley Professional.
[9] MDA Explained: The Model Driven Architecture(TM): Practice and Promise, Anneke G. Kleppe, Jos Warmer, and Wim Bast. Addison-Wesley Professional (2003).
[10] Pattern-Oriented Software Architecture For Dummies. Robert Hanmer. Wiley (2012).
[11] SOA Design Patterns, Thomas Erl. Prentice Hall (2009).
[12] SOA with REST: principles, patterns & constraints for building enterprise solutions with REST. Erl, Thomas. Prentice Hall (2013).
[13] Designing Software Architectures: A Practical Approach (SEI Series in Software Engineering). Humberto Cervantes, Rick Kazman. Addison-Wesley (2016).
[14] Service-Oriented Architecture: Analysis and Design for Services and Microservices. Thomas Erl. Prentice Hall (2016).
[15] Building Microservices: Designing Fine-Grained Systems. Sam Newman. O'Reilly Media (2015).
[16] Cloud Computing For Dummies. Judith Hurwitz, Robin Bloor, Marcia Kaufman, Fern Halper. Wiley Pub (2010).

XI - Resumen de Objetivos

Reconocer, describir, y desarrollar arquitecturas de software

XII - Resumen del Programa

Lenguajes de Descripción Arquitectónica
Análisis y Diseño Arquitectónico
Patrones Arquitectónicos
Arquitecturas Orientadas a Servicio
Arquitectura con Seguridad
Transacciones en SOA
Arquitectura para Aplicaciones Blockchain

XIII - Imprevistos

Modalidad de cursado Presencial.

XIV - Otros

Para comunicarse con el docente responsable de la materia, escribir al email: driesco@unsl.edu.ar

ELEVACIÓN y APROBACIÓN DE ESTE PROGRAMA	
	Profesor Responsable
Firma:	
Aclaración:	
Fecha:	