UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA

Sistemas Distribuidos

Alumnos: Bolaños Guerrero Julián y Zurita Cámara Juan Pablo
Profesor: Gunnar Wolf
Asignatura: Sistemas Operativos
Grupo: 06
Semestre: 2024-2
Fecha de entrega: 16/04/2024

ÍNDICE

1. Introducción

A partir de la mitad de los 1980 ‘s, 2 avances tecnológicos empezaron a cambiar el panorama de la computación. El primero fue el desarrollo de microprocesadores cada vez más potentes. Imaginar que en un inicio las máquinas eran de 8 bits, pero poco después empezaron a surgir los CPUs de 16, 32 y 64 bits. El segundo avance fue la invención de redes computacionales de alta velocidad. Las redes de área local (LAN) permiten que miles de máquinas dentro de un edificio se conecten de tal manera que pequeñas cantidades de información puedan transferirse en unos pocos microsegundos. Las redes de área extendida (WAN) se extienden más allá de un edificio o de un gran recinto para incluir múltiples ubicaciones repartidas a lo largo de una zona geográfica concreta, o incluso del mundo.

Paralelo al desarrollo de máquinas cada vez más potentes y conectadas en red, hemos podido presenciar la miniaturización de sistemas informáticos, con quizás el teléfono inteligente como el resultado más impresionante. Equipados con sensores, mucha memoria y una potente CPU, estos dispositivos no son menos que computadoras completas.

El resultado de estas tecnologías es que ahora no solo es factible, sino fácil, armar un sistema informático compuesto por un gran número de computadoras interconectadas, ya sean grandes o pequeñas. Estas computadoras generalmente están geográficamente dispersas, por lo que suelen decirse que forman un **sistema distribuido**. El tamaño de un sistema distribuido puede variar desde un puñado de dispositivos hasta millones de computadoras. La red de interconexión puede ser cableada, inalámbrica o una combinación de ambas. Además, los sistemas distribuidos suelen ser altamente dinámicos, en el sentido de que las computadoras pueden unirse y salir, con la topología y el rendimiento de la red subyacente casi cambiando continuamente.

2. ¿Qué es un sistema distribuido?

Un sistema distribuido es una colección de elementos informáticos autónomos que aparece ante sus usuarios como un único sistema coherente.

Esta definición se refiere a dos características distintivas de los sistemas distribuidos:

1. Es una colección de elementos informáticos, cada uno capaz de comportarse de manera independiente entre sí. Un elemento informático, usualmente referido como **nodo**, puede ser tanto un dispositivo de hardware como un proceso de software.

2. Los usuarios (ya sean personas o aplicaciones) creen que están tratando con un único sistema. Esto significa que, de una forma u otra, los nodos autónomos necesitan colaborar. Cómo establecer esta colaboración es el corazón del desarrollo de sistemas distribuidos.

2.1 Lo que no se dice

- Como consecuencia de tratar con nodos independientes, cada uno tendrá su propia noción de tiempo. En otras palabras, no siempre podemos asumir que existe algo como un reloj global.
- El esfuerzo por lograr un sistema único y coherente introduce un importante compromiso. Dado que no podemos ignorar el hecho de que un sistema distribuido consiste en múltiples nodos interconectados, es inevitable que en cualquier momento solo una parte del sistema falle. Esto significa que el comportamiento inesperado, por ejemplo, en el que algunas aplicaciones siguen ejecutándose correctamente mientras que otras se detienen por completo es una realidad con la que se tiene que lidiar. Aunque las fallas parciales son inherentes a cualquier sistema complejo, en los sistemas distribuidos son particularmente difíciles de ocultar. Esto llevó al ganador del Premio Turing, Leslie Lamport, a describir un sistema distribuido como "[...] uno en el que la falla de una computadora que ni siquiera sabías que existía puede hacer que tu propia computadora sea inutilizable".

3. ¿Por qué utilizar sistemas distribuidos?

- **Costo**. Comparado con una computadora sistema centralizado, compuesto por varios procesadores, el sistema distribuido está formado por varias computadoras juntas. Este tipo de infraestructura tiene una mejor relación precio/rendimiento.
- **Rendimiento**. Al distribuir la carga de trabajo se logra un uso óptimo de los recursos y permite operar de manera eficiente a pesar de la alta demanda.
- **Escalabilidad**. Los sistemas distribuidos están diseñados por defecto para ser escalables. Solo es necesario adicionar o modificar el elemento de cómputo requerido. Por ejemplo: incrementa la carga de trabajo, se pueden añadir más estaciones de trabajo.
- **Confiabilidad**. Menos propenso a fallas completas o pérdida de datos cuando los datos están duplicados en varios nodos. Incluso si un nodo falla, el sistema puede seguir funcionando con los nodos restantes.
- **Distribución inherente**. Algunas aplicaciones, como el correo electrónico y la Web (donde los usuarios están dispersos por todo el mundo), son naturalmente

distribuidas. Esto incluye casos en los que los usuarios están geográficamente dispersos, así como cuando se necesitan compartir recursos únicos (por ejemplo, impresoras, datos).

A pesar de los diferentes beneficios que introducen los sistemas distribuidos, todavía existen diferentes retos que deben ser resueltos como los siguientes:

- **Software:** La naturaleza distribuida del sistema introduce una complejidad adicional en el desarrollo, implementación y mantenimiento del software. Por ejemplo: La comunicación y sincronización entre distintos elementos con diferencias ya sea en software o en hardware.
- **Redes:** Los sistemas distribuidos son propensos a errores de red que resultan en fallos de comunicación. La información puede no ser entregada o no estar en la secuencia correcta.
- **Seguridad:** Comparado con los sistemas centralizados, los sistemas distribuidos enfrentan más amenazas de seguridad. Puede ser más difícil y vulnerable gestionar el control de acceso, la autenticación y confidencialidad de los datos en varios nodos debido a posibles fallos.
- **Tolerancia a fallas:** Con un mayor número de computadoras, redes y otros periféricos que componen todo el sistema, hay más elementos que pueden fallar. Los sistemas distribuidos deben estar construidos para sobrevivir al fallo de algunos de sus elementos, lo que agrega aún más complejidad al software del sistema..

4. Metas de Diseño de un sistema distribuido

4.1 Transparencia

Es una característica de los sistemas distribuidos para ocultar al usuario la manera en que el sistema funciona o está construido, de tal forma que el usuario tenga la sensación de que todo el sistema está trabajando en una sola máquina local. Entre las principales transparencias deseables en un sistema distribuido están:

- La **transparencia de acceso** trata de ocultar las diferencias en la representación de datos y en la forma en que los objetos pueden ser accedidos. Por ejemplo, un sistema distribuido puede tener sistemas informáticos que ejecutan diferentes sistemas operativos, cada uno con sus propias convenciones de nomenclatura de archivos. Las diferencias en las convenciones de nomenclatura, las diferencias en las operaciones de archivo o las diferencias en cómo se llevará a cabo la comunicación a nivel de bajo nivel con otros procesos,

son ejemplos de problemas de acceso que preferiblemente deberían estar ocultos para los usuarios y las aplicaciones.

- La **transparencia de ubicación** se refiere al hecho de que los usuarios no pueden saber dónde está físicamente ubicado un proceso o recurso en el sistema.
- La **transparencia de replicación** trata de ocultar el hecho de que existen varias copias de un recurso, o que varios procesos están operando de alguna forma en modo sincronizado para que uno pueda tomar el control cuando otro falla.
- La **transparencia de concurrencia** oculta que un proceso o recurso puede ser compartido por varios usuarios. Por ejemplo, dos usuarios independientes pueden haber almacenado sus archivos en el mismo servidor de archivos o pueden estar accediendo a las mismas tablas en una base de datos compartida. En tales casos, es importante que cada usuario no note que el otro está haciendo uso del mismo recurso.
- La **transparencia de fallas.** Esto significa que un usuario o aplicación no nota que alguna parte del sistema falla en funcionar correctamente, y que posteriormente (y automáticamente) el sistema se recupera de ese fallo.

4.2 Escalabilidad

Una de las características de los sistemas distribuidos es su modularidad, lo que le permite una gran flexibilidad y posibilita su escalabilidad, definida como la capacidad del sistema para crecer sin aumentar su complejidad ni disminuir su rendimiento.

- **Escalabilidad en tamaño**: Un sistema puede ser escalable en cuanto a su tamaño, lo que significa que podemos agregar fácilmente más usuarios y recursos al sistema sin ninguna pérdida perceptible de rendimiento.
- **Escalabilidad geográfica**: Un sistema geográficamente escalable es aquel en el que los usuarios y recursos pueden estar muy separados, pero el hecho de que los retrasos en la comunicación puedan ser significativos apenas se nota.

Escalamiento vertical: Comprar hardware más costoso con un mejor rendimiento.
Escalamiento horizontal: Expandir el sistema añadiendo más máquinas.

4.3 Fiabilidad y tolerancia a fallos

La fiabilidad de un sistema puede definirse como su capacidad para realizar correctamente y en todo momento las funciones para las que se ha diseñado.

La tolerancia a fallos expresa la capacidad del sistema para seguir operando correctamente ante el fallo de alguno de sus componentes, enmascarando el fallo al usuario o a la aplicación. Por lo tanto, la tolerancia a fallos implica detectar el fallo, y continuar el servicio, todo ello de forma transparente para la aplicación (transparencia de fallos).

4.4 Seguridad

Es importante considerar todos los factores de riesgo a que se expone la información en un ambiente distribuido, por ello se deben de implementar los mecanismos de seguridad que permitan proteger esta información.

5. Tipos de sistemas distribuidos

5.1 Procesamiento distribuido

Cuando un único conjunto lógico de funciones de procesamiento se implementa en varios dispositivos físicos, de modo que cada uno realiza una parte del procesamiento total requerido. A menudo está acompañado por una base de datos distribuida.

5.1.1 Cluster

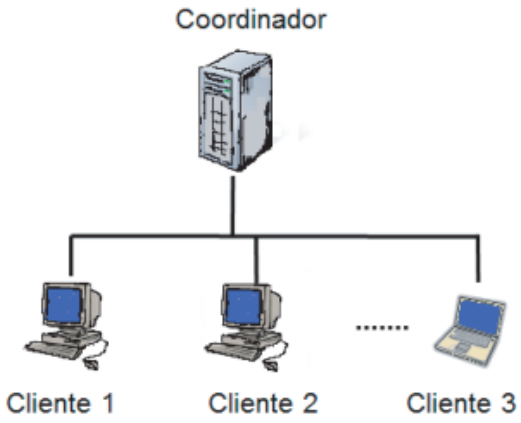

Figura 1. Ejemplo de cluster.

Consiste en una colección de nodos que son controlados y accedidos mediante un único nodo maestro. En prácticamente todos los casos, la computación en clúster se utiliza para la programación paralela, en la que un programa único (intensivo en cómputo) se ejecuta en paralelo en múltiples máquinas. El maestro típicamente maneja la asignación de nodos a un programa paralelo específico, mantiene una cola de trabajos enviados y proporciona una interfaz para los usuarios del sistema. Como tal, el

maestro realmente ejecuta el middleware[1] necesario para la ejecución de programas y la gestión del clúster, mientras que los nodos de cómputo están equipados con un sistema operativo estándar extendido con funciones de middleware típicas para comunicación, almacenamiento, tolerancia a fallos, etc. Una de sus características es la alta heterogeneidad de los nodos que lo componen.

5.2 Sistemas de información distribuidos

También conocida como base de datos distribuida, existe cuando los elementos de datos almacenados en múltiples ubicaciones están interrelacionados, o si un proceso en una ubicación requiere acceso a datos almacenados en otra ubicación.

- Base de datos particionada: Una sola copia de un conjunto de información dividida en fragmentos que residen en múltiples ubicaciones.
- Base de datos replicada: Conjunto de información donde todas o partes seleccionadas de este se copian en dos o más ubicaciones.

5.3 Pervasive computing

La nueva tendencia es incorporar microprocesadores en objetos cotidianos para que puedan comunicar información. Por ejemplo: teléfonos, smartwatches, asistentes virtuales, etc.

6. Estilos de arquitectura

6.1 Arquitectura de capas

El sistema está dividido en cierto número de capas, donde las capas superiores hacen uso de los servicios ofrecidos por las capas inferiores. De esta manera, una determinada capa ofrece una abstracción de software, sin que las capas superiores o inferiores a ésta deban de estar al tanto de los detalles de implementación.

6.2 Arquitectura Basada en Objetos

Esta arquitectura gira en torno a un arreglo de objetos débilmente acoplados. A diferencia de la arquitectura en capas, la arquitectura basada en objetos no tiene que seguir ningún paso en una secuencia. Cada componente es un objeto, y todos los

[1] El middleware es similar a un sistema operativo en un sistema distribuido de la misma manera que un sistema operativo es para una computadora: es un administrador de recursos que ofrece sus aplicaciones para compartir y desplegar eficientemente esos recursos a través de una red.

objetos pueden interactuar a través de una interfaz. Bajo la arquitectura basada en objetos, estas interacciones entre componentes pueden ocurrir mediante una llamada directa a método.

7. Arquitectura de Sistema

7.1 Arquitectura simple cliente-servidor

En el modelo básico cliente-servidor, los procesos en un sistema distribuido se dividen en dos grupos. Un servidor es un proceso que implementa un servicio específico, por ejemplo, un servicio de sistema de archivos o un servicio de base de datos. Un cliente es un proceso que solicita un servicio a un servidor enviándole una solicitud y posteriormente esperando la respuesta del servidor.

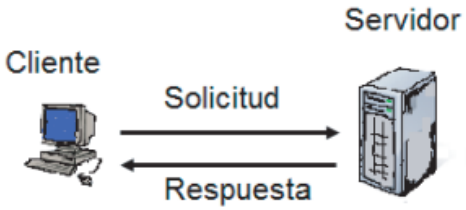

Figura 3. Ejemplo arquitectura simple cliente-servidor

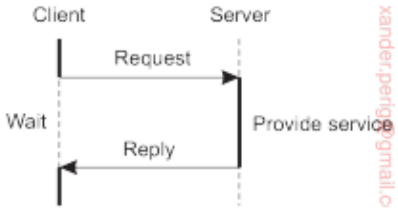

Figura 4. Interacción General cliente-servidor

Cuando un cliente solicita un servicio, simplemente empaqueta un mensaje para el servidor, identificando el servicio que desea, junto con los datos de entrada necesarios. El mensaje luego se envía al servidor. Este último, a su vez, siempre esperará una solicitud entrante, posteriormente la procesará y empaquetará los resultados en un mensaje de respuesta que luego se enviará al cliente. Esta interacción cliente-servidor se conoce como comportamiento de solicitud-respuesta.

7.2 Peer - To - Peer

En el modelo cliente-servidor tradicional, dos tipos de nodos son empleados: clientes y servidores. En este contexto, los clientes solo solicitan servicios y el servidor solo proporciona a los clientes el servicio apropiado.

En contraste, en los sistemas P2P no se requiere una infraestructura dedicada. Los servidores y clientes dedicados no existen, ya que cada peer puede tomar el papel tanto de servidor como de cliente al mismo tiempo.

8. Middleware

En el inicio de los sistemas distribuidos, todos los servicios proporcionados por los servidores debían de programarse a la medida. Así, servicios de acceso a bases de datos, de impresión y transferencias de archivos tenían que ser desarrollados por las propias aplicaciones. Quedó en evidencia la necesidad de crear servicios de uso más común por las aplicaciones, de tal manera que pudieran incluirse en todas las aplicaciones como software prefabricado.

El middleware es una capa de software separada que se coloca lógicamente sobre los sistemas operativos respectivos de las computadoras que forman parte del sistema.

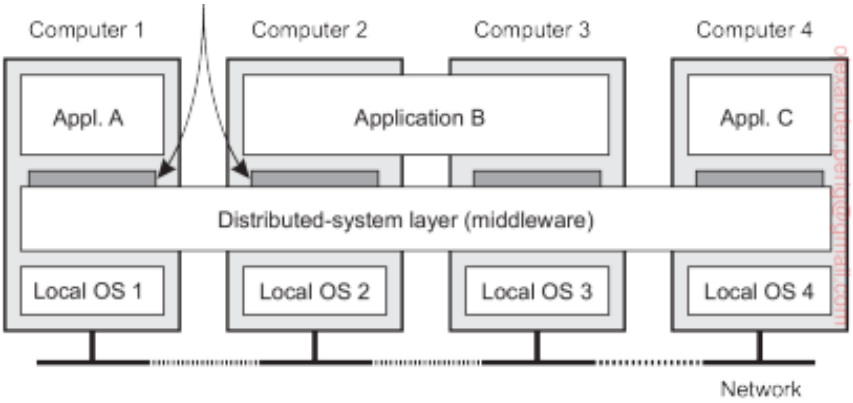

Figura 5

La Figura 5 muestra cuatro computadoras en red y tres aplicaciones, de las cuales la aplicación B está distribuida en las computadoras 2 y 3. Cada aplicación ofrece la misma interfaz. El sistema distribuido proporciona los medios, middleware, para que los componentes de una sola aplicación distribuida se comuniquen entre sí, pero también para permitir que diferentes aplicaciones se comuniquen. Al mismo tiempo, oculta, de la mejor manera y de manera razonable, las diferencias en hardware y sistemas operativos de cada aplicación.

En cierto sentido, el middleware es lo mismo para un sistema distribuido que lo que un sistema operativo es para una computadora: un administrador de recursos que ofrece sus aplicaciones para compartir y desplegar eficientemente esos recursos a través de una red. Además de la gestión de recursos, ofrece servicios que también se encuentran en la mayoría de los sistemas operativos, incluyendo:

- Instalaciones para la comunicación entre aplicaciones.
- Servicios de seguridad.
- Servicios de contabilidad.

- Ocultamiento y recuperación de fallos.

9. Comunicación

9.1 Modelo OSI

Para facilitar el manejo de los numerosos niveles y problemas involucrados en la comunicación, la Organización Internacional de Normalización (ISO) desarrolló un modelo de referencia que identifica claramente los diversos niveles involucrados, les da nombres estándar y señala el trabajo que debe hacer cada nivel.

Este modelo se llama el Modelo de Referencia de Interconexión de Sistemas Abiertos, comúnmente abreviado como ISO OSI o a veces simplemente el modelo OSI.

En el modelo OSI, la comunicación se divide en siete niveles o capas. Cada capa ofrece uno o más servicios de comunicación específicos a la capa superior. De esta manera, el problema de llevar un mensaje de A a B puede dividirse en piezas manejables, cada una de las cuales puede resolverse independientemente de las demás. Cada capa proporciona una interfaz a la capa superior. La interfaz consiste en un conjunto de operaciones que juntas definen el servicio que la capa está preparada para ofrecer.

Figura 6.

- **Capa física:** Se encarga de estandarizar cómo se conectan dos computadoras y cómo se representan los 0s y 1s.
- **Capa de enlace de datos:** Proporciona los medios para detectar y posiblemente corregir errores de transmisión, así como protocolos para mantener un emisor y un receptor en el mismo ritmo.

- **Capa de red:** Contiene los protocolos para enrutar un mensaje a través de una red de computadoras, así como protocolos para manejar la congestión.
- **Capa de transporte:** Principalmente contiene protocolos para admitir directamente aplicaciones, como aquellos que establecen comunicación confiable o admiten transmisión de datos en tiempo real.
- **Capa de sesión:** Proporciona soporte para sesiones entre aplicaciones.
- **Capa de presentación:** Prescribe cómo se representa los datos de una manera independiente de los hosts en los que se ejecutan las aplicaciones de comunicación.
- **Capa de aplicación:** Esencialmente, incluye todo lo demás: protocolos de correo electrónico, acceso web, protocolos de transferencia de archivos, entre otros.

9.2 Protocolos

Es un conjunto bien conocido de reglas y formatos que se utilizan para la comunicación entre procesos que realizan una determinada tarea. Se requieren dos partes:

- Especificación de la secuencia de mensajes que se han de intercambiar.
- Especificación del formato de los datos en los mensajes.

Un protocolo permite que componentes heterogéneos de sistemas distribuidos puedan desarrollarse independientemente, y por medio de módulos de software que componen el protocolo, haya una comunicación transparente entre ambos componentes. Es conveniente mencionar que estos componentes del protocolo deben estar tanto en el receptor como en el emisor.

9.2.1 Ejemplos de protocolos usados en los sistemas distribuidos

- **IP (Protocolo de Internet).** Protocolo de la capa de red, que permite definir la unidad básica de transferencia de datos y se encarga del direccionamiento de la información, para que llegue a su destino en la red.
- **TCP (Protocolo de Control de Transmisión)**. Protocolo de la capa de Transporte, que permite dividir y ordenar la información a transportar en paquetes de menor tamaño para su transporte y recepción.
- **HTTP (Protocolo de Transferencia de Hipertexto)**. Protocolo de la capa de aplicación, que permite el servicio de transferencia de páginas de hipertexto entre el cliente WEB y los servidores.
- **SMTP (Protocolo de Transferencia de Correo Simple)**. Protocolo de la capa de aplicación, que permite el envío de correo electrónico por la red.

- **POP3 (Protocolo de Oficina de Correo)**. Protocolo de la capa de aplicación, que permite la gestión de correos en Internet, es decir, le permite a una estación de trabajo recuperar los correos que están almacenados en el servidor.

9.3 Llamada de procedimiento remoto

Permite que un programa solicite la ejecución de una función o procedimiento en una computadora remota y reciba los resultados de vuelta como si estuviera llamando a una función local.

9.3.1 Operación del RPC

1. El cliente hace la llamada al procedimiento remoto mediante un mensaje a través de la red. Este se detiene ya que es un proceso síncrono, es decir, necesita una respuesta del servidor para poder continuar su ejecución. En esta llamada se incluye un *stub* (o resguardo en español) el cual se encarga de ajustar parámetros y direcciones de memoria de un ambiente, el cliente, a otro, el servidor.
2. El servidor recibe la petición y desempaqueta el mensaje para extraer la información necesaria para realizar la tarea. El stub ayuda a que el servidor sea capaz de convertir parámetros de una representación a otra (de ser necesario), para traducir direcciones de memoria de cliente a servidor, etc.
3. El servidor ejecuta la tarea.
4. El servidor crea un mensaje de respuesta para el cliente en el que incluye el resultado de la tarea que éste le pidió realizar.
5. El cliente recibe y desempaqueta el mensaje de respuesta del servidor. Continúa con su ejecución normal.

El *stub* es la pieza de código que le permite al servidor ejecutar la tarea que se le asignó. Se encarga de proveer la información necesaria para que el servidor convierta las direcciones de los parámetros que el cliente envió en direcciones de memoria válidos dentro del ambiente del servidor.

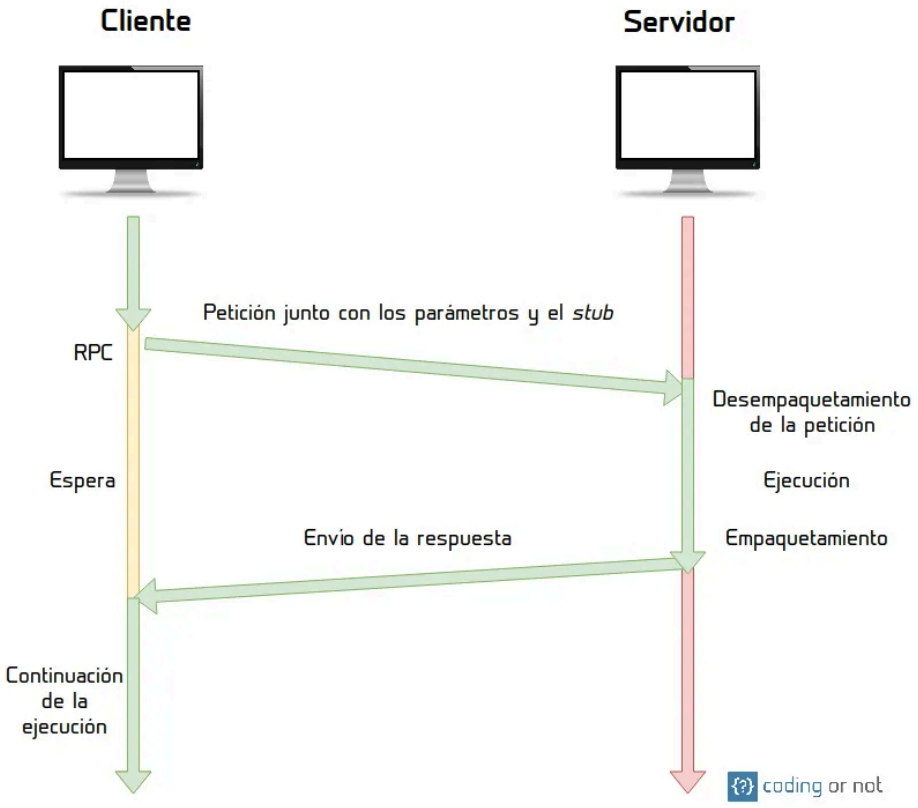

Figura 9.

9.4 Comunicación multicast

La comunicación multicast es un método de comunicación en redes de computadoras donde un único mensaje se envía a múltiples destinatarios simultáneamente. En lugar de enviar copias individuales del mensaje a cada destinatario, como en la comunicación unicast, el mensaje multicast se envía una vez y se entrega a múltiples receptores al mismo tiempo.

La comunicación multicast es útil en situaciones donde se necesita enviar la misma información a múltiples destinatarios, como en la transmisión de video en vivo, la difusión de actualizaciones de software o la distribución de contenido a grupos de usuarios.

10. Relación Sistemas Distribuidos con la materia de Sistemas Operativos

1. Gestión de recursos: Tanto los sistemas distribuidos como los sistemas operativos se ocupan de la gestión de recursos computacionales. Mientras que los sistemas operativos gestionan los recursos locales de una sola máquina, los sistemas

distribuidos extienden esta gestión para incluir recursos distribuidos en una red de computadoras.

2. Abstracción y encapsulación: Los sistemas operativos proporcionan abstracciones de hardware y software que permiten a las aplicaciones interactuar con el hardware subyacente de manera simplificada. Del mismo modo, los sistemas distribuidos abstraen los detalles de la red subyacente y proporcionan una interfaz uniforme para que las aplicaciones accedan a los recursos distribuidos de manera transparente.

3. Gestión de procesos y comunicación: Tanto los sistemas operativos como los sistemas distribuidos gestionan la ejecución de procesos y la comunicación entre ellos. Los sistemas operativos proporcionan mecanismos para la gestión de procesos locales y la comunicación interproceso, mientras que los sistemas distribuidos extienden estas capacidades para gestionar procesos distribuidos y la comunicación entre ellos a través de una red.

4. Servicios: Tanto los sistemas operativos como los sistemas distribuidos proporcionan servicios compartidos y abstracciones que facilitan el desarrollo de aplicaciones. Por ejemplo, los sistemas operativos proporcionan servicios como sistemas de archivos y sistemas de entrada/salida, mientras que los sistemas distribuidos pueden proporcionar servicios como sistemas de archivos distribuidos y middleware para la comunicación entre procesos distribuidos.

11. Conclusión

En conclusión tras haber realizado esta investigación nos dimos cuenta que este es un tema fascinante, pero vaya que es extenso y complicado. Si bien es cierto que posiblemente nos faltó más información por exponer y precisión en los temas expuestos, queremos dejar en claro que este trabajo es un primer acercamiento a los sistemas distributivos, abarcar todo lo relacionado con este tema nos tomaría cientos de páginas, justo como los libros con los que consultamos. Además, muchos de los conceptos no los entendemos por nuestra misma falta de conocimientos más avanzados de computación.

Terminamos este documento recordando a los lectores que este trabajo es un primer acercamiento al tema de sistemas distribuidos interpretado por estudiantes, por lo tanto, este trabajo está orientado a estudiantes que buscan tener una introducción quitando muchos de los tecnicismos que muchas fuentes de información pueden tener.

BIBLIOGRAFÍA

Fuentes de información principales

- Francisco de Asís López Fuentes. (2015). Sistemas Distribuidos: Universidad Nacional Metropolitana. Recuperado el 6 de abril del 2024 de http://dccd.cua.uam.mx/libros/archivos/03IXStream_sistemas_distribuidos.pdf
- Maarten van Steen Andrew S. Tanenbaum. (2018). Distributed Systems. Recuperado el 6 de abril del 2024 de http://www.dgma.donetsk.ua/docs/kafedry/avp/metod/van%20Steen%20-%20Distributed%20Systems.pdf
- Cinvestav Unidad Tamaulipas. Recuperado el 7 de abril del 2024 de https://www.tamps.cinvestav.mx/~vjsosa/clases/sd/sistemas_distribuidos_panorama.pdf

Fuentes de información secundarias

- The University of New South Wales, School of Computer Science & Engineering. Distributed Systems. Recuperado el 6 de abril del 2024 de https://www.cse.unsw.edu.au/~cs9243/20t3/lectures/intro-notes.pdf
- Alberto Lafuente, Mikel Larrea. Introducción a los sistemas distribuidos. UPV/EHU. Recuperado el 7 de abril del 2024 http://www.sc.ehu.es/acwlaalm/sdi/1-Introduccion.pdf
- Gracy M. Booth. Distributed Information Systems: Honeywell Information Systems. Recuperado el 10 de abril del 2024 en https://dl.acm.org/doi/pdf/10.1145/1499799.1499907
- Advantages and Disadvantages of Distributed System. (s/f). Www.javatpoint.com. Recuperado el 14 de abril de 2024, de https://www.javatpoint.com/advantages-and-disadvantages-of-distributed-system

Fuentes complementarias

- Vicky Shiv. Distributed Information Systems: Acadia University. Recuperado el 10 de abril del 2024 en https://scholar.acadiau.ca/islandora/object/theses:2973/datastream/PDF/file.pdf
- Huaman, W. C. (2018, agosto 28). ¿Qué es RPC (llamada a procedimiento remoto)? - Wilber Ccori huaman. Medium. https://medium.com/@maniakhitoccori/qu%C3%A9-es-rpc-llamada-a-procedimiento-remoto-7cbcbe45d8e

- Amazon.com. Recuperado el 6 de abril del 2024 de https://aws.amazon.com/es/compare/the-difference-between-lan-and-wan/#:~:text=LAN%20means%20local%20area%20network,WAN%20means%20wide%20area%20network.&text=LANs%20connect%20users%20and%20applications,locations%20(across%20the%20globe)