



Universidad Nacional de Ingeniería (UNI)

Escuela Profesional de
Ciencia de la Computación
Sílabo 2027-I

1. CURSO

CS3P2. Cloud Computing (Obligatorio)

2. INFORMACIÓN GENERAL

2.1 Curso	: CS3P2. Cloud Computing
2.2 Semestre	: 10 ^{mo} Semestre.
2.3 Créditos	: 3
2.4 Horas	: 1 HT; 4 HP;
2.5 Duración del periodo	: 16 semanas
2.6 Condición	: Obligatorio
2.7 Modalidad de aprendizaje	: Presencial
2.8 Prerrequisitos	: CS370. Big Data. (9 ^{no} Sem)

3. PROFESORES

Atención previa coordinación con el profesor

4. INTRODUCCIÓN AL CURSO

La computación moderna depende en gran medida de arquitecturas distribuidas y plataformas nativas de la nube. Este curso proporciona una exploración en profundidad de los servicios de computación en la nube, conceptos de bases de datos distribuidas y plataformas web. Los estudiantes aprenderán a diseñar arquitecturas escalables, comprender las implicaciones de la consistencia de datos distribuidos y evaluar los desafíos de privacidad y seguridad inherentes a los entornos basados en la nube.

5. OBJETIVOS

- Comprender la arquitectura y los modelos de despliegue de la computación en la nube.
- Diseñar soluciones de bases de datos distribuidas considerando consistencia y disponibilidad.
- Desarrollar aplicaciones utilizando plataformas y frameworks web modernos basados en la nube.
- Analizar las implicaciones de seguridad y privacidad en los servicios distribuidos y en la nube.

6. RESULTADOS DEL ESTUDIANTE

AG-C09) Diseño y Desarrollo de Soluciones: Diseña, implementa y evalúa soluciones para problemas complejos de computación. (Usage)

AG-C11) Uso de Herramientas: Aplica herramientas modernas de computación en la resolución de problemas. (Usage)

7. TEMAS

Unidad 1: Arquitectura Cloud y Gestión de Datos (12 horas)	
Resultados esperados: AG-C09,AG-C11	
Temas	Aprendizaje esperado (<i>Learning Outcomes</i>)
■ Plataformas, frameworks o meta-frameworks web ● Servicios en la nube ● API, Componentes Web (Web Components) ■ Servicios de computación Sistemas NoSQL ● Alojamiento en la Nube (Cloud Hosting) ● Escalabilidad (por ejemplo, Autoescalado (<i>Autoscaling</i>), Clústeres) ● Estimación de costos para servicios ■ Arquitectura ● Monolitos vs Microservicios ● Micro-frontends ● Arquitecturas Dirigidas por Eventos (Event-Driven) vs RESTful: ventajas y desventajas ● Sin servidor (Serverless), computación en la nube bajo demanda	■ Diseñar e implementar una aplicación basada en web utilizando un diseño de arquitectura de microservicios [Diseñar] ■ Describir las restricciones, como alojamiento, servicios y escalabilidad, relacionadas con las plataformas web [Describir]
Lecturas : [Marinescu2022], [Buyya2013]	

Unidad 2: Bases de Datos Distribuidas/Computación en la Nube (10 horas)	
Resultados esperados: AG-C09,AG-C11	
Temas	Aprendizaje esperado (<i>Learning Outcomes</i>)
■ Bases de datos distribuidas/sistemas basados en la nube	■ Describir los componentes clave de un DBMS distribuido, incluyendo almacenamiento de datos distribuido, procesamiento de consultas y gestión de transacciones [Describir] ■ Analizar las ventajas y desventajas entre arquitecturas de DBMS paralelo: memoria compartida, disco compartido y nada compartido [Analizar] ■ Describir estrategias de replicación de datos y modelos de consistencia débil en sistemas de bases de datos distribuidas [Describir]
Lecturas : [Buyya2013]	

Unidad 3: Computación Paralela y Distribuida (8 horas)	
Resultados esperados: AG-C09,AG-C11	
Temas	Aprendizaje esperado (<i>Learning Outcomes</i>)
▪ Seguridad (safety) y vivacidad (liveness): • Condiciones de carrera (race conditions) • Dependencias/precondiciones • Modelos de fallos (fault models) • Terminación ▪ Modelos de programación: • Modelo de actores (<i>actor models</i>) • Modelos procedurales y reactivos • Modelos de programación síncrona/asíncrona • Paralelismo de datos (<i>data parallelism</i>) ▪ Propiedades: • Propiedades basadas en orden: ◦ Conmutatividad ◦ Independencia • Propiedades basadas en consistencia: ◦ Atomicidad ◦ Consenso ▪ Control de ejecución: • Espera asíncrona <i>Async await</i> • Promesas (<i>promises</i>) • Hilos (<i>threads</i>) ▪ Comunicación y coordinación: • Mutexes • Paso de mensajes (<i>message-passing</i>) • Memoria compartida (<i>shared memory</i>) • Cobegin-coend • Monitores • Canales (<i>channels</i>) • Hilos (<i>threads</i>) • Guardas (<i>guards</i>) ▪ Futuros (futures) ▪ Soporte del lenguaje para paralelismo de datos como forall, desenrollado de bucles (loop unrolling), map/reduce ▪ Efecto de los modelos de consistencia de memoria en la semántica del lenguaje y la generación correcta de código ▪ Interfaces de Programación de Aplicaciones de Transferencia de Estado Representacional (REST APIs) ▪ Tecnologías y enfoques: computación en la nube (<i>cloud computing</i>), computación de alto rendimiento, computación cuántica, computación ubíqua	▪ Explicar por qué los lenguajes de programación no garantizan la consistencia secuencial en presencia de condiciones de carrera (data races) y qué deben hacer los programadores como resultado [Explicar] ▪ Implementar programas concurrentes correctos usando múltiples modelos de programación, como memoria compartida, actores, futuros, construcciones de sincronización y primitivas de paralelismo de datos [Implementar] ▪ Usar un modelo de paso de mensajes para analizar un protocolo de comunicación [Usar] ▪ Usar construcciones de sincronización como métodos monitorizados/sincronizados en un programa simple [Usar] ▪ Modelar dependencia de datos usando construcciones de programación simples que involucran variables, lectura y escritura [Implementar] ▪ Modelar dependencia de control usando construcciones simples como selección e iteración [Implementar] ▪ Explicar cómo las APIs REST integran aplicaciones y automatizan procesos [Explicar] ▪ Explicar beneficios, limitaciones y desafíos relacionados con la computación distribuida y paralela [Explicar]

Unidad 4: Aplicaciones en Red (6 horas)	
Resultados esperados: AG-C09,AG-C11	
Temas	Aprendizaje esperado (<i>Learning Outcomes</i>)
■ Paradigmas de aplicaciones distribuidas (por ejemplo, cliente/servidor, peer-to-peer, nube, edge y fog). Ver también: Comunicación, Coordinación	■ Definir los principios de denominación, direccionamiento y localización de recursos [Definir] ■ Analizar las necesidades de demandas específicas de aplicaciones en red [Analizar] ■ Describir los detalles de un protocolo de capa de aplicación [Describir] ■ Implementar una aplicación simple cliente-servidor basada en sockets [Implementar]
Lecturas : [Marinescu2022]	

Unidad 5: Algoritmos: Dominios de Aplicación (4 horas)	
Resultados esperados: AG-C09,AG-C11	
Temas	Aprendizaje esperado (<i>Learning Outcomes</i>)
■ Panorama de dominios de aplicación comunes en multinúcleo, reactivo, datos paralelos, clúster, nube, sistemas distribuidos abiertos y marcos de trabajo (con referencia a la siguiente tabla). ● Multinúcleo: Agentes de ejecución típicos: Hilos. Mecanismos de comunicación típicos: Memoria compartida, Atómicos, bloqueos. Dominios algorítmicos típicos: Gestión de recursos, procesamiento de datos. Objetivos de ingeniería típicos: Rendimiento (<i>throughput</i>), latencia, energía. ● Reactivo: Agentes de ejecución típicos: Manejadores, hilos. Mecanismos de comunicación típicos: Canales de E/S. Dominios algorítmicos típicos: Servicios, tiempo real. Objetivos de ingeniería típicos: Latencia. ● Datos paralelos: Agentes de ejecución típicos: GPU, SIMD, aceleradores, híbridos. Mecanismos de comunicación típicos: Memoria heterogénea. Dominios algorítmicos típicos: álgebra lineal, gráficos, análisis de datos. Objetivos de ingeniería típicos: Rendimiento (<i>throughput</i>), energía. ● Clúster: Agentes de ejecución típicos: Hosts gestionados. Mecanismos de comunicación típicos: Sockets, canales. Dominios algorítmicos típicos: Simulación, análisis de datos. Objetivos de ingeniería típicos: Rendimiento (<i>throughput</i>). ● Nube: Agentes de ejecución típicos: Hosts provisionados. Mecanismos de comunicación típicos: APIs de servicio. Dominios algorítmicos típicos: Aplicaciones web. Objetivos de ingeniería típicos: Escalabilidad. ● Distribuido abierto: Agentes de ejecución típicos: Hosts autónomos. Mecanismos de comunicación típicos: Sockets, Almacenes de datos. Dominios algorítmicos típicos: Almacenes de datos y servicios tolerantes a fallos. Objetivos de ingeniería típicos: Confiabilidad.	■ Extender un programa secuencial dirigido por eventos estableciendo una nueva actividad en un manejador de eventos (por ejemplo, un nuevo hilo en un manejador de acciones de GUI) [Diseñar] ■ Mejorar el rendimiento de un componente secuencial introduciendo paralelismo y/o distribución [Crear] ■ Elegir entre diferentes diseños paralelos/distribuidos para componentes de un sistema dado [Evaluar (valorar)]
Lecturas : [Marinescu2022]	

Unidad 6: Principios y Mentalidad de Seguridad (8 horas)	
Resultados esperados: AG-C09,AG-C11	
Temas	Aprendizaje esperado (<i>Learning Outcomes</i>)
■ Aplicaciones de una mentalidad de seguridad: web, nube y dispositivos móviles Diseño del Sistema, Aspectos Comunes: Plataformas, APIs y Restricciones, Aspectos Comunes: Lenguajes de Plataforma y Patrones Web. ■ Implicaciones de privacidad de la recopilación generalizada de datos, incluyendo pero no limitado a bases de datos transaccionales, almacenes de datos, sistemas de vigilancia, computación en la nube e inteligencia artificial.	■ Analizar los riesgos de seguridad asociados con aplicaciones en la nube distribuidas [Usar]. ■ Evaluar las implicaciones de privacidad del almacenamiento de datos en entornos de nube multiinquilino [Evaluar].
Lecturas : [Marinescu2022]	

8. PLAN DE TRABAJO

8.1 Metodología

Se fomenta la participación individual y en equipo para exponer sus ideas, motivándolos con puntos adicionales en las diferentes etapas de la evaluación del curso.

8.2 Sesiones Teóricas

Las sesiones de teoría se llevan a cabo en clases magistrales donde se realizarán actividades que propicien un aprendizaje activo, con dinámicas que permitan a los estudiantes interiorizar los conceptos.

8.3 Sesiones Prácticas

Las sesiones prácticas se llevan en clase donde se desarrollan una serie de ejercicios y/o conceptos prácticos mediante planteamiento de problemas, la resolución de problemas, ejercicios puntuales y/o en contextos aplicativos.

9. SISTEMA DE EVALUACIÓN

Sesiones Teóricas:

Las sesiones de teoría se llevan a cabo en clases magistrales donde se realizarán actividades que propicien un aprendizaje activo, con dinámicas que permitan a los estudiantes interiorizar los conceptos.

Sesiones de Laboratorio:

Para verificar que los alumnos hayan alcanzado el logro planteado para cada una de las unidades de aprendizaje, realizarán actividades que les permita aplicar los conocimientos adquiridos durante las sesiones de teoría y se les propondrá retos que permitan evaluar el desempeño de los alumnos.

Exposiciones individuales o grupales:

Se fomenta la participación individual y en equipo para exponer sus ideas, motivándolos con puntos adicionales en las diferentes etapas de la evaluación del curso.

Lecturas:

A lo largo del curso se proporcionan diferentes lecturas, las cuales podrían ser evaluadas. El uso de herramientas Online permite a cada estudiante acceder a la información del curso, e interactuar fuera de aula con el profesor y con los otros estudiantes.

La nota final NF depende de de las siguientes notas:

- EP : Examen Parcial.
- EF : Examen Final.
- P : Evaluacion permanente (Promedio de prácticas y Trabajos calificados).

La nota final NF se calcula así:

$$NF = \frac{EP + EF + P}{3}$$

10. BIBLIOGRAFÍA BÁSICA