FCEFyN
FACULTAD DE CIENCIAS EXACTAS, FÍSICAS y NATURALES

UNC Universidad Nacional de Córdoba

Asignatura: **Algoritmos y Estructuras de Datos**

Código:	RTF	7
Semestre: 2	Carga Horaria	96
Bloque: Ciencias Básicas	Horas de Práctica	48
Departamento: Computación		

Correlativas:
- Estructuras Discretas
- Fundamentos de Programación

Contenido Sintético:
- Historia y visión general
- Herramientas relevantes, estándares y/o restricciones de ingeniería
- Estructuras de datos lineales y no lineales
- Análisis de algoritmos básicos
- Estrategias algorítmicas
- Algoritmos clásicos para tareas comunes de ordenamiento y búsqueda
- Análisis y diseño de algoritmos de aplicación específicos
- Complejidad algorítmica

Competencias Genéricas:
- CG1: Identificar, formular y resolver problemas de ingeniería. Media
- CG4: Utilizar de manera efectiva las técnicas y herramientas de aplicación en ingeniería. Alta

Aprobado por HCD: NNNN-HCD-AAAA RES: Fecha: DD/MM/AAAA

Competencias Específicas:
CE1.3 Conocer, desarrollar nuevos e implementar algoritmos y estructuras de datos. Alta

Presentación

Los algoritmos y las estructuras de datos son esenciales a la computación, ambos forman parte de los conceptos centrales en que esta ciencia se basa. Las estructuras de datos han sido desarrolladas para representar adecuadamente la información de ciertos tipos de problemas de procesamiento de datos, de forma tal de poder implementar adecuadamente las soluciones de estos. Los algoritmos realizan el procesamiento sobre los datos, por lo que la combinación adecuada de algoritmos con estructuras permiten desarrollar soluciones informáticas eficientes.
La asignatura comprende la presentación y usos de las estructuras de datos y algoritmos de uso frecuente y generalizado y están destinados a que el alumno adquiera conocimientos y capacidades su aplicación. Además es teniendo en cuenta el uso eficiente tanto de la memoria como de las unidades de procesamiento para el abordaje de las distintas situaciones en las cuales se utilizan dichos algoritmos y estructuras en los sistemas de computación.
El desarrollo de los contenidos es gradual en su complejidad para facilitar el aprendizaje de los mismos y en toda oportunidad se presentan acompañados de su uso concreto de forma tal que los fundamentos de estos puedan contrastarse inmediatamente con su utilización.
Es tenida en cuenta la ubicación de la asignatura dentro del trayecto formativo de los alumnos en cuanto a los requisitos previos para poder tomar el curso adecuadamente y el formato de presentación de los temas. Además, en la preparación y capacitación del alumnado se tienen en cuenta las competencias necesarias para poder abordar las asignaturas posteriores a la presente a lo largo del plan de estudios de la carrera de Ingeniería en Computación.

Contenidos

Unidad 1: Punteros y memorias
1.1 Variables, memoria y direcciones
1.2 Paso de parámetros mediates punteros
1.3 Uso dinámico de la memoria. Reserva y liberación de memoria
1.4 Aritmética de Punteros
1.5 Usos y Ejemplos

Unidad 2: Conceptos básicos de objetos y clases
2.1 Clases y Objetos
2.2 Métodos y atributos privados y públicos
2.3 Herencia
2.4 Constructores
2.5 usos y ejemplos

Unidad 3: Estructuras lineales
3.1 Arreglos y Vectores
3.2 Listas simplemente enlazadas
3.3 Pilas
3.4 Colas
3.5 Usos y Ejemplos

Unidad 4: Complejidad algorítmica
4.1 Complejidad temporal y espacial
4.2 Funciones asintóticas Notación Big O, Omega O y Theta O

4.3 Ejemplos

Unidad 5: Algoritmos de ordenamiento y búsqueda
5.1 Algoritmos de ordenamiento de Orden cuadrático
5.2 Algoritmos de ordenamiento de Orden logarítmico
5.3 Comparación y usos apropiados de cada algoritmo
5.4 Algoritmos de búsqueda: lineal, binaria y otros
5.5 Usos y ejemplos

Unidad 6: Estructuras tipo árbol
6.1 Definición y formas de implementar árboles.
6.2 Árbol binario y m-ario
6.3 Balances y recorridos de arboles binarios
6.4 Otros tipos de árboles usuales
6.5 Usos y ejemplos

Unidad 7: Otras estructuras de datos
7.1 Tablas y funciones de dispersión
7.2 Conjuntos y bolsas (set y bags)
7.3 Mapas de Bits

Unidad 8: Grafos
8.1 Definición y formas de implementar grafos
8.2 Algoritmos sobre grafos dirigidos
8.3 Algoritmos sobre grafos no dirigidos
8.4 Usos ejemplos

Unidad 9: Estrategias algorítmicas
9.1 Algoritmos “Divide y Vencerás”
9.2 Algoritmos voraces
9.3 Algoritmos de *backtracking*

Unidad 10: Visión de conjunto e historia de los algoritmos y las estructuras de datos
10.1 Revisión histórica de los algoritmos y las estructuras de datos

Metodología de enseñanza

Los contenidos de la asignatura serán impartidos bajo una metodología de clases teórico - prácticas con la activa participación del alumnado en las mismas. La asignatura tiene una impronta muy fuerte en el diseño y elaboración de programas, en cuya elaboración los contenidos teóricos son necesarios para la resolución de los mismos pero la importancia de su conocimiento es efectivamente adquirida en oportunidad del desarrollo de los programas, por lo que la combinación de conceptos teóricos con programas que los implementen genera un alto impacto en la capacidades que los alumnos adquieren.

Las clases son impartidas combinando la presentación del tema y su desarrollo junto con la exposición de programas concretos que los implementen, de forma tal que el alumno reconoce inmediatamente el impacto de la aplicación de los conceptos en el desarrollo de

algoritmos, su ventajas, alcances y limitaciones. Dicho esto, en toda oportunidad está presente un entorno de desarrollo y ejecución de programas que determina la condición de una metodología teórico-práctica.

Evaluación

La metodología de la evaluación es por medio del método de la resolución de problemas informáticos y el desafío de diseñar e implementar la solución del mismo. A lo largo del curso se presentan tres proyectos a desarrollar bajo la metodología precedente, donde para su solución se deben aplicar conceptos impartidos en clases. La solución de los mismos puede ser encarada con enfoques diferentes por lo que el alumnado tiene que desarrollar habilidades para comprender el problema y solucionarlo basado en sus conocimientos adquiridos.

A los efectos de evaluar el proceso de aprendizaje del alumno, no solo se tendrá en cuenta la aplicación del algoritmo y la estructura de datos utilizada, sino que también la forma de haber abordado la solución del problema, la justificación de la técnica utilizada y la claridad a la hora de comunicar todo el proceso constructivo del proyecto

Si bien para el desarrollo de los proyectos pueden utilizarse diversos lenguajes de programación, se solicita que en la resolución sea utilizado un lenguaje de programación compilado y usual en el desarrollo de programación de sistemas embebidos para que sea parte de la formación integral del alumno en su carrera.

Condiciones de aprobación

Se presentan tres(3) proyectos a desarrollar a lo largo del curso. Para su ejecución se solicita trabajo en equipo y que la presentación y defensa de los mismos se realice en forma conjunta por parte de los miembros del equipo frente al docente. También se solicita un documento escrito que exponga el diseño y la solución desarrollada.
Para alcanzar la regularidad, deben aprobarse dos(2) de los tres proyectos y asistencia a clases del 80%. La promoción se logra aprobando los tres proyectos.

La nota final en caso de promoción es el promedio de las tres evaluaciones. Para el caso del alumno regular, la nota es el promedio ponderado de las dos evaluaciones aprobadas durante el cursado, a un 25% cada una de estas, y la nota del examen final ponderado al 50%. Al alumno libre le corresponde la nota del examen libre.

Actividades prácticas y de laboratorio

Al ser la metodología de clases teóricas - prácticas, el alumno se acompaña en las mismas por computadoras donde desarrollan los ejemplos planteados y las actividades prescriptas.

Desagregado de competencias y resultados de aprendizaje

Al aprobar el curso el alumno adquiere una serie de habilidades y conocimientos técnicos propios de la carrera como también habilidades blandas. Dentro de los primeros podemos citar la identificación de problemas y diseño y desarrollo de soluciones propios de la Ingeniería en Computación, la aplicación de algoritmos y estructuras de datos de uso frecuente en computación, el diseño, desarrollo y depuración de programas y competencias de adquisición y aplicación de nuevos conceptos. Uso de lenguajes de programación compilables. Dentro de las habilidades genéricas adquiridas podemos citar la capacidad de trabajo en equipo y la documentación y comunicación de los trabajos técnicos realizados. Estos resultados de aprendizaje se evaluarán en las instancias de evaluación previstas mediante el uso de rúbricas diseñadas al efecto.

Bibliografía

Principal

Estructuras de datos con C++. Objetos, abstracciones y diseño, E. Koffman & P.Wolfgang

Introduction to Algorithms (4th Edition), T. Cormen, C. Leiserson, R. Rivest,& C.Stein

Adicional

Algorithms in C++, Robert Sedgewick & Kevin Wayne

Algorithms and Data Structures, Niklaus Wirth

The Design and Analysis of Computer Algorithms (3rd Ed.), A. Aho, J. Hopcroft, J. Ullman

FCEFyN
FACULTAD DE CIENCIAS EXACTAS, FÍSICAS y NATURALES

UNC Universidad Nacional de Córdoba

Asignatura: **Arquitectura de Computadoras**

Código:	RTF	9
Semestre: 8	Carga Horaria	96
Bloque: Tecnologías Aplicadas	Horas de Práctica	48
Departamento: Computación		

Correlativas:

- Sistemas Operativos

Contenido Sintético:

- Historia y descripción general.
- Arquitectura y set de instrucciones.
- Medición del rendimiento
- Organización del procesador
- Organización y arquitectura del sistema de memoria
- Paralelismo a nivel de instrucciones
- Arquitecturas multi-núcleo, multiprocesadores
- Computadoras Vectoriales.
- Arquitecturas de sistemas distribuidos, arquitecturas alternativas
- Caso de estudio realizado con herramientas utilizadas en el diseño de arquitectura de computadoras de diseño de un procesador de arquitectura abierta, haciendo uso de Verilog.

Competencias Genéricas:

- CG1: Identificar, formular y resolver problemas de ingeniería.
- CG2: Concebir, diseñar y desarrollar proyectos de ingeniería (sistemas, componentes, productos o procesos).
- CG5: Contribuir a la generación de desarrollos tecnológicos y/o innovaciones tecnológicas.
- CG6: Desempeñarse de manera efectiva en equipos de trabajo.
- CG7: Comunicarse con efectividad.
- CG10: Actuar con espíritu emprendedor.

Aprobado por HCD: NNNN-HCD-AAAA RES: Fecha: DD/MM/AAAA

Competencias Específicas:

- CE2 Diseñar, desarrollar e implementar diversas Arquitecturas de Computadoras y todos los subsistemas relacionados.
  - CE2.1 Conocer profundamente los principios fundamentales de la arquitectura de computadoras, incluyendo la estructura de la CPU, la memoria, el sistema de entrada/salida y la interconexión entre los componentes principales.
  - CE2.2 Utilizar correctamente lenguajes de descripción de hardware,
  - CE2.3 Poseer habilidades en la escritura y comprensión de código HDL, así como en la simulación y verificación del diseño.
  - CE2.4 Analizar, modelar, diseñar, desarrollar y probar circuitos electrónicos digitales
  - CE2.5 Utilizar técnicas de optimización de rendimiento de las CPUs.
  - CE2.6 Analizar, modelar y diseñar microarquitecturas y pipelines.
  - CE2.7 Conocer e implementar conceptos de organización y gestión de la memoria en un sistema de computadoras.
  - CE2.8 Conocer e implementar los diferentes esquemas de interconexión utilizados en sistemas de computadoras,
  - CE2.9 Conocer e implementar los protocolos de comunicación y las técnicas de enrutamiento utilizadas en estos sistemas.
  - CE2.10 Manejar conceptos de sistemas operativos y su interacción con la arquitectura de computadoras., recursos del sistema, interrupciones y multiprogramación.

Presentación

Objetivos:

- Adquirir: La dimensión tecnológica necesaria para la comprensión, construcción e implementación de las estructuras internas, su organización e interrelación de computadores digitales.
- Comprender: Todas las reglas y principios tecnológicos que nos permita independizarnos de los dispositivos desarrollados en el momento, como así también una profunda práctica en el manejo de dispositivos actuales para así afianzar los conceptos teóricos y un manejo acabado de la realidad tecnología.

LINEAMIENTOS GENERALES

La presente asignatura es una actividad curricular que pertenece al octavo semestre de la carrera de Ingeniería en Computación.

A través del cursado de la asignatura el alumno desarrollará las siguientes competencias:

- Comprensión, construcción e implementación de las estructuras internas, su organización e interrelación de computadores digitales.
- Identificar y construir estructuras internas, su organización e interrelación de computadores digitales con el fin de comprenderlos e implementarlos.
- Mediante el análisis identificar problemas de implementación en distintas arquitecturas.
- Saber identificar los distintos dominios de aplicación de las distintas arquitecturas.

- Conocer la interacción de la arquitectura con su entorno; se pondrá énfasis en la detección y prevención de situaciones problemáticas tales como la performance en pipeline, la eficiencia en threads y en procesos (paralelismo temporal y espacial).
- Buscar la eficiencia y la eficacia del sistema haciendo uso de implementaciones paralelas.
- Mediante el uso de herramientas y criterios, realizar el diseño de arquitecturas paralelas, basado en la definición de thread, procesos, e interacciones.
- Realizar la implementación de un sistema con aplicaciones paralelas

En los últimos años se han producido profundos cambios en las arquitecturas de computadoras. Las arquitecturas súper escalar, múltiples core y NUMA; comenzaron a dominar el mercado, con lo que han emergido un sin numero de nuevos conceptos como predecodificación, Dynamic Brach, Prediction, instruction reordering, speculation execution of loads e implementación de procesadores CISC por un RISC core. Estas ideas han sido implementadas en las principales arquitecturas de procesadores. Más aun, existen lenguajes que explotan estos beneficios, por lo que se hará énfasis en:

- Arquitectura computacionales que facilitan y promueven la programación en paralelo
- Herramientas de depuración que facilitan las pruebas.

El dictado se orienta a capacitar al alumno para Identificar y construir modelos de sistemas donde este involucradas las arquitecturas paralelas, lograr sistemas con eficiencia y eficacia en los cuales las arquitecturas sean concordantes con los requerimientos. Todo materializando un diseño, implementación y las pruebas de sistema.

Contenidos

Unidad 1. Introducción
Tecnología y arquitectura.
Medidas de la calidad de las arquitecturas.
Medidas de performance, precio y consumo.
Factores de influencia de las distintas arquitecturas. Ejemplos.
Unidad 2. Sistema de memoria
Sistema de memoria tecnologias y costo.
Memoria principal. Jerarquia de memoria. Ejemplos.
Unidad 3. Set de Instrucciones
Representación de datos.
Tipos y precisión. Set de registros.
Tipo de instrucciones. Técnicas de direccionamiento.
Diseño de set de instrucciones. Ejemplos.
Unidad 4. Paralelismo a nivel de instrucciones (ILP).
Conceptos basicos de pipelining. Técnicas de diseño de pipeline. Control de pipeline.
Tecnicas de compilador para explotar ILP. Ejemplos.
Unidad 5. Planificacion dinamica
Prediccion de saltos. Especulacion basada en hardware.
Paralelismo a nivel de hilos. Sistemas distribuidos.
Unidad 6. Arquitecturas vectoriales
Procesador vectorial genérico. SIMD. Ejemplos.
Unidad 7. Arquitecturas de proposito general.
MIPS. RISC V. ARM-Cortex A53. Intel Core i7.

Unidad 8. Multiprocesadores
Multiprocesadores perfomance.
Multiprocesadores interconexión.
Coherencia en el cache de multiprocesadores.
Algoritmos de multiprocesadores.
Simples paralelismos, técnicas de sincronización, transformación de algoritmos seriales en paralelos. Ejemplos.
Unidad 9. Arquitecturas Alternativas
Arquitecturas de flujo de datos.
Deep Neural Networks. Tensor Processor Unit. GPUs.

Metodología de enseñanza

Las clases impartidas son teóricas, prácticas y de laboratorio.
Las actividades teóricas se realizan a través de exposiciones dialogadas del docente orientadas a desarrollar en los alumnos la capacidad de saber identificar la concurrencia, la interacción de la aplicación con su entorno, el diseño y las pruebas de sistemas.
Durante el desarrollo de los Trabajos Prácticos se realizan actividades que le permiten al estudiante poner en práctica las habilidades y verificar los criterios y técnicas de modelado, diseño y prueba.
Por otra parte en las clases de Laboratorio el alumno verifica, a través de la implementación, el funcionamiento de sistemas y realizará los casos de pruebas.

Evaluación

La evaluación formativa se realizará mediante la realización de las actividades prácticas. El desarrollo de las competencias se evaluará de forma contínua mediante rúbricas, construida en base a los resultados de aprendizaje propuestos. Se realizarán también evaluaciones parciales.

<u>Condiciones para la promoción de la materia</u>
1. Tener aprobadas las materias correlativas.
2. Asistir al 80% de las clases teóricas y prácticas.
3. Rendir y aprobados los dos parciales con 50% , el alumno podrá recuperar una vez cada parcial.
4. Presentar y aprobar los trabajos que se exijan durante el desarrollo de los trabajos prácticos.
5. Aprobar los trabajos de Laboratorio.
6. Aprobar un coloquio integrador.
7. Alcanzar el criterio de aceptación mínimo para los resultados de aprendizaje propuestos

Los alumnos que cumplan las exigencias referidas en los puntos 1 al 7 serán considerados promocionados. Los alumnos que cumplan las exigencias referidas en los puntos 1 al 5 serán considerados regulares. El resto será considerado libre.

Actividades prácticas y de laboratorio

Actividades Prácticas

1. Herramientas e implementar sistemas digitales. Bases de Verilog, Estructuras de Verilog, Concurrencia y Secuencialidad, Inferencia de elementos de memoria (voluntaria e involuntaria), Instancias de arquitectura, multi-instancias, verificación
2. Herramientas e implementación de sistemas paralelos en el tiempo. Otras estructuras y sentencias, Subprogramas: funciones y procedimientos, Tipos y Packages aritméticos, Uso eficiente de operadores aritméticos, Diseño de máquinas de estado con Verilog. Diferentes técnicas para diseñar arquitecturas mas eficientes
3. Implementación de sistemas paralelos en el tiempo. Herramientas para el diseño jerárquico, Reglas generales de diseño para optimizar la performance, Manejo de archivos en Verilog, Modelado del tiempo en Verilog, Verificación funcional de modelos Verilog, Ejemplos con Xilinx en Verilog.

Actividades de Laboratorio

1. Implementación de distintos diseños por clases
2. Implementación de un ALU
3. Implementación de un modulo de comunicacion UART
4. Implementación del pipeline simplificado de un procesador MIPS / RISC

Desagregado de competencias y resultados de aprendizaje

Al final del curso se espera que un estudiante alcance los siguientes habilidades:

- Interpreta correctamente el dominio de un problema.
- Posee las habilidades comunicacionales suficientes para realizar las preguntas necesarias para desarrollar un diseño completo ajustado a las necesidades del dominio presentes y futuras.
- Cumple en tiempo con los compromisos asumidos con su equipo de trabajo.
- Respeta las pautas de trabajo establecidas en clase para las actividades de equipo.
- Comprender el funcionamiento de los procesadores y computadoras modernas.
- Identificar y construir estructuras internas del computador, su organización e interrelación de computadores digitales.
- Identificar problemas de implementación en distintas arquitecturas.
- Identificar los distintos dominios de aplicación de las distintas arquitecturas.
- Buscar la eficiencia y la eficacia del sistema haciendo uso de implementaciones paralelas.
- Realizar el diseño de arquitecturas paralelas, basado en la definición de thread, procesos, e interacciones.

Bibliografía

- Computer Architecture A Quantitative Approach 5ta Edición, David A. Patterson, John L. Hennerssy
- Computer Architecture A Quantitative Approach 6ta Edición, David A. Patterson, John L. Hennerssy
- Introduction to Parallel Computing, Second Edition, By Ananth Grama, Anshul Gupta, George Karypis, Vipin Kumar
- FPGA Protopyping by Verilog Examples. Pong P. Chu.
- Computer Organization and Design RISC-V Edition: The Hardware Software Interface. 2da Edición. David A. Patterson, John L. Hennerssy
- Computer Organization and Design: The Hardware Software Interface. Revised 4th Edition. David A. Patterson, John L. Hennerssy

FCEFyN
FACULTAD DE CIENCIAS EXACTAS, FÍSICAS y NATURALES

UNC Universidad Nacional de Córdoba

Asignatura: **Bases de Datos**

Código:	RTF	7
Semestre: 5	Carga Horaria	72
Bloque: Tecnologías Aplicadas	Horas de Práctica	30
Departamento: Computación		

Correlativas:
- Algoritmos y Estructuras de Datos

Contenido Sintético:
- Introducción y visión histórica de las bases de datos (BDs).
- Paradigmas Relacional y no Relacionales.
- Modelado Conceptual y Diseño de BDs. relacionales y no relacionales.
- Optimización, normalización y desnormalización.
- Álgebra relacional.
- El Lenguaje SQL.
- Formulación de consultas.
- Diseño físico de Bases de Datos, Índices.
- Transacciones, recuperación y concurrencia.
- Bases de datos distribuidas.
- Seguridad de las bases de datos

Competencias Genéricas:
- CG2: Concebir, diseñar y desarrollar proyectos de ingeniería (sistemas, componentes, productos o procesos). Alta
- CG4: Utilizar de manera efectiva las técnicas y herramientas de aplicación en ingeniería. Alta
- CG6: Desempeñarse de manera efectiva en equipos de trabajo. Alta
- CG8: Actuar con ética, responsabilidad profesional y compromiso social, considerando el impacto económico, social y ambiental de su actividad en el contexto local y global. Baja

Aprobado por HCD: NNNN-HCD-AAAA RES: Fecha: DD/MM/AAAA

Competencias Específicas:

- C.E4.8 Analizar, interpretar, modelar, diseñar, implementar, optimizar y consultar bases de datos.

Presentación

El conocimiento y la aplicación de bases de datos son habilidades esenciales en el mundo informático actual. Las bases de datos desempeñan un papel vital en prácticamente todas las industrias y disciplinas, desde la atención médica y la gestión de inventarios hasta la toma de decisiones estratégicas en organizaciones. La capacidad de gestionar y acceder a datos de manera eficiente es lo que permite optimizar las operaciones, tomar decisiones informadas y ofrecer servicios cada vez más personalizados.

Es en este contexto donde la capacitación en bases de datos se presenta como una necesidad imperante en la formación de profesionales capaces de dar respuesta a las demandas de la sociedad contemporánea

En un entorno académico y científico en constante evolución, las bases de datos son la clave para almacenar y analizar grandes volúmenes de información, lo que impulsa avances significativos en campos como la inteligencia artificial, la bioinformática y la investigación espacial.

En el contexto de los sistemas ciberfísicos, las bases de datos desempeñan un papel esencial al permitir la recopilación, el almacenamiento y el análisis de datos provenientes de sensores, actuadores y otros dispositivos conectados. Esto no solo facilita la supervisión en tiempo real de los sistemas, sino que también habilita la toma de decisiones autónomas y la optimización de procesos de manera eficiente.

Este curso está diseñado para proporcionar a los estudiantes una profunda comprensión de las bases de datos, abarcando desde los conceptos fundamentales hasta las técnicas avanzadas en diseño, implementación y gestión de bases de datos. A lo largo de esta experiencia de aprendizaje, se exploran conceptos teóricos clave relacionados con la modelización de bases de datos, tanto en el contexto de las bases de datos relacionales como no relacionales. Además, se abordan aspectos cruciales como la implementación física, la seguridad de los datos, el manejo de transacciones, la concurrencia y la recuperación.

Esta asignatura no pretende ser un curso de DBA (Administrador de Bases de Datos) especializado en un motor de bases de datos sino ser un curso genérico que permita al profesional especializarse en alguna herramienta específica.

Contenidos

Capítulo 1: Introducción y visión histórica de las bases de datos (BDs)

Datos vs. Información. La importancia de los datos. ¿Qué es una Base de Datos? La persistencia de los datos. La independencia de los datos.
Diferentes tipos de Sistemas de Gestión de Bases de Datos. Historia y evolución. Condiciones a cumplir por los SGBD. Funciones de los SGBD.

Capítulo 2:Paradigmas Relacional y no Relacionales.

Introducción a los paradigmas de BDs. Paradigma Relacional: relaciones, tuplas, atributos, dominios, claves, cardinalidad y grado de las relaciones. Dominios. Tipos de dominios. Relaciones. Propiedades. Tipos de relaciones. Diferentes paradigmas no relacionales: orientado a documentos, a columnas, clave-valor, etc. Comparaciones y usos.

Capítulo 3: Modelado Conceptual y Diseño de BDs. relacionales y no relacionales.

El modelo Entidad – Relación. El modelado semántico. El enfoque general. Cómo modelar entidades. Diagramas E/R. Diseño de Bases de Datos. Claves: Tipos de claves: claves primarias, claves foráneas. Guías de diseño de BDs no relacionales. Diferencias y similitudes. Implementación de Bases de Datos

Capítulo 4: Optimización, normalización y desnormalización.

Formas normales: Concepto de normalización. Primera, segunda y tercera forma normal. Conservación de la dependencia. Forma normal de Boyce Codd. Formas normales adicionales. Desnormalización

Capítulo 5: Álgebra relacional.

Introducción al álgebra relacional. Algebra relacional y teoría de conjuntos. Sintaxis Semántica. Operadores: Unión, Intersección, Diferencia, Producto, Selección, Reunión, División, Asignación

Capítulo 6: El Lenguaje SQL.

Definición. El ANSI SQL. Formulación de consultas simples. La sentencia SELECT. Tipos de juntas. Ordenamiento Las sentencias INSERT, UPDATE, DELETE.

Capítulo 7: Formulación de consultas.

Consultas complejas. Optimización Procesamiento de consultas. La heurística en la optimización de consultas. Uso de índices. Reglas de transformación. Estimación del costo. Pruebas de rendimiento

Capítulo 8: Diseño físico de Bases de Datos, Índices.

Archivos y páginas. Agrupamiento. Índices. Distintos tipos de índices. Compresión de datos

Capítulo 9: Transacciones, recuperación y concurrencia.

Conceptos de transacciones en bases de datos. Recuperación y manejo de errores. Control de concurrencia en bases de datos.

Capítulo 10: Bases de datos distribuidas.

Bases de datos distribuidas: fundamentos. Diseño y desafíos en bases de datos distribuidas. Consistencia y replicación en bases de datos distribuidas. Diferentes arquitecturas. Teorema CAP

Capítulo 11: Seguridad de las bases de datos

Seguridad en bases de datos: autenticación y autorización. Usuarios y roles. Estrategias de seguridad en bases de datos relacionales y NoSQL.

Metodología de enseñanza

En el compromiso por formar profesionales competentes y preparados para los desafíos actuales del mundo de las bases de datos, se aplica una metodología de enseñanza centrada en el estudiante y orientada hacia el desarrollo de competencias profesionales sólidas. Esta

metodología se basa en la premisa de que el aprendizaje efectivo de las bases de datos no solo implica la adquisición de conocimientos teóricos, sino también la capacidad de aplicar esos conocimientos de manera significativa en entornos profesionales.

Enfoque Activo y Participativo: Se fomenta la participación activa de los estudiantes en el proceso de aprendizaje en las clases teórico-prácticas. Se promueve la resolución de problemas, el trabajo en equipo y la discusión de casos prácticos. Los estudiantes son alentados a plantear preguntas, desafiar ideas preconcebidas y explorar nuevas perspectivas, lo que les permite desarrollar habilidades críticas para la toma de decisiones en el ámbito de las bases de datos.

Aprendizaje Basado en Proyectos: Los proyectos prácticos desempeñan un papel central. Los estudiantes tienen la oportunidad de aplicar los conceptos y técnicas aprendidos en situaciones del mundo real. A través de la creación de bases de datos, la optimización de consultas y la resolución de problemas específicos, los estudiantes adquieren habilidades prácticas esenciales para su futura carrera profesional.

Enfoque Interdisciplinario: las bases de datos son una herramienta fundamental en una amplia variedad de campos profesionales. Por lo tanto, fomentamos un enfoque interdisciplinario que permite a los estudiantes explorar cómo se aplican las bases de datos en diferentes sectores, como la salud, la logística, la educación y la investigación científica. Esto les brinda una visión más amplia de las oportunidades profesionales disponibles.

Evaluación Continua y Retroalimentación: La evaluación continua es un pilar pedagógico. Los estudiantes reciben retroalimentación constante a lo largo del curso, lo que les permite identificar áreas de mejora y ajustar su enfoque de aprendizaje. Además, se fomenta la autorreflexión y la autoevaluación como herramientas para el desarrollo profesional.

Recursos Tecnológicos Avanzados: Se utilizan herramientas y recursos tecnológicos de vanguardia para enriquecer la experiencia de aprendizaje. Esto incluye el acceso a bases de datos en la nube, software de gestión de bases de datos, y simulaciones que replican situaciones del mundo real.

Colaboración y Comunicación: Se promueve la comunicación efectiva y la colaboración entre estudiantes, imitando las dinámicas profesionales. El trabajo en equipo y la capacidad para explicar y defender soluciones son competencias esenciales que se desarrollan a lo largo del curso.

Cuestiones Éticas: El manejo y la protección de datos de terceros implica una formación ética y profesional que asegure su correcta administración teniendo en cuenta la responsabilidad ante la sociedad que ello representa.

En resumen, la metodología de enseñanza en bases de datos se enfoca en el estudiante como protagonista activo de su proceso de aprendizaje y en la adquisición de competencias profesionales sólidas. La combinación de teoría sólida y aplicación práctica es esencial para preparar a los estudiantes para tener éxito en un mundo donde las bases de datos desempeñan un papel central en numerosos campos profesionales.

Propuesta de Evaluación

Las evaluaciones están diseñadas para garantizar que los estudiantes adquieran y demuestren de manera efectiva las competencias profesionales necesarias en este campo. Estas evaluaciones están alineadas con la metodología centrada en el estudiante y en el desarrollo de habilidades prácticas.

Proyectos Prácticos de Bases de Datos: Estos proyectos son la piedra angular de la evaluación. Los estudiantes trabajan en equipos para diseñar, implementar y gestionar bases de datos en situaciones del mundo real. Los proyectos incluyen la creación de una base de datos desde cero, la optimización de consultas complejas y/o la resolución de problemas específicos relacionados con bases de datos.
Como parte de los proyectos se presentan casos de uso en los que los estudiantes individualmente analizan y proponen soluciones basadas en bases de datos. Estos estudios de caso requieren que los estudiantes apliquen conceptos teóricos a situaciones concretas y presenten soluciones argumentadas.
Cada grupo de trabajo tiene un hilo conductor de su proyecto a lo largo del curso, previendo dos presentaciones en el semestre.
En cada grupo de trabajo, los estudiantes asumen roles simulados en correspondencia con los que se identifican en el mundo profesional, “product manager”, “designer”, “test”, “security assurance”, etc.
Los proyectos se evalúan en función de la calidad de diseño, rendimiento y solución de problemas.
Los estudiantes son requeridos para presentar y defender sus proyectos y soluciones ante sus compañeros y el profesor. Esto fomenta la habilidad de comunicación y la capacidad de explicar y respaldar sus decisiones de diseño y gestión de bases de datos.

Colaboración y Evaluación por Pares: En conjunto con la presentación de los proyectos, se trabaja con la evaluación entre pares, donde los estudiantes evalúan y proporcionan retroalimentación sobre el trabajo de sus compañeros. Esto promueve la colaboración y la capacidad de evaluar críticamente el trabajo de otros. Se evalúa la pertinencia y completitud de las críticas.

Evaluaciones Continuas: A lo largo de la asignatura, se realizan evaluaciones formativas automatizadas que permiten a los estudiantes recibir retroalimentación regular sobre su progreso. Esto puede incluir ejercicios prácticos y cuestionarios automatizados. La retroalimentación se utiliza para ayudar a los estudiantes a ajustar su enfoque de aprendizaje y mejorar sus habilidades.

Autoevaluación y Reflexión: Los estudiantes son alentados a realizar autoevaluaciones periódicas para reflexionar sobre su progreso y desarrollo de competencias. Esta reflexión personal les ayuda a identificar áreas de mejora y a tomar medidas para abordarlas.

Si bien cada trabajo puede favorecer el desarrollo de una determinada competencia en particular y es de esperar la evidencia de esto hacia la conclusión de dicha actividad, la evaluación será contínua a lo largo de todas las actividades propuestas.

Al final del semestre cada estudiante debe haber demostrado un nivel de desarrollo mínimo de las competencias propuestas a través de los resultados de aprendizaje propuestos.

Cada trabajo será calificado en función de los aspectos disciplinares, así como de la evidencia de desarrollo de las competencias alcanzadas al momento de la finalización del mismo, pudiendo modificar esta calificación si en el transcurso de los trabajos subsiguientes se evidencia un mayor desarrollo de las mismas.

Como herramienta de evaluación del conjunto de competencias propuestas se emplea la siguiente rúbrica:

Competencia	Resultado de Aprendizaje	Mínimo	Valoración
CG2 CE4.8	Interpreta correctamente el dominio de un problema.	2	
CG6 CE4.8	Posee habilidades comunicacionales e interpretativas para realizar las preguntas necesarias para desarrollar un diseño completo ajustado a las necesidades del dominio presentes y futuras.	2	
CG8	Identifica los aspectos de la encomienda profesional que por su naturaleza tienen connotaciones éticas	2	
CG6 CG8	Cumple en tiempo con los compromisos asumidos con su equipo de trabajo.	2	
CG6 CG8	Respeta las pautas de trabajo establecidas en clase para las actividades de equipo.	2	
CG2 CE4.8	Diseña una base de datos tanto relacional como no relacional según buenas prácticas establecidas.	2	
CG2 CG4 CE4.8	Implementa una base de datos relacional y no relacional.	2	
CE4.8	Elabora y ejecuta consultas complejas en lenguaje SQL	2	
CG6	Trabaja en equipo asumiendo distintos roles dentro de un grupo de trabajo	2	
CG6 CE4.8	Detecta y comunica errores y oportunidades de mejoras en diseños de BDs propios y de terceros.	2	
CG4 CE4.8	Conoce los fundamentos de una base de datos distribuida.	2	
CG4 CG8 CE4.8	Asegura una base de datos según buenas prácticas de seguridad de BDs e identifica los compromisos éticos que surgen de implementar seguridad en los sistemas de datos.	2	

Para evaluar los trabajos arriba indicados se utilizarán rúbricas a fin de que el alumno pueda discernir los objetivos que alcanzó y en qué grado como una forma de retroalimentación.

El rango de valoración de la rúbrica es de 1 a 3 u se corresponde a:

1. Insuficiente: No se evidencia el nivel de desarrollo de las competencias esperado a través de los resultado de aprendizaje
2. Suficiente: En la mayoría de las situaciones se evidencia el nivel de desarrollo deseado.
3. Alto: Se evidencia un claro desarrollo de las competencias esperado a través de los resultados de aprendizaje.

Exámenes teóricos-prácticos (ETP): aunque la metodología se centra en la aplicación práctica, también es importante que los estudiantes comprendan los fundamentos teóricos. Los exámenes teóricos-prácticos evalúan el conocimiento conceptual de los estudiantes en áreas como paradigmas, teoría de bases de datos, el diseño de esquemas y la normalización entre otros, se realizan de manera automatizada, dos instancias de evaluación.

La calificación final se compone según la siguiente fórmula polinómica: 0.2 * ETP1 + 0.2 * ETP2 + 0.6 Rúbrica.

Condiciones de aprobación

Para regularizar y promoción se exige un 80% de asistencia y participación en clase.
Para regularizar, además, se exige haber aprobado ambos ETPs
Para promocionar se exige además aprobar la evaluación por rúbrica.

Actividades prácticas y de laboratorio

Además de las actividades indicadas en el punto Evaluación se realizan actividades prácticas en conjunto con el dictado de clases resolviendo problemas de acuerdo a los contenidos que se van desarrollando.

Resultados de aprendizaje

Al final del curso se espera que un estudiante alcance los siguientes habilidades:

- Interpreta correctamente el dominio de un problema.
- Posee las habilidades comunicacionales suficientes para realizar las preguntas necesarias para desarrollar un diseño completo ajustado a las necesidades del dominio presentes y futuras.
- Identifica los aspectos de la encomienda profesional que por su naturaleza tienen connotaciones éticas

- Cumple en tiempo con los compromisos asumidos con su equipo de trabajo.
- Respeta las pautas de trabajo establecidas en clase para las actividades de equipo.
- Diseña una base de datos tanto relacional como no relacional según buenas prácticas establecidas.
- Implementa una base de datos relacional y no relacional.
- Elabora y ejecuta consultas complejas en lenguaje SQL
- Trabaja en equipo asumiendo los distintos roles dentro de un grupo de trabajo
- Detecta y comunica errores y oportunidades de mejoras en diseños de BDs propios y de terceros.
- Conoce los fundamentos de una base de datos distribuida.
- Asegura una base de datos según buenas prácticas de seguridad de BDs e identifica los compromisos éticos que surgen de implementar seguridad en los sistemas de datos.

Bibliografía

"Fundamentos de Bases de Datos" de Abraham Silberschatz, Henry F. Korth y S. Sudarshan
"Bases de Datos. Diseño y gestión" de Richard T. Watson, María Leonor Uscátegui Álvarez y Leonor Pérez Daza
"Introducción a las Bases de Datos" de C.J. Date:
"Sistemas de Bases de Datos" de Elmasri y Navathe
"Bases de Datos" de Ramakrishnan, Gehrke y Donaire:
"Gestión de Bases de Datos con MySQL" de Ignacio Rodríguez Huelva:*
"SQL Para Principiantes" de Don Jones y Dan Sullivan
"Bases de Datos NoSQL" de Carlos Casares Mouriño y Antonio Bahamonde Feijóo
"NoSQL: Bases de Datos No Relacionales" de Ángel Aria
"MongoDB: Aplique el NoSQL en su Empresa" de Jordi Llonch Andreu y Miquel Àngel Capó Granados

FCEFyN
FACULTAD DE CIENCIAS EXACTAS, FÍSICAS y NATURALES

UNC Universidad Nacional de Córdoba

Asignatura: **Calidad de Software y Hardware**

Código:	RTF	6
Semestre: 7	Carga Horaria	72
Bloque: TB	Horas de Práctica	36

Departamento: Computación

Correlativas:
- Ingeniería del Software y Hardware

Contenido Sintético:
- Fundamentos de la calidad.
- Mejora continua.
- Costos y herramientas para la calidad.
- Calidad en el servicio al cliente.
- Calidad personal y liderazgo para la calidad.
- Calidad del software.
- Calidad de sistemas hardware-software.
- Métricas del software.
- Aseguramiento de la calidad del software y hardware-software.
- Estándares, normas y modelos para la calidad

Competencias Genéricas:

- CG2: Concebir, diseñar y desarrollar proyectos de ingeniería (sistemas, componentes, productos o procesos).
- CG3: Gestionar -planificar, ejecutar y controlar- proyectos de ingeniería (sistemas, componentes, productos o procesos).
- CG4: Utilizar de manera efectiva las técnicas y herramientas de aplicación en ingeniería.
- CG7: Comunicarse con efectividad.
- CG8: Actuar con ética, responsabilidad profesional y compromiso social, considerando el impacto económico, social y ambiental de su actividad en el contexto local y global.

Aprobado por HCD: NNNN-HCD-AAAA RES: Fecha: DD/MM/AAAA

Competencias Específicas:

- CE4.3: Analizar, diseñar, programar, implementar, probar, depurar y evaluar hardware y software para sistemas de computación de propósitos específicos.
- CE4.4: Analizar, diseñar, programar, implementar, probar, depurar y evaluar hardware y software para sistemas de computación de propósitos generales.
- CE8 Certificar el funcionamiento, condición de uso o estados de Sistemas de Procesamiento de Señales, Sistemas Embebidos, Sistemas Computarizados de automatización y control, Sistemas Conjuntos de Hardware y Software.
- CE10.1 Conocer profundamente métricas de calidad, estándares, normas y modelos de calidad.
- CE10.2 Asegurar la calidad de los sistemas desarrollados.

Presentación

La asignatura Calidad de Software y Hardware pertenece al 7(séptimo) semestre (4 año) de la carrera de Ingeniería en Computación.
Al inicio de este espacio curricular el estudiante ya cuenta con bases sólidas en las prácticas de Ingeniería de Software y Hardware y de toda la infraestructura necesaria para aplicar dichos procesos ya que lo han desarrollado durante los trabajos de laboratorio. Todo este conocimiento aplicado sirve de base para el desarrollo de la presente propuesta educativa. En ella se busca conocer, descubrir, aprender, implementar, experimentar, emplear y mantener procesos para gestionar la calidad durante todo el ciclo de vida del desarrollo de software y de hardware.
Durante el transcurso de la asignatura exploramos las dificultades, los retos y los desafíos en los procesos para gestionar la calidad en la producción de software y hardware a gran escala, su configuración, su administración, su control y mantenimiento; para finalmente arribar en implementaciones de procesos de ingeniería que brinden soluciones a dichas problemáticas.
La asignatura está diseñada desde un enfoque constructivista donde los estudiantes deben organizarse alrededor de metodologías ágiles para ir cumpliendo los retos y desafíos que proponemos entregar valor al final de cada sprint de 2 semanas de duración.
La cátedra ha establecido pautas para conformar un esquema de autonomía alineada, donde los docentes establecen los lineamientos y los equipos de trabajo tienen autonomía para resolver problemáticas reales y desafíos de la industria en términos de Calidad en el Software y Hardware.
En términos más concretos, los docentes explican, ejemplifican y facilitan herramientas, patrones, métricas, etc, pero son finalmente los alumnos quienes deben implementarlas, adaptarlas, configurarlas, probarlas y mantenerlas de acuerdo con cada singular problema que se desea resolver.
La calidad del software y hardware es un aspecto fundamental en la industria tecnológica, ya que afecta directamente la experiencia del usuario, la eficiencia del sistema y la confiabilidad de los productos.
La calidad del software y hardware se refiere a la medida en que un sistema cumple con los requisitos establecidos y satisface las necesidades del usuario. Un sistema software/hardware de calidad se caracteriza por su funcionalidad, usabilidad, rendimiento, confiabilidad, seguridad, durabilidad, fiabilidad y capacidad de mantenimiento. Para garantizar la calidad del software, es necesario seguir procesos de desarrollo bien definidos, realizar pruebas exhaustivas, gestionar adecuadamente los errores y adoptar buenas prácticas de programación.
La calidad del software y hardware están estrechamente relacionadas, ya que el software se ejecuta en el hardware y ambos interactúan para brindar una experiencia satisfactoria al usuario. Un software bien desarrollado puede ser afectado negativamente si el hardware subyacente es deficiente, y un hardware potente puede ser subutilizado si el software no está optimizado.

Contenidos

Módulo I. Gestión de la Calidad de Software y Hardware

- Unidad 1. Fundamentos de Calidad
    - Definiciones. Paradigmas. Control de la Calidad. Aseguramiento de la Calidad. Sistemas de Gestión de la Calidad. Administración de la Calidad. Mejora Continua.
- Unidad 2. Aspectos Económicos, Riesgos y Viabilidad
    - Costos de la Calidad y no-Calidad. Errores, Fallas y Defectos. Clasificación de las Causas de los errores. Factores y Atributos de la Calidad. Gestión del Riesgo.
- Unidad 3. Integración en el Ciclo de Vida del Proyecto
    - Plan de Aseguramiento de la Calidad. Integración en Modelos Tradicionales. Integración en Metodologías de Desarrollo Ágil. Integración en Modelos Modernos.

Módulo II. Componentes del Aseguramiento de la Calidad de S/H

- Unidad 4. Métricas de Calidad del Software/Hardware
    - Métricas en el Diseño y Desarrollo. Métricas relacionadas a la Resolución de los errores/defectos. Métricas en las etapas de Release. Métricas de experiencia y de cara al Cliente/Usuario. Métricas Intrínsecas.
- Unidad 5. Revisiones, Estrategias e Implementación
    - Revisiones Formales. Revisión de Pares. Opiniones de Expertos. Estrategias de Testeos. Clasificación y Tipos de Testeos. Alcances de los Testeos
- Unidad 6. Herramientas empleadas en el Aseguramiento de la Calidad
    - Plantillas, Diagramas, Gráficas. Herramientas CASE empleadas en el desarrollo, testeo y mantenimiento. Contribución a las mejoras en la calidad en cada una de las etapas del ciclo de vida.

Módulo III. Infraestructura del Aseguramiento de la Calidad de S/H

- Unidad 7. Procesos y Flujos de Trabajo
    - Procedimientos e Instrucciones de Trabajo. Procesos Manuales y Automáticos. Preparación, implementación y actualización.
- Unidad 8. Infraestructura y Automatización
    - Integraciones con el Sistema de Gestión de las Configuraciones. Intervenciones en el Sistema de Integración Contínua. Integración en el sistema de Distribución Continua. Relaciones entre Calidad y Seguridad. Implicancias de Seguridad e intervenciones en el flujo de trabajo y en la infraestructura de despliegue.
- Unidad 9. Aspectos Humanos
    - Definición del Equipo de Aseguramiento de la Calidad. Calidad en el Equipo de Desarrollo. Entrenamiento, Capacitación y Certificaciones. Programas de Actualización. Liderazgo.

Módulo IV. Estándares, Certificaciones y Evaluaciones de Calidad

- Unidad 10. Estándares de Gestión de la Calidad de Software y Hardware. Alcances. Auditorías. Certificaciones. Modelo de Madurez. Evaluaciones de Calidad. Normas.

Metodología de enseñanza

Considerando a los estudiantes como actores principales en el proceso de aprendizaje, es que definimos esta metodología de aprendizaje, donde el objetivo sea la construcción de su propio conocimiento a través de la participación activa, el cuestionamiento y la reflexión crítica.
Por otro lado, también buscamos que el estudiante desarrolle habilidades colaborativas generando espacios donde se promueva la interacción y colaboración entre los estudiantes, permitiéndoles aprender unos de otros, fomentando el trabajo en equipo, y ayudándoles a desarrollar habilidades de comunicación y cooperación. Aquí los estudiantes deben organizarse en equipos y el docente va guiándonos con los objetivos y recursos a fin de concretar en cada uno de los sprints/iteraciones.
Asimismo, trabajamos en darle autonomía alineada a los diferentes equipos, para que ellos mismos diseñen las soluciones planteándose la problemática a resolver. Este enfoque implica presentar a los estudiantes problemas reales o simulados para que los resuelvan, lo que les ayuda a aplicar lo que han aprendido en contextos prácticos, fomentando el pensamiento crítico y la resolución de problemas.
Por último, se busca individualmente que desarrollen habilidades de Aprendizaje Autónomo, promoviendo la independencia y la autodisciplina en el aprendizaje, fomentando que los estudiantes tomen la responsabilidad de su propio proceso de aprendizaje, que establezcan sus propios objetivos, que seleccionen sus estrategias, y que se autoevalúen su progreso.

Evaluación

Las instancias de evaluación de los conocimientos curriculares ocurrirán en dos momentos a lo largo del semestre y las evaluaciones de las habilidades y competencias se realizarán de forma continua al finalizar cada sprint/bloque con las entregas y exposiciones de los trabajos realizados que vayan aportando valor a su trabajo final.
Las actividades propuestas están diseñadas para que su conclusión sea indicadora del desarrollo de las competencias propuestas.
Al finalizar, los equipos deberán exponer una presentación final de su proyecto desde el inicio hasta su estado actual, luego de esas iteraciones.
La calificación final será una combinación de los exámenes, las presentaciones al final de los sprints y la presentación final.

Condiciones de aprobación

Calificación:
La calificación se obtendrá a través del siguiente polinomio:

CALIFICACIÓN= 0,2xP1 + 0,2P2 + 0,4xS + 0,2PF

Donde:
P1: Es la nota del primer examen parcial.
P2: Es la nota del segundo examen parcial.
S: Es el promedio de las calificaciones de las actividades prácticas entregadas al final de los Sprints donde se evalúa a nivel individual y grupal pudiendo diferir entre integrantes del mismo equipo por cuestiones relacionadas con la exposición, el compromiso, la participación, la colaboración con otros equipos, el mentoreo, el liderazgo, etc.
PF: Es la calificación de la presentación final donde se evalúa no solo la exposición, sino también la capacidad de resolución de problemas y los casos de éxito y fracaso; valorando positivamente la capacidad de resiliencia y tolerancia a dichas condiciones adversas. Aquí se busca motivar, premiar y desarrollar esta habilidad muy necesaria para formar futuros emprendedores y líderes
.
Requisitos para alcanzar la regularidad.

- 80% de asistencia.
- Rendir y aprobar los dos parciales con un 60% o más, el alumno podrá recuperar sólo un parcial.
- Aprobación del 100% de las actividades prácticas propuestas al final del Sprint.
- Aprobación de la Presentación Final del Trabajo de Laboratorio.

Requisitos para alcanzar la Promoción.

- Cumplir los requisitos de regularidad
- Tener aprobadas las materias correlativas
- La calificación final debe ser igual o mayor que 7.

Los alumnos que no alcancen la condición de regular al finalizar la Presentación Final serán considerados libres.

Actividades prácticas y de laboratorio

Se requiere emplear el sistema de integración continua y de despliegue contínuo desarrollado en Ingeniería del Software y Hardware, con el fin de ir adicionando herramientas, umbrales de métricas, controles, chequeos, verificaciones, inspecciones, etc para poder asegurar la calidad en todo el ciclo de vida de desarrollo del software/hardware.

Por otro lado, también se trabajará sobre el Sistema de Manejo de las Configuraciones y el Sistema Versionado de Código con el objetivo de inyectar, intervenir, modificar, bloquear, alertar, monitorear, observar, recomendar, etc cualquier desviación, adulteración, defecto, falla o error que pueda ocasionar problemas relacionados con la calidad, con la seguridad y estabilidad de todo el sistema.

Desagregado de competencias y resultados de aprendizaje

Las instancias de evaluación se emplean para garantizar que los estudiantes certifiquen de manera efectiva las competencias profesionales requeridas para el alcance de esta materia. Las evaluaciones fueron alineadas a la metodología centrada en el estudiante y enfocadas en desarrollar habilidades prácticas.

Competencia	Resultado de Aprendizaje	Mínimo	Valoración
CG2	Interpreta correctamente el dominio de un problema.	2	
CG2 CG3 CG7 CE4.3 CE4.4 CE8 CE10.1 CE10.2	Detecta errores, fallas, defectos y oportunidades de mejoras tanto en el proceso como en el producto elaborado bajo estudio.	2	
CG2 CG3 CG7 CE8 CE10.1 CE10.2	Genera adecuadamente reporte de errores, fallas y defectos detectados en el proceso o producto elaborado bajo estudio.		
CG3 CG4 CE4.3 CE4.4 CE10.1 CE10.2	Emplea adecuadamente las técnicas y herramientas propuestas aplicables a Procesos de Aseguramiento de la Calidad de Software y Hardware.	2	
CG4 CG7 CE8 CE10.1 CE10.2	Comunica de Manera Efectiva los resultados de los procesos de implementaciones de herramientas, las mediciones, los inconvenientes, los desafíos y las acciones realizadas para superar los obstáculos	2	
CG3 CG7 CE10.1 CE10.2	Redacta, verifica y valida requerimientos a través de Casos de Pruebas o alguna otra herramienta.	2	
CG8	Identifica los aspectos del desarrollo profesional que por su origen y naturaleza tienen connotaciones éticas	2	
CG3 CG7 CG8	Respeta las pautas de trabajo establecidas en clase para las actividades de equipo.	2	
CG3	Trabaja en equipo asumiendo los distintos	2	

CG7	roles dentro de un grupo de trabajo		
CE10.1 CE10.2	Conoce normas, estándares y modelos de calidad	2	

Se realizan autoevaluaciones periódicas buscando generar un espacio de reflexión sobre su progreso de aprendizaje y su desarrollo en dichas competencias. Esta autoreflexión les permite identificar oportunidades de mejoras y tomar decisiones para poder afrontar dicho desafío.

A los fines de evaluar los trabajos se emplearán rúbricas para que el estudiante pueda discernir los objetivos que alcanzó y en qué grado a modo de retroalimentación.
El rango de valoración es de 1 a 3 y se corresponde a:
1. Insuficiente: No se evidencia el nivel de desarrollo de las competencias esperado a través de los resultados de aprendizaje
2. Suficiente: En la mayoría de las situaciones se evidencia el nivel de desarrollo deseado.
3. Alto: Se evidencia un claro desarrollo de las competencias esperado a través de los resultados de aprendizaje.

Bibliografía

Bibliografía Principal

- Daniel Galin (2003) Software Quality Assurance: From Theory to Implementation (1 Edición) Pearson College

Bibliografía Complementaria

- Guillermo Pantaleo (2016) Calidad en el Desarrollo de Software (2 Edición) Alfaomega
- Ian Sommerville (2016) Ingeniería del Software (10 Edición) Pearson

FCEFyN
FACULTAD DE CIENCIAS EXACTAS, FÍSICAS y NATURALES

UNC Universidad Nacional de Córdoba

Asignatura: **ESTRUCTURAS DISCRETAS**

Código:	RTF	7
Semestre: 1	Carga Horaria	96
Bloque: Ciencias Básicas	Horas de Práctica	0
Departamento: Computación		

Correlativas:
- Matemática

Contenido Sintético:
- Visión de conjunto e historia.
- Herramientas relevantes, estándares y/o restricciones de ingeniería.
- Funciones, relaciones y conjuntos.
- Algebra de Boole.
- Cálculo proposicional.
- Cálculo de predicados.
- Técnicas de demostración.
- Fundamentos del conteo.
- Grafos y árboles.

Competencias Genéricas:
- CG1: Identificar, formular y resolver problemas de ingeniería. (B)
- CG4: Utilizar de manera efectiva las técnicas y herramientas de aplicación en ingeniería. (M)

Aprobado por HCD: NNNN-HCD-AAAA RES: Fecha: DD/MM/AAAA

Competencias Específicas:
N/A

Presentación

La disciplina Estructuras Discretas, objeto de esta asignatura, se ha desarrollado a lo largo del tiempo debido a una conjunción entre la Matemáticas, la Lógica y la Computación, consolidándose en el siglo XX debido al auge de las Tecnologías de la Información y las Comunicaciones y a la necesidad de abordar los requerimientos y problemas computacionales, que surgieron con un desarrollo vertiginoso, con una base matemática sólida. Su objetivo es el estudio de los conjuntos discretos finitos o numerablemente infinitos y está fuertemente relacionado con los números naturales que es un conjunto numerablemente infinito. Establece el fundamento teórico para las ciencias de la computación, porque permite computar funciones u operaciones sobre conjuntos numerablemente infinitos.
Particularmente en el plan de estudios de la carrera de Ingeniería en Computación, de esta unidad académica, se sitúa en el Bloque de Ciencias Básicas y se enseña en el primer semestre de cursado, por lo tanto, los conocimientos, procedimientos y técnicas aprendidas y puestas en práctica serán insumo clave para abordar temas más avanzados, durante el trayecto formativo y profesional.
La asignatura se enseña con enfoque constructivista y aprendizaje centrado en el alumno de tal manera de facilitar el desarrollo de las competencias claves para que los futuros ingenieros posean la capacidad de resolver problemas complejos, diseñar algoritmos eficientes y comprender la ciencia de la computación.
Se abordan los siguientes temas: lógica, conjuntos, relaciones, funciones y grafos, enfocados en la especificación de nuevas aplicaciones y su desarrollo de manera sistémica, con prácticas contextualizadas y acordes al trayecto formativo del alumno, utilizando los softwares de código abierto y gratuitos disponibles para cada unidad temática.

Contenidos

Unidad 1: Visión de conjunto e historia
1.1 Visión global de la matemática en la tecnología.
1.1 Los aportes de los babilonios al desarrollo del lenguaje matemático escrito.
1.2 La medición del tiempo, y las superficies de los egipcios.
1.3 La contribución de los griegos en el desarrollo de la matemática, la lógica, la astronomía, la física, la filosofía, el drama y la política.
1.4 El aporte de los romanos en el cálculo, el comercio y las obras de arquitectura e ingeniería.
1.5 La influencia de la cultura islámica y morisca de España en el cálculo algorítmico, los números y el álgebra.
1.6 La matemática en China y la India.

Unidad 2: Herramientas relevantes, estándares y/o restricciones de ingeniería
2.1 Especificación de programas en lenguaje funcional.
2.2 Aplicación del lenguaje lógico a análisis de proposiciones compuestas, expresiones y tablas de verdad.
2.3 Aplicación de bibliotecas y lenguajes para realizar cálculos de matemática simbólica.
2.4 Especificación de requisitos en el lenguaje formal.

Unidad 3: Conjuntos, relaciones y funciones
3.1 Conjuntos y operaciones de conjuntos.
3.2 Tuplas, sucesiones y conjuntos potencia.
3.3 Relaciones.
3.4 Propiedades de las relaciones.
3.5 Representaciones y manipulaciones que involucran funciones.
3.6 Enumeraciones, isomorfismos y homomorfismos.
3.7 Complejidad computacional.
3.8 Relaciones de Recurrencia.

Unidad 4: Algebra de Boole
4.1 Circuitos combinacionales.
4.2 Propiedades de los circuitos combinacionales.
4.3 Álgebra de Boole.
4.4 Funciones booleanas y síntesis de circuitos.

Unidad 5: Cálculo Proposicional
5.1Cálculo proposicional
5.2 Argumentos y proposiciones lógicas.
5.3 Conexiones lógicas.
5.4 Proposiciones compuestas.
5.5 Tautología y contradicciones.
5.6 Equivalencias lógicas y su utilización
5.7 Implicaciones y derivaciones lógicas.

Unidad 6: Cálculo de Predicados
6.1 Componentes sintácticos del cálculo de predicados.
6.2 Interpretaciones y validez.
6.3 Derivaciones.
6.4 Equivalencias lógicas.
6.5 Lógica de las ecuaciones

Unidad 7: Técnicas de Demostración
7.1 Nociones de implicación, equivalencia, conversa, inversa, contrapositiva, negación y contradicción
7.2 La estructura de las demostraciones matemáticas.
7.3 Demostraciones directas.
7.4 Contra demostración por contraejemplo.
7.5 Demostraciones por contradicción.
7.6 Inducción sobre los números naturales.
7.7 Inducción estructural
7.8 Inducción débil y fuerte.
7.9 Definiciones matemáticas recursivas.

Unidad 8: Fundamentos de Conteo
8.1 Principios básicos del conteo.
8.2 Permutaciones y combinaciones.

8.3 Permutaciones y combinaciones generalizadas.
8.4 Algoritmos para generar permutaciones y combinaciones.
8.5 Introducción a la teoría de la probabilidad discreta.
8.6 Coeficientes binomiales e Identidades combinatorias.
8.7 El principio del Pigeonhole.

Unidad 9: Grafos y árboles
9.1 Introducción y modelado de grafos.
9.2 Definiciones básicas de la teoría de grafos.
9.3 Caminos, accesibilidad y conexiones.
9.4 Cálculo de caminos a partir de una representación matricial de los grafos.
9.5 Recorrido de grafos representados como listas de adyacencia.
9.6 Árboles y árboles de expansión.
9.7 Redes de planificación.

Metodología de enseñanza

La metodología de enseñanza, con foco en competencias y aprendizaje centrado en el estudiante, se desarrolla para cada unidad temática en dos etapas:
En primera instancia a través de clases dialogadas se le enseña al alumno y se discuten los conceptos centrales de cada unidad temática, tanto en su aspecto teórico como práctico.
En una segunda instancia se utilizan estrategias didácticas variadas acordes a cada unidad temática, a las competencias genéricas que se pretenden desarrollar en el alumno y a la complejidad que presentan atendiendo al recorrido curricular recién iniciado. Las actividades se enmarcan, principalmente, en la utilización de software de código abierto y gratuito para la resolución de ejercicios y problemas contextualizados, en forma individual o grupal.
El sentido de uso de las herramientas de software es evitar el trabajo tedioso y repetitivo de un cálculo matemático en papel y permitir una validación automática de lo realizado, fomentando así hábitos de autoevaluación, y a la vez que el estudiante se familiarice con las herramientas disponibles para uso futuro curricular o profesional.
Por lo antes expuesto, las clases se desarrollan en aulas con computadoras con acceso a Internet y proyector.
Se hará uso intensivo del aula virtual para apoyo a la presencialidad, no solo para la distribución de contenidos, sino también para una comunicación fluida, evaluaciones automáticas y registro del recorrido de aprendizaje de cada alumno al interior de la asignatura.
La intención pedagógica de la asignatura es, mediante un proceso constructivista y centrado en el aprendizaje del estudiante, la enseñanza y puesta en práctica de las competencias genéricas, objeto de la asignatura. El proceso será llevado adelante y asistido por el docente desde un rol de facilitador y evaluador.
El recorrido de aprendizaje de cada alumno se registrará en un portafolio personal el cual contendrá el resultado o evidencias de las diferentes actividades didácticas realizadas durante el cursado, ya sea en forma individual o grupal. Este instrumento se utiliza además

durante las etapas de evaluación. Las evidencias se materializan mediante las salidas de los aplicativos en formato adecuado para su almacenaje y posterior revisión desde el portafolio.
En resumen, el diseño metodológico propuesto tiene por objetivo, en primera instancia, a través de las clases expositivas dialogadas, introducir al alumno a los conceptos básicos de cada unidad temática y luego mediante las estrategias didácticas prácticas, enseñar las competencias que le permitan al alumno identificar y resolver los ejercicios y problemas de la asignatura utilizando de manera efectiva las técnicas aprendidas y las herramientas de aplicación.

Evaluación

Las evaluaciones sumativas que se desarrollarán durante el cursado de la asignatura se refieren a dos evaluaciones teórico-prácticas y la evaluación de cuatro actividades prácticas de laboratorio individual o grupal, según la complejidad para la ejecución y acorde al trayecto formativo.
Las instancias de recuperación, a realizar antes de finalizar el cursado, son una para una de las evaluaciones teórico-prácticas y dos para las actividades prácticas de laboratorio.
Atendiendo a la necesidad de evaluar si el alumno aprendió las competencias genéricas, objeto de enseñanza de la asignatura, enumeradas en apartado anterior y desagregadas posteriormente en este documento, se utiliza el instrumento rúbrica. Los criterios pertinentes a cada rúbrica serán informados al alumno en momento oportuno y antes de las evaluaciones.
Las actividades prácticas de laboratorio, individuales o grupales, a evaluar se desarrollarán sobre aplicativos de software específicos para las unidades temáticas referentes a: conjuntos, relaciones y funciones, cálculo proposicional y predicados, grafos y árboles (3-5-6-9), organizadas en un total de cuatro actividades.
Las evidencias de lo realizado por el alumno se resguardan en un portafolio personal en el aula virtual de la asignatura.

Condiciones de aprobación

Al finalizar el semestre, y luego de las instancias recuperatorias, el alumno promociona la asignatura si aprobó los dos parciales teórico-prácticos y las cuatro actividades de laboratorio, con un rendimiento mínimo del 40% en cada una.
La calificación final se obtiene como el promedio de los seis rendimientos transformados en forma directa a valores absolutos en escala de 1 a 10.
Los alumnos lograran su regularidad con la aprobación de las cuatro actividades prácticas de laboratorio, pudiendo rendir en examen final una evaluación teórico-práctica integradora, que se aprueba con un rendimiento mínimo del 40%. La calificación final se obtiene de similar manera a la promoción, tal lo enunciado en el párrafo anterior.
En el caso de presentarse a examen final con condición libre, el alumno deberá realizar una actividad práctica de laboratorio, seleccionada en forma aleatoria entre las cuatro unidades temáticas por el profesor y posteriormente rendir la evaluación teórico-práctica integradora.

Actividades prácticas y de laboratorio

Las actividades prácticas y de laboratorio se desarrollarán en aulas con computadoras, en forma semanal, luego del desarrollo de la clase teórico-práctica y en un día diferente.
La evidencia de cada actividad de cada unidad temática será resguardada en el portafolio personal de cada alumno.
Acorde a lo enunciado en el apartado anterior, sólo las actividades prácticas de laboratorio correspondientes a las unidades temáticas 3-5-6-9 serán evaluadas y calificadas. Se propone, ya sea en forma individual o grupal según la complejidad, la presentación formal de resultados al profesor y alumnos mediante exposición y demostración, y la confección de un informe técnico. En estas unidades se utiliza software de código abierto y uso gratuito para la resolución de los problemas de estructuras discretas.
Para el desarrollo de las actividades prácticas del resto de las unidades temáticas se utilizarán herramientas informáticas acordes, y las evidencias de los resultados obtenidos serán resguardados en los portafolios personales de cada alumno.
El sentido didáctico del uso de portafolio es a modo de autoevaluación y revisión del alumno de su trayecto de aprendizaje, además de los fines administrativos del aula virtual.

Desagregado de competencias y resultados de aprendizaje

El desarrollo de competencias, entendido como un quehacer complejo, conlleva luego de la definición sintética e integrada de cada una de ellas, el desagregado en niveles componentes de capacidades para una correcta implementación curricular y evaluación de los resultados de aprendizaje, según lo antes expresado en el apartado que trata los instrumentos de evaluación.
En tal sentido:

- CG1: Competencia para identificar, formular y resolver problemas de ingeniería. Esta competencia requiere de la articulación efectiva de las siguientes capacidades:
  - Capacidad para identificar problemas a resolver con técnicas de resolución de estructuras discretas y los aplicativos de software asociados.
    Esto implica:
    - Ser capaz de identificar el tipo de problema discreto.
    - Ser capaz de identificar los datos, variables y parámetros del problema discreto.
    - Ser capaz de plantear conceptualmente la solución.
  - Capacidad para implementar la solución con técnicas de resolución de estructuras discretas y los aplicativos asociados.
    Esto implica:
    - Ser capaz de elegir la técnica adecuada a utilizar.
    - Ser capaz de utilizar las herramientas informáticas pertinentes para resolver el problema discreto.
- CG4: Competencia para utilizar de manera efectiva las técnicas y herramientas de la ingeniería. Esta competencia requiere de la articulación efectiva de las siguientes capacidades:
  - Capacidad para identificar y seleccionar las técnicas y herramientas aprendidas
    Esto Implica:

        - Ser capaz de conocer los alcances y limitaciones de las técnicas y herramientas a utilizar en el contexto de las estructuras discretas.
    - Capacidad para utilizar las técnicas y herramientas en forma efectiva y eficiente.

      Esto Implica:
        - Ser capaz de interpretar los resultados que se obtengan de la aplicación de las diferentes técnicas y herramientas utilizadas.
        - Ser capaz de controlar y obtener los resultados correctos.
        - Ser capaz de presentar los resultados en forma adecuada.

Bibliografía

Básica:
Johnsonbaugh, R. (2005). *Matemáticas discretas*. Ed. Prentice-Hall.

Complementaria:
Samuels Epp, S. (2012). *Matemáticas discretas con aplicaciones.* Ed. Cengage Learning.
Gallier, J. (2017). *Discrete Mathematics*. Ed.Springer.
Rosen, K. (2004). *Matemática Discreta y sus Aplicaciones*. Ed. McGraw-Hill.

FCEFyN
FACULTAD DE CIENCIAS EXACTAS, FÍSICAS y NATURALES

UNC Universidad Nacional de Córdoba

Asignatura: **Fundamentos de Programación**

Código:	RTF	7
Semestre: Primero • Ingeniería en Computación	Carga Horaria	96
Bloque: TB (96)	Horas de Práctica	48
Departamento: Computación		

Correlativas:
- Matemática

Contenido Sintético:
- Introducción a la programación.
- Elementos de la programación estructurada.
- Estructuras de control.
- Estructuras de datos.
- Funciones y procedimientos.
- Entrada/salida de información.
- Manejo de errores y excepciones.
- Verificación y validación de programas.

Competencias Genéricas:
- CG1: Identificar, formular y resolver problemas de ingeniería.
- CG4: Competencia para utilizar de manera efectiva las técnicas y herramientas de la ingeniería.
- CG7: Comunicarse con efectividad.

Aprobado por HCD: NNNN-HCD-AAAA RES: Fecha: DD/MM/AAAA

Competencias Específicas para la carrera de Ingeniería en Computación:

- CE7.2.2 Sintetizar, diseñar, desarrollar y analizar programas lenguajes de programación de bajo nivel, como C y C++
- CE7.2.3 Seleccionar y utilizar entornos de desarrollo integrados (IDE) y herramientas de depuración específicas para este tipo de sistemas.

Presentación

La asignatura “Fundamentos de Programación” pertenece al primer año del plan de estudio de Ingeniería en Computación. Al momento de transitar este espacio curricular, el estudiante ha adquirido conocimientos básicos de matemáticas, que le permitirán comprender los conceptos fundamentales de la programación. En particular, la asignatura se enfoca en la programación estructurada que sienta las bases para comprender conceptos más avanzados en programación y diseño de software en asignaturas posteriores. Los estudiantes aprenderán a desarrollar algoritmos y a escribir código claro y legible, habilidades cruciales en su profesión.

El curso no solo establece las bases conceptuales de la programación, sino que también desarrolla competencias prácticas. La capacidad de identificar, formular y resolver problemas de ingeniería se nutre a medida que los estudiantes enfrentan desafíos de programación y desarrollan soluciones efectivas. Además, se fomenta el uso efectivo de técnicas y herramientas de ingeniería, ya que los estudiantes aprenden a utilizar lenguajes de programación y se familiarizan con entornos de desarrollo integrados (IDE) y herramientas de depuración específicas. Por otro lado, el uso de la comunicación efectiva también se integra en el curso a través de la documentación y el trabajo en equipo, habilidades que son esenciales en la colaboración dentro de equipos de desarrollo de software.

En la actualidad la informática y la programación son componentes esenciales en la mayoría de las industrias y sectores, desde la automatización de procesos industriales hasta la resolución de problemas complejos en campos como la inteligencia artificial y la ciberseguridad. Por lo tanto, esta asignatura representa la puerta de entrada al vasto y dinámico campo de la informática, brindando a los futuros profesionales las competencias fundamentales para abordar los desafíos tecnológicos de hoy y adaptarse a los cambios constantes en el futuro. En última instancia, el conocimiento adquirido en esta asignatura sienta las bases para una carrera profesional que pueda contribuir significativamente al desarrollo tecnológico y la innovación en la sociedad actual.

Contenidos

Unidad 1: Programación estructurada
Resolución de problemas y algoritmos. Lenguajes de programación. Concepto de programa y sus elementos. Tipos de datos primitivos. Operaciones aritméticas. Variables, constantes y declaraciones. Operaciones de asignación. Entrada y salida estándar de información. Formato de salida. Funciones de biblioteca. Comentarios y convenciones de nomenclatura. Funciones definidas por el usuario: procedimientos y funciones con parámetros por valor. Alcance de variables. Aplicaciones prácticas. Verificación y validación de programas con flujo secuencial.

Unidad 2: Estructuras de selección
Operaciones relacionales. Operaciones lógicas. Precedencia y asociatividad. Estructuras de selección. La estructura de decisión simple. La estructura de decisión doble. Estructuras de decisión anidadas. La estructura de decisión múltiple. Funciones definidas por el usuario: funciones con parámetros por referencia. Aplicaciones. Verificación y validación de programas con flujo selectivo.

Unidad 3: Estructuras de repetición
Estructuras de repetición. Las estructuras de repetición indefinidas. La estructura de repetición definida. Estructuras de repetición anidadas. Alteraciones del flujo normal. Funciones definidas por el usuario: recursividad. Aplicaciones. Verificación y validación de programas con flujo repetitivo.

Unidad 4: Estructuras de datos y acceso a archivos
Arreglos unidimensionales. Inicialización de arreglos. Arreglos bidimensionales. Arreglos como argumentos de función. Estructuras sencillas. Arreglo de estructuras. Estructuras como argumentos de función. Lectura y escritura de archivos. Acceso aleatorio de archivos. Flujo de archivos como argumento de función. Excepciones y comprobación de archivos. Aplicaciones. Verificación y validación de programas con estructuras de datos y archivos.

Metodología de enseñanza

La asignatura se organiza en cuatro unidades didácticas, las cuales se desarrollan en un rango de entre 6 y 8 clases cada una. El dictado del curso se lleva a cabo a través de 2 clases semanales de carácter teórico-práctico, con una duración de 3 horas cada una. La metodología de enseñanza propuesta integra el modelo de aula invertida, el aprendizaje basado en problemas (ABP), el estudio de casos y el uso del aula virtual como complemento esencial a las clases semanales.

Los estudiantes trabajarán con el material de estudio disponible en el aula virtual, que incluye videos, lecturas y casos prácticos, cubriendo conceptos teóricos y ejemplos básicos de programación. Esta plataforma no solo facilita el acceso a los recursos, sino que también permite que los alumnos lleguen a clase con una base sobre la cual construir. Al inicio de cada sesión en clase, se llevará a cabo una revisión y profundización de este material, asegurando que todos los estudiantes hayan comprendido los conceptos clave y estén listos para aplicarlos.

Durante las clases, el docente propondrá y conducirá la actividad de estudio de casos, que se incorporará como una herramienta para analizar situaciones reales o simuladas en el mundo de la programación. Esta actividad permitirá a los estudiantes aplicar y contextualizar los conceptos aprendidos. Estos casos, junto con otros problemas y programas afines propuestos por el docente, servirán como punto de partida para una serie de preguntas conceptuales que fomentarán la discusión y el análisis de las posibles respuestas entre los estudiantes. Esta dinámica tiene como objetivo profundizar en conceptos fundamentales de la programación, permitiendo a los estudiantes no solo entender las soluciones correctas, sino también identificar y discutir errores comunes.

En las clases presenciales, se dedicará tiempo a resolver dudas, realizar ejercicios prácticos y trabajar en problemas específicos. Estos problemas, presentados al inicio de cada unidad, requerirán soluciones a través de la programación. Los estudiantes trabajarán en grupos para discutir y buscar soluciones, con el docente actuando como guía y facilitador. Al final de cada unidad, los grupos presentarán sus soluciones, propiciando un espacio de retroalimentación

y discusión colectiva, enriqueciendo así el proceso de aprendizaje a través de la experiencia compartida.

Además de las clases semanales, y como parte de la fase posterior de la metodología del aula invertida, se complementará la discusión de problemas y ejercicios y la atención de consultas a través de un foro disponible en el aula virtual. Este espacio permitirá la comunicación asincrónica pero continua entre docentes y estudiantes, fomentando un aprendizaje colaborativo y constante, y permitiendo a los estudiantes profundizar y reflexionar sobre lo aprendido en clase.

Evaluación

La evaluación se estructura en base a un enfoque continuo e integrador, reflejando la naturaleza progresiva y acumulativa del aprendizaje en las áreas de informática y cálculo numérico. Esta metodología de evaluación tiene como objetivo no solo medir el conocimiento adquirido, sino también fomentar una comprensión profunda y aplicada de los conceptos y habilidades aprendidos. En este contexto, la evaluación se realiza a través de una evaluación parcial de cada una de las 4 unidades didácticas a través de las cuales se organiza la asignatura, y una evaluación integradora una vez aprobadas las 4 evaluaciones parciales.
El desarrollo de las competencias se evaluará de forma contínua mediante rúbricas, construida en base a los resultados de aprendizaje propuestos.

- **Evaluación Parcial:** cada una de las cuatro unidades didácticas culmina con una instancia de evaluación la cual es precedida por una actividad práctica.
  - *Trabajo Práctico (TP):* Los estudiantes deberán realizar un trabajo práctico que refleje la aplicación de los conceptos y habilidades aprendidos en la unidad. Este trabajo tiene como objetivo desarrollar la capacidad del estudiante para aplicar de manera práctica y efectiva lo aprendido, y puede incluir ejercicios, problemas, proyectos o simulaciones, según lo que sea más pertinente para la unidad en cuestión. Se dispondrá de al menos 2 semanas para poder realizar esta actividad, debiendo la misma ofrecer realimentación inmediata al estudiante de forma que pueda realizarla de forma asincrónica sin la supervisión del docente dentro del aula virtual.
  - *Evaluación Parcial (EP):* Una vez completado el TP con al menos el 60%, los estudiantes se someterán a una evaluación parcial que medirá su comprensión teórica y práctica de los contenidos de la unidad. Esta evaluación puede incluir preguntas teóricas, problemas prácticos y/o análisis de casos, asegurando una evaluación completa de los conocimientos y habilidades adquiridos en la unidad. La duración de esta actividad no deberá superar los 30 minutos y la misma deberá ser realizada bajo la supervisión del docente durante el horario de clase dentro del aula virtual. Se prevé una instancia de recuperación de esta evaluación al finalizar cada unidad didáctica, y una adicional antes de la evaluación integradora (totalizando 6 instancias de recuperación). En cada instancia será posible recuperar una evaluación parcial de cualquier unidad didáctica ya finalizada.

- ➔ **Evaluación Integradora:** una vez aprobadas las 4 evaluaciones parciales, se propone una instancia de evaluación integradora, de la cual
  - ◆ *Práctica Integradora (PI):* Al finalizar cada unidad didáctica, se propondrá al estudiante una actividad integradora, la cual integra todas las unidades ya trabajadas en el curso y consiste en el desarrollo de un programa informático cuya especificación es dada a los estudiantes como enunciado. Esta actividad es temporizada a los fines de poner en práctica la gestión de los tiempos y prioridades en el desarrollo de un programa. Es posible realizarla de forma asincrónica a través del aula virtual, sin necesidad de supervisión docente, y la misma deberá ofrecer realimentación inmediata al estudiante.
  - ◆ *Evaluación Integradora (EI):* Una vez aprobadas todas las Evaluaciones Parciales, los estudiantes deberán presentarse a una evaluación integradora de todas las unidades didácticas. Esta evaluación tiene como objetivo medir la capacidad del estudiante para conectar y aplicar de manera integrada los conceptos y habilidades aprendidos a lo largo del curso. Esta evaluación es temporizada y tendrá una duración de 2 horas, debiendo ser realizada en una clase bajo la supervisión de un docente dentro del aula virtual. Se prevé una instancia de recuperación de esta evaluación.

La combinación de evaluaciones parciales y una evaluación integradora garantiza que los estudiantes no solo adquieran y retengan el conocimiento, sino que también desarrollen habilidades críticas y aplicadas esenciales para su futuro profesional en el campo de la ingeniería.

Condiciones de aprobación

Condiciones de regularización

Para alcanzar la condición de regular, el estudiante debe cumplir las siguientes condiciones:

1. Asistir al menos al 80% de las clases. La asistencia es registrada en el aula virtual mediante la realización de los TPs y las EPs.
2. Alcanzar un rendimiento no inferior a 60% en cada uno de los Trabajos Prácticos (TP).
3. Alcanzar un rendimiento global no inferior a 60% en cada una de las Evaluaciones Parciales (EP).
4. Alcanzar el criterio de aceptación mínimo para los resultados de aprendizaje propuestos.

Condiciones de promoción

Para alcanzar la promoción, el alumno debe cumplir las siguientes condiciones:

1. Alcanzar la condición de alumno regular para lo cual se deben cumplir las condiciones indicadas al respecto.
2. Alcanzar un rendimiento no inferior al 60% en la Evaluación Integradora (EI).

Cumplidas estas condiciones, la nota final de promoción se calculará según la siguiente fórmula:

Nota Final = redondear((20%*TPs + 20%*EPs + 60%*EI)/10)

Donde TPs y EPs corresponden al resultado de promediar los TP y EP de cada unidad respectivamente.

Examen final

Estudiantes Regulares

Los estudiantes regulares deben rendir un examen equivalente a la actividad Evaluación Integradora (EI), el cual será calificado del mismo modo que para los estudiantes que alcanzaron la promoción, considerando para las variables TPs y EPs el rendimiento alcanzado durante la cursada, y para la variable EI el rendimiento alcanzado en el examen final.

Estudiantes Libres

Los estudiantes libres deben rendir un examen que consta de dos partes:
Una prueba de competencias con la misma metodología y objetivos que la Evaluación Parcial (EP), acotada a ejercicios que serán evaluados automáticamente en el aula virtual. La aprobación de esta primera parte es requisito excluyente para la prosecución del examen, y se deberá obtener un rendimiento no inferior al 60%.
Un examen de Evaluación Integradora (EI), con la misma modalidad y objetivos que el requerido a los estudiantes regulares.

La Nota Final final para los estudiantes libres se obtendrá por la siguiente expresión:

$$\text{Nota Final} = \text{redondear}((40\% \text{ EP} + 60\% \text{ EI})/10)$$

Actividades prácticas y de laboratorio

Las actividades prácticas se desarrollan a partir de trabajos prácticos disponibles en el aula virtual. Cada una de las 4 unidades sobre la que se organiza el recorrido formativo de la asignatura comprende un Trabajo Práctico (TP) compuesto por ejercicios que proponen el desarrollo de programas como solución a diferentes problemas generales. Cada problema es descrito a través de un enunciado y al menos una solución particular es ejemplificada indicando el resultado esperado. Para cada problema enunciado se espera que el estudiante provea un código en lenguaje de programación que será evaluado de forma inmediata a través de diferentes pruebas predefinidas. Estas pruebas comprenden soluciones a diferentes casos que el programa debe resolver de forma satisfactoria, y ofrece al estudiante una realimentación inmediata sobre las pruebas superadas y las fallidas. De esta forma, el estudiante puede realizar un diagnóstico sobre los casos donde falla y buscar así modificar su solución para que la misma sea de carácter general, responda a la consigna, y pueda entonces superar todas las pruebas propuestas.

Desagregado de competencias y resultados de aprendizaje

Los resultados de aprendizaje a promover en el desarrollo de la asignatura relacionados con el descriptor “Lenguajes, algoritmos y estructuras de datos” (Tecnología Básica) de la carrera

de Ingeniería en Computación son X. Estos representan una amplia gama de habilidades y conocimientos relacionados con la programación y su aplicación en la resolución de problemas a través del diseño, codificación y verificación de algoritmos en un lenguaje de programación.

- RA1: Comprender y analizar programas codificados en un lenguaje de programación para identificar su funcionalidad y lógica.
- RA2: Analizar y evaluar problemas para determinar los requisitos de programación y diseñar soluciones efectivas antes de iniciar el proceso de codificación.
- RA3: Diseñar soluciones para problemas complejos y traducirlas en algoritmos codificados en un lenguaje de programación.
- RA4: Gestionar eficientemente la memoria de la computadora utilizando variables, constantes y tipos de datos primitivos para almacenar y manipular información.
- RA5: Aplicar operaciones aritméticas y de asignación de manera precisa para resolver problemas matemáticos y científicos.
- RA6: Gestionar la entrada y salida de datos en programas mediante el uso efectivo de flujos de información.
- RA7: Mejorar la legibilidad y la colaboración en el desarrollo de software mediante el uso de comentarios y el seguimiento de convenciones de nomenclatura.
- RA8: Diseñar, crear y aplicar funciones definidas por el usuario para modular y reutilizar el código de manera eficiente.
- RA9: Desarrollar programas con múltiples flujos de ejecución mediante el uso experto de estructuras de control, como condicionales y bucles.
- RA10: Implementar pruebas rigurosas para verificar la precisión y la conformidad con las especificaciones de los programas, y resolver eficazmente los errores identificados.

A continuación, se indican las competencias genéricas y específicas asociadas a los resultados de aprendizaje relacionados con la carrera de Ingeniería en Computación.

Competencias Genéricas	**Resultados de Aprendizaje**
CG1: Identificar, formular y resolver problemas de ingeniería.	RA1, RA2, RA8, RA9
CG4: Competencia para utilizar de manera efectiva las técnicas y herramientas de la ingeniería.	RA3, RA4, RA5, RA6
CG7: Comunicarse con efectividad.	RA7, RA10

Competencias Específicas (Ingeniería en Computación)	**Resultados de Aprendizaje**
CE7.2.2 Sintetizar, diseñar, desarrollar y analizar programas lenguajes de programación de bajo nivel, como C y C++.	RA1, RA2, RA3, RA4, RA5, RA6, RA8, RA9
CE7.2.3 Seleccionar y utilizar entornos de desarrollo integrados (IDE) y herramientas de depuración específicas	RA7, RA10

para este tipo de sistemas.	

Bibliografía

- Bronson, Gary, *C++ para Ingeniería y Ciencias (2da. edición)*, International Thomson Editores, México, 2007
- Deitel, H. M., Deitel, P. J., *Cómo programar en C++,* Pearson Educación, 2015

FCEFyN
FACULTAD DE CIENCIAS EXACTAS, FÍSICAS y NATURALES

UNC Universidad Nacional de Córdoba

Asignatura: **Ingeniería de Software y Hardware**

Código:	RTF	8
Semestre: 5	Carga Horaria	96
Bloque: TA	Horas de Práctica	48

Departamento: Computación

Correlativas:
- Programación Avanzada
- Electrónica Digital 2

Contenido Sintético:
- Historia y visión general. Herramientas relevantes, estándares y/o restricciones de ingeniería.
- Gestión de proyectos.
- Riesgo, confiabilidad, seguridad y tolerancia a fallos.
- Procesos de hardware y software. Análisis y elicitación de requisitos.
- Especificaciones del sistema.
- Diseño y evaluación arquitectónica del sistema.
- Diseño concurrente de hardware y software.
- Integración, pruebas y validación de sistemas.
- Mantenimiento, sostenibilidad, manufacturabilidad

Competencias Genéricas:
- CG2: Concebir, diseñar y desarrollar proyectos de ingeniería (sistemas, componentes, productos o procesos).
- CG3: Gestionar -planificar, ejecutar y controlar- proyectos de ingeniería (sistemas, componentes, productos o procesos).
- CG4: Utilizar de manera efectiva las técnicas y herramientas de aplicación en ingeniería.
- CG7: Comunicarse con efectividad.
- CG8: Actuar con ética, responsabilidad profesional y compromiso social,
- considerando el impacto económico, social y ambiental de su actividad en el contexto local y global.
- CG10: Actuar con espíritu emprendedor.

Aprobado por HCD: NNNN-HCD-AAAA RES: Fecha: DD/MM/AAAA

Competencias Específicas:

- CE1.2 Analizar, especificar, diseñar y proyectar arquitectura de sistemas informáticos.
- CE1.4 Implementar y mantener sistemas informáticos.
- CE4.3: Analizar, diseñar, programar, implementar, probar, depurar y evaluar hardware y software para sistemas de computación de propósitos específicos.
- CE4.4: Analizar, diseñar, programar, implementar, probar, depurar y evaluar hardware y software para sistemas de computación de propósitos generales.
- CE4.11 Analizar, proyectar y desarrollar proyectos de software y sistemas conjuntos de hardware y software haciendo uso de conceptos, métodos y herramientas de gestión de proyectos, ingeniería de software, elicitación, análisis, especificación y validación de requerimiento.
- CE8 Certificar el funcionamiento, condición de uso o estados de Sistemas de Procesamiento de Señales, Sistemas Embebidos, Sistemas Computarizados de automatización y control, Sistemas Conjuntos de Hardware y Software.

Presentación

La asignatura Ingeniería de Software y Hardware pertenece al 5(quinto) semestre (3 año) de la carrera de Ingeniería en Computación.
Al inicio de éste espacio curricular el estudiante ya cuenta con bases sólidas de algoritmos, programación y de estructuras de datos que sirven de base para el desarrollo de la presente propuesta educativa. En ella se busca conocer, descubrir, aprender, implementar, experimentar, emplear y mantener procesos y metodologías de desarrollo, como así también buenas prácticas de ingeniería ampliamente usados y probados en la industria del desarrollo de software y hardware.

Durante el transcurso de la asignatura exploramos las dificultades de producción de software y hardware a gran escala, su configuración, su administración, su control y mantenimiento; para finalmente arribar en implementaciones de procesos de ingeniería que brinden soluciones a dichas problemáticas.
La asignatura está diseñada desde un enfoque constructivista donde los estudiantes deben organizarse alrededor de metodologías ágiles para ir cumpliendo los retos y desafíos que proponemos entregar valor al final de cada sprint de 2 semanas de duración.
La cátedra ha establecido pautas para conformar un esquema de autonomía alineada, donde los docentes establecen los lineamientos y los equipos de trabajo tienen autonomía para resolver problemáticas reales y desafíos de la industria en términos de Software y Hardware.
En términos más concretos, los docentes explican, ejemplifican y facilitan herramientas, patrones, métricas, etc, pero son finalmente los alumnos quienes deben implementarlas, adaptarlas, configurarlas, probarlas y mantenerlas de acuerdo a cada singular problemática que se desea resolver.

Contenidos

Módulo I. Ingeniería de Software y Hardware
Objetivo: manejar el contexto de la Ingeniería de Software y Hardware, el lenguaje y la terminología asociada a la disciplina.

Unidad 1. Introducción
Contenido: Introducción a la Ingeniería de Software.

Unidad 2 . Procesos de Software
Contenido: Modelos de Desarrollo de Software y Hardware Tradicionales y Metodologías Ágiles. Sus comparaciones. Ventajas y Desventajas. Actividades del Proceso: Especificación del proceso, Diseño e implementación del software, Validación y Verificación del Software y Hardware, evolución del software. Herramientas.

Unidad 3: Metodologías Ágiles

Contenido: Modelos Ágiles de Desarrollo. Manifiesto Agile. Modelos Modernos de Metodologías Ágiles. Combinaciones de Metodologías de acuerdo al contexto y naturaleza de los Proyectos de Ingeniería de Software y Hardware. Métricas. Herramientas. Indicadores. Artefactos. Acuerdos y Definiciones. Buenas Prácticas. Retos y Desafíos Modernos.

Módulo II. Requerimientos y Administración de la Configuración

Objetivo: manejar el contexto de la Ingeniería de Requerimientos, el lenguaje y la terminología asociada a esta disciplina. Establecer las pautas y lineamientos de la administración de la configuración.

Unidad 4: Requerimientos del Software y Hardware
Contenido: Requerimientos funcionales, no funcionales y de dominio. Requerimientos de Usuario. Requerimientos de Sistema. Especificaciones de la Interfaz. Procesos de la Ingeniería de Requerimientos. Adquisición, Elicitación y Análisis de requerimientos. Validación de requerimientos. Gestión de requerimientos.

Unidad 5: Administración de la Configuración
Objetivo: Identificar y establecer los esquemas necesarios para una correcta gestión de las configuraciones, sus implicancias, los procesos y las herramientas de Control de Versiones.
Contenido. Gestión del Versionado. Sistemas de Buildings. Manejo de los Cambios

Módulo III. Diseño, Modelado e Implementación.
Objetivo: desarrollar habilidades y manejar las técnicas de diseño, modelado y construcción de sistemas de software y hardware

Unidad 6: Modelos del Sistema Software y Hardware
Contenido: Modelos de Contexto. Modelos de Comportamiento. Modelos de datos. Modelos de Objetos. Métodos Estructurados. Modelado de Sistemas. UML.

Unidad 7: Diseño arquitectónico del Software
Contenido: Decisiones del diseño arquitectónico. Organización del sistema. Estilos de descomposición modular. Patrones de Arquitecturas Tradicionales. Patrones Modernos. Arquitectura distribuidas.

Unidad 8: Sistemas Basados en Componentes
Contenido: Component-Based Software Engineering (CBSE). Modelos Basados en la reutilización. Procesos y Flujos de Trabajo con Componentes. Patrones de Diseño. Patrones Tradicionales de Diseño. Patrones Creacionales, Estructurales y Conductuales. Patrones Modernos de Diseño.

Módulo IV. Pruebas y Distribución de Software y Hardware

Unidad 9: Pruebas del Software
Contenido: Pruebas Unitarias. Pruebas de Integración. Pruebas de Sistema. Pruebas de Aceptación. Pruebas de UI. Diseño de Casos de Prueba. Escenarios. Automatización. Ambientes de pruebas. Trackeo y seguimiento de Errores, Fallas y Defectos.

Unidad 10: Distribución y Puesta Productiva.
Contenido: Gestión de las Versiones Productivas. Canales Alfa, Beta. Canary Releases. Gestión Remota de las funcionalidades. Dark Launch. Técnicas de Distribución para grandes desarrollos y múltiples equipos. Release Train. Técnicas de Distribución para productos de software/hardware con millones de usuarios.

Metodología de enseñanza

Las clases incluyen actividades donde priman las teóricas, prácticas y de laboratorio alternativamente. Las actividades teóricas se realizan a través de exposición dialogada, orientadas a aproximar a los estudiantes a las metodologías de desarrollo de software y hardware y su ámbito de aplicación.

Durante el desarrollo de los Trabajos Prácticos se realizan actividades que le permiten al estudiante poner en práctica las habilidades y verificar los criterios y técnicas de modelado, diseño y pruebas. Tanto el dominio del problema, como la naturaleza de las soluciones, el alcance y el enfoque del proyecto son ideados y proporcionados por los estudiantes, consiguiendo mayor compromiso y un sin fin de casuísticas y variedad de proyectos únicos para la cátedra. Asimismo ésto permite trabajar con problemáticas reales, con casos de usos concretos y sacados de la realidad de los estudiantes y su entorno, pudiéndose repetir dichos escenarios en su vida profesional.

Respecto de los Prácticos de Laboratorio y el Proyecto Final de la Asignatura, se ha diseñado e implementado un sistema de seguimiento al estilo de la metodología ágil, donde dividimos las clases, los prácticos y las entregas de los trabajos en 7 sprint /7 entregas; es decir una entrega cada 2 clases. Ésto nos permite un seguimiento mucho más profundo y un trackeo más individual, con el objeto de identificar posibles desviaciones al plan original donde se intenta y se logra establecer un objetivo tanto individual como en equipo por sprint, de manera que sea bien claro y acotado. Con ésta metodología original, hemos innovado en la participación y compromiso individual y colectivo desde el sprint 1, generando más motivación y entusiasmo para lograr terminar la propuesta educativa y el programa de la materia. Tanto la cantidad de sprints como el alcance puede ir variando de cohorte en cohorte con el objetivo de ajustar mejor la propuesta a los cambios generacionales de las metodologías.

Evaluación

La evaluación de los conocimientos disciplinares se realizará a través de dos parciales a lo largo del semestre y las evaluaciones de las habilidades y competencias se realizarán de forma continua al finalizar cada sprint/bloque con las entregas y exposiciones de los trabajos realizados que vayan aportando valor a su trabajo final.
Al finalizar, los equipos de desarrollo deberán exponer una presentación final de su proyecto desde el inicio hasta su estado actual, luego de esas iteraciones y haciendo una

distribución/release del software/hardware funcionando con las características/features comprometidas en las etapas tempranas del proyecto.
Las actividades propuestas están diseñadas teniendo en cuenta los resultados de aprendizaje, para que su conclusión sea indicador de haber alcanzado un nivel de desarrollo aceptable de las competencias propuestas.

La calificación final será una combinación de los exámenes, las presentaciones al final de los sprints y la presentación final.

Condiciones de aprobación

Calificación:
La calificación se obtendrá a través del siguiente polinomio:

CALIFICACIÓN= 0,2xP1 + 0,2P2 + 0,4xS + 0,2PF
Donde:
P1: Es la nota del primer examen parcial.
P2: Es la nota del segundo examen parcial.
S: Es el promedio de las calificaciones de las actividades prácticas entregadas al final de los Sprints donde se evalúa a nivel individual y grupal pudiendo diferir entre integrantes del mismo equipo por cuestiones relacionadas con la exposición, el compromiso, la participación, la colaboración con otros equipos, el mentoreo, el liderazgo, etc.
PF: Es la calificación de la presentación final donde se evalúa no solo la exposición, sino también la capacidad de resolución de problemas y los casos de éxito y fracaso; valorando positivamente la capacidad de resiliencia y tolerancia a dichas condiciones adversas. Aquí se busca motivar, premiar y desarrollar esta habilidad muy necesaria para formar futuros emprendedores y líderes
.
Requisitos para alcanzar la regularidad.

- 80% de asistencia.
- Rendir y aprobar los dos parciales con un 60% o más, el alumno podrá recuperar sólo un parcial.
- Aprobación del 100% de las actividades prácticas propuestas al final del Sprint.
- Aprobación de la Presentación Final del Trabajo de Laboratorio.

Requisitos para alcanzar la Promoción.

- Cumplir los requisitos de regularidad
- Tener aprobadas las materias correlativas
- La calificación final debe ser igual o mayor que 7.

Los alumnos que no alcancen la condición de regular al finalizar la Presentación Final serán considerados libres.

Actividades prácticas y de laboratorio

Se espera que los alumnos desarrollen dos trabajos prácticos y un trabajo final durante el transcurso de la asignatura. Para ello se emplea una metodología de revisión semanal y

entrega parcial cada dos semanas y el desarrollo de los trabajos será en grupos de 4 estudiantes como máximo, pudiendo modificarse si la cátedra lo considera necesario. Esto tiene por objetivo que los alumnos empleen técnicas de desarrollo ágil que son prácticas estándares en la industria.
El objeto sobre el cual se desarrollaran las actividades serán de libre elección de los estudiantes, pudiendo tratarse de aplicaciones para teléfonos, tablets o de escritorio, sitios web, mejoras sobre software desarrollado en alguna otra materia e incluso sobre software libre de cualquier índole. En todos los casos, la propuesta será analizada por el docente para determinar que cumpla con lo esperado para la materia según complejidad, posible duración del proyecto, alcance, entre otros parámetros y puntos básicos comunes.
Habiendo definido el tema del trabajo, cada grupo deberá presentar tres documentos.

- ***Plan de Gestión de las Configuraciones.***
- ***Documento de Requerimientos.***
- ***Casos de Pruebas de Sistema***

En las secciones siguientes se detallarán los requisitos mínimos para cada uno de los documentos.

Plan de Gestión de las Configuraciones

Siguiendo los planes de administración y control de las configuraciones mostrados en clase, cada grupo deberá elaborar y presentar un **Plan de Gestión de las Configuraciones** que incluya como ***mínimo*** los siguientes puntos:

1. Dirección y forma de acceso a la herramienta de control de versiones.
2. Dirección y forma de acceso a la herramienta de integración continúa.
3. Dirección y forma de acceso a la herramienta de gestión de defectos.
4. Esquema de directorios y propósito de cada uno.
5. Normas de etiquetado y de nombramiento de los archivos.
6. Plan del esquema de ramas a usar.
7. Políticas de fusión de archivos y de etiquetado de acuerdo al progreso de calidad en los entregables.
8. Forma de entrega de los “releases”, instrucciones mínimas de instalación y formato de entrega.
9. Change Control Board. Se debe incluir el propósito, la lista y forma de los integrantes del equipo y su rol en la CCB, la periodicidad de las reuniones, etcétera.
10. Herramienta de seguimiento de defectos usada para reportar los defectos descubiertos y su estado. Forma de acceso y dirección.
11. Cualquier otra información relevante.

Documento de Requerimientos

Siguiendo los modelos expuestos en clase, cada grupo debe elaborar un **documento de requerimientos**.

- El documento debe presentar el detalle de los requerimientos funcionales y no funcionales.
- También, deberá incluir diagramas de casos de uso, diagramas de actividades, diagramas de secuencia y cualquier otro diagrama que considere necesario para mejorar el detalle y la explicación de los requerimientos del software a construir.

- Alternativamente, se debe generar una matriz de trazabilidad entre los requerimientos y los casos de uso generados.

Casos de Pruebas de Sistema

Además, se deben generar los **Casos de Prueba de Sistema** contra los requerimientos funcionales y no funcionales. Además de los casos de prueba de uso normal, incluir casos de prueba alternativos que prueban valores límites o inusuales y que tratan de generar errores no esperados.

Diseño, Implementación y Pruebas

En éste trabajo práctico, cada grupo presentará una serie de documentos detallando el diseño e implementación de su proyecto además de las pruebas realizadas sobre el sistema.

Arquitectura del Sistema

Se debe añadir al documento de requerimientos creado en el trabajo práctico anterior una sección que incluya un **diagrama de arquitectura preliminar** que permita asociar requerimientos y casos de usos con los sistemas, subsistemas y módulos identificados.

Por otro lado, se debe generar otro documento de **arquitectura** donde se presente:

- Un gráfico de arquitectura general para mostrar los componentes y sus relaciones con las interfaces externas.
- La explicación del **patrón de arquitectura** que fue usado y por qué, haciendo énfasis en cómo resuelve los requerimientos no funcionales.
- Diagramas de despliegue y de componentes.
- Opcionalmente se puede incluir un diagrama de contexto que incluya la relación de los sistemas y los subsistemas con las actividades del dominio del conocimiento de la aplicación.

Se deben definir los casos de **prueba de integración** que verifican la correcta interacción entre todos los componentes del sistema.

Diseño del Sistema

Se debe presentar un **documento de diseño** en el cual se incluyan diagramas de paquetes, diagramas de clases y objetos, diagramas de secuencia y/o todo aquel diagrama que sirva para explicar el diseño del software a construir.
Se debe mostrar también la aplicación de patrones de diseño usando diagramas adecuados que detallen su implementación e indicar por qué se utilizan y qué problemas se solucionan con ellos.

Por otro lado, se deben generar **pruebas unitarias automáticas para el código**, explicando cómo correrlas y verificar su estado.

Opcionalmente, se puede incluir cualquier otra herramienta de gestión de la calidad de software que haya sido usada. Mostrando ejemplos de su uso y se debe agregar al Plan de Gestión de las Configuraciones la dirección y forma de acceso a la herramienta.

Presentación Final

Además de todos los documentos generados en el Trabajo Práctico 1 y en los puntos anteriores del Trabajo Práctico 2, se deben presentar tres documentos:

- Notas de entrega del proyecto (release notes).
- Informe sobre el trabajo realizado.
- Presentación para mostrar ante sus compañeros.

La nota de entrega debe incluir:

- Breve listado de la funcionalidad incluida (con el estado de implementación de cada uno).
- “Pass/Fail Ratio (porcentaje de pruebas pasadas/falladas)” para todas las pruebas realizadas (pruebas de sistema, aceptación, integración y unitarias).
- Número de defectos identificados y corregidos (agrupados por severidad).
- Defectos conocidos (no resueltos) al momento de entregar el proyecto.
- Dirección de acceso o archivo con los archivos del proyecto y sus instrucciones de instalación y ejecución.

El informe sobre el trabajo realizado debe incluir:

- Detalle de dedicación de esfuerzo para realizar el trabajo (en personas horas), distinguiendo la contribución personal de cada miembro del grupo y el esfuerzo invertido por cada tarea realizada. Esto incluye todas las tareas para poder construir la aplicación, documentar y elaborar el informe y la presentación de esta materia.
- Lecciones aprendidas durante la elaboración del práctico y errores cometidos.

Las condiciones requeridas para los prácticos podrían ir actualizándose en función de las necesidades y las modernizaciones en las técnicas/metodologías.

Desagregado de competencias y resultados de aprendizaje

Las instancias de evaluación están desarrolladas para certificar que los estudiantes respalden de manera efectiva las competencias profesionales requeridas para el alcance de ésta materia. Las evaluaciones fueron alineadas a la metodología centrada en el estudiante y enfocadas en desarrollar habilidades prácticas.

Competencia	Resultado de Aprendizaje	Mínimo	Valoración
CG2 CE1.2	Interpreta correctamente el dominio de un problema.	2	
CG2 CG3	Detecta y comunica errores, fallas, defectos y oportunidades de mejoras tanto en el proceso	2	

CG7 CE1.2 CE4.3 CE4.4 CE8	como en el producto elaborado		
CG3 CG4 CE1.4 CE4.3 CE4.4 CE4.11	Selecciona y emplea adecuadamente las técnicas y herramientas a aplicar para un problema dado	2	
CG4 CG7 CE1.2 CE4.11 CE8	Comunica de Manera Efectiva los resultados de los procesos de implementaciones de herramientas, las mediciones, los inconvenientes, los desafíos y las acciones realizadas para superar los obstáculos	2	
CG3 CG7 CE1.2 CE4.11	Desarrolla habilidades comunicacionales efectivas para elicitar, redactar y validar requerimientos.	2	
CG8	Identifica los aspectos del desarrollo profesional que por su origen y naturaleza tienen connotaciones éticas	2	
CG3 CG7 CG8 CG10	Respeta las pautas de trabajo establecidas en clase para las actividades de equipo.	2	
CG3 CG7 CG10	Trabaja en equipo asumiendo los distintos roles dentro de un grupo de trabajo	2	
CG10 CE1.4 CE4.3 CE4.4	Lidera, Transforma, Motiva y/o Propone mejoras, cambios, adecuaciones y soluciones a los paradigmas actuales, a los procesos aprendidos, a los desarrollos actuales con el fin de encontrar propuestas superadoras que generen nuevas oportunidades.	2	

Se realizan autoevaluaciones periódicas buscando generar un espacio de reflexión sobre su progreso de aprendizaje y su desarrollo en dichas competencias. Esta autoreflexión les permite identificar oportunidades de mejoras y tomar decisiones para poder afrontar dicho desafío.

A los fines de evaluar los trabajos se emplearán rúbricas para que el estudiante pueda discernir los objetivos que alcanzó y en qué grado a modo de retroalimentación.

El rango de valoración es de 1 a 3 y se corresponde a:
1. Insuficiente: No se evidencia el nivel de desarrollo de las competencias esperado a través de los resultado de aprendizaje
2. Suficiente: En la mayoría de las situaciones se evidencia el nivel de desarrollo deseado.
3. Alto: Se evidencia un claro desarrollo de las competencias esperado a través de los resultados de aprendizaje.

Bibliografía

BIBLIOGRAFIA Principal

- Ian Sommerville (2016) Ingeniería del Software (10 Edición) Pearson

BIBLIOGRAFIA Complementaria

- Gergely Orosz (2021) Building Mobile Apps at Scale: 39 Engineering Challenges (2 Edición) Primedia E-launch LLC
- Eric Freeman (2021) Head First Design Patterns (2 Edición) O'Reilly Media
- Eric Evans (2003) Domain-Driven Design: Tackling Complexity in the Heart of Software (1 Edición) Pearson Education
- Robert C Martin (2017) Clean Architecture (1 Edición) Pearson
- Robert C Martin (2008) Clean Code (1 Edición) Pearson

FCEFyN
FACULTAD DE CIENCIAS EXACTAS, FÍSICAS y NATURALES

UNC Universidad Nacional de Córdoba

Asignatura: **Inteligencia Artificial**

Código:	RTF	8
Semestre: 7	Carga Horaria	80
Bloque: Tecnologías Aplicadas	Horas de Práctica	40
Departamento: Computación		

Correlativas:
- Análisis Matemático 2
- Probabilidad y Estadística
- Base de Datos

Contenido Sintético:
- Historia, áreas e impacto de la Inteligencia Artificial (IA).
- Fundamentos del aprendizaje automático.
- Nociones de redes neuronales.
- Modelos de IA.
- Arquitecturas avanzadas de IA.
- Los datos y la IA.
- Aplicaciones de la IA.

Competencias Genéricas:
Tecnológicas:
- CG2: Concebir, diseñar y desarrollar proyectos de ingeniería (sistemas, componentes, productos o procesos). (M)
- CG4: Utilizar de manera efectiva las técnicas y herramientas de aplicación en ingeniería. (A)
- CG5: Contribuir a la generación de desarrollos tecnológicos y/o innovaciones tecnológicas. (A)

Sociales, políticas y actitudinales:
- CG8: Actuar con ética, responsabilidad profesional y compromiso social, considerando el impacto económico, social y ambiental de su actividad en el contexto local y global. (A)
- CG9: Aprender en forma continua y autónoma. (A)

Aprobado por HCD: NNNN-HCD-AAAA RES: Fecha: DD/MM/AAAA

Competencias Específicas:
CE1.3 Conocer, desarrollar nuevos e implementar algoritmos y estructuras de datos.

Presentación

La Inteligencia Artificial (IA) es una disciplina multidisciplinaria que está siendo requerida como formación esencial de cualquier profesional, más allá de las Ingenierías, las ciencias de la computación y las restantes áreas STEM. En este escenario, nuestros futuros ingenieros incorporan una nueva perspectiva que pone especial énfasis en acciones de sensibilización hacia su compromiso social, político y ambiental.

En el mismo sentido, incluir los Objetivos de Desarrollo Sostenibles (ODS), específicamente para formar talento humano en IA, es un ambicioso objetivo que trasciende al ODS 4. La IA está inmanente en la gran mayoría de los productos y servicios de la Economía del Conocimiento.

El objetivo principal de la asignatura consiste en promover la adquisición de competencias para apropiarse de estas nuevas tecnologías con el compromiso de trabajar por una IA fiable durante todo su ciclo de vida, centrada en las personas, hacia el bien común, con perspectiva de género, teniendo como horizonte la soberanía tecnológica.

Acompañar a nuestros jóvenes estudiantes en este desafiante camino para que emerjan profesionales tecnológicamente formados y comprometidos como actores clave del ecosistema de la IA, es la misión de esta cátedra que, a su vez, está íntimamente articulada con LIDeSIA[1], de reciente creación.

Inteligencia Artificial es una asignatura que pertenece al cuarto año (séptimo semestre) de la carrera de Ingeniería en Computación, está disponible como materia selectiva para las demás carreras de la Facultad y además mantiene abierta la convocatoria a estudiantes vocacionales de otras disciplinas (los procesos de evaluación se adecuan a sus saberes previos).

Al momento de transitar este espacio curricular el estudiante posee la formación básica requerida de: fundamentos de la programación y bases de datos, análisis matemático, álgebra lineal y estadística.

IA forma parte del bloque de tecnologías aplicadas, constituyendo una de las disciplinas inmanentes en el ejercicio profesional cotidiano del ingeniero actual. Articula con otros actores, a través de LIDeSIA, para ofrecer al estudiante un contexto lo más cercano posible a la realidad actual. La cátedra está abierta a la articulación horizontal y vertical con otros espacios curriculares.

Contenidos

El temario se orienta a los núcleos estratégicos de hegemonía internacional en la materia, con enfoque basado en competencias para saber SER, saber CONOCER y saber HACER; desde una perspectiva multidisciplinaria.

En esta asignatura se describen los diversos paradigmas de la Inteligencia Artificial y se profundiza en Aprendizaje Automático a partir de ejemplos (datos) y en Sistemas

[1] Laboratorio de Investigación y Desarrollo de Software en Inteligencia Artificial

conexionistas, más específicamente Redes Neuronales Artificiales. Ello en razón de los requerimientos actuales.
Se incorporan dimensiones éticas y regulatorias en relación a la IA, basadas en los avances que los organismos internacionales desarrollan para su adopción responsable.
Unidad 1: HISTORIA, ÁREAS E IMPACTO DE LA INTELIGENCIA ARTIFICIAL (1 semana)
Evolución de la Inteligencia Artificial. Problemas y Soluciones. Regresión. Optimización. Inteligencia Computacional, Agentes inteligentes, Sistemas Inteligentes. Aspectos éticos y regulatorio de organismos internacionales para facilitar su adopción responsable, centrada en los derechos fundamentales de las personas y la mitigación de los inequidades.
Generalidades de: Ingeniería del Conocimiento, Aprendizaje Automático Basado en Datos y Descubrimiento de Conocimiento, Sistemas Conexionistas: Redes neuronales y otros modelos computacionales.
Unidad 2: FUNDAMENTOS DEL APRENDIZAJE AUTOMÁTICO. (2 semanas)
Aprendizaje Automático. Representación de la Información. Regresión y Clasificación. Formas de Aprendizaje. El descenso por el Gradiente. Modelos de Machine Learning seleccionados bajo el criterio de un adecuado compromiso entre relevancia y actualidad. Generalización. Complejidad. Hiperparámetros. Técnicas de Regularización. Validación cruzada. Métricas
Aplicaciones en Laboratorio
Unidad 3: LOS DATOS Y LA IA. (3 semanas)
Proceso de Explotación de la Información (Aprendizaje Supervisado): Descubrimiento de Reglas. Algoritmos. Aplicaciones en Laboratorio. Métricas: ROC, matriz de confusión, exhaustividad, energía, asertividad, precisión, etc.
Proceso de Explotación de la Información (Aprendizaje no Supervisado): Descubrimiento de Grupos. Algoritmos de Clustering. Aplicaciones en Laboratorio. Métricas: Elbow y otras
Proceso de Explotación de la Información: Interdependencia de atributos basados en Algoritmos de Redes Bayesianas
Proyecto de IA y DATOS. Elección de los procesos y algoritmos intervinientes.
Aplicaciones en Laboratorio
Unidad 4: NOCIONES DE REDES NEURONALES. (1 semana)
Los modelos conexionistas. Redes Neuronales: El modelo Biológico. Redes Neuronales Artificiales. Computador vs. Cerebro. Simulación Emulación. Inteligencia Computacional. Componentes de un Sistema conexionista. Arquitectura. Reglas de Propagación. Función de Activación. Función de Transferencia. Estado de Activación. Modelos de Neuronas. Modos de operación: Aprendizaje y Recuerdo o Ejecución. Mecanismos de Aprendizaje. Representación de la Información de entrada – salida. Computabilidad Neuronal.
Unidad 5: MODELOS DE IA (3 semanas)
Redes con conexiones hacia adelante. Aprendizaje Supervisado. Algoritmo de entrenamiento del Perceptron simple. Puertas lógicas.
El Perceptron Multicapa. Arquitectura. Aprendizaje. Operación. La red Backppropagation. La regla delta generalizada. Incorporación del momentum. Dimensionamiento de la Red. Cantidad de patrones. Capacidad de generalización. Número de capas ocultas. Número de neuronas por capa oculta. Representación de la información de salida. Hiperparámetros de ajuste de la red. Funciones de transferencia. Desvanecimiento y Explosión del Gradiente.
Aprendizaje no supervisado: Modelo de Hopfield, Mapas auto organizados. Aprendizaje por refuerzo, RNN con LSTM y GRU.
Aplicaciones en Laboratorio
Unidad 6: ARQUITECTURAS AVANZADAS DE IA. (6 semanas)

Aprendizaje profundo: Redes convolucionales. La convolución como filtro automatizado: tamaño de la malla, padding, stride. Los canales RGB. El agrupamiento o pooling. Las funciones de linealización. Capas densas, capas planas, totalmente conectadas. Representación de las etiquetas de salida, las funciones de distribución de probabilidad, softmax; el aprendizaje por corrección del error (Backpropagation), etc.
Aplicaciones en Laboratorio
RBM (máquina de Boltzmann Restringida). Modelos Generativos. Transfer Learning, Autoencoders. Modelos avanzados para lenguaje natural, imágenes y otros tipos de datos. Transformers y mecanismos de atención. Nuevos modelos fundacionales.
Aplicaciones en Laboratorio

Metodología de enseñanza

Las etapas de construcción y elaboración de los saberes son sustentadas mediante la exposición dialogada como estrategia didáctica y el empleo de proyección de diapositivas, filminas y pizarrón (o tableta digitalizadora) como materiales didácticos.

El estudiante, en un **trabajo colaborativo grupal** desde el comienzo de las clases, se aboca a construir sus saberes para atravesar con éxito los desafíos de la asignatura.

En **el aula virtual** se organiza el desarrollo de la asignatura, además la cátedra incorpora grupo de mensajería en WhatsApp. La disponibilidad de: clases actualizadas grabadas, repositorio de actividades prácticas y videos de laboratorio, también contempla la oportunidad de facilitar el acceso de los estudiantes con restricciones horarias para el cursado, a los que se les hace seguimiento personalizado todas las semanas.

Dada la naturaleza de la disciplina en continua evolución, las clases se enriquecen con modelos y herramientas que se actualizan escogiendo entre las versiones libres disponibles y más usadas. Las actividades asíncronas facilitan el cumplimiento de RTF y el aula virtual está organizada en ese sentido.

La dinámica constructivista escogida para el desarrollo de la **primera unidad,** muy extensa y de gran impacto en el componente actitudinal del aprendizaje, permite al estudiante apropiarse de disparadores que van emergiendo en la asignatura.

Las demás unidades se desarrollan con exposición dialogada de aspectos fundamentales de cada uno de los contenidos, inherentes a los componentes esenciales de las tecnologías aplicadas, relativos a: la comprensión del problema en un dominio determinado para construir una solución, la detección de los requerimientos específicos y los actores involucrados durante el ciclo de vida de la IA fiable. Cobran relevancia: los datos, los modelos, la infraestructura y las condiciones éticas, regulatorias y su impacto.

Uno de los ejes centrales de la asignatura son los fundamentos del Aprendizaje automático (Unidad 2). A través de exposición detallada de conceptos y de prácticas de laboratorio, el estudiante se fortalece en saberes relativos a las métricas de evaluación de perfomance, las técnicas de regularización para atenuar la

complejidad de los modelos y el adecuado manejo de los hiperparámetros para optimizar su entrenamiento.

El segundo eje (Unidad 3) se destina a desentrañar los paradigmas del Aprendizaje Basado en Datos que nutre a la Minería de Datos y a la Ingeniería de la Explotación de la Información, rama ingenieril denominada también como Inteligencia de Negocios. La dinámica escogida implica varios aspectos ya que se pretende que el estudiante se familiarice con las herramientas open source disponibles, que contienen librerías de algoritmos, inteligentes o no, utilizados en diversos dominios adonde se aplica Minería de Datos: Data Science y Machine learning. Se destacan los algoritmos asociados, sus limitaciones, potencialidades y usos más habituales y, otros procesos conexos como el pre procesamiento de los datos, la interpretación de la información obtenida de los algoritmos para construir conocimiento, el tipo de resultados que arrojan cada uno de ellos, tales como: reconocimiento de patrones, agrupamientos, descubrimientos de reglas para clasificar, etc. El estudiante se enfrenta a comprender las potencialidades de las distintas técnicas, investigar las herramientas disponibles, para luego documentar un proceso de solución frente a un problema identificado en un dominio adonde cuente con acceso a datos. La Metodología utilizada es la de Aprendizaje Basado en Problemas, colaborativo en equipo multidisciplinario.

Por último, Las Redes Neuronales, los sistemas conexionistas en general y los modelos y arquitecturas más actuales, conforman el tercer eje (Unidades 4, 5 y 6). En exposición dialogada, se abordan las características generales de los sistemas conexionistas y se desarrollan, con distinta profundidad, modelos de Redes Neuronales Artificiales, destacando los saberes que contribuyen a profundizar en otros modelos de Inteligencia Artificial. El trabajo colaborativo facilita el aprendizaje de diseño, desarrollo y despliegue de Modelos fundacionales y otras arquitecturas más avanzadas. Se usan plataformas de IA con modelos pre entrenados y con arquitecturas actuales de RNA, que contemplan mejoras tales como Continued Learning, Reinforcement Learning, Meta Learning, Autoencoders, Fine tuning, sólo por mencionar algunas vigentes; aplicadas a problemas de diversos campos como imágenes y procesamiento de lenguaje natural. De esta manera el estudiante se explica las potencialidades para valorar su uso en el desarrollo de proyectos.

Se invita a ex estudiantes, integrantes de LIDeSIA y de otros espacios, para compartir sus experiencias en Inteligencia Artificial aplicada, constituyendo un aporte en la formación del futuro profesional. Se invita a los estudiantes a sumarse a las líneas de trabajo de LIDeSIA, jerarquizando a la propuesta como facilitadora de egreso.

Evaluación

La estrategia incorpora la evaluación continua, el seguimiento de los estudiantes, el trabajo en equipo y la multidisciplinariedad, como requerimientos del perfil del egresado actual.

Se realizan evaluaciones conceptuales de proceso (ECP), de carácter formativo. Estas impactan en el desempeño del estudiante de forma cualitativa, son requisitos para alcanzar la regularidad y atraviesan todas las jerarquías de la taxonomía de Bloom. Se organizan en cuestionarios que se distribuyen por temas durante el desarrollo del curso.
Para la acreditación de la asignatura, evaluación de carácter sumativo, se implementan instancias de evaluación continua de los trabajos de laboratorio realizados en equipos multidisciplinarios que incluyen aspectos éticos de la Inteligencia Artificial Aplicada. Están distribuidas a lo largo de toda la cursada (se describen en el apartado de Actividades Prácticas y de Laboratorio) bajo la modalidad de coloquio.
El estudiante debe aprobar todas las instancias detalladas, contando para ello con los espacios destinados a la evaluación continua y con reuniones programadas de manera conjunta con la cátedra, por fuera del horario de clases, durante el semestre correspondiente al calendario académico.

Condiciones de aprobación

Las evaluaciones de carácter formativo están diseñadas para alcanzar la condición de alumno regular, en tanto que las sumativas, para acreditar la asignatura. Todas, están concebidas para ser abordadas de forma presencial o virtual.

Para alcanzar la condición de ALUMNO REGULAR se debe:
- Asistir al 80% de las clases teórico-prácticas y de laboratorio.
- Alcanzar un nivel superior al 60% en cada una de las ECP.

Para aprobar la asignatura por promoción se debe:
- Alcanzar la condición de ALUMNO REGULAR.
- Aprobar todas las instancias de evaluación continua de Actividades prácticas y de Laboratorio que la cátedra implementa, alcanzando un nivel de desarrollo aceptable en relación a las competencias generales y específicas que se describen en el apartado “Desagregado de competencias y resultados de aprendizaje”

Los estudiantes cuentan con la oportunidad de recuperar los saberes pendientes para alcanzar requisitos de aprobación a lo largo de todo el semestre.

Actividades prácticas y de laboratorio

Todas las actividades de laboratorio están programadas para ser implementadas la semana siguiente al tratamiento del tema en clase.

La cátedra ofrece acompañamiento en el desarrollo del laboratorio a los grupos de estudiantes durante las clases y fuera de ellas, de 19.30 a 20.30 hs.

La cátedra cuenta con un repositorio de laboratorios desarrollados de todos los temas propuestos en el programa, videos de tratamiento de laboratorio para las actividades y de las clases teórico prácticas. El material se actualiza cada semestre.

Primer trayecto: Machine Learning y Data Science (Unidades: 1, 2 y 3 - Actividades: 4), las actividades desarrolladas deben poner en evidencia el desarrollo de competencias necesarias para:

a. Generar un plan para el tratamiento de los datos basado en la comprensión del problema,
b. la elección de los algoritmos adecuados en una plataforma determinada,
c. elaborar un plan basado en el conocimiento de las prestaciones de esta plataforma,
d. experimentar, basado en el uso adecuado de hiperparámetros, funciones, resultados, técnicas de regularización, etc.,
e. evaluar e interpretar los resultados, basado en las métricas disponibles de validación para los algoritmos.
f. Juzgar, recomendar y valorar los aspectos éticos y regulatorios relativos a los datos y los algoritmos, con una perspectiva crítica, para todo el ciclo de vida de la IA, y todos los actores involucrados

Están programados para ser desarrollados durante seis semanas, a partir de la primera clase.

Segundo trayecto: Sistema Conexionistas (Unidades 1, 4, 5 y 6 – 4 Actividades): las actividades implican aplicaciones en frameworks que sugiere la cátedra para el trabajo colaborativo. Los estudiantes deben poner en evidencia el desarrollo de competencias necesarias para:

a. La construcción del diseño de procesos asociados para encontrar una solución tecnológicamente viable y útil para las restricciones del problema y los datos disponibles,
b. implementar modelos tales como: MLP- CNN- RNN – AUTOENCODERS – TRANSFER LEARNING- TRANSFORMERS,
c. la elección del modelo, basado en la comprensión de los conceptos inherentes a cada modelo abordado y,
d. explicar y justificar su decisión, basada en el conocimiento de modelos de vanguardia en plataformas open source y sus aplicaciones más habituales, actualmente: YOLO, RES-NET, GPT2, BERT, etc.
e. Juzgar, recomendar y valorar los aspectos éticos y regulatorios relativos a los datos y los algoritmos, con una perspectiva crítica, para todo el ciclo de vida de la IA, y todos los actores involucrados

Están programados para ser desarrollados durante ocho semanas.

Desagregado de competencias y resultados de aprendizaje

Competencia	Actividad	Resultado. Aprendizaje 1	Resultado. Aprendizaje 2	Resultado. Aprendizaje 3	Resultado. Aprendizaje 4	Mín.
●**CG2:** Concebir, diseñar y desarrollar proyectos de ingeniería (sistemas, componentes, productos o procesos)	Trayecto 1 – Unidad 3	No comprende los procesos asociados a proyectos de descubrimiento o de conocimiento basado en datos	Comprende el sistema en general pero no logra organizar la estructura del proyecto. Manipula modelos y herramientas	Organiza el proyecto, alcanzando a realizar propuestas de sin comprender la oportunidad de mejora continua	Logra gestionar adecuadamente todo el proeycto desde la comprensión del problema hasta la incorporación de actualización y mejora	2
●**CG4**: Utilizar de manera efectiva las técnicas y herramientas de aplicación en ingeniería.	Trayecto 1 - Unidad 3 y Trayecto 2 - Unidad 5	No comprende los recursos y herramientas para datos y algoritmos	Conoce las herramientas de datos y modelos pero no alcanza a organizar las técnicas	Maneja las herramientas para datos y modelos pero no incorpora métricas	Utiliza de manera responsable las herram. de datos, algoritmos y su evaluación	3
●**CG5**: Contribuir a la generación de desarrollos tecnológicos y/o innovaciones tecnológicas.	Trayecto 2 -Unidades 4, 5 y 6	No alcanza a identificar los componentes centrales de los sistemas conexionistas para proponer alternativa novedosa	Comprende los componentes centrales de los sistemas conexionistas pero no alcanza a ejecutarlos en una propuesta	Ejecuta con alguna dificultad, basado en una propuesta alternativa pero no alcanza a validar ni a interpretar sus resultados	Propone con idoneidad alternativas novedosas, las ejecuta y alcanza a evaluar sus resultados distinguiendo con claridad sus restricciones	3
●**CG8:** Actuar con ética, responsabilidad profesional y compromiso social, considerando el impacto económico, social y ambiental de su actividad en el contexto local y global.	Trayecto 1 y Trayecto 2	No indagó lo suficiente acerca de ética y regulaciones como para comprender la importancia de los datos, los algoritmos, su impacto, sus oportunidades y sus amenazas para una IA fiable hacia el bien común	Si bien indagó en el material básico suministrado por la cátedra, no alcanza a visualizar cómo estos aspectos condicionan y determinan los desarrollos tecnológicos basados en IA	Visualiza con alguna claridad las prácticas necesarias de los ingenieros para desarrollar tecnologías basadas en IA y ser un actor responsable en su ciclo de vida	Comprende la necesidad de acciones de regulación en la materia, las de sensibilización, el riesgo que implica en los ecosistemas y en el medio ambiente; juzga con claridad los aspectos inherentes a datos y algoritmos	3

●**CG9**: Aprender en forma continua y autónoma.	Trayecto 1 y Trayecto 2	No encuentra un rol en su equipo de trabajo, no ha comprendido que los desarrollos se realizan en equipo, razón por la cual no alcanza a aprender de manera continua	Se acomoda bien en un rol dentro de su equipo, interpreta adecuadamente sus errores, aunque no alcanza a tener un rol colaborativo para el trabajo multidisciplinario	Alcanza a trabajar de manera colaborativa, interpreta el diálogo entre pares, propone herramientas aunque no las ejecuta con idoneidad	Aprende en forma continua y autónoma, materializado en sus explicaciones, sus decisiones, su comunicación con sus pares y la forma en que interpreta las limitaciones de las actividades de laboratorio	3
CE1.3 Conocer, desarrollar nuevos e implementar algoritmos y estructuras de datos.	Trayecto 1 Unidades 2 y 3 y Trayecto 2 Unidades 5 y 6	No comprende el adecuado uso de plataformas y frameworks que la cátedra sugiere para trabajar en las actividades de laboratorio	Si bien maneja el ambiente de desarrollo que sugiere la cátedra, no ha realizado una investigación de las librerías en base a la documentación disponible que le permita armar proyectos con alguna facilidad	Manipula las librerías, busca adecuadamente los recursos y los organiza, sólo lo logra en base a un ensayo de prueba error, sin adquirir una actitud crítica que dé cuenta del manejo de la abstracción que estos procesos representan	Si bien no ha adquirido la habilidad requerida para manejarse con solvencia en la plataforma que usa, es capaza de desarrollar y proponer alternativas de solución, basadas en la conceptualización que ha alcanzado de los modelos de IA que manipula	3

Bibliografía

1. About RISC-V – RISC-V International. https://riscv.org/about/ (accedido 9 de julio de 2023).
2. AuthorCorporate: UNESCO, Recomendación sobre la ética de la inteligencia artificial, UNESCO Biblioteca Digital, 2022. https://unesdoc.unesco.org/ark:/48223/pf0000381137_spa (accedido 23 de junio de 2023).
3. AI for Good, AI for Good. https://aiforgood.itu.int/ (accedido 19 de julio de 2023).
4. BRITOS, P., HOSSIAN, GARCIA MARTINEZ, R. y SIERRA. Minería de datos Basada en Sistemas Inteligentes. Nueva Librería. 2005
5. COMUNICACIÓN DE LA COMISIÓN AL PARLAMENTO EUROPEO, AL CONSEJO, AL COMITÉ ECONÓMICO Y SOCIAL EUROPEO Y AL COMITÉ DE LAS REGIONES Fomentar un planteamiento europeo en materia de inteligencia artificial. 2021. Accedido: 24 de agosto de 2023. [En línea]. Disponible en: https://eur-lex.europa.eu/legal-content/ES/TXT/?uri=CELEX:52021DC0205
6. GARCIA MARTINEZ, R, PASQUINI, D. y SERVENTE, M. Sistemas Inteligentes. Nueva Librería. 2013

7. GOODFELLOW, I., BENGIO Y. AND COURVEILLE, A., Deep learning MIT Press, 2016
8. HERTZ, J., KROGH, A., PALMER R.. Introduction to the Theory of Nural Computation. Addison-Wesley. 1991
9. ISO/IEC JTC 1/SC 42 - Artificial intelligence, ISO, 20 de enero de 2022. https://www.iso.org/committee/6794475.html (accedido 18 de julio de 2023).
10. MARTIN DEL BRIO, B., SANZ, M. Redes Neuronales y Sistemas Difusos, (3ra ed). Alfaomega. Ra-Ma. 2006
11. NIELSEN, M., Neural Networks and Deep Learning, Determination Press, 2016
12. N. Dey, «Cerebras-GPT: A Family of Open, Compute-efficient, Large Language Models», Cerebras, 28 de marzo de 2023. https://www.cerebras.net/blog/cerebras-gpt-a-family-of-open-compute-efficient-large-language-models/ (accedido 20 de mayo de 2023).
13. SELECCIÓN DE ESCRITOS SOBRE INTELIGENCIA ARTIFICIAL: INTELIGENCIA ARTIFICIAL: ALGUNOS ASPECTOS DE SU IMPACTO, Centro de Estudios en Tecnologías Inteligentes (CETI). en SERIE CONTRIBUCIONES COMPILADAS, no. 6. Ciudad Autónoma de Buenos Aires: Academia Nacional de Ciencias de Buenos Aires, 2022. [En línea]. Disponible en: https://www.ciencias.org.ar/user/CETI/Compilado%20CETI%20final.pdf
14. Open Sourcing BERT: State-of-the-Art Pre-training for Natural Language Processing, 2 de noviembre de 2018. https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html (accedido 13 de julio de 2023).
15. Vaswani et al., «Attention is All you Need», en Advances in Neural Information Processing Systems, Curran Associates, Inc., 2017. Accedido: 18 de julio de 2023. [En línea]. Disponible en: https://proceedings.neurips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

FCEFyN
FACULTAD DE CIENCIAS EXACTAS, FÍSICAS y NATURALES

UNC Universidad Nacional de Córdoba

Asignatura: **Práctica Profesional Integradora**

Código:	RTF	9
Semestre: 10	Carga Horaria	300
Bloque: Tecnologías Aplicadas	Horas de Práctica	300
Departamento: Computación		

Correlativas:
- Programación Avanzada

Contenido Sintético:
- Trabajo en ambiente profesional.
- Trabajo en equipo.
- Integración de saberes.
- Aprendizaje autónomo.

Competencias Genéricas:: (Contribución: A = Alta; M = Media; B = Baja)

- CG1: Identificar, formular y resolver problemas de ingeniería. (A)
- CG2: Concebir, diseñar y desarrollar proyectos de ingeniería (sistemas, componentes, productos o procesos). (A)
- CG3: Gestionar -planificar, ejecutar y controlar- proyectos de ingeniería (sistemas, componentes, productos o procesos). (A)
- CG4: Utilizar de manera efectiva las técnicas y herramientas de aplicación en ingeniería. (A)
- CG5: Contribuir a la generación de desarrollos tecnológicos y/o innovaciones tecnológicas. (M)
- CG6: Desempeñarse de manera efectiva en equipos de trabajo. (A)
- CG7: Comunicarse con efectividad. (A)
- CG8: Actuar con ética, responsabilidad profesional y compromiso social,
- considerando el impacto económico, social y ambiental de su actividad en el contexto local y global. (A)
- CG9: Aprender en forma continua y autónoma. (A)
- CG10: Actuar con espíritu emprendedor. (M)

Aprobado por HCD: NNNN-HCD-AAAA RES: Fecha: DD/MM/AAAA

Competencias Específicas: N/A

Presentación

La culminación de una carrera de ingeniería con una práctica profesional integradora conlleva una invaluable oportunidad de crecimiento y desarrollo para los estudiantes. En primer lugar, esta experiencia permite a los estudiantes aplicar de manera práctica los conocimientos y competencias técnicas que han adquirido a lo largo de su formación académica. Esto no solo promueve la comprensión de los conceptos teóricos, sino que también les brinda la oportunidad de enfrentarse a desafíos del mundo real, en contextos de diseño equivalentes a su futuro profesional, lo que incluso puede resultar en soluciones innovadoras y eficaces.

Además de su crecimiento técnico, los estudiantes también desarrollan competencias sociales cruciales, como la comunicación efectiva y las habilidades de interacción. A medida que colaboran con profesionales experimentados y equipos multidisciplinarios, aprenden a expresar sus ideas de manera clara, a trabajar de manera colaborativa y a gestionar y resolver problemas de forma eficiente. Estas competencias son esenciales en cualquier entorno laboral y enriquecen la capacidad de los estudiantes para contribuir de manera significativa en su futura carrera.

Asimismo, las prácticas profesionales facilitan la construcción de redes profesionales valiosas, lo que puede abrir puertas a futuras oportunidades laborales. Los estudiantes también tienen la oportunidad de contribuir a proyectos importantes y de impacto real en la empresa o la comunidad, lo que les brinda un sentido de logro y la confianza para abordar desafíos futuros.

En última instancia, las prácticas profesionales no solo benefician a los estudiantes al proporcionarles una transición más fluida al mundo laboral, sino que también enriquecen a la industria al infundir nuevas perspectivas y conocimientos a las empresas. Esta colaboración entre la academia y la industria fortalece la base de talento y la innovación en el campo de la ingeniería, lo que a su vez contribuye al avance continuo de la tecnología y la sociedad en su conjunto.

Contenidos

La siguiente descripción de contenidos se enfoca en los aspectos esenciales de la preparación, ejecución y evaluación de la práctica profesional, así como en la escritura y presentación del informe final. Proporciona una estructura concisa para abordar los aspectos críticos de la asignatura.

Capítulo 1: Introducción a la Práctica Profesional Integradora
Objetivos y Beneficios de la Práctica Profesional Integradora. Requisitos y Expectativas de la Práctica. Selección de la Empresa, Laboratorio o lugar de realización.

Capítulo 2: Competencias Técnicas y Habilidades Profesionales
Metodología de desarrollo. Aplicación de Conocimientos Técnicos. Desarrollo de Competencias Blandas (Comunicación, Trabajo en Equipo, Resolución de Problemas).

Capítulo 3: Realización de la Práctica Profesional

Diseño y Desarrollo. Inmersión en el Entorno Profesional. Contribución a Proyectos y Tareas Relevantes. Superación de Desafíos y Resolución de Problemas.

Capítulo 4: Escritura del Informe Final
Estructura y Contenido del Informe. Recopilación y Análisis de Datos Redacción Técnica y Estilo Académico.

Capítulo 5: Presentación Final y Evaluación
Preparación y Presentación del Informe Final. Evaluación de la Práctica Profesional Integradora. Reflexión sobre el Aprendizaje y el Cierre de la Práctica

Metodología de enseñanza

Dentro del cuerpo de conocimiento dedicado a la pedagogía, se establecen fundamentales criterios didácticos y estrategias de enseñanza. En el contexto de la presente asignatura y en relación a estos criterios, la premisa fundamental es la implementación de una metodología educativa centrada en el estudiante. De esta manera y siguiendo el modelo de enseñanza por explicación se realizan encuentros con exposiciones dialogadas en donde se abordan las temáticas relevantes al proceso de desarrollo de la práctica profesional integradora.
En esta asignatura el aprendizaje basado en proyectos desempeña un rol esencial, debido a que los estudiantes a lo largo de la asignatura deben aplicar e integrar las competencias técnicas específicas adquiridas a lo largo de la carrera de manera recíproca a lo que sucederá en su vida profesional. A su vez enfrentando decisiones de diseño, trade-offs o relaciones de compromiso donde es necesario poder aplicar estas competencias aprendidas pero a su vez aprender o desarrollar las competencias políticas, sociales y actitudinales para argumentar con claridad las decisiones tomadas frente al grupo de pares y frente al tribunal evaluador en la instancia correspondiente.
Se aplican tecnologías modernas, con aplicación de casos y contrastación de ejemplos contra Large Language Models LLM AI. Se actualiza constantemente material audio visual, haciendo uso de las herramientas y el conocimiento obtenido en cursos de mejora continua. A través del campus virtual LEV (Moodle), se establece un canal de contacto bidireccional con material didáctico que en conjunto con las plataformas de mensajería brindan al estudiante la posibilidad de estar en contacto directo con el docente.

Evaluación

Dentro del marco y el contexto del cierre de la carrera de grado el proceso de evaluación de esta asignatura debe cumplir las siguientes etapas:

- Al iniciar el desarrollo de la Práctica, es necesario presentar a la Cátedra un formulario el cual se titula Solicitud de Admisión de Tema (SAT). La SAT tiene como objetivo resumir las características principales del proyecto, a saber:
    - Integrantes del proyecto: director/codirector/estudiantes
    - Representante de la Empresa/Organización/Institución/Laboratorio
    - Area temática abordada
    - Resumen - Objetivos - Hipótesis

- Cronograma detallado de Desarrollo
- Antecedentes académicos
- Lugar de realización (Empresa, laboratorio, etc)
- Material Necesario - Presupuesto
- Bibliografía

Una vez presentada la solicitud de admisión de tema, se asigna un tribunal pertinente formado por 3 docentes de esta casa de estudios para su análisis y evaluación.

- El resultado de la evaluación de la SAT es uno de tres:
  - Se aprueba sin modificaciones
  - Se aprueba con correcciones parciales
  - No se aprueba

- Una vez aprobada la SAT, comienza la etapa de desarrollo de la Práctica. La cual a lo largo de su desarrollo presenta instancias de encuentro con el tribunal para exponer avances y consensuar los siguientes objetivos. El desarrollo implica la participación activa de los estudiantes en el lugar de realización de manera presencial. Estas horas deberán acreditarse con certificado de aval.
- Una vez finalizado el desarrollo, se debe avanzar en la escritura del informe final de la práctica.
- Una vez finalizado el informe y con el aval del Director y/o co-Director de la práctica se eleva a corrección el informe escrito.
- El tribunal evaluador dispone de 15 días hábiles para la corrección, para la corrección se utilizarán rúbricas basadas en los resultados de aprendizaje.
- Una vez aprobado el informe por el tribunal evaluador, el trabajo debe exponerse oral y públicamente para su aprobación final, la nota final será mediante rúbrica.

Condiciones de aprobación

Condiciones:

- Aprobación de SAT.
- Aprobación de informe escrito.
- Aprobación de defensa oral y pública.

Actividades prácticas y de laboratorio

Como actividades prácticas de la materia, durante el desarrollo de la PPI, los estudiantes deben acreditar y presentar certificado de la realización de 150 hs presenciales en el lugar destinado a la realización del proyecto.

Resultados de aprendizaje

Se proponen los siguientes resultados de aprendizaje, no obstante la naturaleza del proyecto o actividad a realizar implica la aplicabilidad o no de la totalidad los mismos:

1. Identifica claramente el problema propuesto para su trabajo. (CG1,CG2)
2. Plantea soluciones concretas para brindar soluciones al problema propuesto. (CG1,CG2)
3. Analiza la factibilidad de las propuestas realizadas. (CG1,CG2)
4. Plantea un modelo de gestión adecuado para la implementación del proyecto propuesto. (CG3)
5. Emplea adecuadamente las técnicas propias del sistema de gestión aplicado. (CG3)
6. Selecciona adecuadamente las herramientas o tecnologías a aplicar en el desarrollo propuesto. (CG4)
7. Emplea adecuadamente las herramientas o tecnologías seleccionadas. (CG4)
8. Propone innovaciones en las soluciones planteadas. (CG5, CG10)
9. Trabaja colaborativamente con su equipo. (CG6)
10. Elabora un informe adecuado, respetando las instrucciones provistas. (CG7)
11. Se expresa oralmente de forma clara, empleando lenguaje propio de la disciplina. (CG7)
12. Documenta adecuadamente el proyecto de acuerdo a los criterios propios de la disciplina.(CG7)
13. Identifica claramente los aspectos de su proyecto que tienen implicancias éticas.(CG8)
14. Identifica claramente los aspectos de su proyecto que pudieran tener impacto social y ambiental.(CG8)
15. Realiza una valoración del impacto social o ambiental de la implementación de su proyecto.(CG8)
16. Recupera e integra saberes adquiridos a lo largo de la carrera. (CG9)
17. Identifica fuentes fiables de información. (CG9)
18. Identifica saberes conceptuales y procedimentales necesarios para la realización del proyecto. (CG9)
19. Incluye los aprendizajes necesarios dentro de la gestión del proyecto. (CG9, CG2)
20. Comunica los aspectos relevantes de su proyecto de forma clara en lenguaje no técnico.(CG7, CG10)
21. Establece estimativamente el costo de la implementación de la solución propuesta. (CG10)
22. Valora la viabilidad económica del proyecto. (CG10)
23. Incorpora dentro de las especificaciones del proyecto aspectos orientados a la comercialización del mismo (CG1, CG2, CG10)

Bibliografía

- "Instructional Design" de Patricia L. Smith y Tillman J. Ragan - 2004
- "Tools for teaching" de Barbara Gross Davis - Año: 2009
- "Developing Professional Skills by Integrating Internships into Academic Programs" de Wendi L. Kappers y Julia R. Pomerenk - 2014
- "Assessing Soft Skills Development in Engineering Students during Internships" de Barbara M. Olds y Cheryl Q. Li - 2015

FCEFyN
FACULTAD DE CIENCIAS EXACTAS, FÍSICAS y NATURALES

UNC Universidad Nacional de Córdoba

Asignatura: **Procesamiento de Señales**

Código:	RTF	10
Semestre: 6	Carga Horaria	96
Bloque: Tecnologías Básicas	Horas de Práctica	38
Departamento: Computación		

Correlativas:
- Señales y Sistemas

Contenido Sintético:
- Visión de conjunto e historia.
- Procesos estocásticos. Estimación. Predicción. Filtrado FIR.
- Densidad espectral de potencia. Filtro blanqueador. Muestreo de procesos aleatorios limitados en banda.
- Estadística suficiente. Diseño de etapas de procesamiento de señales de tiempo continuo. Filtro apareado.
- Test de hipótesis. Diseño de etapas de procesamiento de señales de tiempo discreto. Detección de señales.
- Filtro de Wiener IIR y FIR. Filtro adaptativo. Interferencia intersímbolo. Ecualización.

Competencias Genéricas:
- CG1: Identificar, formular y resolver problemas de ingeniería. Alta.
- CG4: Utilizar de manera efectiva las técnicas y herramientas de aplicación en ingeniería. Alta.
- CG5: Contribuir a la generación de desarrollos tecnológicos y/o innovaciones tecnológicas. Alta.
- CG9: Aprender en forma continua y autónoma. Alta.

Aprobado por HCD: NNNN-HCD-AAAA RES: Fecha: DD/MM/AAAA

Competencias Específicas:

- CE7.1: Proyectar, desarrollar, dirigir, controlar, construir, operar y mantener sistemas de procesamiento de señales
- CE7.1.1 Interpretar y emplear las técnicas, tecnologías, principios físicos y matemáticos y herramientas necesarias para planteo, interpretación, modelización y solución de problemas de detección, estimación y comunicación de señales.
- CE7.1.2 Conocer los principios básicos de los procesamientos de señales y de comunicación digitales
- CE7.1.3 Diseñar y analizar sistemas y componentes para la detección de señales aleatorias
- CE7.1.4 Diseñar y analizar sistemas y componentes para la estimación espectral y de parámetros
- CE7.1.5 Diseñar, implementar y analizar sistemas de procesamiento de señales y de comunicaciones

Presentación

En el ámbito de la ingeniería y la ciencia abundan multitud de señales, que abarcan desde imágenes de sondas espaciales hasta impulsos eléctricos producidos por el corazón y el cerebro, ecos en sistemas de radar y sonar, vibraciones sísmicas y un sinfín de otras aplicaciones. El procesamiento de señales digitales (DSP) es la disciplina que aprovecha las computadoras para dar sentido a dichos datos. Esto abarca una amplia gama de objetivos, incluido el filtrado, el reconocimiento de voz, la mejora de imágenes, la compresión de datos, la utilización de redes neuronales y mucho más. El procesamiento de señales digitales se erige como una de las tecnologías más potentes e influyentes preparadas para dar forma al panorama de la ciencia y la ingeniería en el siglo veintiuno.

Hemos sido testigos de transformaciones innovadoras en un amplio espectro de aplicaciones, incluidas las comunicaciones, las imágenes médicas, el radar, el sonar, la reproducción de música de alta fidelidad, por mencionar sólo algunas. En cada uno de estos dominios, el procesamiento de señales digitales ha desempeñado un papel fundamental, con su propio conjunto de algoritmos, marcos matemáticos y metodologías especializadas. La enorme amplitud e influencia del DSP hacen que sea una tarea insuperable para cualquier individuo convertirse en un maestro de todas las tecnologías DSP que han surgido. La educación en DSP abarca dos esfuerzos esenciales: en primer lugar, comprender los principios generales que son aplicables a todo el campo y, en segundo lugar, profundizar en las técnicas especializadas pertinentes al área de interés específica de cada uno.

En este curso se presentan e interpretan funciones de correlación y densidades espectrales de potencia para describir y procesar señales aleatorias. Los contextos de aplicación incluyen modulación de amplitud de pulso, filtros lineales óptimos para una estimación mínima del error cuadrático medio y filtrado adaptado para la detección de señales. Se enfatizan los enfoques de inferencia basados en modelos, en particular para la estimación de señales y la detección de señales. El curso explora ideas, métodos y herramientas comunes a numerosos campos que involucran señales, sistemas e inferencia: procesamiento de señales, comunicación, análisis de series temporales, biomedicina y muchos otros.

La educación en DSP implica dos tareas: aprender conceptos generales que se aplican al campo en su conjunto y aprender técnicas especializadas para su área de interés particular. Este capítulo inicia nuestro viaje hacia el mundo del procesamiento de señales digitales describiendo el espectacular efecto que DSP ha tenido en varios campos diversos. La revolución ha comenzado.

Contenidos

Modelos probabilísticos

El modelo de probabilidad básico. Probabilidad condicional, regla de Bayes e independencia. Variables aleatorias. Distribución acumulativa, densidad de probabilidad y función de masa de probabilidad para variables aleatorias. Variables aleatorias distribuidas conjuntamente.

Expectativas, momentos y variaciones. Correlación y covarianza para variables aleatorias bivariadas. Una imagen en el espacio vectorial para las propiedades de correlación de variables aleatorias.

Estimación con error cuadrático medio mínimo.
Estimación de una variable aleatoria continua. De estimaciones a estimador. Ortogonalidad. Estimación de mínimo Error cuadrático medio lineal.

Procesos aleatorios.
Definición y ejemplos de un proceso aleatorio. Estacionariedad en sentido estricto. Estacionariedad de sentido amplio. Algunas propiedades de las funciones de correlación y covarianza de WSS. Resumen de definiciones y notación. Otros ejemplos. Ergodicidad. Estimación lineal de procesos aleatorios. Predicción lineal. Filtrado FIR lineal. El efecto de los sistemas LTI en los procesos WSS.

Densidad espectral de potencia
Potencia instantánea esperada y densidad espectral de potencia. Teorema de Einstein-Wiener-Khinchin sobre la potencia esperada promediada en el tiempo. Identificación del sistema utilizando procesos aleatorios como entrada. Invocando la ergodicidad. Filtros de modelado y filtros blanqueadores. Muestreo de procesos aleatorios de banda limitada

Filtro Wiener
Filtro Wiener de tiempo discreto no causal. Filtro Wiener de tiempo continuo no causal. Propiedad de ortogonalidad. Filtrado Wiener causal. Tratamiento con media distinta de cero.

Modulación de amplitud de pulso (PAM), modulación de amplitud en cuadratura (QAM)
Modulación de amplitud de pulso. La señal transmitida. La señal recibida. Caracterizaciones en el dominio de la frecuencia. Interferencia entre símbolos en el receptor. Pulsos de Nyquist. Transmisión portadora. FSK. PSK. QAM.

Evaluación de la hipótesis
Modulación de amplitud de pulso binario en ruido. Prueba de hipótesis binaria. Decidir con mínima probabilidad de error (Pe): la regla MAP. Comprensión de Pe: falsa alarma, fallo y detección. Test de verosimilitud. Otros escenarios. Características operativas del receptor y detección de Neyman-Pearson. Decisiones de Riesgo Mínimo. Pruebas de hipótesis en comunicación digital codificada. Decisión óptima a priori. El modelo de transmisión. Decisión óptima a posteriori.

Detección de señal.
Detección de señales como prueba de hipótesis. Detección óptima de ruido blanco gaussiano. Filtrado adaptado. Clasificación de señales. Una estructura general de detectores. Detección de pulsos en ruido blanco. Maximizando la SNR. Filtros adaptado de tiempo continuo. Detección de pulsos en ruido coloreado.

Metodología de enseñanza

Enfoque Activo y Participativo: Se fomenta la participación activa de los estudiantes en el proceso de aprendizaje en las clases teórico-prácticas. Se promueve la resolución de problemas, el trabajo en equipo y la discusión de casos prácticos. Los estudiantes son alentados a plantear preguntas, desafiar ideas preconcebidas y explorar nuevas perspectivas, lo que les permite desarrollar habilidades críticas para el diseño de sistemas de procesamiento de señales.
Aprendizaje Basado en Proyectos: Los proyectos prácticos desempeñan un papel central. Los estudiantes tienen la oportunidad de aplicar los conceptos y técnicas aprendidos en situaciones del mundo real. A través del diseño de algoritmos de procesamiento de señales, la optimización de parámetros y la resolución de problemas específicos, los estudiantes adquieren habilidades prácticas esenciales para su futura carrera profesional.
Enfoque Interdisciplinario: el procesamiento de señales es una herramienta fundamental en una amplia variedad de campos profesionales. Por lo tanto, fomentamos un enfoque interdisciplinario que permite a los estudiantes explorar cómo se aplican las técnicas de procesamiento de señales en diferentes sectores, como la salud, la tecnología de la información y comunicación, y la investigación científica. Esto les brinda una visión más amplia de las oportunidades profesionales disponibles.
Evaluación Continua y Retroalimentación: La evaluación continua es un pilar pedagógico. Los estudiantes reciben retroalimentación constante a lo largo del curso, lo que les permite identificar áreas de mejora y ajustar su enfoque de aprendizaje. Además, se fomenta la autorreflexión y la autoevaluación como herramientas para el desarrollo profesional.
Recursos Tecnológicos Avanzados: Se utilizan herramientas y recursos tecnológicos de vanguardia para enriquecer la experiencia de aprendizaje. Esto incluye el acceso a repositorios en la nube de algoritmos de procesamiento de señales, laboratorio de radio definida por software con acceso remoto y software de simulación que replica situaciones del mundo real.
Colaboración y Comunicación: Se promueve la comunicación efectiva y la colaboración entre estudiantes, imitando las dinámicas profesionales. El trabajo en equipo y la capacidad para explicar y defender soluciones son competencias esenciales que se desarrollan a lo largo del curso.
Cuestiones Eticas: El manejo y la interpretación de los resultados implica una formación ética y profesional que asegure su correcta administración teniendo en cuenta la responsabilidad ante la sociedad que ello representa.

Evaluación

Evaluaciones Continuas: A lo largo de la asignatura, se realizan evaluaciones formativas automatizadas que permiten a los estudiantes recibir retroalimentación regular sobre su progreso. Esto puede incluir ejercicios prácticos y cuestionarios automatizados. La retroalimentación se utiliza para ayudar a los estudiantes a ajustar su enfoque de aprendizaje y mejorar sus habilidades. Como parte de los ejercicios se presentan casos de uso en los que los estudiantes individualmente analizan y proponen soluciones basadas en procesamiento

de señales. Estos estudios de caso requieren que los estudiantes apliquen conceptos teóricos a situaciones concretas y presenten soluciones argumentadas.
Exámenes teóricos-prácticos: Los exámenes teóricos-prácticos evalúan el conocimiento conceptual de los estudiantes en áreas como detección, filtrado y estimación de señales, y el diseño de sistemas de procesamiento de señales. Se realizan de manera automatizada y comprende dos instancias de evaluación.
Proyectos de Laboratorio de Procesamiento de Señales: Los estudiantes trabajan en equipos para diseñar, implementar y analizar sistemas de procesamiento de señales del mundo real. Los proyectos incluyen el diseño de algoritmos basados en transformaciones matemáticas de las señales, la simulación del escenario de trabajo, la implementación en hardware y su posterior análisis de desempeño y complejidad computacional.

Los resultados de aprendizaje serán evaluados mediante rúbrica basada en los resultados de aprendizajes propuestos.

Condiciones de aprobación

Para regularizar y promoción se exige un 80% de asistencia y participación en clase.
Para regularizar, además, se exige un 60% de la calificación en las evaluaciones continuas y en los exámenes teóricos prácticos.
Para promoción, se exige además un 70% de calificación en el proyecto de laboratorio de procesamiento de señales y haber alcanzado un nivel mínimo de desarrollo en los resultados de aprendizaje propuestos.

Actividades prácticas y de laboratorio

Estas actividades forman parte de los Recursos Tecnológicos Avanzados que se emplean durante el cursado de la asignatura, y que permiten trabajar con señales adquiridas del mundo real.

Desagregado de competencias y resultados de aprendizaje

Al final del curso se espera que un estudiante alcance los siguientes habilidades:
● Interpreta correctamente el dominio de un problema.
● Posee las habilidades comunicacionales suficientes para realizar las preguntas necesarias para desarrollar un diseño completo ajustado a las necesidades del dominio presentes y futuras.
● Identifica los aspectos de la encomienda profesional que por su naturaleza tienen connotaciones éticas
● Cumple en tiempo con los compromisos asumidos con su equipo de trabajo.
● Respeta las pautas de trabajo establecidas en clase para las actividades de equipo.
● Diseña un sistema de procesamiento de señales que permita estimar o detectar la información.
● Implementa un sistema de procesamiento de señal.
● Elabora y ejecuta procesamiento digital de señales en lenguajes de programación de alto nivel y evalúa su desempeño y complejidad computacional

● Trabaja en equipo asumiendo los distintos roles dentro de un grupo de trabajo
● Detecta y comunica errores y oportunidades de mejoras en diseños de propios y de terceros.
● Conoce los fundamentos de un sistema de procesamiento de señales.
● Asegura un procesamiento de señales según buenas prácticas de seguridad e identifica los compromisos éticos que surgen de implementar seguridad en los sistemas de procesamiento de señales.
● identifica y evalúa de manera crítica diversos recursos de aprendizaje, como libros, artículos, tutoriales en línea y otros, para seleccionar aquellos que mejor se adapten a sus necesidades de aprendizaje.

Bibliografía

“Signals, Systems and Inference” de Alan Oppenheim y George Verghese. Prentice Hall, 2015. ISBN: 9780133943283
“The Scientist and Engineer's Guide to Digital Signal Processing” de Steven W. Smith. California Technical Pub. http://www.dspguide.com/
“Principles of Digital Communication: A Top-Down Approach” de Bixio Rimoldi”. Cambridge University Press.

FCEFyN
FACULTAD DE CIENCIAS EXACTAS, FÍSICAS y NATURALES

UNC Universidad Nacional de Córdoba

Asignatura: **Programación Avanzada**

Código:	RTF	7
Semestre: Tercero	Carga Horaria	96
Bloque: Tecnologías Básicas	Horas de Práctica	48
Departamento: Computación		

Correlativas:
- Algoritmos y Estructuras de Datos

Contenido Sintético:
- Visión de conjunto e Historia
- Diseño y Programación Orientada a Objetos
- Prueba, verificación y validación de software
- Uso de interfaces de programación de aplicaciones
- Patrones de diseño de software
- Diseño de aplicaciones. Visualización de datos.
- Experiencia de Usuario

Competencias Genéricas:
- CG1: Identificar, formular y resolver problemas de ingeniería.(B)
- CG2: Concebir, diseñar y desarrollar proyectos de ingeniería (sistemas, componentes, productos o procesos).(B)
- CG9: Aprender en forma continua y autónoma.(B)

Aprobado por HCD: NNNN-HCD-AAAA RES: Fecha: DD/MM/AAAA

Competencias Específicas:

CE1.1 Analizar, especificar, diseñar, proyectar y desarrollar programas de computadoras en lenguajes de alto y bajo nivel.
CE1.2 Analizar, especificar, diseñar y proyectar arquitectura de sistemas informáticos.
CE1.3 Conocer, desarrollar nuevos e implementar algoritmos y estructuras de datos.

CE4.3 Analizar, diseñar, programar, implementar, probar, depurar y evaluar hardware y software para sistemas de computación de propósitos específicos.
CE4.10 Analizar, interpretar, modelar, diseñar interfaces humano-máquina en sistemas de software y software-hardware optimizando la experiencia de usuario.

Presentación

La Programación Orientada a Objetos se inicia con el sistema de simulación denominado Simula en los años 60 y posteriormente recibe un gran impulso teórico de sus fundamentos con el grupo de investigación del Xerox Parc Place y su desarrollo del lenguaje Smalltalk en los 70. La pureza del paradigma implementado en Smalltalk se ha convertido en su punto débil ya que se debe abandonar completamente el paradigma procedural y también la sintaxis de los lenguajes descendientes del Algol como Pascal y particularmente la familia del C, constituida por el C mismo, C++, C# y el Java. Un caso particular es el de Eiffel, más cercano a las ideas del Pascal y Algol, pero implementa el paradigma de objetos en forma pura, lo que lo lleva ser de difícil distribución masiva, no obstante, muchas de sus características han sido implementadas actualmente en Java y C#.

Otro aspecto que ha colaborado en la difusión del paradigma orientado a objetos han sido los Sistemas Operativos con interfaces gráficas de usuario en las cuales la metáfora de los objetos y los eventos permite elaborar una interacción con el usuario en forma natural y flexible.

Simultáneamente con el crecimiento de la Programación Orientada a Objetos se da el Análisis y Diseño Orientados a Objetos hasta desembocar en el Proceso Unificado de Desarrollo y el Lenguaje de Modelado (UML) que permite actualmente disponer de una Metodología que guíe el proceso de diseño e implementación. Al igual que en los proyectos de ingeniería, se ha descubierto que existen patrones de diseño a partir de los cuales se puede construir el software sin necesidad de tener que reinventarlo y, además, se ha consolidado la construcción de software a partir de componentes reusables durante los 90 y principios de este siglo. Actualmente, el desarrollo profesional de software se somete a estos principios que denominamos Ingeniería de Software.

En cuanto a la pedagogía de la asignatura, se ha seguido el concepto de introducir los Objetos Primero y para ello el estudiante deberá acompañar el estudio teórico del diseño orientado a objetos con la práctica de la elaboración de partes de un proyecto integral que justifique plenamente por su complejidad el paradigma de análisis, diseño e implementación adoptados. También se pone énfasis en aceptar que el desarrollo industrial de software requiere metodologías de Ingeniería de Software y de herramientas de desarrollo que amplifiquen la eficiencia y la productividad. Esto no significa que se abordará en forma completa ningún lenguaje orientado a objetos en particular ya que se enfocará en los criterios de diseño.

Contenidos

Introducción a los lenguajes de programación

Historia de los lenguajes de programación y los distintos paradigmas. Comparación entre intérpretes y compiladores; fases de la traducción de lenguajes; aspectos independientes y dependientes de la máquina. El concepto de máquina virtual, jerarquía de máquinas virtuales, lenguajes intermediarios

Objetos y Clases
Definición de clases y de objetos. Llamado de métodos y sus parámetros. Tipos de datos. Instancia de objetos y estado. Interacción básica entre objetos. Objetos como parámetros de métodos. Declaración y definición de clases: Campos, constructores y métodos.

Interacción entre objetos
Abstracción y modularización de software. Diagramas de clases y de objetos. Tipos primitivos y tipos de objetos. Creación de objetos y constructores múltiples. Llamado interno y externo de métodos. La autorreferencia a un objeto.

Agrupamiento de objetos
Agrupamiento de objetos en colecciones de tamaño flexible. Elementos básicos de una biblioteca de clases. Clases genéricas. Enumeración y gestión de colecciones. Iteradores. Colecciones fijas de objetos. Información y ocultamiento de campos y métodos. Variables de clase y constantes. Documentación de bibliotecas de clases estándar. Interfaces o implementación de métodos. Lectura y redacción de documentación de clases parametrizadas. Programación defensiva. Manejo de excepciones. Reporte de errores.

Procesamiento funcional de colecciones
Un primer vistazo a las funciones lambda. El método de colecciones forEach. Streams.

Prueba, depuración y mantenimiento de software
Pruebas de unidad. Inspectores. Pruebas positivas y negativas. Automatización de las pruebas. Pruebas de regresión. Escenarios y registro de pruebas. Uso de depuradores. Uso de aserciones.

Diseño de clases
Introducción al acoplamiento y la cohesión. Duplicación y extensión de código. Uso de encapsulado. Diseño basado en responsabilidades. Refactorización. Guías de diseño.

Herencia de clases
Jerarquías de herencia de campos y métodos de clases. Derechos de acceso a la herencia. Inicialización de instancias con herencia. Superclase y tipo de dato. Subtipos y subclases. Variables polimórficas. Conversión de tipo (Casting). Tipos estáticos y dinámicos. Sobrecarga de métodos. Búsqueda dinámica de métodos. Llamado a la superclase en los métodos. Polimorfismo de métodos. Acceso protegido. Clases abstractas. Métodos abstractos. Interfaces. Interfaces como tipos. Interfaces como especificaciones.

Construcción de interfaces gráficas de usuario

Construcción de Interfaces Gráficas de Usuario. Componentes y despliegue de la interfaz. Manejo de eventos. Bibliotecas de manejo abstracto de ventanas e Interfaces Gráficas de Usuario.

Diseño de aplicaciones
Análisis y diseño. El método de verbos/sustantivos. Descubrimiento de clases. Escenarios. Colaboración entre clases. Diseño de interfaz de clases y usuarios. Documentación. Cooperación. Modelos de desarrollo de software. Patrones básicos de diseño.

Metodología de enseñanza

Cada nuevo tema se abordará mediante clases expositivas y exposición dialogada, a fin de introducir a los estudiantes en la temática.

Los estudiantes deberán afrontar diversas actividades prácticas con los saberes conceptuales y procedimentales adquiridos previamente. El docente seguirá el proceso y orientará al estudiante en la ejecución de las actividades de práctica y en los trabajos prácticos.

Todas las actividades y material utilizado se encontrará disponible en el Aula Virtual de la Asignatura.

Evaluación

Los estudiantes deberán demostrar los conocimientos y habilidades adquiridas en tres tipos de instancias de evaluación:

Evaluación conceptual (EC): Consiste en una serie de evaluaciones breves de ejercicios que permitan demostrar comprensión de temas puntuales desarrollados en las clases anteriores. La evaluación conceptual consiste en ejercicios de programación que serán evaluados objetivamente mediante pruebas de software basadas en test unitarios, verificando el cumplimiento de los requerimientos, de forma automática a través del Aula Virtual. Para aprobar el conjunto de todas las Evaluaciones Conceptuales, se deberá alcanzar un rendimiento igual o superior al 60% del puntaje total. Se prevé un recuperatorio del 30% del puntaje total que se considerará como puntaje adicional y se suma a los ya obtenidos, no pudiendo pasar la suma del 100% del puntaje. Tienen carácter acreditativo.

Trabajo Práctico (TP): Consiste en una serie de actividades de desarrollo de la solución algorítmica a problemas propuestos por la Cátedra. Las soluciones deberán estar libres de errores sintácticos que impidan su compilación y generación de código ejecutable. La solución propuesta por el estudiante se evaluará objetivamente mediante pruebas de software basadas en test unitarios, verificando el cumplimiento de los requerimientos, de forma automática a través

del Aula Virtual. Para aprobar cada trabajo práctico, se deberá alcanzar individualmente un rendimiento igual o superior al 60%. Estas actividades son de carácter extra-áulico, para las cuales los alumnos dispondrán de no menos de 72 horas para su resolución y envío. Tienen carácter acreditativo. Si bien cada trabajo puede favorecer el desarrollo de una determinada competencia en particular y es de esperar la evidencia de esto hacia la conclusión de dicha actividad, la evaluación será contínua a lo largo de todas las actividades propuestas y se hará mediante el uso de rúbricas.

Examen de Promoción (EP): en la última clase, se realizará una ejercitación consistente en el desarrollo de un programa correspondiente al enunciado de un algoritmo. La implementación deberá estar libre de errores sintácticos que impidan su ejecución por parte del compilador, como requisito para su evaluación. La solución propuesta por el estudiante se evaluará objetivamente mediante pruebas de software basadas en test unitarios, verificando el cumplimiento de los requerimientos, de forma automática a través del LEV. Para ser aprobado, se deberá alcanzar un rendimiento igual o superior al 60%. Tiene carácter acreditativo. Al final del semestre cada estudiante debe haber demostrado un nivel de desarrollo mínimo de las competencias propuestas a través de los resultados de aprendizaje propuestos.
El diseño del examen permitirá evaluar si se alcanzaron los resultados de aprendizaje abajo indicados.

Condiciones de aprobación

Condiciones de regularización
El estudiante alcanzará la condición de regular al cumplir las siguientes condiciones:

1. Asistir al 80% de las clases. La asistencia será registrada en el aula virtual mediante la presentación del 60% de las EC y del 60% de los TP.
2. Alcanzar un rendimiento global igual o superior al 60% de los puntos en las EC. La presentación a la EC confiere asistencia a clase.
3. Aprobar el 60% de los TP propuestos.
4. Alcanzar el nivel de desarrollo mínimo en los resultados de aprendizaje propuestos.

Condiciones de promoción
El estudiante alcanzará la condición de promocionado al cumplir las siguientes condiciones:

1. Alcanzar la condición de alumno regular para lo cual se deben cumplir las condiciones indicadas al respecto.
2. Aprobar la actividad EP.

Cumplidas estas condiciones, la nota final de promoción se calculará según la siguiente fórmula:

Nota Final = redondear((0,3 * EC + 0,7 * EP)/10)

Examen final
Los estudiantes regulares rendirán un examen equivalente a la actividad Examen de Promoción (EP), la cual será calificada del mismo modo que para los alumnos promocionados, considerando para la variable EC el rendimiento alcanzado durante la cursada, y para la variable EP el rendimiento alcanzado en el examen final.

Los estudiantes libres rendirán un examen que constará de dos partes:

1. Una prueba de competencias con la misma metodología y objetivos que la Evaluación Conceptual (EC) que serán evaluados automáticamente en el aula virtual. La aprobación de esta primera parte es requisito excluyente para la prosecución del examen, y se deberá obtener un rendimiento igual o superior al 60%.
2. Un examen equivalente al Examen de Promoción (EP), con la misma modalidad y objetivos que el requerido a los alumnos regulares.

La Nota Final final de examen para los casos 1 y 2 se obtendrá por la siguiente expresión:

Nota Final = redondear ((0,3 * EC + 0,7 * EP)/10)

Actividades prácticas y de laboratorio

Las clases serán de carácter teórico-práctico en laboratorio de computación, en las cuales se presentarán los temas teóricos, se dará lugar al diálogo e intercambio de ideas y se resolverán actividades específicas a la temática abordada cada clase.

Resultados de aprendizaje

- Identificar problemas en el planteo de diseño y desarrollo de software.
- Formular soluciones efectivas y aplicar métodos de resolución de problemas
- Diseñar software aplicando los principios de modular, cohesión y acoplamiento adecuados
- Resolver problemas de programación aplicando nuevas herramientas y paradigmas de programación al problema más allá de los abordados en clase
- Implementar el concepto de tipo de dato y tipo abstracto de dato e identificar las características principales de un sistema de tipos.
- Explicar los fundamentos de la orientación a objetos y ser capaz de identificar las diferencias entre la representación basada en objetos y los modelos de flujo de datos.

- Aplicar el diseño modular y los conceptos de cohesión y acoplamiento.
- Desarrollar sistemas aplicando técnicas y metodologías de desarrollo: especificación de requisitos, análisis, diseño, prueba y depuración de aplicaciones orientadas a objetos.
- Utilizar un entorno de desarrollo para aplicar los fundamentos del paradigma orientado a objetos mediante un lenguaje de programación orientado a objetos..
- Diseñar software empleando lenguajes de modelado.
- Desarrollar aplicaciones informáticas utilizando diferentes estructuras de datos, aplicando algoritmos de ordenación y búsqueda sobre ellas.
- Utilizar patrones estándar de diseño para el desarrollo de aplicaciones.
- Implementar soluciones algorítmicas a problemas y ser capaz de representarlas como un software orientado a objetos.

Bibliografía

- David J. Barnes y Michael Kölling (2017), “Programación orientada a objetos con Java usando BlueJ” (Sexta edición), Pearson https://efn.biblio.unc.edu.ar/cgi-bin/koha/opac-detail.pl?biblionumber=17531
- Robert C. Martin (2012), “Código limpio: manual de estilo para el desarrollo ágil de software”, Anaya https://efn.biblio.unc.edu.ar/cgi-bin/koha/opac-detail.pl?biblionumber=17536
- Bertrand Meyer (2009), “Touch of class : learning to program well with objects and contracts”, Springer https://efn.biblio.unc.edu.ar/cgi-bin/koha/opac-detail.pl?biblionumber=9937

FCEFyN
FACULTAD DE CIENCIAS EXACTAS, FÍSICAS y NATURALES

UNC Universidad Nacional de Córdoba

Asignatura: **Programación Concurrente y Paralela**

Código:	RTF	8
Semestre: Sexto	Carga Horaria	96
Bloque: Tecnologías Básicas	Horas de Práctica	45
Departamento: Computación		

Correlativas:
- Programación Avanzada

Contenido Sintético:
- Visión e Historia de los sistemas concurrentes.
- Elementos, soporte y herramientas relevantes de la programación concurrente.
- Paradigmas, algoritmos y patrones de programación concurrente.
- Aspectos formales y especificación de concurrencia, paralelismo, sincronización, eventos y conflictos en sistemas embebidos y reactivos.
- Gestión del tamaño y complejidad de los modelos.
- Modelado de sistemas para evaluar el rendimiento de la dinámica de sistemas concurrentes.
- Nociones de programación paralela: estructura paralela de un algoritmo, coordinación de procesos paralelos, speedup y escalabilidad, ejemplos en sistemas multihilo.
- Aplicación práctica de integración de hardware software de sistemas.

Competencias Genéricas:: (Contribución: A = Alta; M = Media; B = Baja)
- CG1: Identificar, formular y resolver problemas de ingeniería. (A)
- CG4: Utilizar de manera efectiva las técnicas y herramientas de aplicación en ingeniería. (A)

Aprobado por HCD: NNNN-HCD-AAAA RES: Fecha: DD/MM/AAAA

Competencias Específicas: (Contribuciones: A = Alto; M = Medio; B = Bajo)

- CE6.1: Conocer y poder diseñar la estructura de sistemas operativos, la administración de procesos, la gestión de memoria, la administración de archivos, la gestión de dispositivos de entrada/salida y la implementación de políticas de seguridad. (A)
- CE6.2: Analizar, conocer, interpretar y aplicar mecanismos de comunicación y sincronización. (A)
- CE6.3: Proyectar, desarrollar, dirigir, controlar, construir, operar y mantener sistemas operativos embebidos, para dispositivos móviles y de tiempo real. (M)
- CE7.2.2: Sintetizar, diseñar, desarrollar y analizar programas lenguajes de programación de bajo nivel, como C y C++. (A)
- CE7.3.1: Sintetizar, diseñar, simular, construir y analizar circuitos y sistemas de control en tiempo continuo y tiempo discreto, aplicables a cualquier área del alcance de la profesión.

Presentación

La computación concurrente y paralela es de gran relevancia en la actualidad debido a que los sistemas informáticos modernos disponen de procesadores multinúcleo, los cuales pueden ejecutar tareas heterogéneas simultáneamente. Estos sistemas multinúcleo, requieren técnicas formales de diseño y modelado para su desarrollo e implementación. Ejemplos de estos sistemas son los sistemas ciber físicos (CF), reactivos (RS), embebidos, críticos y dirigidos por eventos los cuales interactúan con variables y eventos externos heterogéneos y no determinísticos.

En este contexto la programación concurrente es fundamental en educación informática y requiere abordar comunicación, coordinación, modelos de memoria compartida, atomicidad, sincronización y espera condicional para garantizar eficacia y eficiencia en el diseño y desarrollo de este tipo de sistemas.

Programación concurrente y paralela es una asignatura que aborda la comprensión y resolución de las problemáticas informáticas inherentes a las aplicaciones con ejecución de varios subprocesos simultáneos o tareas en paralelo para aprovechar al máximo los recursos. De esta manera es posible mejorar el rendimiento, acelerar la ejecución de los programas y resolver la problemática del uso de los recursos informáticos disponibles. Esta asignatura pertenece al tercer año (sexto semestre) de la carrera Ingeniería en Computación y en esta instancia de cursado se espera que el estudiante haya adquirido competencias en programación avanzada.

Las líneas conceptuales abordadas en esta asignatura permiten inferir que un estudiante que certifique la promoción de la asignatura habrá desarrollado habilidades para: identificar situaciones donde el uso compartido de recursos tiene potencial impacto en el funcionamiento del sistema y diseñar una solución a través de la gestión, la creación y el manejo de subrutinas o hilos de ejecución (threads), implementar la sincronización de subprocesos, crear y administrar ejecutores (thread pool executors), controlar el flujo de su aplicación mediante Fork / Join framework, analizar el paralelismo y explicar los factores que limitan la aceleración alcanzable, utilizar estructuras de datos para programas concurrentes, adaptar el comportamiento predeterminado de algunas clases de concurrencia a sus necesidades y diseñar, verificar y validar casos de aplicación de concurrencia con el lenguaje Java.

Contenidos

Capítulo 1: Introducción a la programación concurrente y paralela

Introducción y visión histórica de los sistemas concurrentes. Concurrencia y paralelismo. Testing, simulación y sus limitaciones. El problema de la corrección del software y ejemplos de fallos.

Capítulo 2: Elementos, soporte y herramientas relevantes de la programación concurrente.

Conceptos y definiciones de Sistemas Críticos y reactivos. Interleaving. Problemas comunes de los programas concurrentes. Ejecución de programas concurrentes. El enfoque de modelar. Diagramas UML.

Capítulo 3: Descripción y control de procesos
Definición de un proceso y de una subrutina o thread. Descripción de un proceso. Estados de un proceso y estados de un thread. Control y gestión de threads.

Capítulo 4: Autómatas finitos y Lenguajes Regulares
Introducción e historia de los lenguajes y gramáticas formales. Teoría de autómatas. Símbolo, vocabulario, cadena, lenguaje gramática y autómata. Definición formal de gramática.

Capítulo 5. Jerarquía de las gramáticas y Máquina de Turing.
Jerarquía de las gramáticas. Expresiones regulares. Máquinas de Mealy y Moore. Máquina de Turing y equivalencias. Autómatas finitos deterministas y no deterministas.

Capítulo 6. Concurrencia: exclusión mutua y sincronización.
Principios generales del paradigma de la concurrencia. Conceptos y definiciones. Procesos concurrentes y memoria compartida. Herramientas de sincronización. Exclusión mutua. Sección crítica y condición de carrera.

Capítulo 7. Sincronización, eventos y conflictos en sistemas embebidos y reactivos.
Conceptos y definiciones. Primitivas de sincronización de hilos (threads). Semáforos. Monitores. Paso de mensajes. Ejecutores y pool de hilos Problema de los lectores/escritores. Problema de la cena de los filósofos. Problema de la barbería y otros.

Capítulo 8. El enfoque de modelar: Redes de Petri
Redes de Petri ordinarias. Matriz de incidencia. Ecuación general de estado. Propiedades de las Redes de Petri (vivacidad, interbloqueo, etc). Modelo de proceso concurrente. Invariante de plaza y de transición. Grafo de alcanzabilidad y cobertura. Herramientas para el análisis de sistemas modelados con Redes de Petri.

Capítulo 9. Redes de Petri extendidas. Evaluación de rendimiento.
Redes de Petri temporales y formalismos. Semántica de las Redes de Petri temporales. Propiedades. Grado de Sensibilizado y prioridad. Semántica de tiempo débil y fuerte. Modelado de sistemas de tiempo real mediante Redes de Petri con tiempo. Redes de Petri coloreadas y su aplicación.

Capítulo 10. Integración
Análisis y desarrollo de un sistema concurrente y paralelo, caso de aplicación. Descripción de propiedades. Diseño y determinación de los threads necesarios y máximos en ejecución para la aplicación. Análisis de rendimiento. Validación formal de operatividad. Aplicación práctica de integración de hardware software de sistemas.

Metodología de enseñanza

Dentro del cuerpo de conocimiento dedicado a la pedagogía, se establecen fundamentales criterios didácticos y estrategias de enseñanza. En el contexto de la presente asignatura y en relación a estos criterios, la premisa fundamental es la implementación de una metodología educativa centrada en el estudiante, con un enfoque decidido en el desarrollo de competencias. Esta perspectiva pedagógica se sustenta en los principios del constructivismo. De esta manera y siguiendo el modelo de enseñanza por explicación y contrastación de casos para que el estudiante pueda relacionar el conocimiento cotidiano con los conocimientos de ingeniería se alcanza el progreso en el aprendizaje. A su vez en los prácticos se aplica un enfoque activo-participativo donde se fomenta la resolución de problemas, el trabajo en equipo y el pensamiento crítico frente a problemas de diseño para alcanzar el desarrollo de habilidades y la toma de decisiones en grupo.
En esta asignatura el aprendizaje basado en proyectos desempeña un rol esencial, debido a que los estudiantes a lo largo de la asignatura deben resolver trabajos prácticos en los cuales deben aplicar los conceptos aprendidos y las técnicas de manera recíproca a lo que sucedería en la vida real. Enfrentando decisiones de diseño, trade-offs o relaciones relaciones de compromiso donde es necesario poder aplicar las competencias aprendidas para argumentar las decisiones de diseño tomadas frente al grupo de pares y frente al docente.
Las nuevas tecnologías se utilizan activamente, con aplicación de casos y contrastación de ejemplos contra Large Language Models LLM AI (Chat GPT). Se actualiza constantemente material audio visual, haciendo uso de las herramientas y el conocimiento obtenido en cursos de mejora continua. A través del campus virtual LEV (Moodle), se establece un canal de contacto bidireccional con amplio material didáctico que en conjunto con las plataformas de mensajería brindan al estudiante la posibilidad de estar en contacto directo con el docente.

Evaluación

Dentro del marco de esta propuesta teórico-práctica, esta cátedra ha optado por un proceso de evaluación mixto que incluye dos (2) exámenes parciales con oportunidad de recuperación, desarrollo de dos (2) trabajos prácticos de laboratorio y examen coloquio final integrador.
Los exámenes parciales están diseñados para evidenciar tanto el conocimiento como las competencias adquiridas y sus contenidos implican tanto conceptos teóricos como prácticos. Los trabajos prácticos de laboratorio representan el verdadero desafío de aplicación de las competencias adquiridas como un todo. Los estudiantes trabajan en equipo para diseñar, desarrollar, implementar y documentar soluciones a sistemas concurrentes y paralelos. Los proyectos incluyen el análisis de las características comportamentales del sistema, comprender la interacción de la aplicación, poner especial atención en la detección y prevención de situaciones problemáticas relacionadas con la concurrencia, el paralelismo y los recursos compartidos en el sistema así como la decisión de las técnicas utilizadas para resolverlos.

Estos trabajos tienen instancia de entrega y de defensa grupal, en la cual los estudiantes deben articular las competencias técnicas específicas adquiridas y las competencias genéricas, para argumentar y validar el desarrollo propuesto.
Estas entregas se evalúan individualmente con los siguientes criterios prefijados:

- Calidad del diseño
- Organización
- Estructura de la solución
- Calidad de la implementación (código)
- Escritura académica y documentación
- Claridad en la defensa-exposición.

Esta modalidad fomenta la habilidad comunicacional y la capacidad de argumentación de decisiones de diseño frente a terceros.
En la instancia final el estudiante debe realizar un coloquio integrador en el cual mediante una exposición dialogada con el docente, se recorren los conocimientos adquiridos y analizan las soluciones implementadas en los trabajos prácticos de laboratorios.

Si bien cada trabajo puede favorecer el desarrollo de una determinada competencia en particular y es de esperar la evidencia de esto hacia la conclusión de dicha actividad, la evaluación será contínua a lo largo de todas las actividades propuestas.

Si bien es posible inferir que cada actividad propuesta puede impactar sobre el desarrollo de competencias particulares ya sean específicas o genéricas y se anticipa que esto será evidente al finalizar cada actividad; es importante destacar que la evaluación se llevará a cabo de manera continua a lo largo de todas las actividades propuestas.
Las actividades propuestas están diseñadas para que su conclusión sea indicativo del desarrollo de los resultados de aprendizaje propuestos.

Condiciones de aprobación

Condición de regularización:
Para alcanzar esta condición el estudiante debe cumplir con:

- Asistencia y participación a clases mayor o igual al 80%.
- Aprobación de ambos exámenes Teórico-Prácticos con porcentaje mayor o igual a 60%.
- Aprobación de un trabajo práctico de laboratorio.

Condición de promoción:
Para alcanzar esta condición el estudiante debe cumplir con:

- Todos los requisitos indicados en la condición de alumno regular.
- Debe aprobar ambos trabajos prácticos de laboratorio.
- Debe aprobar el coloquio final integrador.

Actividades prácticas y de laboratorio

Además de las actividades indicadas en el punto Evaluación, en las clases prácticas se desarrollan actividades en conjunto con el dictado de clases resolviendo problemas de

acuerdo a los contenidos que se van desarrollando entre los cuales se destacan las siguientes actividades:

- Desarrollo de diagramas UML - Casos de aplicación
- Creación de hilos (Threads) y su interacción. Debug.
- Control de un thread (Interrupt, sleep, join)
- Herramientas de sincronización (Synchronized, locks)
- Herramientas avanzadas de sincronización (Read_write_locks, fairness, semáforos)
- Herramientas avanzadas de concurrencia (Uncont_exceptions, daemon_threads, local_thread_variables, count_down_latch)
- Herramientas avanzadas de sincronización en java (Cyclic_barrier, phaser, exchanger)
- Control y gestión de subrutinas concurrentes y paralelas (Factory, thread_factory, executors)

Resultados de aprendizaje

Los resultados de aprendizaje esperados de la asignatura son:

1. Analizar e interpretar correctamente el dominio de un problema
2. Evaluar las características y requerimientos de un sistema concurrente y paralelo de mediana dimensión.
3. Identificar y comprender situaciones donde el uso compartido de recursos tiene potencial impacto en el funcionamiento del sistema
4. Analizar el paralelismo inherente a un simple algoritmo secuencial.
5. Seleccionar métodos apropiados para medir el rendimiento de algoritmos multiproceso.
6. Explicar y calcular los tiempos de respuesta de un sistema concurrente y paralelo y explicar los factores que limitan la aceleración alcanzable.
7. Explicar las limitaciones de la escalabilidad.
8. Reconocer el potencial del co-diseño hardware-software en circunstancias de complejidad modesta.
9. Gestionar la creación y el manejo de subrutinas o hilos de ejecución (threads),
10. Identificar, diseñar e implementar la sincronización de subprocesos reconociendo condiciones de carrera y exclusión mutua.
11. Gestionar un conjunto de threads, crear y administrar ejecutores (thread pool executors)
12. Crear un plan de prueba y generar casos de prueba para un sistema basado en computadora de complejidad media, seleccionando una combinación apropiada de cantidad y niveles de pruebas para garantizar la calidad del sistema.
13. Adaptar el comportamiento predeterminado de algunas clases de concurrencia a sus necesidades y diseñar, verificar y validar casos de aplicación de concurrencia con el lenguaje Java.
14. Realizar, como parte de una actividad de equipo, una inspección de un diseño de sistema informático de tamaño mediano.

Estos resultados de aprendizaje cubren las competencias genéricas y específicas y serán evaluados en las instancias de evaluación previstas mediante rúbricas elaboradas a tal efecto.

Bibliografía

- J. T. P. Méndez, Programación concurrente: Editorial Paraninfo, 2003.
- J. F. González, Java 9 Concurrency Cookbook, Second Edition, Packt Publishing Ltd, 2017.
- D. Lea, Concurrent Programming in Java™: Design Principles and Patterns, Second Edition, 1999.
- W. Stallings, Sistemas operativos vol. 2: Prentice Hall, 2004.
- M. Diaz, Petri Nets Fundamental Models, Verification and Applications. NJ USA: John Wiley & Sons, Inc, 2009.
- R. David and H. Alla, Discrete, continuous, and hybrid Petri nets. Springer Science & Business Media, 2010.

FCEFyN
FACULTAD DE CIENCIAS EXACTAS, FÍSICAS y NATURALES

UNC Universidad Nacional de Córdoba

Asignatura: Redes de Computadoras

Código:	RTF	9
Semestre: Sexto	Carga Horaria	96
Bloque: Tecnologías Básicas	Horas de Práctica	40
Departamento: Computación		

Correlativas:
- Electrónica Digital 2

Contenido Sintético:
- Historia y descripción general.
- Herramientas relevantes, estándares y / o restricciones de ingeniería.
- Arquitectura de red.
- Redes locales y de área amplia.
- Redes inalámbricas y móviles.
- Protocolos de red.
- Aplicaciones de red.
- Gestión de redes.

Competencias Genéricas:
- CG.1. Identificar, formular y resolver problemas de ingeniería.
- CG.2. Concebir, diseñar y desarrollar proyectos de ingeniería (sistemas, componentes, productos o procesos).
- CG.4. Utilizar de manera efectiva las técnicas y herramientas de aplicación en ingeniería.
- CG.6. Desempeñarse de manera efectiva en equipos de trabajo.
- CG.7. Comunicarse con efectividad.
- CG.9. Aprender en forma continua y autónoma.

Aprobado por HCD: NNNN-HCD-AAAA RES: Fecha: DD/MM/AAAA

Competencias Específicas:

- CE5.1 Analizar, interpretar, diseñar e implementar arquitecturas de redes de computadoras.
- CE5.2 Seleccionar e implementar los componentes de red adecuados.
- CE5.3 Configurar, administrar y mantener redes de computadoras.
- CE5.4 Conocer e implementar protocolos y tecnologías móviles.
- CE5.5 Conocer e implementar protocolos de comunicación utilizados en redes de computadoras.

Presentación

Redes de Computadoras es una asignatura que pertenece al tercer año (sexto cuatrimestre) de la carrera de Ingeniería en Computación. Al momento de transitar este espacio curricular, el estudiante ha cursado las primeras materias de física, matemáticas y fue introducido en la teoría de señales y sistemas, siendo en Redes de Computadoras, el primer espacio en el que se aborda la temática de las comunicaciones de datos, y en particular, las redes de computadoras. Así, en esta asignatura se integran conocimientos de las ciencias básicas en el desarrollo de soluciones aplicando conocimientos y tecnologías propias de las comunicaciones de datos y las redes de computadoras.

Contenidos

Capítulo 1: Introducción a los Sistemas de Comunicaciones

- Modelo de un sistema de comunicación.
- Teoría de la información: Entropía, Señal/Ruido, Ley de Shannon-Hartley.
- Elementos de Comunicaciones de Datos: Transmisión analógica y digital, sincrónica y asincrónica. Modulaciones. Multiplexación.

Capítulo 2: Introducción a las Redes de Datos

- Concepto de Red de telecomunicación. Arquitecturas.
- Modelos de referencia OSI y TCP/IP.
- Internet: Orígenes, servicios básicos, arquitectura.
- Organismos Internacionales de Normalización.

Capítulo 3: Capa Física: Medios de Transmisión

- Medios físicos de transmisión: Cables, coaxiales, fibras ópticas, transmisiones inalámbricas.
- Propagación inalámbrica, tipos de interferencias.
- Unidades de medida.

Capítulo 4: Capa de Acceso en Redes Locales

- Métodos de acceso al medio.
- Estándares IEEE 802.3 y 802.11.
- Dispositivos: NIC, Hub, Bridge, Switch, Access Point.
- Protocolo de Trunk (802.1Q), Spanning Tree Protocol (STP).

Capítulo 5: Capa de Interred – Direccionamiento – Encaminamiento (Ruteo)

- Direccionamiento IP: IPv4, IPv6, VLSM, CIDR.
- Protocolos ICMP, ARP, BOOTP.
- Encaminamiento interno y externo: RIP, EIGRP, OSPF, BGP.
- Routers y enrutamiento entre VLANs.

Capítulo 6: Capa de Transporte

- Protocolos TCP y UDP.
- Traducción de Direcciones de Red (NAT, PAT).

Capítulo 7: Capa de Aplicación
- Protocolos DHCP, DNS, FTP, TFTP, SNMP, Correo Electrónico, WWW.
- Protocolos en aplicaciones específicas: Voz sobre IP, IoT, tecnologías móviles.
- Control de congestión, calidad de servicio, herramientas de administración de red.

Metodología de enseñanza

La asignatura Redes de Computadoras, es una asignatura teórico-práctica, donde el conjunto de actividades que se presentan, están enfocadas hacia el Aprendizaje Centrado en el Estudiante y el desarrollo de las competencias.

Durante las clases se emplearán distintas estrategias como exposición dialogada, resolución de problemas, actividades prácticas, aprendizaje basado en proyectos y visitas.

Evaluación

La evaluación se realiza en forma continua, durante las actividades de clases, trabajos prácticos y de laboratorio y proyecto final.

La evaluación de las competencias propuestas se realiza de forma contínua a través del desarrollo de las actividades propuestas.

Para las competencias CG6, CG7 y CG9 se propone el empleo de rúbrica como instrumento de evaluación, mientras que para las demás competencias se considera que el completar las actividades propuestas es indicador de haber alcanzado un nivel de desarrollo satisfactorio de las mismas.

Rúbrica de evaluación:

Competencia	Resultado de aprendizaje	Nivel de aceptación mínimo	Nivel alcanzado 1 - 3
CG.6	Participa de manera activa y colaborativa en proyectos prácticos de redes de computadoras.	2	
CG.6	Contribuye efectivamente a la resolución de problemas en el ámbito de las redes, demostrando habilidades de cooperación.	2	

CG.7	Explicar claramente conceptos técnicos relacionados con redes de computadoras tanto de forma oral como escrita	2	
CG.7	Demostrar habilidades de comunicación en la presentación de proyectos, informes y documentación técnica.	2	
CG.9	Muestra iniciativa en la búsqueda y adquisición de conocimientos adicionales sobre tecnologías emergentes en redes de computadoras	2	
CG.9	Identifica fuentes fiables de información.	2	

Niveles de desarrollo:

1: No evidencia haber alcanzado el resultado esperado.

2: Evidencia haber alcanzado el desarrollo mínimo esperado.

3: Evidencia haber alcanzado un desarrollo satisfactorio.

Condiciones de aprobación

Para obtener la regularidad:

- 80% de asistencia a las clases.
- Aprobación de los trabajos prácticos con el 60% cada uno, incluidas dos instancias de recuperación.
- Alcanzar un nivel de desempeño mínimo en los resultados de aprendizaje propuestos.

Para obtener la Promoción:

- Cumplir las condiciones necesarias para alcanzar la regularidad.
- Aprobar los trabajos finales de laboratorio y de investigación.

Se conformará una nota promedio final, de las notas obtenidas en las distintas instancias.

Actividades Prácticas propuestas

1. Medios físicos de comunicación
2. Análisis de tramas en Capa 2 en modelo OSI
3. Introducción a Capa 3 en modelo OSI
4. Ruteo interno dinámico y Ruteo Externo
5. Capa de transporte.

6. Capa de Aplicación.
7. Trabajo práctico y de laboratorio grupal integrador Final: Análisis, diseño e implementación de una red de datos.
8. Trabajo grupal de investigación.

resultados de aprendizaje

Competencia	Resultado de aprendizaje
CG.6	Participa de manera activa y colaborativa en proyectos prácticos de redes de computadoras.
CG.6	Contribuye efectivamente a la resolución de problemas en el ámbito de las redes, demostrando habilidades de cooperación.
CG.7	Explicar claramente conceptos técnicos relacionados con redes de computadoras tanto de forma oral como escrita
CG.7	Demostrar habilidades de comunicación en la presentación de proyectos, informes y documentación técnica.
CG.9	Muestra iniciativa en la búsqueda y adquisición de conocimientos adicionales sobre tecnologías emergentes en redes de computadoras
CG.9	Identifica fuentes fiables de información.
CG.1 CE5.2	Analiza problemas específicos en redes de computadoras y propone soluciones efectivas.
CG.1 CG.2	Aborda los desafíos técnicos propuestos mediante la aplicación de conocimientos teóricos a las situaciones prácticas.
CG.4	Aplica herramientas de simulación y configuración para el análisis y diseño de redes.
CE5.1	Evalúa arquitecturas existentes y propone mejoras basadas en las necesidades del entorno.
CE5.1 CG.2	Plantea arquitecturas de red diseñadas para satisfacer requerimientos específicos.
CE5.2	Selecciona dispositivos de red según las especificaciones del proyecto.
CE5.3 CG.4	Conoce los criterios y herramientas para administrar y mantener redes operativas, aplicando prácticas de seguridad y resolviendo problemas operativos.
CE5.3	Configura dispositivos y servicios de red para garantizar su funcionalidad y rendimiento.

CE5.4 CE5.5	Conoce protocolos y tecnologías móviles en el diseño y configuración de redes.
CE5.4	Evalúa y selecciona soluciones móviles adecuadas para entornos específicos.
CE5.5	Conoce protocolos de comunicación estándar para lograr la interoperabilidad en redes heterogéneas.

Bibliografía

Bibliografía Obligatoria:

1. STALLINGS William. Comunicaciones y Redes de Computadoras. 7ma. Edición (o superior). Pearson Prentice Hall.

2. COMER Douglas. Internetworking with TCPIP Vol1. 6Ed (o superior). Pearson.

3. Kurose J. y Ross K. Redes de Computadores. Un Enfoque Descendente. 7ma (o superior). Edición. Addison Wesley.

Bibliografía optativa y otros materiales a utilizar en la asignatura:

1. TANENBAUM Andrew S. Redes de Computadoras. 4ta. Edición (o superior). Prentice Hall.

2. TANENBAUM A. y WETHERALL D. Redes de Computadoras. 5ta. Edición (o superior). Pearson Educación.

3. Revistas especializadas y manuales de instalación y configuración de dispositivos.

FCEFyN
FACULTAD DE CIENCIAS EXACTAS, FÍSICAS y NATURALES

UNC Universidad Nacional de Córdoba

Asignatura: Seguridad Informática

Código:	RTF	8
Semestre: Octavo	Carga Horaria	80
Bloque: Tecnologías Aplicadas (TA)	Horas de Práctica	40
Departamento: Computación		

Correlativas:

- Redes de Computadoras
- Calidad de Software y Hardware-Software

Contenido Sintético:

- Historia y descripción general.
- Herramientas relevantes, estándares y / o restricciones de ingeniería.
- Seguridad e integridad de los datos.
- Vulnerabilidades: factores técnicos y humanos.
- Modelos de protección de recursos.
- Criptografía de clave pública y secreta.
- Códigos de autenticación de mensajes.
- Seguridad de red y web.
- Autenticación.
- Informática de confianza.
- Ataques de canal lateral.
- Desarrollo de software seguro.

Competencias Genéricas: (Contribuciones: A = Alto; M = Medio; B = Bajo)

- CG2: Concebir, diseñar y desarrollar proyectos de ingeniería (sistemas, componentes, productos o procesos).(B)
- CG4: Utilizar de manera efectiva las técnicas y herramientas de aplicación en ingeniería. (A)
- CG6: Desempeñarse de manera efectiva en equipos de trabajo. (M)
- CG7: Comunicarse con efectividad. (A)
- CG8: Actuar con ética, responsabilidad profesional y compromiso social, considerando el impacto económico, social y ambiental de su actividad en el contexto local y global. (A)

Aprobado por HCD: NNNN-HCD-AAAA RES: Fecha: DD/MM/AAAA

Competencias Específicas: (Contribuciones: A = Alto; M = Medio; B = Bajo)

- CE10 Proyecto, Dirección y Aseguramiento de la calidad en lo referido a Seguridad Informática. (A)
- CE10.3 Asegurar la seguridad informática de los sistemas proyectados y desarrollados. (A)

Presentación

La humanidad depende cada vez más de la infraestructura informática para desarrollar prácticamente todas las facetas de la vida moderna: transporte, comunicaciones, atención médica, educación, generación y distribución de energía, solo por nombrar algunas. También se ha puesto en evidencia, con ataques desenfrenados y violaciones de esta infraestructura, que los graduados en ingeniería en computación tienen un papel cada vez más importante en el diseño e implementación de sistemas que sean seguros y puedan mantener la información privada.

La seguridad informática representa un tema transversal omnipresente en todas las demás áreas de la currícula de ingeniería en computación, incluidos los fundamentos del desarrollo de software, la gestión de datos, los sistemas operativos, las redes y las comunicaciones, la informática paralela y distribuida, los fundamentos de los sistemas y la inteligencia artificial. Como consecuencia, la seguridad informática debe incorporarse a la mentalidad filosófica de los graduados en ingeniería en computación para que todo trabajo que se espera de un graduado sea inherentemente seguro.

El área de conocimiento de la seguridad o “Security Knowledge Area” (SKA) es un nombre actualizado en el año 2013 para el área de conocimiento anteriormente conocida como “Information Assurance and Security” (IAS). Ahora bien, desde el año 2017 el SKA pasó a denominarse Ciberseguridad y se constituyó en una disciplina informática con sus propias directrices[1][2]. Los seis temas transversales de la ciberseguridad, relevantes para los graduados en ingeniería en computación son: confidencialidad, integridad, disponibilidad, riesgo, pensamiento sistémico y pensamiento del adversario. De estos temas transversales, la mentalidad o pensamiento del adversario no suele estar cubierto en las otras áreas de conocimiento y debe incluirse en esta unidad referida a seguridad informática o de ahora en más, ciberseguridad.

Los estudiantes también deben aprender conceptos de seguridad informática como autenticación, autorización y no repudio. Necesitan aprender sobre vulnerabilidades y riesgos del sistema y comprender las amenazas contra los sistemas de información. Como tal, es necesario cubrir los principios para proteger los sistemas y complementar los principios de diseño de sistemas de computación cubiertos en las asignaturas correlativas, incluidos principios como seguridad por diseño, privacidad por diseño o defensa en profundidad. Otro concepto importante es la noción de aseguramiento en términos de garantizar una certificación y de que los mecanismos implementados están a la altura de las políticas de seguridad establecidas para datos, procesos y sistemas.

[1] https://www.acm.org/binaries/content/assets/education/curricula-recommendations/csec2017.pdf
[2] https://www.cybok.org/

Contenidos

Unidad 1. Peligros y combatientes en la guerra contra la ciberdelincuencia
Historias de guerra. Actores maliciosos. Impacto de las amenazas. El centro de operaciones de seguridad moderno. Trayectos de formación como defensor.

Unidad 2. Criptografía
Criptografía histórica. Criptografía simétrica. Criptografía de bloque moderna. Criptografía de clave pública. Funciones de hash. Infraestructuras de clave pública.

Unidad 3. Principios de la seguridad de red
ICMP. Utilidades Ping y Traceroute. Verificación de conectividad. MAC e IP. ARP. Problemas de seguridad de ARP. Capa de transporte. Confiabilidad de la capa de transporte. Dispositivos de seguridad de red. Firewall. IDS. IPS. WAF. IPsec. Concentradores VPN.

Unidad 4. Amenazas y ataques comunes
Malware: virus, troyanos, gusanos, ransomware. Ataques comunes: reconocimiento, acceso e ingeniería social. Denegación de servicio, desbordamientos del búfer y evasión.

Unidad 5. Ataques a la red y a sus servicios
Detalles del PDU de IP. Vulnerabilidad de IP. Vulnerabilidades de TCP y UDP. Servicios IP. Vulnerabilidad de las aplicaciones de red. Tipos de datos de seguridad. Registros de terminales. Registros de red.

Unidad 6. Defensa
Estrategias de defensa de una red. Defensa en profundidad. Conceptos del control de acceso. Uso y funcionamiento de autenticación, autorización y registro. Políticas, regulaciones y estándares de seguridad.

Unidad 7. Observación de la operación de red
Monitoreo de red. Herramientas de monitoreo. Inteligencia de amenazas. Fuentes de información. Servicios de inteligencia de amenazas. Protocolos de monitoreo. Tecnologías de seguridad.

Unidad 8. Evaluación de vulnerabilidades
Vulnerabilidad, amenaza y riesgo. Perfiles de redes y servidores. Sistema de puntuación de vulnerabilidades comunes. CVSS. Registros de vulnerabilidades, debilidades y patrones de ataque. Administrador de dispositivos de seguridad.

Unidad 9. Alertas y respuesta a incidentes de seguridad
Fuentes de alertas. Evaluación de alertas. Investigación de datos de red. Manejo de evidencia y atribución de ataques. Cyber Kill Chain. Mitre Att@ck. Respuesta ante incidentes.

Unidad 10. Construcción de software seguro
Seguridad en el ciclo de desarrollo de software. Seguridad como requerimiento funcional. Casos de mal uso. Estándares de codificación segura. Análisis estático de código. Análisis de vulnerabilidades.

Metodología de enseñanza

El desarrollo general de la materia se sustenta en clases teórico-prácticas. Por ello las estrategias de enseñanza seleccionadas para llevar adelante esta propuesta son la exposición dialogada, el estudio de casos y la resolución de problemas. Cada unidad se desarrollará a partir de material bibliográfico propuesto por el equipo de cátedra más los que aporte el estudiante al buscar contenidos en internet a partir de las referencias. Se ofrecerán trabajos prácticos que orientarán el proceso de lectura y análisis del contenido como forma de evaluación y acreditación de cada unidad. Los trabajos prácticos estarán guiados por los casos de estudio presentados en cada unidad y se orientan a resolver problemas concretos aportando los contenidos teóricos necesarios para su resolución.

En ambientes de laboratorio el estudiante podrá representar casos de estudio, verificar su implementación a partir de consignas y aproximarse a la realidad de los problemas planteados.

Evaluación

En el marco de la propuesta, el equipo de cátedra ha decidido realizar el seguimiento de los alumnos con una propuesta mixta que incluye: toma de un mínimo de cuatro exámenes parciales con un recuperatorio y evaluación formativa. Los parciales buscan evidenciar los conocimientos y competencias adquiridas a través de presentación de casos prácticos (análisis, resolución de problemas, etc.) y/o respuestas a preguntas conceptuales. Constituyen en sí mismos una instancia de evaluación formativa ya que luego los estudiantes reciben realimentación de los errores cometidos.

Cada Actividad Práctica o de Laboratorio grupal propuesta (ver debajo) demanda la presentación de implementaciones funcionales e informes, donde los estudiantes explican individualmente las diferentes partes de la solución propuesta y particularmente sus detalles constructivos. Es el momento donde el estudiante pone en juego su participación en el equipo, el rol y peso de sus decisiones, su capacidad de comunicar detalles de diseño, y su correcto manejo y comprensión de las decisiones tecnológicas presentes en la solución. Para la entrega que demande cada prototipo el equipo de cátedra genera un repositorio virtual con control de versiones y los criterios de evaluación sobre esta producción del estudiante son los siguientes:

- Puntualidad en la entrega de las producciones.
- Uso de la escritura académica.
- Integración y pertinencia de conceptos.
- Claridad en la formulación de las producciones.
- Vinculación teoría práctica.

La evaluación de las competencias se realizará mediante una rúbrica construida en base a los resultados de aprendizaje propuestos.

Condiciones de aprobación

Condiciones de regularización

- Asistir al 80% de las clases.
- Aprobar todas las actividades prácticas de laboratorio con al menos el 60% o más de los criterios de evaluación expresados en la sección anterior.
- Alcanzar el criterio de aceptación mínimo para los resultados de aprendizaje propuestos.
- Aprobar al menos tres parciales, Incluidos los recuperatorios que correspondan según el régimen de estudiantes, con el 60% o más de los contenidos evaluados.

Condiciones de aprobación por promoción (no requiere examen final)

- Cumplir con todas las condiciones de regularización.
- Aprobar cada uno de los cuatro parciales propuestos, Incluidos los recuperatorios que correspondan según el régimen de estudiantes, con el 60% o más de los contenidos evaluados.

Condiciones de aprobación por examen final

- Todas las condiciones de regularización expuestas anteriormente.
- Aprobación de un examen final con el 60% o más de los contenidos evaluados.

Para la nota final se promedian las notas obtenidas en cada una de las actividades prácticas y de laboratorio, y el resultado se promedia con las notas de los parciales.

Actividades prácticas y de laboratorio

Se propone la realización de 6 (seis) actividades donde se pongan en práctica los conocimientos adquiridos en la asignatura y se desarrollen las competencias esperadas. Estos trabajos se realizan en grupos lo que permitirá que desarrollen competencias de trabajo en equipo y coordinación de tareas.

APL1 – Malware STUXNET: En esta propuesta los estudiantes profundizan sobre el contexto geopolítico que motivó la construcción del malware STUXNET, los detalles de su diseño, el objetivo perseguido y las consecuencias posteriores al ataque.

APL2 – Encriptar y desencriptar: En este trabajo los estudiantes utilizan diferentes aplicaciones para encriptar, desencriptar, producir y verificar hashes, utilizando criptografía simétrica y asimétrica. Se enfrentan con el problema de la generación de claves robustas y descubren herramientas para administrar múltiples claves robustas de forma segura y aplicable al trabajo profesional cotidiano.

APL3 – Ataque a una base de datos SQL: En esta propuesta los estudiantes analizan casos de ataques de inyección de código SQL, que permite escribir sentencias SQL en un sitio web y recibir una respuesta de la base de datos. El objetivo es visualizar los riesgos de manipulación de una base de datos. El equipo de cátedra entrega un escenario virtual para llevar adelante el ataque, concretarlo, capturar el tráfico en un archivo PCAP y luego analizar con Wireshark los detalles del ataque y elaborar conclusiones.

APL4 – Investigación de tráfico de red: En este caso los estudiantes deben investigar un malware exploit, interpretar datos HTTP y DNS para aislar al actor de amenazas y aislar terminales afectadas en un escenario virtual entregado por el equipo de cátedra.

APL5 – Monitoreo de red: En este trabajo deben preparar un ambiente virtual para monitoreo de alertas con un sistema SIEM (Security Information and Event Management). Se entregarán diferentes escenarios recogidos en máquinas virtuales para que respondan preguntas a fin de tratar alertas de seguridad.

APL6 – Construcción de software seguro: En este laboratorio la propuesta es indagar sobre las consecuencias del tratamiento de los requerimientos de seguridad de una aplicación en términos de requerimientos funcionales. Descubrir alternativas open source para la generación de código seguro y utilizar herramientas para escaneo de vulnerabilidades de una aplicación conteniendo código seguro.

Desagregado de competencias y resultados de aprendizaje

Los resultados de aprendizaje a promover en el desarrollo de la asignatura son 26 (veintiséis) agrupados en 8 (ocho) categorías, en relación con el descriptor "**Conceptos de Seguridad Informática**" dentro del bloque de "**Tecnologías Aplicadas**".

1. Conceptos básicos de criptografía

    1.1. Describir el propósito de la criptografía y enumerar las formas en que se utiliza en las comunicaciones de datos.
    1.2. Describir los siguientes términos: cifrado, criptoanálisis, algoritmo criptográfico y criptología, y describir los dos métodos básicos (cifrados) para transformar texto sin formato en texto cifrado.
    1.3. Explicar cómo la infraestructura de clave pública admite la firma y el cifrado digitales y analizar limitaciones y vulnerabilidades.
    1.4. Discutir los peligros de inventar sus propios métodos criptográficos. Describir qué protocolos, herramientas y técnicas criptográficas son apropiados para una situación determinada.

2. Integridad y autenticación de datos

    2.1. Explicar los conceptos de autenticación, autorización, control de acceso e integridad de datos.
    2.2. Explicar las diversas técnicas de autenticación con sus fortalezas y debilidades.
    2.3. Explicar ataques posibles a contraseñas.

3. Requerimientos de seguridad y el papel que desempeñan en el diseño.

    3.1. Explicar por qué los requerimientos de seguridad son importantes.
    3.2. Identificar vectores de ataque comunes.
    3.3. Describir la importancia de escribir programas seguros y robustos.
    3.4. Describir el concepto de privacidad, incluida la información de identificación personal.

4. Problemas de implementación de software

    4.1. Diferenciar entre codificación segura y parcheo y explicar la ventaja de utilizar técnicas de codificación segura.
    4.2. Describir un desbordamiento de búfer y por qué es un posible problema de seguridad.

5. Configurar y parchear software

5.1. Analizar la necesidad de actualizar el software para corregir vulnerabilidades de seguridad.
5.2. Explicar la necesidad de probar el software después de una actualización pero antes de distribuir el parche.
5.3. Explicar la importancia de configurar correctamente el software.

6. Ética, especialmente en el desarrollo, las pruebas y la divulgación de vulnerabilidades

6.1. Explicar el concepto de que el hecho de que pueda hacerlo no significa que deba hacerlo.
6.2. Discutir las cuestiones éticas al revelar vulnerabilidades.
6.3. Analizar la ética de las pruebas exhaustivas, especialmente los casos extremos.
6.4. Identificar los efectos e impactos éticos de las decisiones de diseño.

7. Monitoreo de tráfico

7.1. Analizar cómo los sistemas de detección de intrusos contribuyen a la seguridad.
7.2. Describir los límites del software antimalware, como los programas antivirus.
7.3. Analizar los usos de sistemas de monitoreo.

8. Pruebas de sistemas

8.1. Describir qué es una prueba de penetración y por qué es valiosa.
8.2. Analizar cómo documentar una prueba que revele una vulnerabilidad.
8.3. Discutir la importancia de validar requerimientos.

A continuación, se muestra en la Tabla 1, las competencias específicas desagregadas y los resultados de aprendizaje relacionados.

Desagregación de Competencias Específicas	Resultados de Aprendizaje
CE10.3_A Proyectar y diseñar la Seguridad Informática. (A)	(1) (2) (3)
CE10.3_B Dirigir e implementar la Seguridad Informática. (A)	(4) (5) (6)
CE10.3_C Asegurar y garantizar la Seguridad Informática. (A)	(7)(8)

Tabla 1: Desagregación de competencias y resultados de aprendizaje

Las competencias genéricas CG2, CG4 y CG8 se encuentran cubiertas por los resultados de aprendizaje propuestos anteriormente.

Para las competencias CG6 y CG7 se proponen los siguientes resultados de aprendizaje:

1. Redacta informes técnicos precisos y con lenguaje claro.
2. Interpreta correctamente una consigna compleja.
3. Respeta y se desempeña adecuadamente en el rol asignado dentro de su equipo.

Bibliografía

- Chapple M., Stewart J., Gibson D., Seidl D., (ISC)2 CISSP Certified Information Systems Security Professional Official Study Guide & Practice Tests Bundle 3rd Edición. Editorial Sybex 2021. ISBN 978-1119790020.
- CyberOps Associate, 2020, Academia Cisco UNC.
- Brooks C., Grow C., Craig P.Jr., Short D., Cybersecurity Essentials 1st Edición, Editorial Sybex 2018. ISBN 978-1119362395.
- https://www.cybok.org/

FCEFyN
FACULTAD DE CIENCIAS EXACTAS, FÍSICAS y NATURALES

UNC Universidad Nacional de Córdoba

Asignatura: **Sistemas Ciberfísicos**

Código:	RTF	8
Semestre: Noveno • Ingeniería en Computación	Carga Horaria	80
Bloque: Tecnologías Aplcadas	Horas de Práctica	48
Departamento: Computación		
Correlativas: • Sistemas de Control 2		
Contenido Sintético: • Introducción a los sistemas ciberfísicos. • Elementos de computación y de comunicación. • Protocolos de comunicación en sistemas ciberfísicos. • Internet de las cosas (IoT). • Control y automatización. • Interacción humano-máquina. • Robótica y movilidad. • Seguridad y privacidad.		
Competencias Genéricas: • CG1: Identificar, formular y resolver problemas de ingeniería. • CG2: Concebir, diseñar y desarrollar proyectos de ingeniería (sistemas, componentes, productos o procesos). • CG3: Gestionar -planificar, ejecutar y controlar- proyectos de ingeniería (sistemas, componentes, productos o procesos). • CG4: Utilizar de manera efectiva las técnicas y herramientas de aplicación en ingeniería. • CG5: Contribuir a la generación de desarrollos tecnológicos y/o innovaciones tecnológicas. • CG10: Actuar con espíritu emprendedor.		
Aprobado por HCD: NNNN-HCD-AAAA RES: Fecha: DD/MM/AAAA		

Competencias Específicas para la carrera de Ingeniería en Computación:

- CE4.3: Analizar, diseñar, programar, implementar, probar, depurar y evaluar hardware y software para sistemas de computación de propósitos específicos.
- CE4.4: Analizar, diseñar, programar, implementar, probar, depurar y evaluar hardware y software para sistemas de computación de propósitos generales.
- CE4.10 Analizar, interpretar, modelar, diseñar interfaces humano-máquina en sistemas de software y software-hardware optimizando la experiencia de usuario.
- CE4.11 Analizar, proyectar y desarrollar proyectos de software y sistemas conjuntos de hardware y software haciendo uso de conceptos, métodos y herramientas de gestión de proyectos, ingeniería de software, elicitación, análisis, especificación y validación de requerimiento.
- CE7.2.6 Diseñar circuitos impresos que integren componentes electrónicos y periféricos en un sistema embebido.
- CE7.2.7 Comprender, configurar y utilizar los protocolos de comunicación utilizados en sistemas embebidos.
- CE7.3.4: Diseñar y desarrollar sistemas que impliquen el uso de sensores y actuadores.
- CE7.3.6: Implementar, operar y mantener Sistemas Computarizados de automatización y control y sistemas conjuntos de hardware y software.

Presentación

La asignatura “Sistemas Ciberfísicos” pertenece al último año del plan de estudio de Ingeniería en Computación, situándose en el primer cuatrimestre del mismo como una materia integradora del trayecto recorrido por el estudiante. Este espacio curricular busca desarrollar la capacidad de aplicar de forma práctica conocimientos en áreas afines al software, el hardware y las comunicaciones en soluciones innovadoras que den respuesta a problemáticas actuales.

Los Sistemas Ciberfísicos (Cyber-Physical Systems o CPS, por sus siglas en inglés) son sistemas integrados que combinan componentes computacionales, comunicación y control con elementos físicos del mundo real. Los CPS permiten que los dispositivos y sistemas físicos se conecten, interactúen y se controlen a través de la computación y la comunicación, lo que los hace altamente inteligentes y adaptables. Estos sistemas pueden ser encontrados en una amplia variedad de aplicaciones, desde la industria manufacturera y la salud hasta el transporte y la energía:

- *Industria 4.0:* Los CPS son un pilar clave de la Industria 4.0, la cuarta revolución industrial. Son esenciales para la automatización de procesos industriales, el mantenimiento predictivo y la optimización de la producción.
- *IoT y Smart Cities*: Los dispositivos de Internet de las Cosas (Internet of Things o IoT, por sus siglas en inglés) y las ciudades inteligentes dependen de los CPS para la recopilación, análisis y toma de decisiones basadas en datos en tiempo real.
- *Salud y Medicina*: Los CPS se utilizan en dispositivos médicos avanzados, hospitales inteligentes y telemedicina para mejorar la atención médica y la calidad de vida.
- *Transporte Autónomo*: Los vehículos autónomos dependen de sistemas ciberfísicos para la navegación, el control y la toma de decisiones seguras.
- *Energía Sostenible*: Los CPS permiten una gestión eficiente y sostenible de la energía, incluyendo la generación distribuida y las redes eléctricas inteligentes.

El profesional de Ingeniería en Computación desempeña un papel esencial en el diseño, desarrollo y gestión de Sistemas Ciberfísicos. Su capacidad para integrar conocimientos en informática, comunicaciones y control le permite abordar desafíos críticos en la automatización industrial, la eficiencia energética, la atención médica avanzada y la movilidad inteligente entre otros. La creciente demanda de estos sistemas en la industria y la sociedad hace imperativo que los futuros profesionales en el campo de la Ingeniería en Computación transiten un curso dedicado a los sistemas ciberfísicos, que les brinde las herramientas necesarias para explorar y poner en práctica proyectos donde el mundo digital y el físico interactúen de forma innovadora y complementaria.

Contenidos

Unidad 1: Análisis de Sistemas Ciberfísicos
Origen de los Sistemas Ciberfísicos y su evolución. Importancia y aplicaciones de los Sistemas Ciberfísicos. Identificación de problemas y oportunidades en el mundo real.

Recopilación y análisis de datos relevantes. Especificación de requisitos funcionales y no funcionales. Análisis de procesos físicos y digitales en Sistemas Ciberfísicos. Documentación de resultados de análisis y especificaciones.

Unidad 2: Modelado de Sistemas Ciberfísicos
Modelado de componentes físicos y sus interacciones. Modelado de componentes lógicos y su integración. Uso de herramientas de simulación para Sistemas Ciberfísicos. Validación y verificación de modelos. Análisis de sensibilidad y escenarios de simulación. Interpretación de resultados de simulación. Documentación y presentación de modelos y resultados.

Unidad 3: Diseño de Sistemas Ciberfísicos
Diseño de arquitecturas para Sistemas Ciberfísicos. Selección de componentes y dispositivos adecuados. Integración de sensores y actuadores en el diseño. Definición de interfaces de comunicación. Elección de tecnologías de comunicación y protocolos. Diseño de sistemas de control y algoritmos. Planificación de la administración de energía y recursos. Evaluación de la escalabilidad y flexibilidad del diseño.

Unidad 4: Implementación de Sistemas Ciberfísicos
Programación de dispositivos y componentes. Desarrollo de software para Sistemas Ciberfísicos. Integración y ensamblaje de componentes. Implementación de algoritmos de control. Configuración de la comunicación entre dispositivos. Prototipado rápido y desarrollo iterativo de modelos. Pruebas unitarias y de integración. Monitoreo y gestión del sistema en tiempo real. Evaluación del desempeño y eficiencia del sistema.

Metodología de enseñanza

La asignatura se organiza mediante cuatro unidades didácticas que guían de forma progresiva al estudiante a través de las diferentes etapas necesarias para el desarrollo de los sistemas ciberfísicos: análisis, modelado, diseño e implementación. Cada unidad propone trabajar sobre una etapa durante un período de entre 3 y 5 semanas. La metodología de enseñanza propuesta integra el modelo de aula invertida, el aprendizaje basado en proyectos (ABP), el estudio de casos y el uso del aula virtual como complemento esencial a las clases semanales. Durante las clases presenciales se priorizará el aprendizaje a través del trabajo en grupos y la experimentación con diferentes tecnologías propuestas en las unidades, desarrollando algunos temas a través de exposiciones breves por parte del docente y del debate de diferentes casos de estudio moderados por el docente.

El curso propone al estudiante el desarrollo de un sistema ciberfísico, el cual es trabajado como proyecto de forma progresiva en cada una de las unidades didácticas del curso. Al finalizar cada unidad, el trabajo desarrollado por el estudiante es revisado y evaluado por el docente a los fines de poder continuar con la siguiente unidad.

La primera unidad consiste en una fase inicial de análisis que busca definir una solución a una problemática donde un sistema ciberfísico pueda ofrecer ventajas sobre soluciones tradicionales. En esta etapa se espera que el estudiante logre identificar las principales características de estos sistemas y pueda formular posibles soluciones basadas en estos a problemáticas actuales. Aun cuando ya existieran sistemas similares, se espera que el

estudiante pueda realizar un análisis propio y compararlo con las soluciones existentes, proponiendo mejoras o simplemente generando una propuesta independiente de las existentes. En una segunda etapa, se propone al estudiante modelar el sistema ciberfísico a través de diferentes herramientas que logren replicar su funcionamiento e interacción con la problemática bajo estudio. El uso de herramientas de simulación y/o emulación es trabajado para lograr este objetivo. En una tercera etapa el estudiante aborda el diseño del sistema considerando aspectos tecnológicos que la solución requiere y justificando sus decisiones. Este diseño es revisado y realimentado tanto por el docente como por sus pares, incentivando la colaboración y trabajo en grupo. Por último, la etapa final consiste en una demostración tecnológica del sistema a través de la implementación del mismo o de una parte. Dada la complejidad que estos sistemas pueden tener, esta etapa busca principalmente alcanzar la implementación de una prueba de concepto o eventualmente un prototipo del sistema.

Evaluación

La evaluación se estructura en base a un enfoque continuo e integrador, reflejando la naturaleza progresiva y acumulativa del aprendizaje basado en proyectos. La misma se organiza a través de la presentación de un informe de avance para cada una de las cuatro etapas de desarrollo del proyecto: análisis, modelado, diseño e implementación. Además de la presentación, cada informe estará acompañado de una exposición a cargo del estudiante del avance del proyecto la cual es realizada durante las clases presenciales donde se fomentará el debate y la participación del resto de los estudiantes. El informe y la exposición serán evaluados de manera conjunta en una escala de 1 a 10 como una actividad de Evaluación de Proyecto (EP), resultando en un total de cuatro evaluaciones, una por cada fase: EP1, EP2, EP3 y EP4. Estas actividades permiten evaluar la comprensión y aplicación de los conceptos principales, así como la progresión del proyecto.
El desarrollo de las competencias se evaluará de forma continua mediante rúbricas, construida en base a los resultados de aprendizaje propuestos.

Condiciones de aprobación

Condiciones de regularización

Para alcanzar la condición de regular, el estudiante debe cumplir las siguientes condiciones:

1. Asistir al menos al 80% de las clases.
2. Alcanzar una calificación no inferior a 4 (cuatro) en cada uno de Evaluaciones de Proyecto (EP).
3. Alcanzar el criterio de aceptación mínimo para los resultados de aprendizaje propuestos.

Condiciones de promoción

Para alcanzar la promoción, el alumno debe cumplir las siguientes condiciones:

1. Alcanzar la condición de alumno regular para lo cual se deben cumplir las condiciones indicadas al respecto.
2. Alcanzar una calificación no inferior a 7 (siete) en la última Evaluación de Proyecto (EP4).

Cumplidas estas condiciones, la nota final de promoción se calculará según la siguiente fórmula:

Nota Final = redondear(10%*EP1 + 20%*EP2 + 30%*EP3 + 40%*EP4)

Examen final

Estudiantes Regulares

Los estudiantes regulares deben rendir un examen equivalente a la última actividad de Evaluación Proyecto (EP4), la cual evaluará el grado de implementación de un sistema ciberfísico propuesto a través de una prueba de concepto o prototipo. El examen consistirá en la entrega de un informe de dicha actividad y la exposición oral del mismo en la fecha del final. La nota final se obtendrá considerando las calificaciones EP1, EP2 y EP3 obtenidas durante la cursada y la calificación de la actividad EP4 obtenida en la fecha del final a través de la siguiente fórmula:

Nota Final = redondear(10%*EP1 + 20%*EP2 + 30%*EP3 + 40%*EP4)

Estudiantes Libres

Los estudiantes libres deben rendir un examen que comprende la evaluación de las cuatro diferentes fases para el desarrollo de un sistema ciberfísico correspondientes a las cuatros actividades indicadas como EP1, EP2, EP3 y EP4.

La Nota Final final para los estudiantes libres se obtendrá por la siguiente expresión:

Nota Final = redondear(10%*EP1 + 20%*EP2 + 30%*EP3 + 40%*EP4)

Actividades prácticas y de laboratorio

Las actividades prácticas y de laboratorio desempeñan un papel fundamental en el curso. Estas actividades se diseñan en consonancia con los objetivos específicos de cada unidad didáctica y se orientan hacia la aplicación práctica de los conceptos y herramientas aprendidos en el aula. A lo largo del curso, los estudiantes trabajarán en proyectos relacionados con sistemas ciberfísicos, lo que les brindará la oportunidad de adquirir experiencia práctica en el desarrollo de soluciones innovadoras.

En cada unidad, se presentarán diversas herramientas y tecnologías que los estudiantes deberán utilizar para avanzar en sus proyectos. Durante las clases presenciales, el docente utilizará ejemplos y casos prácticos para demostrar el uso adecuado de estas herramientas. Los estudiantes podrán replicar estos ejemplos en el entorno de laboratorio, lo que les permitirá familiarizarse con las tecnologías y ganar confianza en su aplicación.El aspecto más destacado de las actividades prácticas y de laboratorio es su integración en el proyecto central del curso. Los estudiantes aplicarán las herramientas y tecnologías aprendidas en clase a sus propios proyectos de sistemas ciberfísicos. Esto implica no sólo la comprensión teórica de los conceptos, sino también su aplicación en un contexto real.

Las actividades prácticas y de laboratorio permitirán a los estudiantes evaluar y ajustar sus proyectos a medida que avanzan en el curso. Podrán experimentar con diferentes enfoques y soluciones, lo que enriquecerá su comprensión y habilidades en la materia. Además, tendrán la oportunidad de colaborar con sus compañeros, lo que fomentará el aprendizaje cooperativo y la resolución conjunta de problemas.

Al finalizar cada actividad práctica, los estudiantes documentarán sus resultados y reflexionarán sobre las lecciones aprendidas. Esto les ayudará a mejorar continuamente sus proyectos y a mantener un registro detallado de su progreso a lo largo del curso.

Desagregado de competencias y resultados de aprendizaje

Los resultados de aprendizaje a promover en el desarrollo de la asignatura relacionados con los siguientes descriptores de Tecnologías Aplicadas (TA)

- Especificación, proyecto y Desarrollo de Software y Sistemas Conjuntos de Hardware y Software haciendo uso de conceptos, métodos y herramientas de gestión de proyectos, ingeniería de software, base de datos, experiencia del usuario, elicitación, análisis, especificación y validación de requerimiento.
- Proyecto, desarrollo, dirección, control, construcción, operación y mantenimiento de Sistemas de Procesamiento de Señales, Sistemas Embebidos y sus periféricos incluido en software de soporte, Sistemas Computarizados de automatización y control y Sistemas Conjuntos de Hardware y Software.

y con los siguientes descriptores de Ciencias y Tecnologías Complementarias (CTC)

- Identificación, formulación y resolución de problemas de ingeniería en computación.
- Concepción, diseño y desarrollo de proyectos de ingeniería en computación.
- Gestión, planificación, ejecución y control de proyectos de ingeniería en computación.
- Generación de desarrollos tecnológicos y/o innovaciones tecnológicas.
- Desarrollo de una actitud profesional emprendedora.

Los siguientes resultados de aprendizaje (RA) representan una amplia gama de habilidades y conocimientos relacionados con el desarrollo de sistemas ciberfísicos a través de sus diferentes etapas: análisis, modelado, diseño e implementación.

- RA1: Comprender los conceptos fundamentales de los sistemas ciberfísicos, incluyendo su definición, evolución histórica y aplicaciones en diversas industrias.
- RA2: Identificar problemas y oportunidades en el mundo real que pueden abordarse mediante la implementación de sistemas ciberfísicos.
- RA3: Realizar un análisis completo de un sistema ciberfísico propuesto, incluyendo la recopilación y análisis de datos relevantes, la especificación de requisitos funcionales y no funcionales, y la documentación de los resultados.
- RA4: Modelar componentes físicos y lógicos de un sistema ciberfísico, utilizando herramientas de simulación para representar su funcionamiento y sus interacciones.
- RA5: Validar y verificar modelos de sistemas ciberfísicos, así como analizar la sensibilidad de los mismos y evaluar escenarios de simulación.
- RA6: Diseñar arquitecturas para sistemas ciberfísicos, seleccionar componentes y dispositivos adecuados, e integrar sensores y actuadores en el diseño.

- RA7: Definir interfaces de comunicación, seleccionar tecnologías y protocolos de comunicación, y diseñar sistemas de control y algoritmos para sistemas ciberfísicos.
- RA8: Planificar la administración de energía y recursos en sistemas ciberfísicos, evaluando la escalabilidad y flexibilidad del diseño.
- RA9: Programar dispositivos y componentes, desarrollar software para sistemas ciberfísicos, integrar y ensamblar componentes, implementar algoritmos de control y configurar la comunicación entre dispositivos.
- RA10: Realizar pruebas unitarias y de integración, monitorear y gestionar sistemas ciberfísicos en tiempo real, y evaluar el desempeño y la eficiencia del sistema.
- RA11: Documentar de manera efectiva todas las etapas del desarrollo de un sistema ciberfísico, incluyendo análisis, modelado, diseño e implementación.
- RA12: Colaborar de manera efectiva en equipos de proyecto, fomentando el aprendizaje cooperativo y la resolución conjunta de problemas.

A continuación, se indican las competencias específicas asociadas a los resultados de aprendizaje relacionados con la carrera de Ingeniería en Computación.

Competencias Genéricas	**Resultados de Aprendizaje**
CG1: Identificar, formular y resolver problemas de ingeniería.	RA2, RA3, RA5, RA6: Estos resultados de aprendizaje se enfocan en la capacidad de identificar problemas en sistemas ciberfísicos, formular soluciones y resolver problemas relacionados con el diseño y desarrollo de estos sistemas..
CG2: Concebir, diseñar y desarrollar proyectos de ingeniería (sistemas, componentes, productos o procesos).	RA4, RA6, RA7, RA8, RA9: Estos resultados de aprendizaje se centran en la capacidad de concebir, diseñar y desarrollar proyectos de sistemas ciberfísicos, incluyendo la planificación de la arquitectura, la selección de componentes y la definición de interfaces.
CG3: Gestionar -planificar, ejecutar y controlar- proyectos de ingeniería (sistemas, componentes, productos o procesos).	RA6, RA7, RA8, RA9, RA10: Estos resultados de aprendizaje se relacionan con la capacidad de gestionar proyectos de sistemas ciberfísicos, incluyendo la planificación, ejecución y control de las actividades relacionadas con el diseño, desarrollo e implementación de estos sistemas.
CG4: Utilizar de manera efectiva las técnicas y herramientas de aplicación en ingeniería.	RA4, RA9: Estos resultados de aprendizaje se enfocan en la utilización efectiva de técnicas y herramientas de simulación, modelado y diseño en la ingeniería de sistemas ciberfísicos.
CG5: Contribuir a la generación de desarrollos tecnológicos y/o innovaciones tecnológicas.	RA1, RA2, RA3, RA5, RA6, RA7, RA9, RA11, RA13: Estos resultados de aprendizaje están directamente relacionados con la contribución a la generación de desarrollos tecnológicos e innovaciones tecnológicas en el campo de los sistemas ciberfísicos.
CG10: Actuar con espíritu emprendedor.	RA2, RA3, RA6, RA11, RA13: Estos resultados de aprendizaje se centran en la capacidad de actuar con espíritu emprendedor al identificar oportunidades, formular soluciones innovadoras y colaborar en proyectos relacionados con sistemas ciberfísicos.

Competencias Específicas (Ingeniería en Computación)	Resultados de Aprendizaje
CE4.3: Analizar, diseñar, programar, implementar, probar, depurar y evaluar hardware y software para sistemas de computación de propósitos específicos.	RA6, RA9: Estos resultados de aprendizaje se relacionan con la capacidad de programar, implementar y evaluar el software y hardware específico necesario para sistemas ciberfísicos.
CE4.4: Analizar, diseñar, programar, implementar, probar, depurar y evaluar hardware y software para sistemas de computación de propósitos generales.	RA6, RA9: Estos resultados de aprendizaje se centran en la capacidad de programar, implementar y evaluar el software y hardware de propósito general utilizado en sistemas ciberfísicos.
CE4.10 Analizar, interpretar, modelar, diseñar interfaces humano-máquina en sistemas de software y software-hardware optimizando la experiencia de usuario.	RA7, RA11: Estos resultados de aprendizaje se enfocan en la capacidad de diseñar interfaces humano-máquina efectivas y optimizar la experiencia del usuario en sistemas ciberfísicos.
CE4.11 Analizar, proyectar y desarrollar proyectos de software y sistemas conjuntos de hardware y software haciendo uso de conceptos, métodos y herramientas de gestión de proyectos, ingeniería de software, elicitación, análisis, especificación y validación de requerimiento.	RA2, RA3, RA6, RA7, RA8, RA9, RA10, RA11: Estos resultados de aprendizaje están directamente relacionados con la capacidad de analizar, proyectar y desarrollar proyectos de sistemas ciberfísicos, utilizando una amplia gama de conceptos y herramientas de ingeniería de software y gestión de proyectos.
CE7.2.6 Diseñar circuitos impresos que integren componentes electrónicos y periféricos en un sistema embebido.	RA6, RA9: Estos resultados de aprendizaje contribuyen al diseño de circuitos impresos y la integración de componentes electrónicos en sistemas ciberfísicos.
CE7.2.7 Comprender, configurar y utilizar los protocolos de comunicación utilizados en sistemas embebidos.	RA7, RA9: Estos resultados de aprendizaje se relacionan con la comprensión y configuración de protocolos de comunicación utilizados en sistemas ciberfísicos.
CE7.3.4: Diseñar y desarrollar sistemas que impliquen el uso de sensores y actuadores.	RA6, RA7, RA8, RA9, RA10: Estos resultados de aprendizaje están directamente relacionados con la capacidad de diseñar y desarrollar sistemas que involucran sensores y actuadores en sistemas ciberfísicos.
CE7.3.6: Implementar, operar y mantener Sistemas Computarizados de automatización y control y sistemas conjuntos de hardware y software.	RA9, RA10: Estos resultados de aprendizaje se centran en la implementación, operación y mantenimiento de sistemas computarizados de automatización y control, así como en sistemas ciberfísicos conjuntos de hardware y software.

Bibliografía

- Edward A. Lee & Sanjit A. Seshia, *Introduction to Embedded Systems, A Cyber-Physical Systems Approach (Second Edition)*, MIT Press, ISBN 978-0-262-53381-2, 2017

- Taha, W. M., Taha, A. E. M., & Thunberg, J., *Cyber-Physical Systems: A Model-Based Approach*, Springer Nature, 2021

FCEFyN
FACULTAD DE CIENCIAS EXACTAS, FÍSICAS y NATURALES

UNC Universidad Nacional de Córdoba

Asignatura: **Sistemas Distribuidos**

Código:	RTF	10
Semestre: 8	Carga Horaria	96
Bloque: Tecnologías Aplicadas	Horas de Práctica	40
Departamento: Computación		

Correlativas:
- Redes de Computadores
- Sistemas Operativos

Contenido Sintético:
1) Introducción, evolución y visión histórica de los sistemas distribuidos.
2) Modelo de sistemas distribuidos.
3) Comunicaciones entre procesos distribuidos.
4) Objetos distribuidos e invocación de métodos remotos.
5) Algoritmos de sistemas distribuidos.
6) Soporte del sistema operativo
7) Sistemas de archivos distribuidos
8) Sincronización y estados globales
9) Transacciones y control de concurrencia
10) Redes Inalámbricas de Sensores
11) Sistemas de Memoria y Archivos distribuidos
12) Replicación y tolerancia a fallas

Competencias Genéricas:
- CG2: Concebir, diseñar y desarrollar proyectos de ingeniería (sistemas, componentes, productos o procesos).
- CG6: Desempeñarse de manera efectiva en equipos de trabajo.
- CG9: Aprender en forma continua y autónoma.

Aprobado por HCD: NNNN-HCD-AAAA RES: Fecha: DD/MM/AAAA

Competencias Específicas:
CE5.1 Analizar, interpretar, diseñar e implementar arquitecturas de redes de computadoras. (M)
CE6.1 Conocer y poder diseñar la estructura de sistemas operativos, la administración de procesos, la gestión de memoria, la administración de archivos, la gestión de dispositivos de entrada/salida y la implementación de políticas de seguridad. (M)
CE6.4 Proyectar, desarrollar, dirigir, controlar, construir, operar y mantener sistemas operativos distribuidos (A)

Presentación

Desde el inicio de la era de la computadora moderna (1945), hasta cerca de 1985, sólo se conocía la computación centralizada. A partir de la mitad de la década de los ochenta aparecen microprocesadores poderosos y económicos y comienza el desarrollo de las redes con posibilidad de conectar cientos de máquinas. Entonces aparecen los primeros sistemas distribuidos, en contraste con los sistemas centralizados.

Un sistema distribuido es un sistema en el que los componentes hardware o software se encuentran separados lógica o físicamente y que se comunican y coordinan sus acciones. Esta definición simple cubre toda la gama de sistemas en los que se pueden implementar de manera útil computadoras conectadas.
Esta definición de sistema distribuido tiene las siguientes consecuencias significativas: concurrencia, inexistencia de un reloj global, necesidad de temporalidad para coordinación/sincronización y la posibilidad de fallos independientes (desconexión, problemas de hardware o problemas de software).

La principal motivación para la construcción y uso de los sistemas distribuidos proviene del deseo de compartir los recursos. El término «recurso» es un poco abstracto, pero es el que mejor caracteriza a la variedad de cosas que pueden compartirse útilmente en un sistema de computadoras conectadas. Se extiende desde los componentes de hardware, como discos e impresoras a entidades definidas por software como archivos, bases de datos y objetos de datos de todo tipo. También incluye el flujo de tramas de vídeo que proviene de una cámara de vídeo digital y la conexión de audio que representa una llamada de teléfono móvil.

Muchas de las empresas más grandes del mundo han puesto un esfuerzo significativo en el diseño de una infraestructura sofisticada de sistema distribuido para soportar la mayoría de los servicios que hoy utilizamos como buscadores, editores de texto colaborativos, videojuegos y un sin número de herramientas que utilizamos a diario en las que sin el concepto de sistemas distribuidos todo esto sería impensado. Otro ejemplo de sistema distribuido es el de blockchain que logró sintetizar en él la solución a casi todos los problemas que presentan los sistemas distribuidos logrando una red de pares robusta y tolerante a fallas sin precedentes.

Este curso está diseñado para proporcionar a los estudiantes una profunda comprensión de los sistemas distribuidos, abarcando desde los conceptos fundamentales hasta las técnicas avanzadas en diseño, implementación y gestión. A lo largo de esta experiencia de aprendizaje, se exploran conceptos teóricos clave relacionados con la modelización de sistemas como así también aspectos relacionados con la seguridad tanto en sistemas de gran escala como en sistemas embebidos.

Contenidos

Capítulo 1) Introducción, evolución y visión histórica de los sistemas distribuidos.

Introducción. Ejemplos de sistemas distribuidos. Búsqueda web. Juegos online masivamente multijugador (MMOGs). Comercio financiero. Tendencias de los sistemas distribuidos. La importancia de las redes y el Internet moderno. Computación móvil y ubicua.

Capítulo 2) Modelo de sistemas distribuidos.

Modelos físicos de sistemas distribuidos. Introducción. Modelos físicos. Modelos arquitectónicos de sistemas distribuidos. Introducción. Elementos arquitectónicos. Patrones arquitectónicos. Soluciones middleware asociadas. Modelos fundamentales de sistemas distribuidos. Modelos fundamentales. Modelo de interacción. Modelo de fallas. Modelo de seguridad.

Capítulo 3) Comunicaciones entre procesos distribuidos.

Comunicación interproceso. Las características de la comunicación interproceso. Sockets. Protocolos de red distribuidos. Comunicación Multicast. Redes de pares mesh. Invocación remota. Introducción. Protocolos de solicitud-respuesta. Invocación de Método Remoto RMI.

Otras distinciones clave. Sistemas Publicar-Suscribir. Aplicaciones de los sistemas publicar-suscribir. Características de los sistemas publicar-suscribir. El Modelo de programación. Cola de mensajes. El modelo de programación. Aproximaciones de Memoria Compartida. Memoria Compartida Distribuida.

Capítulo 4) Objetos distribuidos e invocación de métodos remotos.

Objetos distribuidos. El modelo de objetos distribuidos. Acciones en un sistema de objetos distribuidos. Desacoplamiento espacial y temporal. La relación con la comunicación asincrónica. El modelo de programación. Grupos de procesos y grupos de objetos

Capítulo 5) Algoritmos de sistemas distribuidos.

Algoritmos diseñados para ejecutarse en múltiples procesadores, sin un control centralizado estricto, auto estabilización, computabilidad sin esperas y detectores de fallas, y material nuevo sobre programación concurrente escalable de memoria compartida.

Capítulo 6) Soporte del sistema operativo, Sistemas de Memoria y de archivos distribuidos

Características, Modelos, Acceso a Archivos, acceso a memoria Caching, Replicación, tolerancia a las Fallas

Capítulo 7) Sincronización y estados globales ,Transacciones y control de concurrencia

Exclusión mutua distribuida. Algoritmos para exclusión mutua. El algoritmo de servidor central. Elecciones. Un algoritmo de elección basado en anillo. Coordinación y acuerdo en comunicación de grupo. Modelo del sistema. Multicast básico. Multicast confiable.

Capítulo 8) Blockchain

Tipos de redes, mecanismos de consenso, smart contracts, trazabilidad.

Capítulo 9) Redes Inalámbricas de Sensores

Protocolos, capa física, microcontroladores y sistemas p2p, seguridad

Metodología de enseñanza

Se aplica una metodología de enseñanza centrada en el estudiante y orientada hacia el desarrollo de competencias profesionales sólidas. Esta metodología se basa en la premisa de que el aprendizaje efectivo de los sistemas distribuidos no solo implica la adquisición de conocimientos teóricos, sino también la capacidad de aplicar esos conocimientos de manera significativa en entornos profesionales. Se fomenta la participación activa de los estudiantes en el proceso de aprendizaje en las clases teórico-prácticas. Se promueve la resolución de problemas, el trabajo en equipo y la discusión de casos prácticos. Los estudiantes son alentados a plantear preguntas, desafiar ideas preconcebidas y explorar nuevas perspectivas, lo que les permite desarrollar habilidades críticas para la toma de decisiones en el ámbito de los sistemas distribuidos. Esto cobra vital relevancia ya que la tecnología en los sistemas distribuidos evoluciona constantemente.
Los proyectos prácticos desempeñan un papel central ya que los estudiantes tienen la oportunidad de aplicar los conceptos y técnicas aprendidos en situaciones del mundo real. A través de la creación de bases de datos, la optimización de consultas y la resolución de problemas específicos, los estudiantes adquieren habilidades prácticas esenciales para su futura carrera profesional.
Los sistemas distribuidos están presentes en un sin número de situaciones del día a día y su comprensión es fundamental en el desarrollo de un ingeniero en computación, esta comprensión implica conocimientos de redes, concurrencia y sistemas operativos, por lo tanto, fomentamos un enfoque interdisciplinario que permite a los estudiantes explorar cómo se aplican los sistemas distribuidos en diferentes sectores, como la salud, la logística, la educación y la investigación científica. Esto les brinda una visión más amplia de las oportunidades profesionales disponibles.
Visitas Técnicas y entrevistas: se realizarán visitas programadas a datacenters y entrevistas técnicas con profesionales que utilizan sistemas operativos distribuidos.
Evaluación Continua y Retroalimentación son un pilar pedagógico de esta asignatura. Los estudiantes reciben retroalimentación constante a lo largo del curso, lo que les permite identificar áreas de mejora y ajustar su enfoque de aprendizaje. Además, se fomenta la autorreflexión y la autoevaluación como herramientas para el desarrollo profesional.
Para poder alcanzar las competencias requeridas se utilizan herramientas y recursos tecnológicos de vanguardia para enriquecer la experiencia de aprendizaje.
Colaboración y Comunicación: Se promueve la comunicación efectiva y la colaboración entre estudiantes, imitando las dinámicas profesionales. Cada proyecto será abordado como un proyecto de ingeniería donde podrán elaborar las capacidades necesarias para desempeñarse de manera segura en ambientes profesionales.
Cuestiones Éticas: La privacidad, el anonimato y la seguridad serán abordados a lo largo de la currícula. Los sistemas distribuidos y en especial las redes de pares proponen una forma ética y sana de relación con la tecnología que un profesional de ingeniería en computación debe tener capacidad de implementar y administrar de manera adecuada teniendo en cuenta la responsabilidad ante la sociedad que ello representa.

Propuesta de Evaluación

Las evaluaciones están diseñadas para garantizar que los estudiantes adquieran y demuestren de manera efectiva las competencias profesionales necesarias en este campo.
Estas evaluaciones están alineadas con la metodología descrita anteriormente.

Proyectos Prácticos: Estos proyectos son la piedra angular de la evaluación. Los estudiantes trabajan en equipos para diseñar, implementar y administrar sistemas distribuidos con aplicación en el mundo real. Los proyectos incluyen la creación de una un sistema de registro distribuido, la implementación de una red de sensores distribuidas de pares con el uso de un registro de datos distribuido, la implementación de un sistema con control de acceso federado entre otros.
Como parte de los proyectos se presentan casos de uso reales de código abierto en los que los estudiantes deberán analizar y proponer soluciones y mejoras basadas en los conceptos estudiados. Los estudios de caso requieren que los estudiantes apliquen conceptos teóricos a situaciones concretas y presenten soluciones argumentadas. Las mejores soluciones serán propuestas a los proyectos originales.
Cada grupo de trabajo deberá presentar sus avances de manera semanal de forma sucinta, y con dos presentaciones formales emulando un proceso de desarrollo basado en metodologías ágiles.
En cada grupo de trabajo, los estudiantes asumen roles simulados en correspondencia con los que se identifican en el mundo profesional.
Los proyectos se evalúan en función de la calidad de diseño, rendimiento y solución de problemas.
Los estudiantes son requeridos para presentar y defender sus proyectos y soluciones ante sus compañeros y el profesor. Esto fomenta la habilidad de comunicación y la capacidad de explicar y respaldar sus decisiones de diseño y gestión de bases de datos.
Colaboración y Evaluación por Pares: En conjunto con la presentación de los proyectos, se trabaja con la evaluación entre pares, donde los estudiantes preparan al tema corto y puntual, luego evalúan y proporcionan retroalimentación sobre el trabajo de sus compañeros. Esto promueve la colaboración y la capacidad de evaluar críticamente el trabajo de otros. Se evalúa la pertinencia y completitud de las críticas.
Evaluaciones Continuas: A lo largo de la asignatura, se realizan evaluaciones formativas automatizadas que permiten a los estudiantes recibir retroalimentación regular sobre su progreso. Esto incluye ejercicios prácticos y cuestionarios automatizados de los temas troncales de cada unidad.
Exámenes teóricos-prácticos: aunque la metodología se centra en la aplicación práctica, también es importante que los estudiantes comprendan los fundamentos teóricos. Los exámenes teóricos-prácticos evalúan el conocimiento conceptual de los estudiantes. De esta manera cada trabajo puede favorecer el desarrollo de una determinada competencia en particular y es de esperar la evidencia de esto hacia la conclusión de dicha actividad, la evaluación será contínua a lo largo de todas las actividades propuestas. Al final del semestre cada estudiante debe haber demostrado un nivel de desarrollo de las competencias propuestas a través de los resultados de aprendizaje propuestos.

Cada trabajo será calificado en función de los aspectos disciplinares, así como de la evidencia de desarrollo de las competencias alcanzadas al momento de la finalización del mismo, pudiendo modificar esta calificación si en el transcurso de

los trabajos subsiguientes se evidencia un mayor desarrollo de las mismas.
Como herramienta de evaluación del conjunto de competencias propuestas se emplea la siguiente rúbrica:

Competencia	Resultado de Aprendizaje	Mínimo	Valoración
CG2, CG6	Interpreta correctamente el dominio de un problema y diseña soluciones acordes y es capaz de establecer procedimientos de trabajo acordes		
CG6	Se comunica de manera efectiva con sus pares y con el docente, utiliza las herramientas de comunicación adecuadas y respeta los procedimientos de trabajo establecidos para el proyecto.		
CG6	Cumple en tiempo con los compromisos asumidos con su equipo de trabajo.		
CG9, CG6	Estudia, prepara y expone un tema pequeño y novedoso, frente a sus pares resaltando los aspectos fundamentales del tema, citando las referencias utilizadas y extrayendo conclusiones propias.		
CE5.1,CG9	Diseña una solución eficiente adecuada a un problema de sensado distribuido y determina las fortalezas y debilidades del diseño.		
CE5.1,CE6.1	Diseña una red para un sistema distribuido eligiendo los protocolos adecuados y evalúa la performance de su implementación y evalúa la seguridad del sistema.		
CE6.4	Proyecta diseña y evalúa una implementación de un sistema operativo distribuido para una arquitectura propuesta.		

Aunque la metodología se centra en la aplicación práctica, también es importante que los estudiantes comprendan los fundamentos teóricos. Los exámenes teóricos-prácticos evalúan el conocimiento conceptual de los estudiantes en áreas como paradigmas, teoría de bases de datos, el diseño de esquemas y la normalización, se realizan de manera automatizada, dos instancias de evaluación.

Autoevaluación y Reflexión: Los estudiantes son alentados a realizar autoevaluaciones periódicas para reflexionar sobre su progreso y desarrollo de competencias. Esta reflexión personal les ayuda a identificar áreas de mejora y a tomar medidas para abordarlas. Para evaluar los trabajos arriba indicados se utilizarán rúbricas a fin de que el alumno pueda discernir los objetivos que alcanzó y en qué grado como una forma de retroalimentación. El rango de valoración de la rúbrica es de 1 a 3 u se corresponde a:
1. Insuficiente: No se evidencia el nivel de desarrollo de las competencias esperado a través de los resultado de aprendizaje
2. Suficiente: En la mayoría de las situaciones se evidencia el nivel de desarrollo deseado.
3. Alto: Se evidencia un claro desarrollo de las competencias esperado a través de los resultados de aprendizaje.

La calificación final se compone ponderando los resultados de las actividades prácticas
(40%) y el resultado de la rúbrica (60%)

Condiciones de aprobación

Para regularizar y promoción se exige un 80% de asistencia y participación en clase. Para regularizar, además, se exige un 40% de la calificación global según los criterios de evaluación. Esto incluye la entrega de la totalidad de los prácticos durante el cursado.
Para promoción, se exige un 70% de calificación.

Actividades prácticas y de laboratorio

Además de las actividades indicadas en el punto Evaluación se realizan actividades prácticas en conjunto con el dictado de clases resolviendo problemas de acuerdo a los contenidos que se van desarrollando.

Resultados de aprendizaje

- Interpreta correctamente el dominio de un problema y diseña soluciones acordes y es capaz de establecer procedimientos de trabajo acordes
- Se comunica de manera efectiva con sus pares y con el docente, utiliza las herramientas de comunicación adecuadas y respeta los procedimientos de trabajo establecidos para el proyecto.
- Cumple en tiempo con los compromisos asumidos con su equipo de trabajo.
- Estudia, prepara y expone un tema pequeño y novedoso, frente a sus pares resaltando los aspectos fundamentales del tema, citando las referencias utilizadas y extrayendo conclusiones propias.
- Diseña una solución eficiente adecuada a un problema de sensado distribuido y determina las fortalezas y debilidades del diseño.
- Diseña una red para un sistema distribuido eligiendo los protocolos adecuados y evalúa la performance de su implementación y evalúa la seguridad del sistema.
- Proyecta diseña y evalúa una implementación de un sistema operativo distribuido para una arquitectura propuesta.

Bibliografía

Coulouris,G (2012). DISTRIBUTED SYSTEMS,Concepts and Design (Fifth Edition). Addison-Wesley
Lynch, N (1996). Distributed Algorithms, O'Reilly
D. Gollmann, The Cyber Security Body of Knowledge. University of Bristol, 2021, https://www.cybok.org/
C. Cachin, R. Guerraoui, and L. Rodrigues, Introduction to Reliable and Secure Distributed Programming. Springer, 2011.
Sukumar G.(2014) Distributed Systems: An Algorithmic Approach.
Cachin, C. Introduction to Reliable and Secure Distributed Programming (Second Edition)

FCEFyN
FACULTAD DE CIENCIAS EXACTAS, FÍSICAS y NATURALES

UNC Universidad Nacional de Córdoba

Asignatura: **Sistemas Embebidos**

Código:	RTF	5
Semestre: 7	Carga Horaria	80
Bloque: Tecnologías Aplicadas	Horas de Práctica	40
Departamento: Computación		

Correlativas:
- Electrónica Digital 3

Contenido Sintético:
- Visión histórica de los sistemas embebidos y de tiempo real
- Aplicar estándares, herramientas relevantes y criterios de ingeniería.
- Interfaz de entrada / salida y comunicación
- Técnicas para el funcionamiento de baja potencia
- Sistemas integrados móviles y en red
- Subsistemas periféricos
- Estrategias de Implementación para Sistemas Embebidos Complejos
- Problemas avanzados de entrada / salidas
- Integración, pruebas y validación de sistemas embebidos
- Mantenimiento, sostenibilidad, manufacturabilidad de sistemas embebidos

Competencias Genéricas:
- CG1: Identificar, formular y resolver problemas de ingeniería.
- CG2: Concebir, diseñar y desarrollar proyectos de ingeniería (sistemas, componentes, productos o procesos).
- CG6: Desempeñarse de manera efectiva en equipos de trabajo.

Aprobado por HCD: NNNN-HCD-AAAA RES: Fecha: DD/MM/AAAA

Competencias Específicas:

CE4.3: Analizar, diseñar, programar, implementar, probar, depurar y evaluar hardware y software para sistemas de computación de propósitos específicos. (M)
CE4.4: Analizar, diseñar, programar, implementar, probar, depurar y evaluar hardware y software para sistemas de computación de propósitos generales
CE4.5 Analizar, diseñar, implementar y probar sistemas embebidos y su software asociado. (A)
CE6.3 Proyectar, desarrollar, dirigir, controlar, construir, operar y mantener sistemas operativos embebidos, para dispositivos móviles y de tiempo real. (A)
CE7.2 Proyectar, desarrollar, dirigir, controlar, construir, operar y mantener Sistemas Embebidos, sus periféricos y software de soporte. (A)
CE7.2.6 Diseñar circuitos impresos que integren componentes electrónicos y periféricos en un sistema embebido. (A)
CE7.2.7 Comprender, configurar y utilizar los protocolos de comunicación utilizados en sistemas embebidos. (A)

Presentación

Esta asignatura se centra en los sistemas embebidos, los elementos que se utilizan para su construcción y los conocimientos necesarios para comprender su funcionamiento, afrontar su diseño y tomar decisiones sobre su implementación.

La asignatura comenzará con una introducción dónde se presentan las características fundamentales de los sistemas embebidos, explicando los elementos principales que los forman, las métricas más importantes que afectan a su desarrollo, así como las distintas posibilidades de abordar la implementación de su hardware de cómputo. Tras la introducción, se expondrán los distintos niveles de diseño de un sistema embebido, así como las técnicas empleadas, tanto de diseño como de validación, que permiten mejorar la productividad y la calidad de los sistemas desarrollados. Entre las técnicas explicadas, se describe, mediante un caso de uso, la técnica de co-diseño hardware-software, con el fin de que el alumno pueda conocer los retos y beneficios que supone la utilización de la misma en el desarrollo de sistemas empotrados.

A continuación se definirá el concepto de arquitectura de un sistema embebido, y se describirán los elementos más característicos que se emplean en las estructuras que forman parte de la arquitectura, así como la funcionalidad que proporcionan, justificando su idoneidad para formar parte de este tipo de sistemas.

Finalmente, se estudiará el proceso de desarrollo de los sistemas embebidos, definiendo el ciclo de vida que permite su especificación y validación, así como la posterior implementación y despliegue del sistema. Se hará, además, una breve introducción sobre algunas herramientas que facilitan el proceso. Paralelamente a la exposición teórica, el alumno llevará a cabo prácticas que refuercen los contenidos teóricos. En este sentido tendrá que ser capaz de desarrollar, desplegar y depurar software sobre el hardware propio de los sistemas embebidos, así como familiarizarse con la simulación un sistema embebido que implique aspectos hardware y software. Para las prácticas se utilizará primeramente el hardware básico de un sistema embebido, implementando drivers sencillos que permitan una comunicación serie, la gestión de las interrupciones y los servicios básicos de temporización.

Posteriormente se utilizarán servicios más avanzados, que faciliten la programación de funciones en el tiempo sobre un ejecutivo cíclico. Finalmente, el alumno realizará prácticas sobre sistemas multitarea, familiarizándose con la utilización de servicios de temporización, mecanismos de protección de acceso a

recursos compartidos, y mecanismos de comunicación, tanto síncrona como asíncrona, entre esas tareas.

Contenidos

Unidad 1. Introducción a los sistemas embebidos
Visión histórica. Sistemas embebidos y de tiempo real. Fundamentos de los sistemas embebidos. Estándares, herramientas relevantes y criterios de ingeniería. Interfaz de entrada/salida. Comunicación serie entre una PC y un sistema embebido. Gestión del tiempo y señales analógicas. Técnicas para el funcionamiento de baja potencia.

Unidad 2. Técnicas de diseño y validación de sistemas embebidos
Estrategias de Implementación para Sistemas Embebidos Complejos. Máquinas de estado finato y Real-Time Clock. Modularización aplicada a los sistemas embebidos. procesos de diseño e implementación de un sistema embebido. Elicitación de requerimientos del proyecto y casos de uso. Diseño del hardware. Diseño del software. Implementación de la interfaz de usuario. Implementación de lectura de sensores. Control de actuadores. Implementación del comportamiento del sistema. Verificación del comportamiento del sistema. Documentación del Sistema

Unidad 3. Arquitectura de un sistema embebido
Sistemas integrados móviles y en red. Subsistemas periféricos. Estrategias de Implementación para Sistemas Embebidos Complejos. Problemas avanzados de entrada/salidas. Comunicación con otros sistemas embebidos, comunicación con sistemas más complejos. Comunicaciones interplacas, comunicaciones de corta distancia y larga distancia.

Unidad 4. Sistemas operativos en sistemas embebidos y seguridad
Cargadores de arranque, Sistemas operativos genéricos, sistemas de tiempo real, planificación de tareas y comunicación entre tareas. Gestión de memoria y drivers de dispositivos seguridad en sistemas embebidos.

Unidad 5. Proceso y entornos de desarrollo de los sistemas embebidos
Integración continua y devops, pruebas y validación de sistemas embebidos. Mantenimiento, sostenibilidad, manufacturabilidad de sistemas embebidos. Actualizaciones de manera remota.

Metodología

El desarrollo general de la materia se sustenta en clases teórico-prácticas. Por ello las estrategias de enseñanza que hemos seleccionado para llevar adelante nuestra propuesta son la exposición dialogada, el estudio de casos y la resolución de problemas. Cada unidad se desarrolla a partir de un material bibliográfico obligatorio. A su vez se ofrecerán trabajos prácticos que orientan el proceso de lectura y análisis del contenido como forma de evaluación y acreditación de cada unidad. Los trabajos prácticos estarán guiados por los casos de estudio presentados en cada unidad y se orientan a resolver problemas concretos aportando los contenidos teóricos necesarios para su resolución.

En ambientes de laboratorio el estudiante podrá realizar casos de estudio, verificar su implementación con el cumplimientos de los requerimientos funcionales a nivel software y componente de hardware.

Evaluación

La evaluación está inspirada en los criterios de evaluación continua. Se valora el desarrollo de las competencias durante todo el proceso de aprendizaje de la asignatura mediante una serie de pruebas de carácter sumativo distribuidas a lo largo del curso, que permiten al estudiante abordar la asignatura de forma progresiva. Se garantiza la retroalimentación temprana en el proceso de aprendizaje del alumno y permite a los profesores hacer un seguimiento global,

con la posibilidad de actuar en caso de que lo aconsejen indicadores o situaciones determinadas. La evaluación de la parte relacionada con las prácticas se realizará al finalizar la actividad de laboratorio correspondiente.

Para la evaluación de la asignatura se utilizará la rúbrica, compuesta por los siguientes criterios, relacionados con los resultados del aprendizaje:

RA1. Enumerar las características específicas de los sistemas embebidos y describir cómo éstas afectan al su proceso de desarrollo.

RA2. Enumerar las distintas métricas de diseño de los sistemas embebidos.

RA3. Describir el papel de la unidad de procesamiento en el contexto de un sistema completo con E/S y memoria.

RA4. Describir cómo es la comunicación de la unidad de procesamiento con el mundo exterior por medio de los periféricos.

RA5. Identificar las alternativas de implementación de un sistema embebido y ser capaz de seleccionar la más adecuada acorde a las métricas de diseño marcadas como objetivo.

RA6. Identificar los distintos niveles de diseño de un sistema embebido y describir las técnicas utilizadas para desarrollarlos.

RA7. Definir el concepto de arquitectura de los sistemas embebidos y describir el proceso seguido para su especificación y validación.

RA8. Ser capaz de explicar el diseño de un sistema embebido.

RA9. Ser capaz de programar un sistema embebido y describir cómo el lenguaje de alto nivel se convierte en código ejecutable.

RA10. Ser capaz de emplear compiladores y entornos de desarrollo propios de los sistemas embebidos.

Instrumentos de evaluación

El rendimiento de los alumnos será valorado mediante la toma de tres exámenes parciales con un recuperatorio y evaluación formativa. Los parciales buscan evidenciar los conocimientos y competencias adquiridas a través de presentación de casos prácticos (análisis, resolución de problemas, etc.) y/o respuestas a preguntas conceptuales. Constituyen en sí mismos una instancia de evaluación formativa ya que luego los estudiantes reciben una realimentación de los errores cometidos, además de resolver los temas del parcial en la clase siguiente.

Cada Actividad Práctica o de Laboratorio grupal propuesta (ver más adelante) demanda la presentación de prototipos funcionales e informes, donde los estudiantes explican individualmente las diferentes partes que componen el prototipo (criterios de diseño, fuentes consultadas, etc.), y particularmente sus detalles constructivos. Es el momento donde el estudiante pone en juego su participación en el equipo, el rol y peso de sus decisiones, su capacidad de comunicar detalles de diseño, y su correcto manejo y comprensión de las decisiones tecnológicas presentes en el prototipo. Para la entrega que demande

cada prototipo el equipo de cátedra genera un repositorio virtual con control de versiones.

Condiciones de aprobación

Condiciones de regularización

- Asistir al 80% de las clases.
- Aprobar todas las actividades prácticas de laboratorio con al menos el 60% o más de los criterios de evaluación expresados en la sección anterior.
- Aprobar los resultados de aprendizaje, con el 60% o más.
- Aprobar al menos dos parciales, incluido un recuperatorio, con el 60% o más de los contenidos evaluados.

Condiciones de aprobación por promoción (no requiere examen final)

- Cumplir con todas las condiciones de regularización.
- Aprobar cada uno de los tres parciales propuestos, o un recuperatorio, con el 60% o más de los contenidos evaluados.

Condiciones de aprobación por examen final

- Todas las condiciones de regularización expuestas anteriormente.
- Aprobación de un examen final con el 60% o más de los contenidos evaluados.

Para la nota final se promedian las notas obtenidas en cada una de las actividades prácticas y de laboratorio, y el resultado se promedia con las notas de los parciales.

Actividades prácticas y de laboratorio

Se propone la realización de 5 (cinco) actividades donde se lleven a la práctica los conocimientos adquiridos en la asignatura y se desarrollen las competencias esperadas. Estos trabajos se realizan en grupos lo que permitirá que desarrollen competencias de trabajo en equipo y coordinación de tareas.

APL1 – Cross Compilation, toolchain y testing en sistemas embebidos
En este trabajo los estudiantes aplican los conceptos de introducción a los sistemas embebidos en hardware real y hardware virtualizado. Especificación de requerimientos, pruebas unitarias, de integración y de sistema utilizando mocks y stubs.

APL2 – Sistemas operativos de tiempo real: En este trabajo los estudiantes aplican los conceptos básicos de conmutación de tareas y organización de memoria. Continúan mejorando sus herramientas de depuración y profiling para medir tiempos de latencia de una aplicación simple en un microcontrolador y lo comparan con un microprocesador con un sistema operativo de propósito general.

APL3 – Programación de drivers para sistemas embebidos: Todo componente electrónico que se conecte a una computadora necesita de un driver para interactuar directamente con el sistema operativo. En este trabajo práctico se compilara un bootloader, un kernel y luego se desarrolla un driver sencillo para un

sistema operativo de propósito general que adquiera datos de un sensor y comunique los datos por algún canal como bluetooth o zigbee.

APL4 – API Rest y su integración con lenguajes de bajo nivel en hardware virtualizado: En este trabajo los estudiantes desarrollan actividades que les permitirán aplicar los conocimientos de redes e integrar el resto de los conceptos aprendidos en los trabajos anteriores. Desarrollan un servidor simple en un microcontrolador virtualizado y otro en un microprocesador de propósitos generales virtualizado.

APL5 – Implementación de un sistema de computación complejo: En este trabajo práctico de laboratorio los estudiantes diseñan e implementan un sistema de computación complejo que incluye la adquisición, almacenamiento, transmisión y visualización de datos distribuidos. Para ejercitar e integrar los conocimientos adquiridos, se realizan ejercicios con diferentes herramientas para cada una de las funcionalidades, se acompaña y guía a los estudiantes en el proceso de diseño e implementación. Finalmente deben presentar una prueba de concepto funcional junto a la documentación del proyecto y los casos de prueba diseñados, implementados y ejecutados.

Resultados de aprendizaje

Al terminar con éxito esta asignatura/enseñanza, los estudiantes serán capaces de:

RA1. Enumerar las características específicas de los sistemas embebidos y describir cómo éstas afectan al su proceso de desarrollo.

RA2. Enumerar las distintas métricas de diseño de los sistemas embebidos.

RA3. Describir el papel de la unidad de procesamiento en el contexto de un sistema completo con E/S y memoria.

RA4. Describir cómo es la comunicación de la unidad de procesamiento con el mundo exterior por medio de los periféricos.

RA5. Identificar las alternativas de implementación de un sistema embebido y ser capaz de seleccionar la más adecuada acorde a las métricas de diseño marcadas como objetivo.

RA6. Identificar los distintos niveles de diseño de un sistema embebido y describir las técnicas utilizadas para desarrollarlos.

RA7. Definir el concepto de arquitectura de los sistemas embebidos y describir el proceso seguido para su especificación y validación.

RA8. Ser capaz de explicar el diseño de un sistema embebido.

RA9. Ser capaz de programar un sistema embebido y describir cómo el lenguaje de alto nivel se convierte en código ejecutable.

RA10. Ser capaz de emplear compiladores y entornos de desarrollo propios de los sistemas embebidos.

RA11. Interactuar en equipos variados, con objetivos claros, utilizando herramientas acordes para la modalidad de trabajo acordado, logrando comunicación efectiva y concreción de resultados

Desagregación de Competencias Específicas	Resultados de Aprendizaje
CE4.5.A Analizar sistemas embebidos y su software asociado.	RA1,RA2
CE4.5.B Diseñar sistemas embebidos y su software asociado.	RA1, RA3,RA4,RA5
CE4.5.C Implementar sistemas embebidos y su software asociado.	RA1, RA2, RA9
CE4.5.D Probar sistemas embebidos y su software asociado.	RA2
CE7.2.A Proyectar sistemas embebidos, sus periféricos y software de soporte.	RA3, RA4, RA5, RA6
CE7.2.B Desarrollar sistemas embebidos, sus periféricos y software de soporte.	RA3, RA4
CE7.2.C Dirigir proyectos de diseño de sistemas embebidos, sus periféricos y software de soporte.	RA4,RA7
CE7.2.D Controlar sistemas embebidos, sus periféricos y software de soporte.	RA4
CE7.2.E Construir sistemas embebidos, sus periféricos y software de soporte.	RA4, RA8, RA9
CE7.2.F Operar sistemas embebidos, sus periféricos y software de soporte.	RA7
CE7.2.G Mantener sistemas embebidos, sus periféricos y software de soporte.	RA7,RA8
CG6: Desempeñarse de manera efectiva en equipos de trabajo.	RA11

Tabla 1: Desagregación de competencias y resultados de aprendizaje

Bibliografía

- Edward A. Lee and Sanjit A. Seshia, Introduction to Embedded Systems, A Cyber-Physical Systems Approach, Second Edition, MIT Press, ISBN 978-0-262-53381-2, 2017.
- R. Barry, Mastering the FreeRTOS™ Real Time Kernel, 2016
- M. VanderVoord, Embedded Testing with Unity and CMock
- Beningo Jacob.; Reusable Firmware Development, A Practical Approach to APIs, HALs and Drivers; editorial apress; ISBN-13 (electronic): 978-1-4842-3297-2; 2017.-

FCEFyN
FACULTAD DE CIENCIAS EXACTAS, FÍSICAS y NATURALES

UNC Universidad Nacional de Córdoba

Asignatura: **Sistemas Informáticos**

Código:	RTF	8
Semestre: 9	Carga Horaria	80
Bloque: Tecnologías Aplicadas	Horas de Práctica	48
Departamento: Computación		

Correlativas:
- Seguridad informática

Contenido Sintético:
- Gestión de la configuración.
- Técnicas para la escalabilidad e interoperabilidad.
- Técnicas para la tolerancia a fallas y robustez.
- Gestión de la infraestructura.
- Gestión del proceso de construcción y despliegue.
- Arquitecturas de grandes datos.
- Licenciamiento y distribución de software.

Competencias Genéricas:
- CG1: Identificar, formular y resolver problemas de ingeniería.
- CG2: Concebir, diseñar y desarrollar proyectos de ingeniería (sistemas, componentes, productos o procesos).
- CG3: Gestionar -planificar, ejecutar y controlar- proyectos de ingeniería (sistemas, componentes, productos o procesos).
- CG4: Utilizar de manera efectiva las técnicas y herramientas de aplicación en ingeniería.

● CG5: Contribuir a la generación de desarrollos tecnológicos y/o innovaciones tecnológicas. ● CG10: Actuar con espíritu emprendedor.
Aprobado por HCD: NNNN-HCD-AAAA RES: Fecha: DD/MM/AAAA

Competencias Específicas:

- CE1.4 Implementar y mantener sistemas informáticos.

Presentación

Sistemas Informáticos es una asignatura del último año (noveno semestre) de la carrera de Ingeniería en Computación. Dado el esquema de correlatividades, al momento de transitar esta asignatura los estudiantes han cursado todas las asignaturas fundamentales relativas a la programación (algoritmos y estructuras de datos), así como también aquellas centradas en los aspectos de la ingeniería de software, el aseguramiento de su calidad, y la seguridad de los sistemas informáticos. Esta asignatura complementa la formación antes mencionada incorporando dimensiones asociadas a la gestión del proceso de desarrollo de sistemas, el diseño de los mismos con la mira puesta en la integración con otros sistemas, su eficiencia computacional, su robustez, y en su despliegue en entornos productivos.

En este sentido, el objetivo del curso es facilitar herramientas tanto conceptuales como tecnológicas para dar cuenta de:

- Una gestión adecuada de los artefactos que componen un sistema informático y su proceso de desarrollo y construcción.
- Una gestión eficiente de la infraestructura para la ejecución de sistemas y el despliegue de sistemas en la misma.
- El diseño de sistemas informáticos que se ejecutan de forma eficiente, robusta y escalable frente a una demanda creciente y fallas externas.
- El diseño de sistemas informáticos que se integran de forma eficaz con otros sistemas.
- Las características y tecnologías necesarias para el desarrollo de sistemas que manejan grandes volúmenes de datos.
- Las estrategias y formas de licenciamiento para la distribución de sistemas informáticos.

Esta asignatura corona los conocimientos necesarios para desarrollar sistemas informáticos que se implementan en entornos productivos realistas. De esta manera, contribuye a desarrollar competencias fundamentales para que los estudiantes puedan desempeñarse efectivamente en el ámbito laboral relacionado al desarrollo de sistemas informáticos.

Teniendo en cuenta que las dimensiones abordadas en esta asignatura están relacionadas a un ejercicio de la profesión que suele desarrollarse en equipo, la asignatura propone casos de estudio realistas y trabajos prácticos a desarrollar en grupo, donde la discusión, la formulación de soluciones acordadas y la gestión de las mismas en equipo es parte intrínseca de las competencias que se espera desarrollar.

Contenidos

Unidad 1: Gestión de la Configuración
Introducción a la gestión de la configuración de sistemas informáticos. Conceptos básicos de control de versiones. Herramientas de control de versiones. Administración de cambios y versionado de software. Prácticas recomendadas en la gestión de configuración.

Unidad 2: Técnicas para la Escalabilidad e Interoperabilidad
Escalabilidad de sistemas informáticos: conceptos y estrategias. Interoperabilidad entre sistemas y aplicaciones. Uso de estándares y protocolos para mejorar la interoperabilidad.

Técnicas de diseño orientado a servicios. Prácticas recomendadas para la escalabilidad e interoperabilidad.

Unidad 3: Técnicas para la Tolerancia a Fallas y Robustez
Tolerancia a fallas en sistemas informáticos. Estrategias de redundancia y alta disponibilidad. Diseño de sistemas robustos y resistentes. Pruebas de resistencia y recuperación ante desastres. Monitoreo y detección de fallas en tiempo real.

Unidad 4: Gestión de la Infraestructura
Automatización de la infraestructura. Despliegue y gestión de máquinas virtuales y contenedores. Administración de recursos en la nube. Orquestación de servicios y aplicaciones en la nube. Políticas de seguridad y acceso a la infraestructura.

Unidad 5: Gestión del Proceso de Construcción y Despliegue
Introducción a la integración continua (CI) y entrega continua (CD). Automatización de pruebas de software. Despliegue automatizado en diferentes entornos (desarrollo, prueba, producción). Estrategias de rollback y rollforward. Herramientas para CI/CD.

Unidad 6: Arquitecturas de Grandes Datos
Fundamentos de arquitecturas de grandes datos. Tecnologías y plataformas para el procesamiento de grandes volúmenes de datos. Almacenamiento y análisis distribuido de datos. Desafíos y consideraciones de seguridad en arquitecturas de grandes datos.

Unidad 7: Licenciamiento y Distribución de Software
Conceptos de licenciamiento de software. Tipos de licencias y su impacto en la distribución. Estrategias de licenciamiento abierto y comercial. Cumplimiento de licencias y gestión de activos de software. Tendencias actuales en licenciamiento de software.

Metodología de enseñanza

El curso se estructura en base a dos clases semanales de naturaleza teórico-práctica, cada una con una duración de 2,5 horas. La metodología de enseñanza propuesta se apoya en el Aprendizaje Basado en Problemas (ABP), el análisis de casos prácticos realistas, y la utilización del aula virtual como recurso esencial complementario a las clases presenciales.

Los estudiantes trabajan a partir de los conceptos y herramientas presentados por el docente en el aula y los materiales disponibles en el entorno virtual de aprendizaje, que incluyen videos, lecturas y ejercicios prácticos, abarcando tanto aspectos teóricos como ejemplos concretos. El aula virtual facilita el acceso a estos recursos y permite a los alumnos establecer un ritmo de aprendizaje personalizado, independiente del que se imparte en las sesiones presenciales.

En las clases presenciales, el docente introduce los conceptos teóricos y prácticos clave, así como coordina la exploración de casos de estudio que presentan diferentes problemáticas. Esta actividad permite a los estudiantes aplicar y contextualizar los conceptos previamente aprendidos. Además de los casos, el docente plantea una serie de preguntas conceptuales

que fomentan la discusión y el análisis de posibles soluciones entre los estudiantes. El objetivo es profundizar en los conceptos fundamentales, ayudando a los alumnos a comprender la aplicación adecuada de soluciones y a identificar y debatir errores comunes.

Durante las clases presenciales, se brinda orientación y se resuelven dudas relacionadas con los trabajos prácticos, abordando los problemas específicos planteados por estos. Los estudiantes trabajan en grupos para discutir y buscar soluciones, con el docente ejerciendo como guía y facilitador. Al finalizar cada trabajo práctico, los grupos presentan sus soluciones, generando así un espacio de retroalimentación y discusión colectiva que enriquece el proceso de aprendizaje a través del intercambio de experiencias.

Además de las clases semanales, se fomenta el debate en torno a conceptos y problemas de los trabajos prácticos mediante un foro disponible en el aula virtual. Este espacio permite una comunicación continua pero asincrónica entre docentes y estudiantes, promoviendo un aprendizaje colaborativo y constante.

Evaluación

La evaluación se basa en un enfoque continuo e integrador que refleja su naturaleza progresiva. El propósito de esta metodología de evaluación es no solo medir el conocimiento adquirido, sino también fomentar una comprensión profunda y aplicada de los conceptos y habilidades aprendidos. En este contexto, la evaluación se lleva a cabo a lo largo del período del curso mediante tres evaluaciones parciales presenciales y cuatro trabajos prácticos.

Cada evaluación parcial consta de preguntas teóricas y análisis de casos prácticos, lo que garantiza una evaluación integral de los conocimientos y habilidades adquiridos hasta ese momento. Está prevista una evaluación hacia la mitad del curso y otra hacia la finalización del mismo. La tercera evaluación parcial, también prevista hacia el final del curso, sirve como instancia de recuperación para cualquiera de las dos evaluaciones parciales anteriores.

Los resultados principales de los trabajos prácticos se presentan de manera oral por parte de los grupos de estudiantes durante las clases, tal como se menciona en el apartado anterior. Además, los estudiantes deben entregar los trabajos prácticos a los docentes para su evaluación detallada. En caso de ser necesario, los trabajos prácticos pueden ser corregidos y reenviados para su revisión.

Condiciones de aprobación

Condiciones de regularización

Para alcanzar la condición de regular, el estudiante debe cumplir las siguientes condiciones:

1. Asistir al menos al 80% de las clases.
2. Alcanzar un rendimiento igual o superior a 60% en cada uno de los Trabajos Prácticos (TP).
3. Alcanzar un rendimiento igual o superior a 60% en dos de las Evaluaciones Parciales (EP).
4. Alcanzar el criterio de aceptación mínimo para los resultados de aprendizaje propuestos.

Condiciones de promoción
Para alcanzar la promoción, el alumno debe cumplir las siguientes condiciones:

1. Asistir al menos al 80% de las clases.
2. Alcanzar un rendimiento igual o superior a 80% en cada uno de los Trabajos Prácticos (TP).
3. Alcanzar un rendimiento igual o superior a 80% en al menos dos de las Evaluaciones Parciales (EP).
4. Alcanzar el criterio de aceptación mínimo para los resultados de aprendizaje propuestos.

Cumplidas estas condiciones, la nota final de promoción se calculará según la siguiente fórmula:

Calificación = redondear((0,5 * TPs + 0,5 * EPs) / 10)

donde TPs es el resultado de promediar el rendimiento de todos los trabajos prácticos, y EPs es el resultado de promediar el rendimiento de las dos evaluaciones parciales con mayor rendimiento.

Examen final
Los estudiantes regulares rinden un examen escrito individual con características similares a las evaluaciones parciales. La calificación final es el rendimiento obtenido en el examen.

Los estudiantes libres rinden en dos etapas. En la primera, presentan y defienden de forma oral los trabajos prácticos. Superada esta etapa, rinden un examen escrito individual equivalente a los estudiantes regulares.

Actividades prácticas y de laboratorio

Las actividades prácticas se desarrollan a partir de trabajos prácticos, que se presentan durante las clases presenciales, y cuyos recursos se disponibilizan en el aula virtual. Los trabajos prácticos abordan principalmente temas de diferentes unidades conceptuales:

- TP1: unidad 1
- TP2: unidades 2 y 3
- TP4: unidades 4 y 5
- TP5: unidades 6 y 7

Sin embargo, algunos aspectos de las unidades son transversales y se integran a todos los trabajos prácticos.

Los trabajos prácticos plantean situaciones y problemáticas a resolver en torno al desarrollo de un proyecto de software. Se espera que se simule un equipo de desarrollo y por lo tanto, los trabajos prácticos se diseñan para que sean trabajados en grupos.

Desagregado de competencias y resultados de aprendizaje

Los siguientes son las competencias esperables como resultado del aprendizaje, organizadas de acuerdo a las unidades de contenido:

1. Conocer los fundamentos de la gestión de la configuración en sistemas informáticos.
2. Comprender y aplicar conceptos básicos de control de versiones.
3. Utilizar herramientas de control de versiones de manera efectiva.
4. Evaluar y comparar diferentes enfoques de administración de cambios y versionado de software
5. Diseñar un sistema de gestión de configuración basado en prácticas recomendadas.

6. Definir y aplicar estrategias de escalabilidad en sistemas informáticos.
7. Diseñar sistemas orientados a servicios para mejorar la escalabilidad e interoperabilidad.
8. Aplicar estándares y protocolos para mejorar la interoperabilidad.
9. Aplicar técnicas de diseño orientado a servicios.
10. Identificar y aplicar prácticas recomendadas para la escalabilidad y la interoperabilidad.

11. Comprender el concepto de tolerancia a fallas en sistemas informáticos.
12. Diseñar estrategias de redundancia y alta disponibilidad.
13. Desarrollar sistemas robustos y resistentes a fallas.
14. Realizar pruebas de resistencia y planificar la recuperación ante desastres.
15. Implementar sistemas de monitoreo y detección de fallas en tiempo real.

16. Automatizar tareas relacionadas con la infraestructura de sistemas informáticos
17. Desplegar y administrar máquinas virtuales y contenedores de manera eficiente.
18. Gestionar recursos en entornos de nube de forma efectiva.
19. Orquestar servicios y aplicaciones en la nube.
20. Establecer políticas de seguridad y acceso para la infraestructura.

21. Comprender los conceptos de integración continua (CI) y entrega continua (CD).
22. Automatizar pruebas de software para garantizar la calidad del código.
23. Desplegar de manera automatizada en diferentes entornos.
24. Utilizar herramientas de CI/CD para optimizar el proceso de construcción y despliegue.

25. Comprender los fundamentos de las arquitecturas de grandes datos.
26. Evaluar y seleccionar tecnologías y plataformas para el procesamiento de grandes volúmenes de datos.
27. Diseñar soluciones de almacenamiento y análisis distribuido de datos.
28. Abordar desafíos y consideraciones de seguridad en arquitecturas de grandes datos.
29. Aplicar prácticas recomendadas en el manejo de grandes conjuntos de datos.

30. Comprender los conceptos fundamentales de licenciamiento de software.
31. Analizar los tipos de licencias y su impacto en la distribución de software.
32. Evaluar estrategias de licenciamiento tanto en el ámbito abierto como comercial.
33. Estar al tanto de las tendencias actuales en licenciamiento de software.

La siguiente matriz desagrega las competencias genéricas y específicas en los resultados anteriores:

	CG1	CG2	CG3	CG4	CG5	CG10	CE1.4
1			x	x		x	
2			x	x		x	
3			x	x		x	
4			x	x		x	
5			x	x		x	
6	x	x		x	x	x	x
7	x	x		x	x	x	x
8	x	x		x	x	x	x
9	x	x		x	x	x	x
10	x	x		x	x	x	x
11	x	x		x	x	x	x
12	x	x		x	x	x	x
13	x	x		x	x	x	x
14	x	x		x	x	x	x
15	x	x		x	x	x	x
16	x		x	x		x	
17	x		x	x		x	
18	x		x	x		x	
19	x		x	x		x	
20	x		x	x		x	
21	x		x	x		x	
22	x		x	x		x	
23	x		x	x		x	
24	x		x	x		x	
25	x	x		x	x	x	x

26	x	x		x	x	x	x
27	x	x		x	x	x	x
28	x	x		x	x	x	x
29	x	x		x	x	x	x
30					x	x	
31					x	x	
32					x	x	
33					x	x	
34					x	x	

Bibliografía

- Chacon, S., & Straub, B. (2014). Pro Git (2a. ed.). [Berkeley, California]: Apress.
- Nygard, M. T. (2007). Release It! Design and Deploy Production-Ready Software. Raleigh, NC: Pragmatic Bookshelf. ISBN: 978-0-9787-3921-8
- Humble, J., Farley, D. G. (2010). Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation. Upper Saddle River, NJ: Addison-Wesley. ISBN: 978-0-321-60191-9
- Kim, G., Humble, J., Debois, P., Willis, J., & Allspaw, J.. (2016). The devOps handbook :how to create world-class agility, reliability, and security in technology organizations . Portland, OR :IT Revolution Press
- Jacobson, D., Brail, G., & Woods, D. (2011). APIs: A Strategy Guide.
- Marz, N., Warren, J. (2015). Big data: principles and best practices of scalable real-time data systems. Manning. ISBN: 9781617290343 1617290343
- Stallman R. & Free Software Foundation (Cambridge Mass). (2002). Free software free society : selected essays of Richard M. Stallman. Free Software Foundation.

FCEFyN
FACULTAD DE CIENCIAS EXACTAS, FÍSICAS y NATURALES

UNC Universidad Nacional de Córdoba

Asignatura: **Sistemas Operativos**

Código:	RTF	8
Semestre: 7	Carga Horaria	80
Bloque:TB-TA	Horas de Práctica	40
Departamento: Computación		

Correlativas:
- Programación Concurrente y Paralela

Contenido Sintético:
- Introducción, evolución y visión histórica de los sistemas operativos.
- Principios de diseño.
- Gestión y control de procesos.
- Gestión de recursos de hardware y software.
- Planificación: monoprocesador, multiprocesador.
- Gestión de memoria.
- Comunicación y sincronización entre procesos.
- Entrada / salida en el sistema operativo
- Gestión de archivos.
- Sistemas operativos para dispositivos móviles y de tiempo real.
- Virtualización.

Competencias Genéricas:
- CG2: Concebir, diseñar y desarrollar proyectos de ingeniería (sistemas, componentes, productos o procesos).
- CG3: Gestionar -planificar, ejecutar y controlar- proyectos de ingeniería (sistemas, componentes, productos o procesos).
- CG6: Desempeñarse de manera efectiva en equipos de trabajo.

Aprobado por HCD: NNNN-HCD-AAAA RES: Fecha: DD/MM/AAAA

Competencias Específicas:

- CE2.10 Manejar conceptos de sistemas operativos y su interacción con la arquitectura de computadoras., recursos del sistema, interrupciones y multiprogramación.
- CE6.1 Conocer y poder diseñar la estructura de sistemas operativos, la administración de procesos, la gestión de memoria, la administración de archivos, la gestión de dispositivos de entrada/salida y la implementación de políticas de seguridad.
- CE6.2 Analizar, conocer, interpretar y aplicar mecanismos de comunicación y sincronización.
- CE6.3 Proyectar, desarrollar, dirigir, controlar, construir, operar y mantener sistemas operativos embebidos, para dispositivos móviles y de tiempo real.

Presentación

Un sistema operativo es la pieza de software más utilizada en cualquier computadora. Un conocimiento profundo y con fundamentos sobre su funcionamiento es una condición esencial para el egresado de Ingeniería en Computación.
El curso apunta a enseñar los conceptos principales de los Sistemas Operativos y mostrar cuales son las diferentes opciones de diseño para la implementación de sus componentes.
A través del dictado de las clases teóricas se le explicará al alumno cada uno de los conceptos y las opciones de diseño disponibles para la implementación de los mismos, así como las diferentes implementaciones existentes en el mercado.
La materia hace especial énfasis en las actividades prácticas de programación. Aquí el alumno puede aplicar y desarrollar los conceptos aprendidos en el teórico y además practicar y experimentar con técnicas aceptadas de organización de proyectos de software, trabajo de programación realizado en equipo y control de versiones de software.

Contenidos

Unidad 1. Introducción a los sistemas operativos
Objetivos y funciones de los sistemas operativos. Evolución de los sistemas operativos. Logros principales.
Características de los sistemas operativos modernos.
Unidad 2. Descripción y control de procesos
Estados de un proceso. Descripción de procesos. Control de procesos.
Procesos e hilos. Multiproceso simétrico.
Unidad 3. Gestión de memoria
Requisitos para la gestión de memoria. Particionamiento de la memoria.
Paginación. Segmentación.
Unidad 4. Memoria Virtual
Hardware y estructuras de control.
Software del sistema operativo.
Unidad 5. Planificación de procesador
Tipos de planificación del procesador. Algoritmos de planificación uniprocesador. Algoritmos de planificación multiprocesador.
Unidad 6. Comunicación entre procesos
Necesidades y objetivos. Semáforos. Señales, FIFOS, Pipes. Colas de mensajes. Sockets.
Unidad 7. Entrada/Salida
Dispositivos de E/S. Organización del sistema de E/S. Planificación del disco. RAID. Caché de disco.
Unidad 8 Manejo de archivos
Directorios y estructuras de directorios. Archivos, organización y permisos. Gestión del almacenamiento. Métodos de asignación.
Unidad 9 Tiempo Real
Requerimientos de tiempo real. Métodos de planificación. Organización y manejo de memoria. Prioridades.

Unidad 10 Virtualización
Máquinas virtuales y contenedores. Virtualización de CPU, Memoria y comunicaciones Características del Hardware que soporta la virtualización.Centros de datos y Virtualización.

Metodología de enseñanza

El desarrollo de la materia se presentará al estudiante en clases teórico prácticas. Estas serán exposiciones dialogadas de los contenidos básicos, con búsqueda y discusión de ejemplos concretos en sistemas operativos modernos y análisis de problemas.

Partiendo de la bibliografía se plantean los primeros principios del tema para luego analizar casos de aplicación, ventajas y desventajas de diferentes enfoques resolutivos.
Se combinan además los desarrollos prácticos en grupo simulando actividades del mundo real, estos se aproximan utilizando las metodologías y herramientas de desarrollo empleadas en la industria para el trabajo en equipo.

Evaluación

La evaluación de los conocimientos adquiridos y las competencias desarrolladas se llevará a cabo con los siguientes métodos.

- El desarrollo y la entrega en tiempo y forma de las actividades prácticas propuestas.
- Evaluaciones parciales de enfoque práctico que permitan al alumno mostrar la construcción individual de conocimientos y técnicas.
- Instancia de discusión oral o coloquio.
- El completar las actividades propuestas es indicativo de haber alcanzado un nivel de desarrollo aceptable en las competencias propuestas.

La evaluación se llevará a cabo mediante la utilización de rúbricas basadas en los resultados de aprendizaje.

Condiciones de aprobación

Condiciones para la regularidad:

- 80% de asistencia a clase
- Aprobación de las 2 evaluaciones parciales incluida recuperatorio sobre una de ellas
- Aprobación de las actividades prácticas
-

Condiciones para la promoción:

- 80% de asistencia a clase
- Aprobación de las 2 evaluaciones parciales incluida recuperatorio sobre una de ellas
- Aprobación de las actividades prácticas
- Aprobación del coloquio oral

Calificación:
La calificación se obtendrá a través del siguiente polinomio:
CALIFICACIÓN= 0,5xP1 + 0,3xP2 + 0,2xP3
Donde:

P1: Es el promedio de las calificaciones de los exámenes parciales
P2: Es el promedio de la calificación de las actividades prácticas.
P3: Es la valoración numérica obtenida en el coloquio

Condiciones de regularidad

- 80% de asistencia a clase
- Aprobación de 1 evaluación parcial
- Aprobación de las actividades prácticas

Actividades prácticas y de laboratorio

1. Gnu Toolchain y Linux Software, Sentar las bases del diseño e implementación de librerías estáticas y dinámicas.
2. Myshell, Diseñar un intérprete de línea de comandos al estilo Bourne shell.
3. Comunicación entre procesos, mmap, mensajes, FIFO, sockets
4. Paralelismo y memoria compartida
5. Sistemas de tiempo real, desarrollo de aplicaciones de tiempo real

Resultados de aprendizaje

Al aprobar la materia el alumno será capaz de:

- Diseñar, ejecutar y controlar proyectos de software de complejidad media
- Proyectar, desarrollar, dirigir, controlar, construir, operar y mantener sistemas operativos embebidos, para dispositivos móviles y de tiempo real.
- Emplea en los trabajos propuestos las herramientas de desarrollo propuestas en la asignatura, integrando de manera efectiva los conceptos y técnicas aprendidas.
- Colaborar de manera efectiva en roles diversos dentro de equipos de trabajo durante las actividades propuestas.
- Demostrar habilidades de comunicación y colaboración en entornos colaborativos, contribuyendo al logro de metas comunes.
- Comprende y aplica los conceptos fundamentales de sistemas operativos, recursos del sistema y multiprogramación.
- Conocer la estructura de sistemas operativos, la administración de procesos, la gestión de memoria, la administración de archivos, la gestión de dispositivos de entrada/salida y la implementación de políticas de seguridad.
- Conocer y comprender los conceptos fundamentales de los sistemas operativos, recursos del sistema y multiprogramación .
- Analizar, conocer, interpretar y aplicar mecanismos de comunicación y sincronización.
- Desarrollar programas que aprovechen las funcionalidades provistas por los sistemas operativos modernos.

Bibliografía

- Operating Systems, Principles / Practice, Anderson y Dahlin segunda edición
- Operating Systems: Internals and Design Principles, Stallings, novena edición

Universidad Nacional de Córdoba
1983/2023 - 40 AÑOS DE DEMOCRACIA

Hoja Adicional de Firmas
Informe Gráfico

Número: IF-2023-01019967-UNC-EICO#FCEFYN

CORDOBA, CORDOBA
Martes 21 de Noviembre de 2023

Referencia: Programas IComp 285-25

El documento fue importado por el sistema GEDO con un total de 155 pagina/s.

Digitally signed by GDE UNC
DN: cn=GDE UNC, c=AR, o=Universidad Nacional de Cordoba, ou=Prosecretaria de Informatica, serialNumber=CUIT 30546670623
Date: 2023.11.21 23:12:21 -03'00'

Aldo Marcelo Algorry
Director de Escuela
Escuela de Ingeniería en Computación Facultad de Ciencias Exactas, Físicas y Naturales
Universidad Nacional de Córdoba

Digitally signed by GDE UNC
DN: cn=GDE UNC, c=AR, o=Universidad Nacional de Cordoba, ou=Prosecretaria de Informatica, serialNumber=CUIT 30546670623
Date: 2023.11.21 23:12:27 -03'00'