

**Comenzado el** viernes, 11 de abril de 2025, 16:36

**Estado** Finalizado

**Finalizado en** viernes, 11 de abril de 2025, 17:58

**Tiempo empleado** 1 hora 22 minutos

**Calificación** Sin calificar aún

**Pregunta 1**

Correcta

Se puntuó 0,03 sobre 0,03

¿Cuál de las siguientes opciones es una plataforma de mensajería que implementa el patrón **publicación-suscripción** de forma distribuida?

Seleccione una:

- a. Kafka ✓ Correcto.
- b. SSH
- c. FTP
- d. HTTP/2

Apache Kafka es ampliamente utilizado para este fin.

La respuesta correcta es: Kafka

**Pregunta 2**

Incorrecta

Se puntuó 0,00 sobre 0,03

¿En qué consiste la **elección de líder** (leader election) y por qué es importante en un clúster distribuido?

Respuesta: La elección de líder es un proceso en el que los nodos de un ×

El líder coordina tareas críticas y asegura la coherencia entre nodos, evitando conflictos.

La respuesta correcta es: es el mecanismo para designar un nodo coordinador que asegura la coherencia y evita conflictos en el clúster

**Pregunta 3**

Incorrecta

Se puntuá 0,00 sobre 0,03

Menciona **dos ventajas y un desafío** clave de las aplicaciones distribuidas.

Respuesta: Ventajas: 1. Procesadores más poderosos y a menos costos ×

Se espera que menciones, por ejemplo: ventajas como escalabilidad y tolerancia a fallos; y desafíos como la complejidad en la comunicación entre nodos.

La respuesta correcta es: escalabilidad, tolerancia a fallos; complejidad de la comunicación

**Pregunta 4**

Incorrecta

Se puntuá 0,00 sobre 0,03

Mencione dos tipos de servicios que ofrece el middleware.

Respuesta: 1. Servicios de desarrollo, 2. Servicios de administración ×

Generalmente se distingue entre servicios de desarrollo y servicios de administración.

La respuesta correcta es: desarrollo, administración

**Pregunta 5**

Correcta

Se puntuá 0,03 sobre 0,03

¿Qué característica define al modelo **Peer-to-Peer** en contraste con el modelo Cliente-Servidor?

Seleccione una:

- a. No hay un servidor centralizado; cada nodo actúa tanto como cliente como servidor. ✓ Correcto.
- b. La comunicación siempre es asíncrona.
- c. Se requiere de un único punto de falla para garantizar la alta disponibilidad.
- d. El sistema no puede escalar más allá de un solo servidor.

En el modelo Peer-to-Peer no existe un servidor centralizado; cada nodo actúa como cliente y servidor.

La respuesta correcta es: No hay un servidor centralizado; cada nodo actúa tanto como cliente como servidor.

**Pregunta 6**

Incorrecta

Se puntuá 0,00 sobre 0,03

Defina brevemente qué es la **Arquitectura Orientada a Servicios (SOA)**.

Respuesta: La Arquitectura Orientada a Servicios (SOA) es un estilo de c ×

SOA es un modelo de diseño de software que organiza las funcionalidades en servicios independientes, comunicados a través de interfaces bien definidas y protocolos comunes, permitiendo la reutilización y el desacoplamiento.

La respuesta correcta es: modelo de diseño que organiza servicios independientes

**Pregunta 7**

Incorrecta

Se puntuá 0,00 sobre 0,03

Mencione un beneficio principal de utilizar una arquitectura de microservicios.

Respuesta: Permite desarrollar, desplegar y escalar servicios de forma ii ×

Un beneficio clave es la capacidad de escalar servicios de forma independiente y la agilidad que otorga en el desarrollo y despliegue.

La respuesta correcta es: escalabilidad independiente

**Pregunta 8**

Incorrecta

Se puntuá 0,00 sobre 0,04

¿Cuál es el propósito principal de configurar IIS en la publicación de una aplicación C/S?

Respuesta: Permitir que la aplicación se aloje y sea accesible a través d ×

Configurar IIS permite alojar y servir la aplicación en un entorno web, garantizando que los clientes puedan acceder a ella a través de HTTP.

La respuesta correcta es: alojar la aplicación en un servidor web para servir solicitudes HTTP

**Pregunta 9**

Finalizado

Se puntúa como 0 sobre 0,03

Explique la diferencia fundamental entre la comunicación basada en Remote Procedure Call (RPC) y la comunicación basada en mensajes en sistemas distribuidos.

- **RPC (Llamada a Procedimiento Remoto):**

Simula una llamada local a funciones/servicios remotos, ocultando la red mediante invocaciones síncronas (el cliente espera una respuesta inmediata).

- **Comunicación basada en mensajes:**

Usa colas o brokers (como RabbitMQ/Kafka) para intercambiar mensajes asíncronos, permitiendo desacoplamiento, escalabilidad y tolerancia a fallos (el cliente no espera respuestas inmediatas).

La respuesta debe resaltar que en RPC el cliente invoca procedimientos remotos como si fueran locales, mientras que en la comunicación basada en mensajes se envían mensajes que son gestionados por colas u otros mecanismos, posibilitando mayor asincronía.

**Pregunta 10**

Incorrecta

Se puntúa 0,00 sobre 0,03

Describa brevemente el objetivo principal del Algoritmo de Lamport en la sincronización de relojes en sistemas distribuidos.

Respuesta: El objetivo principal es ordenar eventos en sistemas distribuidos ×

Se espera que se mencione que el Algoritmo de Lamport asigna marcas temporales lógicas a eventos para establecer un orden causal en la ejecución de procesos distribuidos.

La respuesta correcta es: asigna marcas temporales lógicas para ordenar los eventos y establecer relaciones causales

**Pregunta 11**

Incorrecta

Se puntuó 0,00 sobre 0,03

Defina brevemente qué son los **microservicios**.

Respuesta: Los microservicios son una arquitectura de software que div 

Los microservicios son un enfoque arquitectónico en el que la aplicación se compone de pequeños servicios independientes que se comunican mediante APIs bien definidas, permitiendo la escalabilidad y el despliegue autónomo de cada función.

La respuesta correcta es: servicios pequeños, independientes, que se comunican mediante API

**Pregunta 12**

Finalizado

Se puntuó como 0 sobre 0,03

Explique qué es la llamada a procedimiento remoto (RPC) y cómo logra la transparencia en la invocación de procedimientos remotos.

La **Llamada a Procedimiento Remoto (RPC)** permite ejecutar funciones en otro sistema como si fueran locales, ocultando la complejidad de la red. Esto se logra mediante:

- **Interfaz uniforme:** el cliente usa la misma sintaxis que en una llamada local.
- **Stub/Proxy:** maneja la comunicación, serialización y recepción de datos.
- **Ocultamiento de red:** el sistema gestiona automáticamente aspectos como el direccionamiento y los errores.

RPC permite llamar a funciones o procedimientos ubicados en otra máquina de forma similar a una llamada local, utilizando un resguardo que empaqueta los parámetros y espera la respuesta, lo que oculta la complejidad del envío y recepción de mensajes.

**Pregunta 13**

Finalizado

Se puntuá como 0 sobre 0,05

Describa brevemente los tres modelos de envío de mensajes en el contexto de RPC: comunicación asíncrona, comunicación síncrona y la invocación remota. Compare sus principales características.

La comunicación síncrona bloquea al cliente hasta recibir respuesta, siendo simple pero ineficiente.

La asíncrona permite continuar la ejecución, mejorando rendimiento pero con mayor complejidad.

RPC clásico simula llamadas locales con comunicación de red, normalmente síncrona.

Comparando, la síncrona es bloqueante y acoplada; la asíncrona es no bloqueante y más escalable.

La respuesta debe incluir que en la comunicación asíncrona el emisor continúa su ejecución sin esperar respuesta, en la comunicación síncrona el emisor se bloquea esperando que el receptor reciba el mensaje, y en la invocación remota se espera la respuesta del procedimiento remoto, lo que implica una mayor transparencia en la llamada.

**Pregunta 14**

Incorrecta

Se puntuá 0,00 sobre 0,03

Defina brevemente qué es una **aplicación distribuida** e indique al menos un componente fundamental.

Respuesta: Una aplicación distribuida es aquella que se ejecuta en múltiples sistemas.

Una aplicación distribuida es aquella en la que sus componentes se ejecutan en varias computadoras conectadas en red, permitiendo la compartición de recursos. Entre sus componentes se destaca la división entre cliente y servidor, y el uso de protocolos para la comunicación.

La respuesta correcta es: aplicación que se ejecuta en varias computadoras, cliente-servidor, red

**Pregunta 15**

Incorrecta

Se puntuá 0,00 sobre 0,03

Define en una frase **comunicación síncrona** y **comunicación asíncrona** en sistemas distribuidos.

Respuesta: La comunicación síncrona requiere que ambos procesos est X

Se espera que el estudiante resuma que la comunicación síncrona es bloqueante (espera respuesta) y la asíncrona utiliza colas o mensajes sin bloqueo.

La respuesta correcta es: síncrona bloquea mientras espera la respuesta; asíncrona utiliza colas o mensajes sin bloqueo

**Pregunta 16**

Finalizado

Se puntúa como 0 sobre 0,03

Explique brevemente cómo se implementa un servicio API RESTful basado en SOA usando .NET. En su respuesta, mencione la estructura del proyecto, uso de controladores, y cómo se publicaría la API en un entorno de producción (por ejemplo, en Linux).

**1. Estructura del Proyecto:**

- Se crea un proyecto ASP.NET Core (Web API).
- El código se organiza en capas como Controllers, Services, Models y Repositories para mantener una separación clara de responsabilidades.
- Se utilizan DTOs (Data Transfer Objects) para facilitar la transferencia de datos entre las diferentes capas.

**2. Controladores:**

- Los controladores se encargan de gestionar las solicitudes HTTP y devolver las respuestas correspondientes.
- Se emplean atributos como `HttpGet`, `HttpPost`, entre otros, para definir los endpoints de la API.
- Los servicios (SOA) se injecan en los controladores para manejar la lógica de negocio, siguiendo los principios de la inyección de dependencias.

**3. Publicación en Producción (Linux):**

- La API se publica como una aplicación autónoma (`self-contained`) o dependiente del runtime, según los requisitos del entorno.
- Se utiliza el comando `dotnet publish` para generar los archivos necesarios para el despliegue.
- Se configura un servidor web como Nginx o Apache como proxy inverso para redirigir las solicitudes a la API.
- Para gestionar el proceso en Linux, se emplea `systemd` o un contenedor Docker, garantizando su ejecución continua.

Comando para publicar:

```
dotnet publish -c Release -r linux-x64
```

La respuesta debe incluir que se crea un proyecto ASP.NET Core Web API, se implementan controladores (ejemplo: `UsuariosController`) para exponer servicios, se prueba en `localhost` y luego se publica (por ejemplo, copiando la publicación a un entorno Linux y configurando un servicio `systemd`) para que la API esté disponible en producción.

**Pregunta 17**

Finalizado

Se puntuá como 0 sobre 0,03

Explica brevemente la diferencia entre **modelos centralizados** y **modelos distribuidos** y menciona un ejemplo de cada uno.

**1. Modelo Centralizado:**

- Todo el procesamiento y gestión de datos se realiza en un único sistema o servidor central.
- Ventaja: Simple de administrar y mantener.
- Desventaja: Puede convertirse en un cuello de botella y tener un único punto de fallo.
- **Ejemplo:** Una base de datos tradicional en un solo servidor (como MySQL en una máquina única).

**2. Modelo Distribuido:**

- Los recursos y procesos se dividen entre múltiples nodos o sistemas que trabajan en conjunto.
- Ventaja: Mayor escalabilidad, tolerancia a fallos y rendimiento.
- Desventaja: Complejidad en la coordinación y sincronización.
- **Ejemplo:** Un sistema de microservicios en la nube (como Kubernetes gestionando contenedores en múltiples servidores).

Se debe explicar qué es un sistema centralizado (todos los recursos en un único nodo) versus uno distribuido (recursos distribuidos en múltiples nodos) y exemplificar cada caso.

**Pregunta 18**

Incorrecta

Se puntuá 0,00 sobre 0,03

Define qué es una aplicación distribuida e indica cuáles son sus componentes fundamentales.

Respuesta: Una aplicación distribuida es un sistema que ejecuta múltipl ×

Se espera que se mencione que una aplicación distribuida es aquella en la que sus componentes se ejecutan en distintos entornos (equipos, plataformas) conectados por una red, distinguiendo entre el lado cliente, el lado servidor y el protocolo utilizado para la comunicación.

La respuesta correcta es: Es una aplicación cuyos componentes se ejecutan en equipos distintos conectados por una red, incluyendo un lado cliente, un lado servidor y protocolos para intercambiar mensajes

**Pregunta 19**

Finalizado

Se puntuó como 0 sobre 0,03

Defina qué es el middleware y explique cuál es su objetivo principal en sistemas distribuidos.

El middleware es un software intermediario que facilita la comunicación entre aplicaciones y sistemas en entornos distribuidos, ocultando la complejidad de red, coordinando tareas como autenticación y balanceo, y asegurando la interoperabilidad entre plataformas distintas.

El middleware es un conjunto de servicios intermedios que facilitan la comunicación e integración entre aplicaciones, permitiendo la transparencia y liberando a los desarrolladores de problemas relacionados con la complejidad del sistema operativo y la red.

**Pregunta 20**

Incorrecta

Se puntuó 0,00 sobre 0,04

Mencione dos ventajas de usar un middleware basado en mensajes (por ejemplo, Apache Kafka o RabbitMQ) para la comunicación asíncrona en sistemas distribuidos.

Respuesta: Dos ventajas de usar un middleware basado en mensajes sc ×

Se deben resaltar ventajas como el desacoplamiento de servicios, alta escalabilidad, tolerancia a fallos y la posibilidad de procesar eventos en paralelo.

La respuesta correcta es: desacoplamiento, escalabilidad; tolerancia a fallos

**Pregunta 21**

Correcta

Se puntuá 0,03 sobre 0,03

¿Cuál de las siguientes afirmaciones es característica de una arquitectura basada en microservicios?

Seleccione una:

- a. Se basa en un único proceso centralizado para garantizar la coherencia.
- b. Se pueden desplegar, escalar y actualizar de forma independiente. ✓ Correcto.
- c. Todos los servicios deben ser implementados en el mismo lenguaje de programación.

Los microservicios son independientes, se pueden desplegar y escalar de forma autónoma y están diseñados para cumplir funciones específicas.

La respuesta correcta es: Se pueden desplegar, escalar y actualizar de forma independiente.

**Pregunta 22**

Finalizado

Se puntuá como 0 sobre 0,03

Defina qué es la Arquitectura Orientada a Servicios (SOA) y describa sus principales características.

SOA es un paradigma que organiza funcionalidades en servicios reutilizables e independientes, que interactúan mediante protocolos estandarizados. Sus características incluyen servicios modulares, interoperabilidad, acoplamiento débil, descubrimiento dinámico, contratos bien definidos y la capacidad de componer servicios para procesos complejos.

SOA es una aproximación arquitectónica que integra las metas de negocio con los sistemas de software mediante servicios que representan procesos reales. Se caracteriza por la reutilización, flexibilidad, uso de estándares abiertos y un fuerte gobierno de servicios.

**Pregunta 23**

Incorrecta

Se puntuá 0,00 sobre 0,03

¿Cuál es el rol del middleware en arquitecturas distribuidas, especialmente en entornos de microservicios?

Respuesta: El rol del middleware en arquitecturas distribuidas, especialr ×

Se espera que se mencione que el middleware actúa como capa intermedia para facilitar la comunicación, coordinación y seguridad entre servicios, desacoplando las aplicaciones y permitiendo la integración de funcionalidades comunes.

La respuesta correcta es: facilita la comunicación e integración entre servicios desacoplándolos; se usa para gestionar seguridad, mensajería y transacciones

**Pregunta 24**

Finalizado

Se puntuá como 0 sobre 0,03

¿Qué es el middleware y cuál es su rol en una arquitectura basada en microservicios?

El middleware en microservicios facilita la comunicación, gestión de concurrencia, resiliencia, mensajería asíncrona, autenticación, monitorización, balanceo de carga y descubrimiento de servicios, utilizando herramientas como API Gateways, RabbitMQ, OAuth2 y Prometheus.

Se debe explicar que el middleware actúa como una capa intermedia que facilita la comunicación, integración y gestión de servicios, permitiendo la interoperabilidad y coordinando aspectos comunes como seguridad y transacciones en un entorno de microservicios.

**Pregunta 25**

Finalizado

Se puntuá como 0 sobre 0,03

Explica un ejemplo de flujo de mensajería con **Apache Kafka** y menciona en qué escenarios se recomienda su uso.

En un flujo de Kafka, el servicio de pedidos publica un evento "PedidoCreado" en un tópico, que es consumido por servicios como inventario, notificaciones y logística. Kafka es ideal para procesamiento en tiempo real, microservicios desacoplados, replicación de datos y procesamiento de streams.

Es importante que el estudiante explique la interacción entre productores, topics y consumidores, y destaque escenarios como el procesamiento de streaming y análisis en tiempo real.

**Pregunta 26**

Incorrecta

Se puntuá 0,00 sobre 0,03

¿Qué significa RPC (Remote Procedure Call) y mencione una tecnología o ejemplo que lo implemente.

Respuesta: RPC (Remote Procedure Call) es un protocolo que permite a ×

Se espera que se defina RPC como la capacidad de invocar procedimientos en máquinas remotas como si fueran locales. Ejemplos incluyen gRPC, XML-RPC o SOAP.

La respuesta correcta es: invoca procedimientos remotos; ejemplo: gRPC

**Pregunta 27**

Correcta

Se puntuá 0,05 sobre 0,05

¿Para qué se utiliza el mecanismo de **heartbeats** en sistemas distribuidos?

Seleccione una:

- a. Para sincronizar los relojes de todos los nodos en el clúster.
- b. Para compartir la carga de trabajo en el servidor principal.
- c. Para encriptar las comunicaciones entre servidores.
- d. Para verificar periódicamente que los nodos estén activos y detectar fallos o desconexiones. ✓ Correcto.

Los heartbeats se utilizan para monitorizar la disponibilidad de los nodos en el sistema.

La respuesta correcta es: Para verificar periódicamente que los nodos estén activos y detectar fallos o desconexiones.

**Pregunta 28**

Finalizado

Se puntuá como 0 sobre 0,03

En el contexto de **microservicios**, ¿por qué se suele recomendar la comunicación a través de APIs ligeras? Indica los beneficios y desafíos que esta decisión conlleva.

Los microservicios ofrecen desacople, escalabilidad independiente, interoperabilidad, facilidad de desarrollo y resiliencia, pero enfrentan desafíos como latencia en llamadas síncronas, gestión de fallos, versionado, seguridad y documentación coherente.

Se espera que el estudiante analice aspectos como la modularidad, facilidad de mantenimiento, pero también retos en la orquestación y trazabilidad.

**Pregunta 29**

Finalizado

Se puntuó como 0 sobre 0,04

Describa las principales diferencias entre la comunicación síncrona (por ejemplo, utilizando gRPC o REST) y la comunicación asíncrona (por ejemplo, utilizando colas de mensajes o Apache Kafka) en sistemas distribuidos, haciendo énfasis en acoplamiento, escalabilidad y tolerancia a fallos.

La comunicación síncrona (gRPC, REST) requiere que el cliente espere la respuesta del servidor, lo que genera un alto acoplamiento y limitaciones en escalabilidad. Es ideal para operaciones que requieren confirmación inmediata.

La comunicación asíncrona (Kafka, colas de mensajes) permite que los servicios operen independientemente, con alta escalabilidad y resiliencia, y es adecuada para procesamiento en segundo plano y arquitecturas basadas en eventos.

La respuesta debe explicar que en la comunicación síncrona el cliente espera la respuesta del servicio, lo que genera un acoplamiento más fuerte y dependencia inmediata, mientras que la asíncrona desacopla los servicios, permitiendo mayor escalabilidad y tolerancia a fallos, aunque con mayor complejidad en el rastreo de mensajes.

**Pregunta 30**

Finalizado

Se puntuá como 0 sobre 0,03

Explique el principio de **interoperabilidad** en SOA y por qué es fundamental para que diferentes sistemas se comuniquen.

La interoperabilidad en SOA permite que sistemas con diferentes lenguajes y plataformas colaboren mediante estándares comunes como REST o SOAP, favoreciendo la integración, independencia tecnológica y reutilización de servicios.

Se debe explicar que la interoperabilidad permite que servicios desarrollados en lenguajes o plataformas diferentes se comuniquen mediante estándares abiertos, lo cual facilita la integración entre sistemas heterogéneos.

**Pregunta 31**

Correcta

Se puntuá 0,03 sobre 0,03

¿Cuál de las siguientes afirmaciones describe mejor una aplicación distribuida?

Seleccione una:

- a. Una aplicación cuyas partes lógicas y de procesamiento se reparten en múltiples nodos o máquinas, comunicándose a través de una red. ✓ Correcto.
- b. Una aplicación que se ejecuta solamente en un único proceso y en un único computador.
- c. Una aplicación que no requiere conectividad de red y opera en modo local sin intercambio de datos.
- d. Una aplicación que no tiene componentes desacoplados ni necesidad de escalado.

Una aplicación distribuida reparte sus componentes en múltiples nodos, comunicándose a través de la red.

La respuesta correcta es: Una aplicación cuyas partes lógicas y de procesamiento se reparten en múltiples nodos o máquinas, comunicándose a través de una red.

**Pregunta 32**

Finalizado

Se puntuá como 0 sobre 0,05

Explique qué es el modelo Cliente-Servidor en las aplicaciones distribuidas y describa brevemente las funciones que corresponden a cada parte (cliente y servidor).

El modelo Cliente-Servidor es una arquitectura en la que el cliente solicita recursos y el servidor procesa estas solicitudes, comunicándose a través de protocolos estándar. El cliente gestiona la interfaz y envía peticiones, mientras que el servidor ejecuta la lógica de negocio, gestiona recursos y devuelve respuestas. Ejemplos incluyen navegadores web y servidores web o de bases de datos.

La respuesta debe incluir que en el modelo Cliente-Servidor el cliente se encarga de la presentación (interfaz de usuario) y el servidor de la lógica de negocio y el acceso a datos, permitiendo la separación de responsabilidades.

**Pregunta 33**

Finalizado

Se puntuá como 0 sobre 0,03

¿Por qué es importante la sincronización en sistemas distribuidos y qué desafíos implica lograr un orden adecuado en la ejecución de procesos?

La sincronización en sistemas distribuidos es crucial para garantizar la coordinación entre procesos en nodos diferentes, evitando conflictos y asegurando coherencia. Es vital para la coherencia de datos, eficiencia operativa y prevención de conflictos como condiciones de carrera. Los desafíos incluyen la latencia de red, fallos en nodos, exclusión mutua y la sincronización de relojes distribuidos.

Es fundamental para garantizar que los procesos se ejecuten en un orden coherente y que la referencia temporal se comparta entre todos los nodos, enfrentándose a desafíos como la latencia, la ausencia de un reloj global confiable y la complejidad de algoritmos de consenso.

**Pregunta 34**

Finalizado

Se puntuá como 0 sobre 0,03

Compare brevemente los modelos de computación monolítica, paralela y distribuida, señalando sus características principales en cuanto a uso de procesadores, enlace entre componentes y requerimientos de hardware.

La computación monolítica utiliza un único procesador y componentes integrados en un solo sistema, requiriendo menos recursos. La computación paralela emplea múltiples procesadores para realizar tareas simultáneamente, con componentes coordinados, y necesita arquitecturas especializadas. La computación distribuida utiliza procesadores en diferentes nodos conectados por red, con componentes desacoplados, y requiere múltiples máquinas capaces de manejar heterogeneidad.

Se debe identificar que en la computación monolítica el programa corre en un único procesador con recursos internos, en la computación paralela se aprovechan varios procesadores fuertemente acoplados y en la computación distribuida la ejecución se realiza en equipos conectados a través de una red (ligeramente acoplados).

**Pregunta 35**

Correcta

Se puntuá 0,03 sobre 0,03

Si un servidor maneja múltiples clientes al mismo tiempo usando **hilos**, ¿qué ventaja principal aporta este enfoque?

Seleccione una:

- a. Deshabilita la comunicación a través de sockets.
- b. Garantiza que no haya bloqueos ni problemas de concurrencia.
- c. Aumenta el rendimiento al permitir la ejecución concurrente, aprovechando mejor los recursos del procesador. ✓ Correcto.
- d. Reduce el consumo de memoria considerablemente.

La concurrencia con hilos permite aprovechar mejor los recursos del procesador.

La respuesta correcta es: Aumenta el rendimiento al permitir la ejecución concurrente, aprovechando mejor los recursos del procesador.

**Pregunta 36**

Incorrecta

Se puntuó 0,00 sobre 0,03

Mencione un ejemplo práctico en el que se haya implementado una Arquitectura Orientada a Servicios (SOA) en el desarrollo de aplicaciones distribuidas.

Respuesta: Un ejemplo práctico de SOA es el uso de servicios web para X

Puede mencionarse, por ejemplo, un sistema de gestión empresarial (ERP) que utiliza servicios web para integrar diferentes áreas como finanzas, recursos humanos y ventas.

La respuesta correcta es: sistema ERP basado en servicios web

**Pregunta 37**

Correcta

Se puntuó 0,04 sobre 0,04

El protocolo HTTP y el modelo REST son intercambiables y técnicamente equivalentes.

- Verdadero  
 Falso ✓

Correcto.

HTTP es un protocolo; REST es un estilo de arquitectura basado en ciertos principios de HTTP.

La respuesta correcta es 'Falso'

**Pregunta 38**

Incorrecta

Se puntuó 0,00 sobre 0,03

¿Cuál es la diferencia fundamental entre un **proceso** y un **hilo** en un servidor?

Respuesta: La diferencia fundamental es que un proceso es una instanc X

Un proceso tiene su propia memoria, mientras que un hilo comparte el espacio de memoria del proceso.

La respuesta correcta es: un proceso tiene memoria independiente mientras que un hilo comparte la memoria del proceso

**Pregunta 39**

Finalizado

Se puntuá como 0 sobre 0,03

Compare brevemente la arquitectura por capas con la arquitectura basada en microservicios. Mencione al menos una ventaja y una desventaja de cada enfoque.

La arquitectura por capas facilita la organización y separación de responsabilidades, pero puede ser menos flexible y escalable debido a la interdependencia entre capas.

La arquitectura basada en microservicios ofrece alta escalabilidad y flexibilidad, permitiendo que cada microservicio se maneje de manera independiente, pero introduce mayor complejidad en la gestión y comunicación entre servicios.

La respuesta debe resaltar que la arquitectura por capas organiza el sistema en niveles (presentación, negocio, datos) con una relación jerárquica, lo que facilita la modularidad pero puede afectar el rendimiento; en cambio, los microservicios dividen la funcionalidad en servicios independientes, permitiendo escalabilidad y despliegue independiente, a costa de una mayor complejidad operativa y de comunicación entre servicios.

**Pregunta 40**

Incorrecta

Se puntuá como 0,00 sobre 0,03

Menciona al menos dos ventajas y dos desventajas de utilizar computación distribuida.

Respuesta: Ventajas: 1) Escalabilidad mejorada, 2) Alta disponibilidad. ×

Entre las ventajas se pueden citar el aprovechamiento de recursos distribuidos, escalabilidad y tolerancia a fallos; entre las desventajas, la complejidad de la comunicación y la dificultad en la gestión de fallos y seguridad.

La respuesta correcta es: escalabilidad, tolerancia a fallos; complejidad en la comunicación, dificultad en la sincronización y seguridad

**Pregunta 41**

Incorrecta

Se puntuó 0,00 sobre 0,03

Mencione una ventaja y una desventaja de utilizar SOA en el desarrollo de aplicaciones.

Respuesta: Ventaja: Mejora la reutilización de servicios. Desventaja: Ma ×

Una ventaja es la facilidad de mantenimiento y actualización al trabajar con servicios pequeños; una desventaja puede ser el aumento en las interdependencias que complica la gestión si no se diseña correctamente.

La respuesta correcta es: mantenimiento eficiente; aumento en interdependencias

**Pregunta 42**

Incorrecta

Se puntuó 0,00 sobre 0,04

Explique brevemente cómo se implementa un ejemplo de aplicación distribuida en el ámbito bancario o en videojuegos multijugador, mencionando el rol del cliente y del servidor.

Respuesta: En el ámbito bancario, una aplicación distribuida permite qu ×

Se espera que se describa, por ejemplo, en una aplicación bancaria el cliente (aplicación móvil) realiza consultas o transacciones, y el servidor procesa y gestiona bases de datos; o bien, en un videojuego multijugador, cada instancia del juego en el cliente se conecta a servidores que coordinan la lógica y el estado de la partida.

La respuesta correcta es: el cliente solicita operaciones (consultas, transacciones o interacciones) y el servidor procesa la lógica, gestiona la base de datos y retorna respuestas

**Pregunta 43**

Correcta

Se puntuá 0,03 sobre 0,03

¿Cuál es la característica principal de un protocolo basado en **Remote Procedure Call (RPC)**?

Seleccione una:

- a. Requiere obligatoriamente el uso de JSON para el intercambio de datos.
- b. Está diseñado específicamente para el uso exclusivo de aplicaciones web monolíticas.
- c. Utiliza únicamente texto plano para la comunicación.
- d. Se basa en el modelo request-response pero permite ✓ Correcto. que el cliente invoque métodos como si fueran locales.

RPC permite que el cliente invoque métodos remotos como si fueran locales.

La respuesta correcta es: Se basa en el modelo request-response pero permite que el cliente invoque métodos como si fueran locales.

**Pregunta 44**

Correcta

Se puntuá 0,03 sobre 0,03

¿Cuál de la siguiente lista identifica correctamente cuatro componentes básicos en una arquitectura SOA?

Seleccione una:

- a. Servicio, Contrato del servicio, Interfaz del servicio y ✓ Correcto. Registro de servicios
- b. Cliente, Servidor, Proveedor y Consumidor
- c. Servicio, Middleware, Proveedor y Consumidor

La respuesta correcta es: Servicio, Contrato del servicio, Interfaz del servicio y Registro de servicios.

La respuesta correcta es: Servicio, Contrato del servicio, Interfaz del servicio y Registro de servicios

**Pregunta 45**

Finalizado

Se puntuá como 0 sobre 0,05

Describe los diferentes modelos de consistencia (consistencia fuerte, consistencia débil, consistencia eventual) y brinda un ejemplo de situación en la que utilizarías la consistencia eventual en lugar de la consistencia fuerte.

La consistencia fuerte garantiza que todas las lecturas devuelvan los datos más recientes, ofreciendo fiabilidad, pero puede causar alta latencia. Ejemplo: sistemas bancarios. La consistencia débil no asegura lecturas actuales y mejora el rendimiento, pero puede causar inconsistencias temporales. Ejemplo: juegos en línea. La consistencia eventual asegura que todos los nodos convergerán en el mismo estado con el tiempo, mejorando disponibilidad y rendimiento, pero puede haber retrasos en la sincronización. Ejemplo: redes sociales.

Se debe valorar la comprensión de cada modelo y la capacidad para relacionarlo con ejemplos prácticos, como en sistemas de redes sociales u otros entornos distribuidos.

**Pregunta 46**

Finalizado

Se puntuá como 0 sobre 0,03

Explique cómo el middleware contribuye a la transparencia en sistemas distribuidos.

El middleware contribuye a la transparencia en sistemas distribuidos al ocultar la complejidad de la comunicación y gestión de recursos entre los diferentes nodos. Facilita la interacción entre componentes dispersos sin que los desarrolladores o usuarios finales necesiten preocuparse por detalles como la ubicación de los servicios, el manejo de fallos o la heterogeneidad de las plataformas. A través de servicios como la autenticación, el balanceo de carga y la gestión de sesiones, el middleware asegura que los sistemas distribuidos funcionen de manera coherente y eficiente, sin exponer la complejidad interna.

Se espera que se explique que el middleware oculta la complejidad de la comunicación y la localización de recursos, permitiendo a los usuarios interactuar con el sistema como si todo estuviera en una sola máquina, lo que incluye transparencias de ubicación, de réplica y de fallos.

**Pregunta 47**

Finalizado

Se puntuá como 0 sobre 0,03

Describa la arquitectura del modelo Cliente-Servidor en aplicaciones distribuidas, explicando las funciones principales del cliente y del servidor.

La arquitectura Cliente-Servidor en aplicaciones distribuidas organiza las tareas entre dos componentes clave: el cliente y el servidor. El **cliente** inicia solicitudes al servidor, interactúa con el usuario y proporciona la interfaz para enviar peticiones y recibir respuestas. Ejemplos incluyen navegadores web y aplicaciones móviles. El **servidor** responde a las solicitudes procesando las peticiones, realizando operaciones y devolviendo los resultados. Aloja recursos como bases de datos y aplicaciones para cumplir con las solicitudes del cliente.

Se debe explicar que el cliente es responsable de solicitar servicios (por ejemplo, mediante una interfaz de usuario) y el servidor de procesar las solicitudes, gestionar la lógica de negocio y proporcionar acceso a datos o servicios.

**Pregunta 48**

Correcta

Se puntuá 0,03 sobre 0,03

¿Cuál es una ventaja clave del uso de Web Sockets para la comunicación en aplicaciones distribuidas?

Seleccione una:

- a. Funcionan únicamente con datos en formato XML, lo que aumenta la compatibilidad.
- b. Permiten una comunicación bidireccional en tiempo real con menos latencia. ✓ Correcto.
- c. Requieren menos recursos del servidor eliminando la necesidad de mantenimiento de conexiones persistentes.

Los Web Sockets permiten una comunicación bidireccional y en tiempo real, reduciendo la latencia en el intercambio continuo de datos.

La respuesta correcta es: Permiten una comunicación bidireccional en tiempo real con menos latencia.

**Pregunta 49**

Correcta

Se puntuá 0,05 sobre 0,05

¿Cuál de los siguientes enunciados describe mejor la **fragmentación (sharding)** en bases de datos?

Seleccione una:

- a. Replicar tablas en distintos servidores para tener redundancia de datos.
- b. Mantener una copia de seguridad para recuperación de desastres.
- c. Particionar lógicamente los datos en múltiples nodos ✓ Correcto. para repartir la carga y lograr escalabilidad horizontal.
- d. Copiar toda la base de datos en un único servidor para reducir la latencia.

El sharding consiste en particionar lógicamente la base de datos entre varios nodos para repartir la carga.

La respuesta correcta es: Particionar lógicamente los datos en múltiples nodos para repartir la carga y lograr escalabilidad horizontal.

**Pregunta 50**

Finalizado

Se puntuá como 0 sobre 0,03

Explique los mecanismos de comunicación en sistemas distribuidos. En su respuesta incluya ejemplos de comunicación directa (cliente-servidor) e indirecta (a través de middleware), así como la diferencia entre comunicación síncrona y asíncrona.

La **comunicación directa** (cliente-servidor) implica que los procesos interactúan directamente, como cuando un navegador web solicita una página a un servidor HTTP.

En la **comunicación indirecta** (a través de middleware), un intermediario, como un sistema de colas de mensajes, gestiona la comunicación entre los procesos.

La **comunicación síncrona** ocurre cuando el emisor espera una respuesta inmediata antes de continuar, como en una llamada RPC. En la **comunicación asíncrona**, el emisor no espera una respuesta inmediata y puede seguir con otras tareas, como en el caso del envío de correos electrónicos.

La respuesta debe abarcar aspectos como RPC, sockets, servicios web (REST API), colas de mensajes (RabbitMQ, Apache Kafka) y diferenciar que en la comunicación síncrona el emisor espera respuesta inmediata, mientras que en la asíncrona se procesa la solicitud sin esperar una respuesta al instante.

**Pregunta 51**

Finalizado

Se puntuá como 0 sobre 0,03

Describe un caso práctico en el que la elección de líder sea esencial para el correcto funcionamiento de un sistema distribuido. Explica cómo se manejarían los fallos de dicho líder.

En bases de datos distribuidas, el líder coordina las escrituras para mantener la consistencia. Si falla, los nodos detectan el fallo, eligen un nuevo líder con algoritmos como Bully o Paxos y reconfiguran el sistema para continuar operando.

La respuesta debe incluir un escenario práctico (por ejemplo, coordinación en un clúster de servidores), la función del líder, y mecanismos de detección y reelección (como Raft o Paxos).

**Pregunta 52**

Finalizado

Se puntuá como 0 sobre 0,03

¿Qué es la Arquitectura Orientada a Servicios (SOA) y cuáles son algunas de sus ventajas principales en el desarrollo de aplicaciones distribuidas?

SOA es un enfoque de diseño que organiza funcionalidades en servicios reutilizables e independientes, comunicados por interfaces definidas. Sus ventajas son reutilización, flexibilidad, escalabilidad y mantenimiento eficiente.

Se espera que se defina SOA como el modelo de diseño de software que organiza la funcionalidad en servicios independientes comunicados mediante interfaces bien definidas (por ejemplo, usando XML o JSON sobre HTTP), y se mencionen ventajas como la reutilización, la separación de responsabilidades, y la flexibilidad para escalar o actualizar servicios de manera individual.

**Pregunta 53**

Correcta

Se puntuá 0,03 sobre 0,03

En sistemas distribuidos, un proceso puede desempeñar distintos roles.  
¿Cuál de las siguientes opciones lista correctamente los roles básicos?

Seleccione una:

- a. Servidor y Middleware solamente.
- b. Cliente, Servidor y Peer (par). ✓ Correcto.
- c. Cliente y Servidor solamente.

Se deben considerar al menos los roles de cliente, servidor y peer (par) que permite que un proceso actúe como solicitante y proveedor de servicios.

La respuesta correcta es: Cliente, Servidor y Peer (par).

**Pregunta 54**

Finalizado

Se puntuá como 0 sobre 0,03

Explica el rol de los **sockets** en la comunicación entre servidores y clientes.  
¿Qué tipo de socket usarías para una aplicación de mensajería en tiempo real y por qué?

Los sockets permiten la comunicación entre cliente y servidor en sistemas distribuidos; para mensajería en tiempo real se usa TCP si se necesita confiabilidad o UDP si se prioriza la velocidad.

Se debe explicar que los sockets actúan como puntos finales para la comunicación. Se pueden considerar TCP (para fiabilidad) o UDP (para baja latencia) según el caso.

**Pregunta 55**

Finalizado

Se puntuá como 0 sobre 0,03

Compare brevemente la arquitectura monolítica con la basada en microservicios en cuanto a escalabilidad.

La arquitectura monolítica escala verticalmente, lo que puede ser costoso e ineficiente, mientras que los microservicios permiten escalabilidad horizontal, adaptando solo los servicios necesarios, de forma más flexible y eficiente.

Se debe explicar que en una arquitectura monolítica, para escalar una parte se debe escalar toda la aplicación, mientras que en los microservicios cada componente se puede escalar de manera independiente según la demanda.

**Pregunta 56**

Finalizado

Se puntuá como 0 sobre 0,05

Compara **HTTP** y **gRPC** en cuanto a la eficiencia y uso de recursos en aplicaciones distribuidas. ¿En qué tipo de escenarios recomendarías gRPC por encima de HTTP/REST?

gRPC es más eficiente que HTTP/REST al usar Protobuf y HTTP/2, ideal para microservicios de alto rendimiento, entornos con poco ancho de banda e interoperabilidad entre lenguajes, mientras que REST es más flexible pero menos eficiente.

Valora el conocimiento sobre serialización (JSON vs Protobuf), latencia, eficiencia y escenarios de microservicios con alta demanda de rendimiento.

**Pregunta 57**

Correcta

Se puntuá 0,03 sobre 0,03

¿Cuál de las siguientes afirmaciones es falsa respecto a la publicación de aplicaciones en IIS?

Seleccione una:

- a. Se debe probar la aplicación en localhost antes de la publicación
- b. No se necesita configurar carpetas de aplicación para que IIS funcione ✓ Incorrecto. Esta afirmación es falsa.
- c. Es necesario activar las características de IIS en el Panel de Control

La afirmación falsa es que "No se necesita configurar carpetas de aplicación para que IIS funcione", ya que sí es necesario definir las carpetas y asignarles los permisos adecuados.

La respuesta correcta es: No se necesita configurar carpetas de aplicación para que IIS funcione

**Pregunta 58**

Correcta

Se puntuá 0,03 sobre 0,03

¿Cuál de las siguientes afirmaciones compara correctamente SOAP y REST en el contexto de SOA?

Seleccione una:

- a. SOAP y REST son idénticos en su funcionamiento, diferenciándose solo en nomenclatura.
- b. REST utiliza únicamente XML para el intercambio de datos, a diferencia de SOAP.
- c. SOAP es un protocolo basado en XML con estructura estandarizada, mientras que REST es un estilo arquitectónico que aprovecha HTTP para operaciones ligeras. ✓ Correcto.

SOAP es un protocolo basado en XML diseñado para operaciones específicas y orientadas a servicios, mientras que REST es un estilo arquitectónico que utiliza los verbos HTTP y se caracteriza por su sencillez y ligereza.

La respuesta correcta es: SOAP es un protocolo basado en XML con estructura estandarizada, mientras que REST es un estilo arquitectónico que aprovecha HTTP para operaciones ligeras.

**Pregunta 59**

Incorrecta

Se puntuá 0,00 sobre 0,03

¿Cuáles son los pasos básicos para configurar una carpeta compartida en Ubuntu utilizando Samba, de modo que pueda ser accedida desde Windows?

Respuesta: Instalar Samba con sudo apt install samba. Crear una carpeta X

Se espera mencionar: instalar Samba, crear y asignar permisos a la carpeta compartida, editar el archivo de configuración (smb.conf) agregando la sección correspondiente, reiniciar el servicio smbd y acceder desde Windows mediante la dirección \\[IP\_del\_servidor].

La respuesta correcta es: instalar samba, configurar smb.conf con la ruta y permisos adecuados, reiniciar el servicio y acceder vía \\ip\_del\_ubuntu

**Pregunta 60**

Incorrecta

Se puntuá 0,00 sobre 0,03

¿Cuál es uno de los principales objetivos de construir aplicaciones distribuidas?

Respuesta: Uno de los principales objetivos de construir aplicaciones di X

Uno de los principales objetivos es compartir y aprovechar recursos (tanto hardware como software) de forma eficiente entre diferentes nodos.

La respuesta correcta es: compartir recursos

**Pregunta 61**

Incorrecta

Se puntuá 0,00 sobre 0,03

Defina brevemente qué es una **aplicación distribuida** y mencione al menos dos componentes o características fundamentales.

Respuesta: Una aplicación distribuida es un sistema que ejecuta proces X

Se espera que se mencione que una aplicación distribuida está compuesta por componentes que se ejecutan en distintos equipos o plataformas conectadas por red y que la comunicación entre estos componentes es clave (por ejemplo, cliente, servidor, y uso de protocolos o middleware).

La respuesta correcta es: componentes distribuidos, cliente-servidor, red

**Pregunta 62**

Correcta

Se puntuá 0,03 sobre 0,03

En el ejemplo del proyecto de combustible XYZ, ¿cuándo se recomienda usar comunicación síncrona (por ejemplo, gRPC) en lugar de asíncrona (por ejemplo, Kafka)?

Seleccione una:

- a. Para registrar consumo de combustible en tiempo real, se usa comunicación síncrona.
- b. Para consultas inmediatas, como verificar disponibilidad o asignar choferes, se usa comunicación síncrona. ✓ Correcto.
- c. No existe diferencia; ambos métodos se pueden usar indistintamente para cualquier operación.

Se recomienda usar comunicación síncrona para operaciones que requieren respuesta inmediata (por ejemplo, consulta de disponibilidad de choferes o asignación directa de ruta) y comunicación asíncrona para registrar eventos (como consumo de combustible) o procesos en segundo plano.

La respuesta correcta es: Para consultas inmediatas, como verificar disponibilidad o asignar choferes, se usa comunicación síncrona.

**Pregunta 63**

Incorrecta

Se puntuá 0,00 sobre 0,03

En una aplicación distribuida que sigue el modelo Cliente-Servidor, mencione los componentes principales.

Respuesta: Los componentes principales en una aplicación distribuida c ×

La respuesta debe incluir el **lado cliente** (que realiza solicitudes), el **lado servidor** (que procesa las peticiones) y el **protocolo de comunicación** que se utiliza entre ambos.

La respuesta correcta es: cliente, servidor, protocolo de comunicación

**Pregunta 64**

Finalizado

Se puntuá como 0 sobre 0,03

Defina el concepto de microservicios y explique una de sus principales ventajas frente a la arquitectura monolítica.

Los **microservicios** son un enfoque arquitectónico en el que una aplicación se divide en servicios pequeños e independientes, cada uno con una funcionalidad específica. Estos servicios se comunican entre sí a través de APIs bien definidas y pueden desarrollarse, implementarse y escalarse de manera autónoma.

**Ventaja principal frente a la arquitectura monolítica:**

Una de las principales ventajas de los microservicios es su **escalabilidad independiente**. A diferencia de la arquitectura monolítica, donde toda la aplicación debe escalarse como un todo, los microservicios permiten escalar únicamente los componentes que lo necesitan.

La respuesta debe definir los microservicios como una forma de diseñar aplicaciones en las que la funcionalidad se divide en pequeños servicios independientes y destacar ventajas como la escalabilidad, facilidad de mantenimiento y despliegue independiente.

**Pregunta 65**

Finalizado

Se puntuá como 0 sobre 0,03

Explique al menos dos ventajas y dos desventajas de utilizar aplicaciones distribuidas.

**Ventajas:**

1. **Escalabilidad:** Las aplicaciones distribuidas pueden manejar un aumento en la carga de trabajo al añadir más nodos a la red, lo que permite una expansión eficiente.
2. **Disponibilidad:** Al distribuir los componentes en múltiples nodos, el sistema puede seguir funcionando incluso si uno de los nodos falla, mejorando la tolerancia a fallos.

**Desventajas:**

1. **Complejidad:** La gestión y coordinación de múltiples nodos requiere herramientas avanzadas y puede ser más difícil de implementar y mantener.
2. **Latencia:** La comunicación entre nodos puede generar retrasos, especialmente en redes con alta carga o en sistemas distribuidos geográficamente.

Se espera mencionar ventajas como la ejecución concurrente en varios nodos, mayor confiabilidad y escalabilidad; y desventajas como la complejidad en la sincronización, control de acceso y propagación lenta de información en algunos casos.

**Pregunta 66**

Incorrecta

Se puntuá 0,00 sobre 0,03

Explique brevemente la diferencia principal entre RPC y RMI.

Respuesta: La diferencia principal es que RPC (Remote Procedure Call) : X

RPC es un método general para llamadas remotas en diferentes plataformas, mientras que RMI es específico de Java para invocar métodos en objetos remotos en la JVM.

La respuesta correcta es: RPC es genérico; RMI es propio de Java para invocar métodos remotos

**Pregunta 67**

Correcta

Se puntuá 0,04 sobre 0,04

En el teorema CAP, "disponibilidad" significa que si un nodo falla, todo el sistema se detiene, pues no hay copias ni réplicas.

- Verdadero  
 Falso ✓

Correcto.

La disponibilidad implica que el sistema responde aun en caso de fallos, aunque pudiera devolver datos ligeramente desactualizados.

La respuesta correcta es 'Falso'

**Pregunta 68**

Correcta

Se puntuá 0,04 sobre 0,04

La **replicación** de bases de datos siempre requiere que todas las copias estén sincronizadas en tiempo real, sin permitir ningún retraso en la actualización de los nodos réplicas.

- Verdadero  
 Falso ✓

Correcto.

Existen estrategias de replicación asincrónica que permiten retrasos controlados.

La respuesta correcta es 'Falso'

**Pregunta 69**

Finalizado

Se puntuá como 0 sobre 0,03

Describa los pasos principales para publicar un servicio API RESTful desarrollado en .NET en un entorno Linux, incluyendo la configuración de publicación, traslado de archivos y creación de un servicio systemd.

1. Publica la app con dotnet publish en modo Release.
2. Traslada los archivos generados al servidor Linux (vía SCP o FTP).
3. Crea un archivo de servicio systemd para ejecutar la app como servicio.
4. Habilita y arranca el servicio con systemctl enable y systemctl start.

Se espera que se mencionen pasos como: configurar el proyecto para publicar (seleccionar carpeta destino, configuración de Release y runtime linux-x64), copiar los archivos publicados al servidor Linux, configurar el firewall (abrir puertos), crear un archivo de servicio systemd para ejecutar el servicio con dotnet, y activar el servicio para que arranque automáticamente.

**Pregunta 70**

Correcta

Se puntuá 0,03 sobre 0,03

Según el teorema CAP, en un sistema distribuido no es posible garantizar simultáneamente:

Seleccione una:

- a. Consistencia, disponibilidad y tolerancia a particiones. ✓ Correcto.
- b. Almacenamiento, escalabilidad y rendimiento.
- c. Consistencia, aislamiento y durabilidad.
- d. Coherencia, disponibilidad y partición.

El teorema CAP señala que no se puede garantizar consistencia, disponibilidad y tolerancia a particiones al mismo tiempo.

La respuesta correcta es: Consistencia, disponibilidad y tolerancia a particiones.