

SISTEMAS OPERATIVOS DISTRIBUIDOS

OBJETOS DISTRIBUIDOS E INVOCACION DE METODOS

ALUMNO: ERICK DANIEL GALLEGOS LOPEZ

DOCENTE: ANDRES ALEJANDRO REYES MOLINA

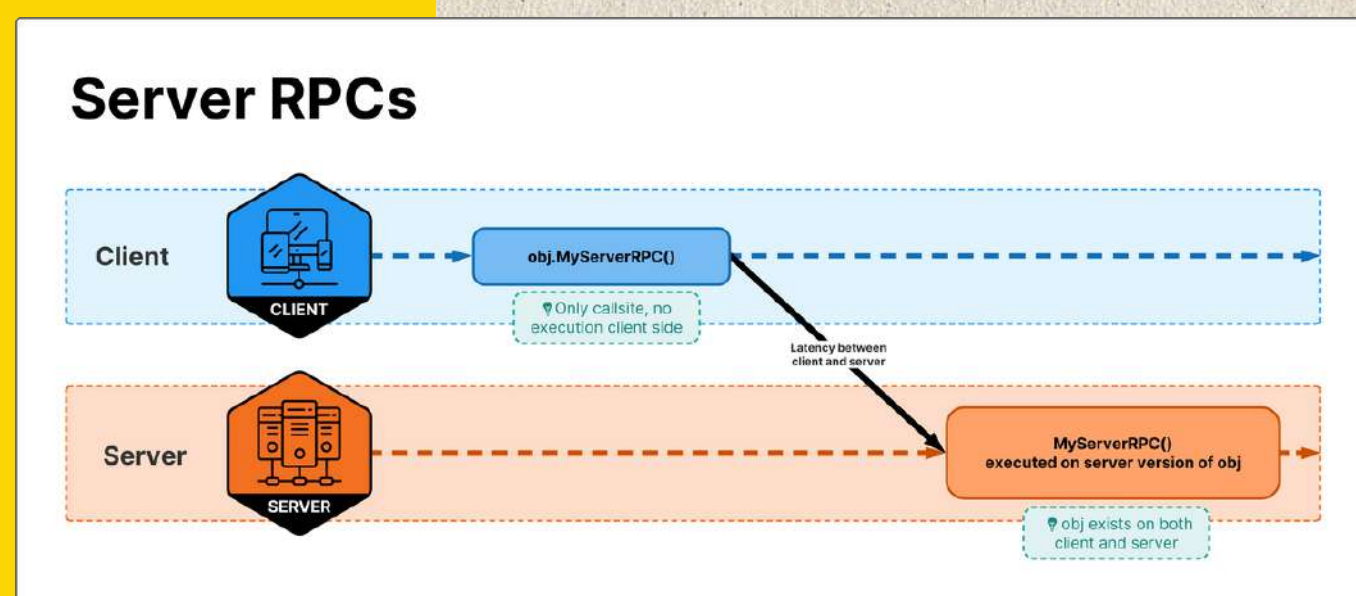


LLAMADA A PROCEDIMIENTO REMOTO RPC

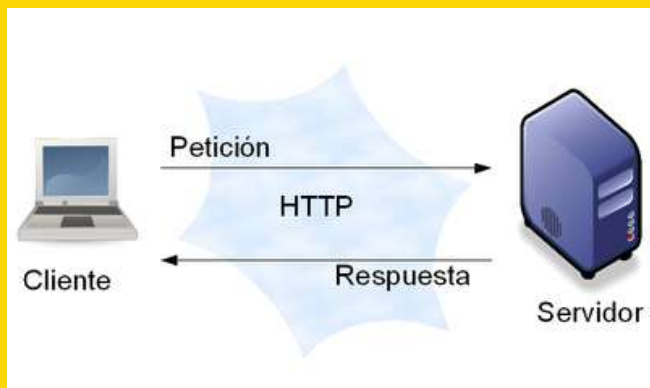
¿QUE ES?

ES UN PROGRAMA QUE UTILIZA UNA COMPUTADORA PARA EJECUTAR CODIGO EN OTRA MAQUIA REMOTA SIN TENER QUE PRECUPARCE POR LAS COMUNICACIONES ENTRE AMBAS . GRACIAS A EL RPC EL PROGRAMADOR NO TINES QUE ESTAR PENDIENTE DE LAS COMUNICACIONES, ESTANDO ENCAPSULADAS DENTRO DE AS RPC

HUY EN DIA SE ESTA UTILIZANDO EL XML COMO LENGUAJE PARA EFINIR EL IDL Y EL HTTP COMO PROTOCOLO DE APLICACION DANO LUGAR A LO QUE SE CONOCE COMO SERVIDOR WEB. EJEMPLOS DE ESTOS PUEDEN SER SOAP O XML- RPC



RPC ACTÚA COMO UN PROTOCOLO DE TRANSPORTE DE BAJO NIVEL PARA TRANSPORTAR PAQUETES DE DATOS ENTRE PROGRAMAS EN COMUNICACIÓN. PARA ELLO, UTILIZA UN MODELO CLIENTE-SERVIDOR



EL PROGRAMA SOLICITANTE SE DENOMINA CLIENTE, MIENTRAS QUE EL PROGRAMA PROVEEDOR DEL SERVICIO SE DENOMINA SERVIDOR. AL IGUAL QUE UNA LLAMADA A UN PROCEDIMIENTO LOCAL, UNA RPC ES UNA OPERACIÓN SÍNCRONA QUE REQUIERE LA SUSPENSIÓN DEL PROGRAMA SOLICITANTE HASTA QUE SE DEVUELVAN LOS RESULTADOS DEL PROCEDIMIENTO REMOTO.



EL LENGUAJE DE DEFINICIÓN DE INTERFAZ (IDL), EL LENGUAJE DE ESPECIFICACIÓN UTILIZADO PARA DESCRIBIR LA INTERFAZ DE PROGRAMACIÓN DE APLICACIONES (API) DE UN COMPONENTE DE SOFTWARE

EN ESTE CASO, IDL PROPORCIONA UN PUENTE ENTRE LAS MÁQUINAS EN AMBOS EXTREMOS DEL ENLACE QUE PODRÍAN USAR DIFERENTES SISTEMAS OPERATIVOS Y LENGUAJES DE PROGRAMACIÓN. ESTO SIGNIFICA QUE RPC ES COMPATIBLE CON SISTEMAS OPERATIVOS COMO WINDOWS, UNIX Y MACOS.



```
public class GreeterService : Greeter.GreeterBase
{
    private readonly ILogger<GreeterService> _logger;
    public GreeterService(ILogger<GreeterService> logger)
    {
        _logger = logger;
    }

    public override Task<HelloReply> SayHello>HelloRequest request, ServerCallContext>
    {
        return Task.FromResult(new HelloReply
        {
            Message = "Hello " + request.Name
        });
    }
}
```

PROCESO DE UNA RPC

1. EL CLIENTE LLAMA AL STUB DEL CLIENTE. LA LLAMADA E UN PROCEDIMIENTO LOCAL CUYOS PARAMETROS SE INSERTAN EN LA PILA DE LA FORMA HABITUAL
2. EL STUB DEL CLIENTE EMPAQUETA LOS PARAMETROS DEL PROCEIMIENTO EN UN MENSAJE Y REALIZA UNA LLAMADA AL SISTEMA PARA ENVIARLO (SEREALIZACION)
3. EL SISTEMA OPERATIVO LOCAL, EL CLIENTE ENVIA EL MENAJE DEDE EL DISPOSITIVO CLIENTE AL DISPOSITIVO SERVIDOR REMOTO
4. LA CAPA DE TRANPORTE DEL SISTEMA OPERATIVO DEL SERVIDOR ENTRA ANTES AL STUB DEL SERVIDOR
5. EL TROZO DEL SERVIDOR DESCROPIME LOS PARAMETROS DEL MENSAJE
6. AL FINALIZAR EL PROCEDIMIENTO DEL SERVIDOR, REGRESA AL STUB DEL SERVIDOR, ORDENANDO LOS VALORES DE RETORNO EN UN MENSAJE. EL STUB DEL SERVIDOR ENVÍA EL MENSAJE A LA CAPA DE TRANSPORTE.
7. LA CAPA DE TRANPORTE ENVIA EL MENSAJE A LA CAPA DE TRANPORTE DEL CLIENTE, QUE LUEGO DEVUELVE EL MENSAJE AL STUB DEL CLIENTE
8. EL STUB DEL CLIENTE DESORGANIZA LOS PARAMETROS DE RETORNO Y LA EJECUCION REGRESA AL LLAMADOR

TIPOS DE RPC

EXISTEN VARIOS MODELOS DE RPC E IMPLEMENTACIONES DE COMPUTACION DISTRIBUIDA.. EL INSTITUTO DE INGENIEROS ELECTRICOS Y ELECTRONICOS DEFINE RPC EN SU ESPECIFICCION DE LLAMADA A PROCEDIMIENO REMOTO ISO " ISO/IEC 11579:1996".
ALGUNOS EJEMPLOS DE CONFIGURACIONES DE RPC INCLUYEN LOS SIGUIENTES :

- SÍNCRONO :ESTE ES EL MÉTODO ESTÁNDAR DE RPC. EL CLIENTE REALIZA UNA LLAMADA Y ESPERA LA RESPUESTA DEL SERVIDOR.
- SINBLOQUEO: EL CLIENTE REALIZA UNA LLAMADA, PERO EN LUGAR DE ESPERAR UNA RESPUESTA, CONTINÚA CON SU PROPIO PROCESAMIENTO.
- MODO POR LOTES: UN CLIENTE ENVÍA VARIAS LLAMADAS SIN BLOQUEO EN UN GRUPO.
- DIFUSIÓN. UN CLIENTE ENVÍA UN MENSAJE A VARIOS SERVIDORES Y RECIBE TODAS LAS RESPUESTAS RESULTANTES.
-

JAVA RMI

ES UN MECANISMO OFRECIDO POR JAVA PARA INVOCAR UN METDOD DE MANERA REMOTA, RMI SE CARACTRIZA POR LA FACILISAD DE SU USO EN ÑA PROGRAMCION POR ESTAR ESPECIFICAMENTE DISEÑADO PARA JAVA GRACIAS RMI UN PROGRAMA JAVA PUEDE EXPORTAR UN OBJETO, CON LO QUE DICHO ONJETO ESTARA ACCSECIBLE A TRAVEZ DE LA RED Y EL PROGRAMA PERMANECE A LA ESPERA DE PETICIONES DE UN PUERTO TCP

EN COMPUTACIÓN, RMI (REMOTE METHOD INVOCATION) SIGNIFICA INVOCACIÓN REMOTA DE MÉTODOS. ES UN MECANISMO EN JAVA QUE PERMITE QUE UN OBJETO EN UNA MÁQUINA JAVA INVOQUE MÉTODOS DE OBJETOS EN OTRAS MÁQUINAS JAVA, INCLUSO SI ESTÁN EN DIFERENTES SISTEMAS FÍSICOS O REDES. BÁSICAMENTE, FACILITA LA CREACIÓN DE APLICACIONES DISTRIBUIDAS DONDE OBJETOS REMOTOS PUEDEN SER INVOCADOS COMO SI FUERAN LOCALES.

PRIMER CAPA

SEGUNDA CAPA

ARQUITECTURA

TERCER CAPA

CUARTA CAPA

LA PRIMERA CAPA ES LA APLICACION Y SE CORRESPONDE CON LA IMPLEMENTACION REAL DE LAS APLICACIONES CLINTES SERVIDOR. AQUI TIENEN LUAR LAS LLAMADAS A ALTO NIVEL PARA ACCEDERY EXPORTAR OBJETOS REMOTOS. JAVA.RMI.REMOTE SE USA BASICAMENTE PARA MRCAR UN OBJTO COMO REMOTAMENTE ACCESIBLE.

LA CAPA 2 ES LA CAPA PROXY, O CAPA STUB-SKELETON. ESTA CAPA ES LA QUE INTERACTÚA DIRECTAMENTE CON LA CAPA DE APLICACIÓN. TODAS LAS LLAMADAS A OBJETOS REMOTOS Y ACCIONES JUNTO CON SUS PARÁMETROS Y RETORNO DE OBJETOS TIENEN LUGAR EN ESTA CAPA.

ES LA DE REFERENCIA Y ES RESPONSABLE DEL MNEJO DE LA PARTE SEMANTICA DE LAS INVOCACIONES REMOTAS. TAMBIEN ES LA RESPONSABLE DE LA GESTIO DE LA REPLICACION DE OBJETOS Y REALIZACION E TAREAS ESPECIFICAS DE LA IMPLEMENTACION CON LOS OBJETOS REMOTOS. EN ESTA CAPA SE ESPERA UNA CONEION DE TIPO STREAM DESDE LA APA DE TRANSPORTE

ES LA RESPONSABLE DE REALIZAR LAS CONEXIONES NESESARIAS Y MANEJO DEL TRANPORTE DE LOS DATOS DE UNA MAQUINA A OTRA. EL PROTOCOLO DE TRANSPORTE SUB YACENTE PARA RMI

LO PRIMERO QUE TENEMOS QUE HACER ES UNA INTERFACE CON LOS MÉTODOS QUE QUEREMOS QUE SE PUEDAN LLAMAR REMOTAMENTE. ESTA INTERFACE SERÍA COMO LA SIGUIENTE:

```
import java.rmi.Remote;
public interface InterfaceRemota extends Remote
{
    public int suma (int a, int b) throws java.rmi.RemoteException;
}
```

LA FORMA SENCILLA DE HACER ESTO ES HACER QUE LA CLASE HEREDE DE UNICASTREMOTEOBJECT. SERÍA MÁS O MENOS LA SIGUIENTE:

```
import java.io.Serializable;
public class ObjetoRemoto extends UnicastRemoteObject implements
InterfaceRemota
{
    public int suma(int a, int b)
    {
        System.out.println ("sumando " + a + " + " + b + "...");
        return a+b;
    }
}
```

Característica	RPC	RMI
Lenguaje de programación	Independiente del lenguaje (comúnmente usado con C)	Específico de Java
Complejidad	Generalmente más simple y con menos funciones.	Más complejo con más funciones.
Tipo de llamadas	Llamadas de función	Llamadas a métodos de objetos
Codificación de datos	Utiliza formatos estandarizados como XDR	Utiliza la serialización incorporada de Java
Transparencia de la red	Sí, pero menos que RMI	Sin fisuras en el entorno Java
Manejo de excepciones	Básico, principalmente relacionado con la red y la codificación de datos.	Rich, permite el manejo de todas las excepciones de Java

DETECCIÓN Y RESPUESTA EXTENDIDAS, A MENUDO ABREVIADA COMO XDR, ES UNA PLATAFORMA UNIFICADA DE INCIDENTES DE SEGURIDAD QUE USA INTELIGENCIA ARTIFICIAL Y AUTOMATIZACIÓN.

SOPORTE DEL SISTEMA OPERATIVO

el soporte del sistema operativo se refiere al conjunto de servicios y actualizaciones que un proveedor ofrece para garantizar el correcto funcionamiento y la seguridad de un sistema operativo, este incluye actualizaciones de seguridad, corrección de errores y asistencia técnica. el fin del soporte implica que el sistema operativo dejara de recibir estas actualizaciones y asistencias, lo cual puede dejar al sistema vulnerable a problemas de seguridad y errores. tal el caso de cuando whatsapp dejó de dar su servicios a ciertos sistemas operativos antiguos y quedar obsoletos.



MECANISMOS DE SINCRONIZACIÓN

LOS MECANISMOS DE SINCRONIZACIÓN SON HERRAMIENTAS QUE PERMITEN COORDINAR LA EJECUCIÓN DE MÚLTIPLES PROCESOS O HILOS PARA ACCEDER A RECURSOS COMPARTIDOS DE MANERA SEGURA Y EVITAR PROBLEMAS COMO CONDICIONES DE CARRERA. ESTOS MECANISMOS ASEGURAN LA CONSISTENCIA DE LOS DATOS Y LA CORRECTA EJECUCIÓN DE LAS OPERACIONES EN ENTORNOS CONCURRENTES.

TIPOS DE SISTEMA OPERATIVO

SOPORTE TECNICO: ASISTENCIA PARA RESOLVER PROBLEMAS O DUDAS SOBRE EL FUNCIONAMIENTO DEL SISTEMA OPERATIVO.

ACTUALIZACIONES E SEGURIDAD: PARCHES Y MEJORAS QUE CORRIJEN VULNERABILIDADES Y ERRORES QUE PODRIAN SER EXPLOTADOS POR MALWARE O CIBERATAQUES.

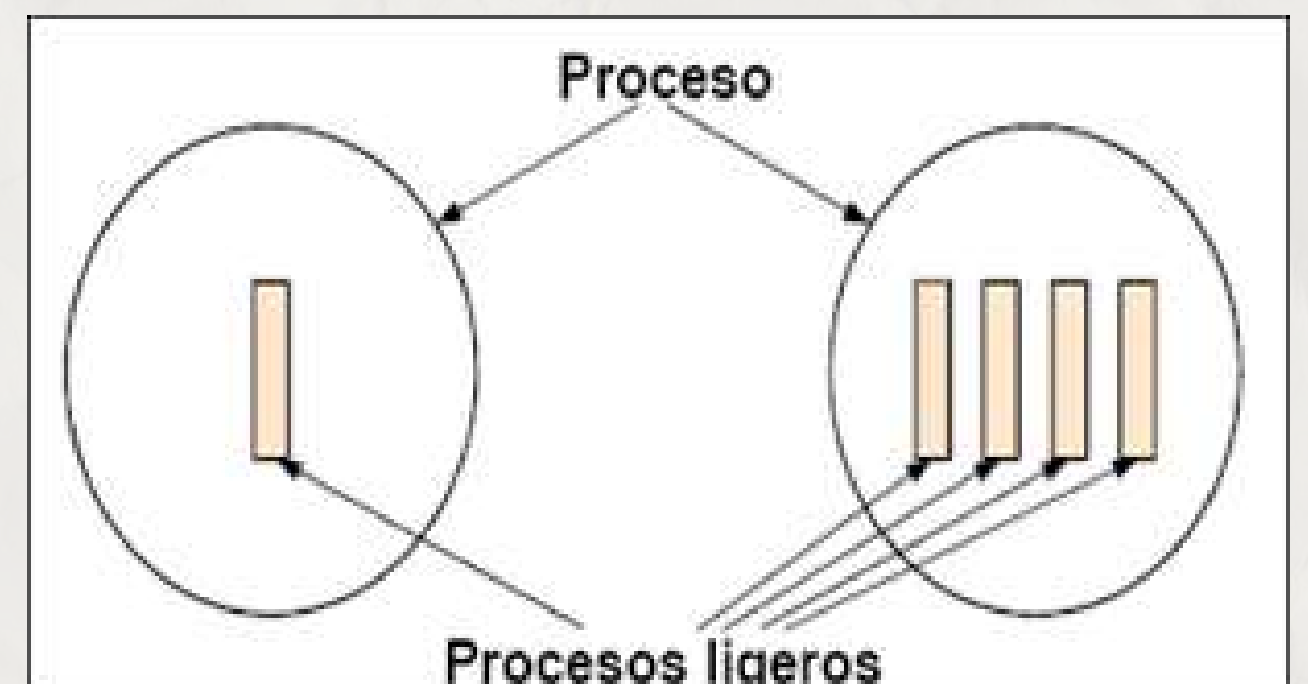
CORRECCIONES DE ERRORES: SOLUCIONES A PROBLEMAS ESPECÍFICOS QUE AFECTAN AL FUNCIONAMIENTO DEL SISTEMA OPERATIVO.

EJEMPLOS

- MICROSOFT FINALIZÓ EL SOPORTE DE WINDOWS 7 EN ENERO DE 2020 Y FINALIZARÁ EL SOPORTE DE WINDOWS 10 EN OCTUBRE DE 2025.
- APPLE TAMBIÉN FINALIZA EL SOPORTE DE VERSIONES ANTIGUAS DE MACOS PERIÓDICAMENTE. ESTO QUIERE DECIR QUE PUES LOS DISPOSITIVOS CON LOS SISTEMAS OPERATIVOS DICHO ANTES PUEDEN PRESENTAR FALLAS Y PUEDEN SER HACKEABLES Y PERDER INFORMACION.



EN SISTEMAS OPERATIVOS, UN HILO (O THREAD) ES LA UNIDAD DE EJECUCIÓN MÁS PEQUEÑA DENTRO DE UN PROCESO. ES UNA SECUENCIA DE INSTRUCCIONES QUE PUEDE SER EJECUTADA CONCURRENTEMENTE CON OTRAS DENTRO DEL MISMO PROCESO. PIENSA EN UN PROCESO COMO UN PROGRAMA EN EJECUCIÓN, Y LOS HILOS COMO LAS DIFERENTES TAREAS QUE ESE PROGRAMA PUEDE REALIZAR SIMULTÁNEAMENTE.



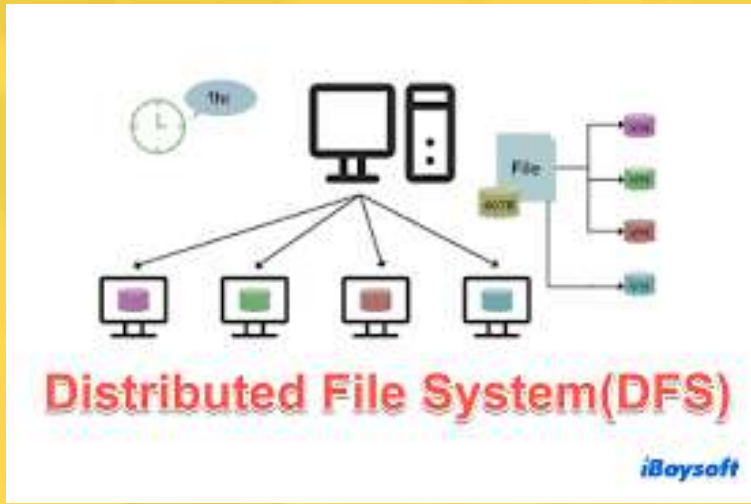
PARA QUE ESTO QUEDE CLARO VAMOS HABLAR EN TÉRMINOS SENCILLOS, UN PROCESO ES UN PROGRAMA EN EJECUCIÓN, MIENTRAS QUE UN HILO ES UNA UNIDAD DE EJECUCIÓN MÁS PEQUEÑA DENTRO DE UN PROCESO, EN POCAS PALABRAS UN PROCESO TIENE SU PROPIO ESPACIO Y LE COMPARTE A LOS HILOS RECURSOS Y MEMORIA



LOS SISTEMAS OPERATIVOS UTILIZAN PROCESOS E HILOS PARA USAR MULTITAREA

SISTEMAS DE ARCHIVOS DISTRIBUIDOS.

UN SISTEMA DE ARCHIVO DISTRIBUIDO (POR SUS SIGLAS EN INGLES) PERMITE QUE MULTIPLES COMPUTADORAS ACCEDAN Y PROCECEN DATOS ALMACENADOS EN UNA RED, PRESENTANDOLOS A USUARIOS Y APLICACIONES COMO SI ESTUVIERAN EN UN SOLO SISTEMA DE ARCHIVOS



UN DFS ES UN TIPO DE SISTEMA DISTRIBUIDO DONDE LOS ARCHIVOS SE ALMACENAN EN MULTIPLES SERVIDORES O UBICACIONES Y SE ACCEDEN A ELLOS COMO SI FUERAN PARTE DE UN UNICO SISTEMA, SE ACCE DE FORMA REMOTA DE DONDE SEA QUE ESTE EL USUARIO MEJORANDO LA CONFIABILIDAD, LA ECALIBILIDAD Y LA COLABORACIÓN

BENEFICIOS DE UN DFS

- ACCESO Y COLOBORACION EFICIENTE: LOS USUARIOS PUEDEN ACCEDER A LOS ARCHIVOS DE MANERA TRANSPARENTES, SIN PREOCUPARSE POR SU UBICACION FISICA, LO QUE FACILITA LA COLABORACIÓN EN PROYECTOS COMPARTIDOS
- ESCABILIDAD : LOS DFS PUEDEN ECALAR FCILMENTE PARA ADAPTARCE A LAS NESECIDADES CAMBIANTES DE ALMACENAMIENTO Y PROCESAMIENTOS DE DATOS, AÑDIENDO O ELIMINANDO NODOS SEA NESESARIOS
- REDUNDANCIA Y CONFIABILIDAD: AL DISTRIBUIR LOS DATOS ENTRE MÚLTIPLES SERVIDORES, UN DFS PUEDE TOLERAR FALLOS EN ALGUNOS NODOS, EVITANDO LA PÉRDIDA DE DATOS Y GARANTIZANDO LA CONTINUIDAD DEL SERVICIO.

DESVENTAJAS

- COMPLEJIDAD: LA IMPLEMENTACIÓN Y GESTIÓN DE UN DFS PUEDEN SER COMPLEJAS, ESPECIALMENTE EN ENTORNOS GRANDES Y HETEROGÉNEOS.
- CUELLOS DE BOTELLA: LA SINCRONIZACIÓN Y REPLICACIÓN DE DATOS ENTRE LOS NODOS PUEDEN GENERAR CUELLOS DE BOTELLA EN EL RENDIMIENTO, ESPECIALMENTE EN REDES CON ANCHO DE BANDA LIMITADO.
- SEGURIDAD: LA DISTRIBUCIÓN DE DATOS EN MÚLTIPLES NODOS PUEDE REQUERIR MEDIDAS DE SEGURIDAD MÁS ROBUSTAS PARA PROTEGER LA INFORMACIÓN SENSIBLE.

ARQUITECTURA DEL SERVICIO DE ARHIVO

LA ARQUITECTURA DE UN SISTEMA DE ARCHIVOS SE REFIERE A SU ESTRUCTURA Y ORGANIZACIÓN INTERNAS, INCLUYENDO LAS ESTRUCTURAS DE DATOS, LOS ALGORITMOS Y LOS MECANISMOS UTILIZADOS PARA ALMACENAR, ACCEDER Y GESTIONAR ARCHIVOS Y DIRECTORIOS EN UN MEDIO DE ALMACENAMIENTO. UNA ARQUITECTURA DE SISTEMA DE ARCHIVOS BIEN DISEÑADA ES ESENCIAL PARA LOGRAR UN RENDIMIENTO EFICIENTE, FIABILIDAD Y ESCALABILIDAD.

- ESTRUCTURA DE DIRECTORIOS Y ARCHIVOS: ORGANIZACIÓN JERÁRQUICA DE ARCHIVOS Y DIRECTORIOS PARA FACILITAR LA NAVEGACIÓN Y LA BÚSQUEDA.
- SISTEMA DE ARCHIVOS: EL SISTEMA QUE GESTIONA LA FORMA EN QUE SE ALMACENAN, RECUPERAN Y MODIFICAN LOS DATOS EN EL ALMACENAMIENTO FÍSICO.
- METADATOS: INFORMACIÓN DESCRIPTIVA SOBRE LOS ARCHIVOS, COMO NOMBRE, TIPO, FECHA DE CREACIÓN, ETC., QUE FACILITA SU ORGANIZACIÓN Y BÚSQUEDA.
- ÍNDICES: ESTRUCTURAS QUE PERMITEN UN ACCESO RÁPIDO A LOS ARCHIVOS Y A LA INFORMACIÓN RELACIONADA.
- CONTROL DE ACCESO: MECANISMOS PARA PROTEGER LOS ARCHIVOS Y DIRECTORIOS DEL ACCESO NO AUTORIZADO.
- CAPA DE ABSTRACCIÓN: EN SISTEMAS MÁS COMPLEJOS, COMO SISTEMAS DE GESTIÓN DOCUMENTAL, UNA CAPA DE ABSTRACCIÓN PUEDE SEPARAR LA LÓGICA DE LA APLICACIÓN DE LA GESTIÓN FÍSICA DE LOS ARCHIVOS, PERMITIENDO FLEXIBILIDAD Y ESCALABILIDAD.

TIPOS DE ARQUITECTURAS

- ARQUITECTURA MONOLÍTICA: UN ÚNICO COMPONENTE QUE CONTIENE TODA LA LÓGICA DEL SERVICIO DE ARCHIVOS. PUEDE SER SIMPLE DE IMPLEMENTAR, PERO MENOS FLEXIBLE Y ESCALABLE.
- ARQUITECTURA ORIENTADA A SERVICIOS (SOA): DIVIDE EL SERVICIO EN COMPONENTES MÁS PEQUEÑOS Y REUTILIZABLES (SERVICIOS) QUE INTERACTÚAN ENTRE SÍ. PERMITE MAYOR FLEXIBILIDAD, ESCALABILIDAD Y REUTILIZACIÓN.
- ARQUITECTURA DE MICROSERVICIOS: UNA EVOLUCIÓN DE SOA, DONDE LOS SERVICIOS SON MÁS PEQUEÑOS Y AUTÓNOMOS, LO QUE PERMITE UNA MAYOR GRANULARIDAD Y FLEXIBILIDAD.

bibliografia

<https://www.techtarget.com/searchapparchitecture/definition/Remote-Procedure-Call->

[RPC#:~:text=A%20Remote%20Procedure%20Call%20\(RPC,to%20understand%20the%20network's%20details](https://www.techtarget.com/searchapparchitecture/definition/Remote-Procedure-Call-)

<https://www.thejat.in/learn/file-system-architecture#:~:text=reliability%2C%20and%20scalability.-,The%20architecture%20of%20a%20file%20system%20refers%20to%20its%20internal,integrity%20and%20ease%20of%20accessibility.>

antologia

https://es.wikipedia.org/wiki/Sistema_de_archivos_distribuido

<https://es.scribd.com/document/601902738/Mecanismos-de-sincronizacion>

