INSTITUTO/S: Tecnología e Ingeniería

CARRERA/S: Licenciatura en informática

MATERIA: Arquitectura de software I

NOMBRE DEL RESPONSABLE DE LA ASIGNATURA: Ricardo Brea

EQUIPO DOCENTE: Ricardo Brea

CUATRIMESTRE: 1ro

AÑO: 4to

PROGRAMA N°: 35 **(Aprob. Por Cons.Directivo fecha XX)**

Instituto/s: Tecnología e Ingeniería
Carrera/s: Licenciatura en informática
Nombre de la materia: Arquitectura de software I
Responsable de la asignatura y equipo docente: Ricardo Brea
Cuatrimestre y año: 1ro cuat. del 4to año
Carga horaria semanal: 4 hs
Programa N°: 35
Código de la materia en SIU: 778

Arquitectura de software I

1. Fundamentación

La arquitectura de software es una disciplina que integra los conocimientos adquiridos en muchas de las asignaturas de la carrera. Aportando sinergia a los distintos componentes de un sistema.
La materia se centra en los elementos fundamentales de la arquitectura como ejes conductores de los distintos temas. Incorporando ejemplos prácticos utilizando tecnologías actuales. Con el objetivo de aportar herramientas para un análisis crítico entre distintas alternativas y opciones de acuerdo a posibles escenarios que se pueden encontrar en la práctica profesional.

2. Propósitos y/u objetivos

Propósitos

- Articular los conocimientos adquiridos en materias correlativas, tanto de programación como en sistemas de información y calidad.
- Fomentar el espíritu crítico para la toma decisiones.
- Transmitir la importancia de evaluar los aspectos de seguridad del sistema de información al momento de diseñar la arquitectura.

Objetivos

Se espera que el/la alumno/a:

- Comprenda la importancia de la arquitectura de software y su relación con el diseño a lo largo del ciclo de vida de desarrollo.
- Conozca el rol y las responsabilidades del arquitecto de software y su relación con los diversos responsables del proyecto.

- Identifique las mejores opciones de definición de arquitectura de acuerdo a los requerimientos del sistema de información a desarrollar.
- Conozca la evolución de la arquitectura de software en conjunción metodologías de desarrollo.
- Conozca las diferentes herramientas conceptuales para ayudar a definir una arquitectura de software.
- Sea capaz de determinar la mejor arquitectura de acuerdo a los requerimientos del sistema y las restricciones del proyecto.
- Pueda comunicar en forma efectiva la arquitectura de software, la toma de decisiones, su relación con los requerimientos funcionales y la importancia de los requerimientos no funcionales.
- Conozca la importancia de evaluar los aspectos de seguridad del sistema de información al momento de diseñar la arquitectura.
- Conozca el estado actual de los patrones arquitectónicos más usados para diferentes tipos de sistemas de software.
- Conozca las herramientas para ayudar en la implementación y verificación del diseño arquitectónico especificado.

3. Programa sintético:

Arquitectura de software y arquitectura de sistemas. Proceso de definición y evolución de una arquitectura en diferentes metodologías de desarrollo. Requerimientos funcionales y no funcionales, restricciones, influencias, entorno social y técnico, estándares, herramientas disponibles. Objetivos de una arquitectura. Estilos arquitectónicos. Arquitectura de dominio. Arquitectura y Diseño. Patrones. Patrones arquitecturales para la interfaz de usuario. Integración con el dominio. Internacionalización. Arquitecturas orientadas a servicios. Arquitecturas extensibles. Arquitecturas basadas en plugins. Arquitecturas de seguridad. Estrategias de verificación de arquitecturas. Arquitecturas concurrentes y distribuidas. Herramientas tecnológicas para soportar las decisiones arquitectónicas.

4. Programa analítico

4.1 Organización del contenido:

Unidad 1: Introducción a la arquitectura de software

- Concepto de Arquitectura de Software y su Relación con el Diseño.
- Responsabilidades del Arquitecto.
- Principios de Arquitectura.
- Requerimientos no funcionales y restricciones

Unidad 2: Modelos arquitectónicos estructurales

- Sinergia, acoplamiento, cohesión y quantum de una arquitectura
- Diagrama de arquitectura
- Método de comparación de arquitecturas

    - Características de una arquitectura
    - Matriz de Pugh
- Patrones de diseño estructurales:
    - Por capas
    - Microkernel
    - Orientada a servicios y microservicios
    - Orientada a eventos
    - Space-based

Unidad 3: Integración de los componentes de una arquitectura

- Componentes: lógica de dominio, interfaz de usuario, persistencia, seguridad, etc.
- Cohesión, acoplamiento y comunicación entre los componentes de la arquitectura.
- Integración basada en aspectos (AOP) y 12 Factor APPS
- Transporte de la información.
- Manejo de transacciones.

Unidad 4: Capa de negocio, lógica, patrones y frameworks

- Cohesión y acoplamiento a nivel de componentes
- Concepto de framework
- SOLID
    - Responsabilidad única.
    - Principio de Abierto/Cerrado.
    - Sustitución de Liskov.
    - Segregación de la interface.
    - Inversión de dependencia.
- Organización de la Lógica de Negocio (Arquitectura no Intrusiva, Domain Driven Design).

Unidad 5: Implementación de persistencia

- Diseños de capas de persistencia aptas para testeo unitario y simulación.
- Patron Active Record.
- ORM: Entity Framework.
- Configuración. Convención sobre configuración

Unidad 6: Desarrollo de la capa de presentación y patrones de interface de usuario

- Patrón Observer
- Internacionalización
- Accesibilidad y web semántica.
- Clientes livianos y pesados
- Integración con el dominio.
    - MVC
    - MVVM

Unidad 7: Arquitecturas de seguridad.

- Autenticación y Autorización.

- Acceso basado en roles o RBAC.
- Patrones y estándares Single Sign-on con OIDC y SAML.
- Perspectivas de seguridad de una aplicación: seguridad web, sistema operativo, base de datos, middleware.
- Seguridad a nivel de transporte / TLS
- Seguridad multifactorial: OTP, WebAuthn

Unidad 8: Diseño de API

- Diseño de API estáticas, Fluent API.
- Diseño de HTTP/ WEB APIs
    - SOAP
    - REST
    - ODATA
    - gRPC
    - Buenas prácticas y rfc7807
    - OpenAPI
    - Versionado

Unidad 9: Integración de aplicaciones

- Clasificación de los mecanismos de integración: base de datos, dominio, servicios, interfaz de usuario.
- Estrategias de integración apropiadas para ambientes compatibles e incompatibles entre sí
- Integración sincrónica y asincrónica.
    - Colas de mensajes.
    - Callbacks.
    - Arquitecturas orientadas a servicios. Web services.
- Patrones para la integración: punto a punto, middleware, Enterprise Service Bus.
- Definición de interfaces y conectores.
- Definición de procesos de negocio.
- Coreografía y orquestación.
- Manejo de transacciones y compensaciones.
- Servicios de directorio (JNDI, UDDI, etc).
- Seguridad M2M

Unidad 10: Estrategias de verificación de arquitecturas

- Procesos formales de evaluación de la arquitectura y de los requerimientos no funcionales.
- Aseguramiento de la adecuación de un sistema a la arquitectura definida, automatización de aseguramiento.
- Herramientas arquitecturales para la automatización de pruebas de dominio.

Unidad 11: Herramientas tecnológicas para soportar las decisiones arquitectónicas.

ADR: Registro de decisiones arquitectónicas

Unidad 12: Cuestiones organizacionales, humanas y sociales relativas a la arquitectura de software.

- Relación entre la arquitectura y el grupo de desarrollo.
- Diciplinas que complementan una arquitectura para poder llevar adelante un desarrollo grande y/o complejo.
    - DevOps
    - DevSecOps

Unidad 13: Arquitecturas concurrentes y distribuidas.

- Objetos distribuidos.
- Máquinas virtuales distribuidas.
- Programación orientada a agentes.

4.2 Bibliografía y recursos obligatorios:

Richards, Mark; Ford, Neal. (2020). Fundamentals of Software Architecture (4ta Edición 2021). O'Reilly Media, Inc.

Bass,Len; Clements, Paul; Kazman, Rick. (2021). "*Software Architecture in Practice*". 4ta Edicion. Addison-Wesley Professional.

Fowler, Martin; Rice, David; Foemmel, Matthew; Hieatt, Edward; Mee, Robert; Stafford, Randy. (2002). Patterns of Enterprise Application Architecture. Addison-Wesley.

Hohpe, Gregor; Woolf, Bobby. (2003). Enterprise Integration Patterns: Designing, Building and Deploying Messaging Solutions. Addison-Wesley.

4.3 Bibliografía optativa:

Martin, Robert Cecil. (2018). Clean Architecture: A CRAFTSMAN'S GUIDE TO SOFTWARE STRUCTURE AND DESIGN.

Richards, Mark; Ford, Neal; Sadalage, Pramod y Dehghani, Zhamak. (2021). Software Architecture: The Hard Parts. O'Reilly Media, Inc.

Fowler, Martin. (2002). Who Needs an Architect. Martin Fowlers's web. https://martinfowler.com/ieeeSoftware/whoNeedsArchitect.pdf

Richards, Mark. (2016**).** Microservices vs. Service-Oriented Architecture. Oreilly web. https://www.oreilly.com/radar/microservices-vs-service-oriented-architecture/

Mell, Peter; Grance, Timothy. (2011). "The NIST Definition of Cloud Computing". NIST Technical Series Publications.
https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf

Fowler, Martin. (2004). Presentation Model. Martin Fowlers's web. https://martinfowler.com/eaaDev/PresentationModel.html

5. Metodología de enseñanza:

Las clases están divididas en dos mitades. Una mitad teórica y la otra mitad práctica. Durante la mitad teórica se presenta el contenido formal del próximo tema práctico a realizar. Durante las porciones práctica se muestran ejemplos prácticos de la teoría que luego deberán ser aplicación en el trabajo practico final.

Al finalizar cada clase las/os alumnos disponen de un cuestionario de auto evaluación. En base a estos cuestionarios se hará la evaluación teórica.

Plan de trabajo en el campus:

En el aula virtual se propondrá material educativo;

- Presentaciones que el/la docente emplea durante las clases.
- Vinculo al repositorio de código (abierto) con los ejemplos prácticos presentados en clase.
- Videos y textos introductorios o ampliatorios de los temas tratados.
- Guías de trabajos prácticos y todo material que el/la alumno/a deba entregar.
- También incluirá un foro de consultas, el programa, el cronograma de la asignatura y cualquier tipo de información adicional que sea necesaria.

6. Actividades de investigación y extensión (si hubiera)

No Aplica

7. Evaluación y régimen de aprobación

7.1 Aprobación de la cursada

Para aprobar la cursada y obtener la condición de regular, el régimen académico establece que debe obtenerse una nota no inferior a cuatro (4) puntos. Todas las instancias evaluativas deberán tener una instancia de recuperatorio. Podrán acceder a la administración de esta modalidad solo aquellos y aquellas estudiantes que hayan obtenido una nota inferior o igual a 6 (seis) puntos en el examen parcial.

Siempre que se realice una evaluación de carácter recuperatorio, la calificación que los/as estudiantes obtengan reemplazará la calificación obtenida en el examen que se ha recuperado y será la considerada definitiva a los efectos de la aprobación.
El/La alumno/a deberá poseer una asistencia no inferior al 75% en las clases presenciales.

En cuanto a las cursadas de materias virtuales se requerirá que el estudiante ingrese al aula virtual como mínimo una vez por semana.

7.2 Aprobación de la materia

La materia puede aprobarse por promoción, evaluación integradora, examen final o libre.

Promoción directa: tal como lo establece el art°17 del Régimen Académico, para acceder a esta modalidad, el/la estudiante deberá aprobar la cursada de la materia con una nota no inferior a siete (7) puntos, no obteniendo en ninguna de las instancias de evaluación parcial menos de seis (6) puntos, sean evaluaciones parciales o recuperatorios. El promedio estricto resultante deberá ser una nota igual o superior a siete (7) sin mediar ningún redondeo.

Evaluación integradora: tal como lo establece el art°18 del Régimen Académico, podrán acceder a esta evaluación aquellos/as estudiantes que hayan aprobado la cursado con una nota de entre cuatro (4) y seis (6) puntos.
La evaluación integradora tendrá lugar por única vez en el primer llamado a exámenes finales posterior al término de la cursada. Deberá tener lugar en el mismo día y horario de la cursada y será administrado, preferentemente, por el/la docente a cargo de la comisión. Se aprobará tal instancia con una nota igual o superior a cuatro (4) puntos, significando la aprobación de la materia.
La nota obtenida se promediará con la nota de la cursada.

Examen final: Instancia destinada a quienes opten por no rendir la evaluación integradora o hayan regularizado la materia en cuatrimestres anteriores. Se evalúa la totalidad de los contenidos del programa de la materia y se aprueba con una calificación igual o superior a cuatro (4) puntos. Esta nota no se promedia con la cursada.

7.3 Criterios de calificación

La calificación de cada evaluación se determinará en la escala 0 a10, con los siguientes valores: 0, 1, 2 y 3: insuficientes; 4 y 5 regular; 6 y 7 bueno; 8 y 9 distinguido; 10 sobresaliente.

8. Cronograma

CLASE	UNIDAD	TIPO DE CLASE	MODALIDAD	ACTIVIDAD
1	1	TEORICA Y PRACTICA	VIRTUAL	INTRODUCCION A DOCKER
2	2	TEORICA Y PRACTICA	VIRTUAL	PRACTICA CON DOCKER COMPOSE
3	3	TEORICA Y PRACTICA	**PRESENCIAL**	CUESTIONARIO DE AUTO-EVALUACION
4	4	TEORICA Y PRACTICA	VIRTUAL	PRACTICA CON .NET

5	5	TEORICA Y PRACTICA	VIRTUAL	PRACTICA CON .NET Y ENTITY FRAMEWORK
6	6	TEORICA Y PRACTICA	VIRTUAL	PRACTICA CON .NET Y ANGULAR
7	7	TEORICA Y PRACTICA	VIRTUAL	IMPLEMENTACION DE ARQUITECTURA DE SEGURIDAD
8	**PARCIAL**		**PRESENCIAL**	-
9	8	TEORICA Y PRACTICA	VIRTUAL	PRACTICA CON .NET Y REPORTE DE AVANCE TP
15	**RECUPERATORIO**		**PRESENCIAL**	-
10	9	TEORICA Y PRACTICA	VIRTUAL	PRACTICA .NET + DOCKER COMPOSE REPORTE DE AVANCE TP
11	10	TEORICA Y PRACTICA	VIRTUAL	PRACTICA SELENIUM / POSTMAN Y REPORTE DE AVANCE TP
12	11	TEORICA Y PRACTICA	VIRTUAL	REALIZAR EL ADR DEL TRABAJO PRACTICO
13	12	TEORICA Y PRACTICA	VIRTUAL	CUESTIONARIO DE AUTO-EVALUACION Y REPORTE DE AVANCE TP
14	13	TEORICA Y PRACTICA	**PRESENCIAL**	PRESENTACION TP
16	**NOTAS Y FEEDBACK DE LA MATERIA**		**PRESENCIAL**	-