Sistemas distribuidos

M.C. Jesus Humberto Abundis Patiño

jesus.abundis@info.uas.edu.mx

Referencias Básicas

- **Distributed Systems: Concepts and Design**
- G. Coulouris, J. Dollimore, T. Kindberg; Addison-Wesley, 2001
- **Distributed Systems: Principles and Paradigms**

A. S. Tanenbaum, M. Van Steen; Prentice-Hall, 2002

- **Distributed Operating Systems: Concepts & Practice**
- D. L. Galli; Prentice-Hall, 2000
- **Distributed Operating Systems & Algorithms**
- R. Chow, T. Johnson; Addison-Wesley, 1997
- **Distributed Computing: Principles and Applications**
- M.L. Liu; Addison-Wesley, 2004

Contenido de la unidad I

Introducción a los sistemas distribuidos

1.1. Definición de sistema distribuido.

1.2. Ventajas y desventajas de los sistemas distribuidos.

1.3. Sistemas operativos distribuidos.

1.4. Sistemas operativos de red.

1.5. Objetivos de un sistema operativo distribuido.

1.6. Componentes de un sistema operativo distribuido

Evolución en la arquitectura de Sistemas de Cómputo

Sistemas Centralizados

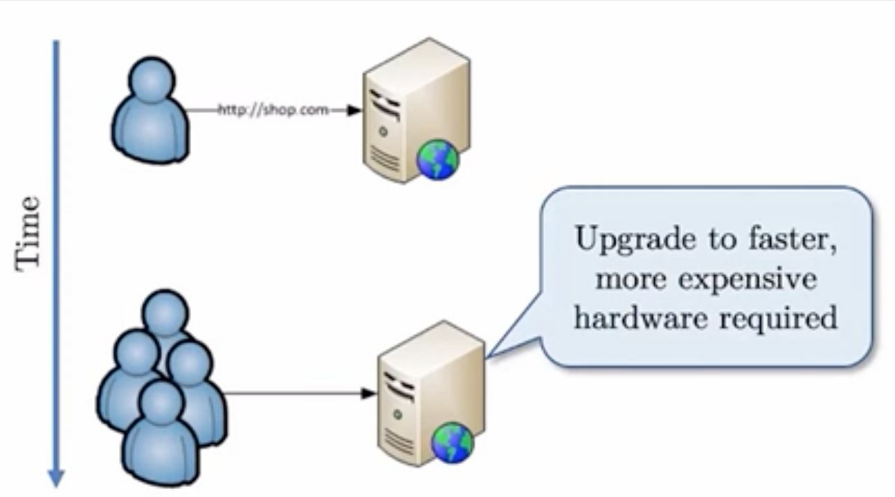
- Principios de la era del internet

Sistemas Distribuidos

- Nos rodea en el siglo 21
- ¿Qué problemas resuelve?
- Características
- Desafíos

Sistemas Centralizados

Los sistemas centralizados aprovechan la arquitectura cliente-servidor, dónde múltiples usuarios se conectan a un único servidor, el cual se encarga de todos los requests

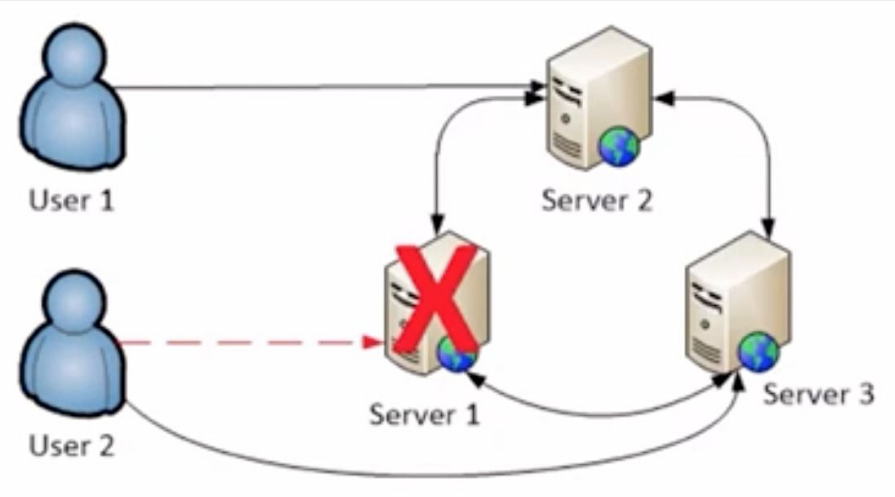


Desventajas

- La mejoras en el rendimiento están limitadas a escalamiento vertical
- Punto único para fallos (fallas totales)
- Muy dificil tener disponibilidad de 99.999%

Sistemas Distribuidos

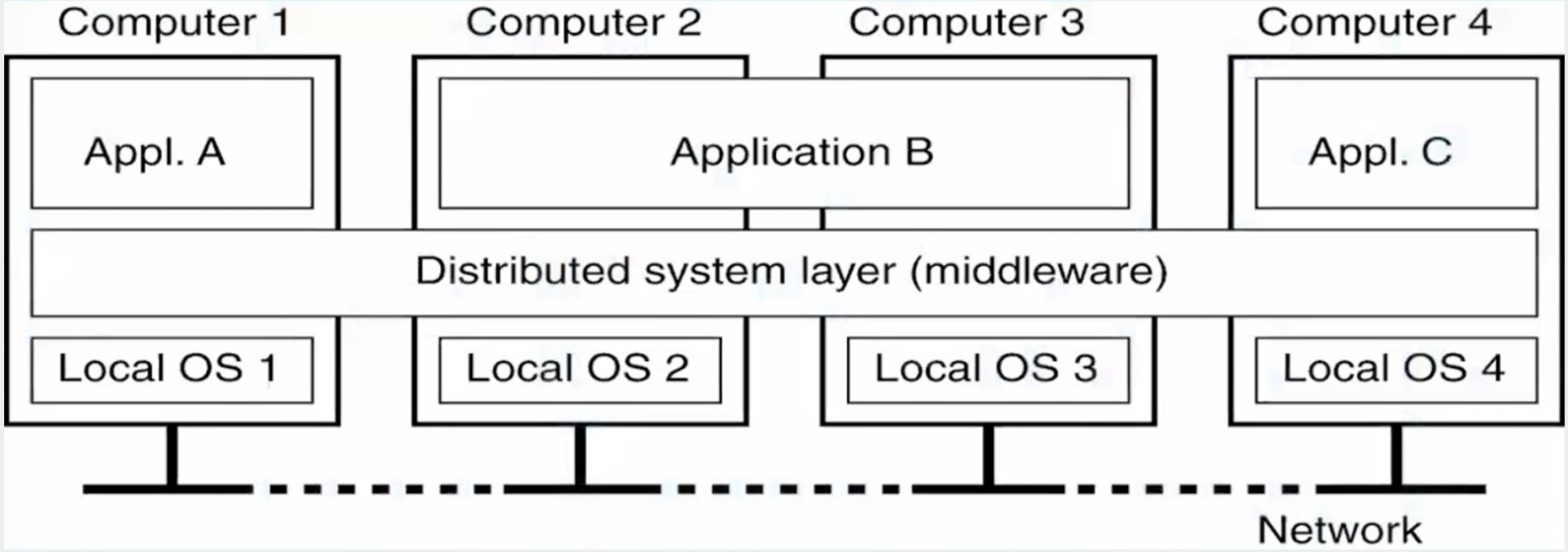
Un sistema distribuido es un ambiente de cómputo dónde múltiple procesos, los cuales corren en diferentes equipos, se comunican a través de la red y coordinan las acciones de tal manera que este aparezca al usuario como un único sistema coherente



Ventajas (hecho “bien”)

- Rendimiento casi infinito gracias a la escalabilidad vertical
- Resistente y tolerante a las fallas
- Siempre disponible

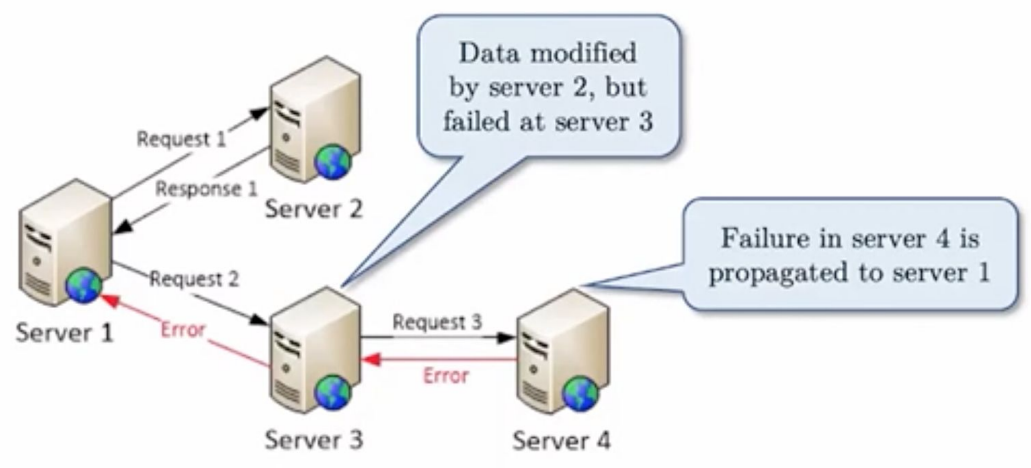
Sistemas Distribuidos



- La capa middleware se extiende sobre múltiples computadoras
- Ofrece a cada aplicación la misma interface

Manipulación de los fallos

- Una llamada a un procedimiento remoto (RPC) puede fallar por múltiples razones
- Existe propagación de fallas de locaciones remotas en el sistema
- Falla parcial vs consistencia en los datos



Una función local regresa una respuesta o una excepción. La llamada a una función remota puede sufrir un timeout.

```
sayHello(“Lukasz”);

private String sayHello(String name) {
        return String.format(
                “Hola %s!”, name
                );
}
```

Desafíos

Los sistemas distribuidos se desarrollan para aplicaciones de gran escala

⇒ Cuando múltiples usuarios acceden a recursos compartidos, estos necesitan ser sincronizados.

Sistema Centralizado

Los lenguajes de programación modernos cuentan con herramientas eficientes tales como; mutexes, semáforos, monitores, para implementar aplicaciones con múltiples hilos y sincronizar el acceso dentro de un único proceso del sistema operativo.

Sistema Distribuido

Se coordinan procesos que se ejecutan en diferentes servidores conectados por una red que no es confiable.

No hay un lenguaje de programación que provea herramientas para sincronizar el acceso a recursos compartidos en el caso distribuido

Desafíos

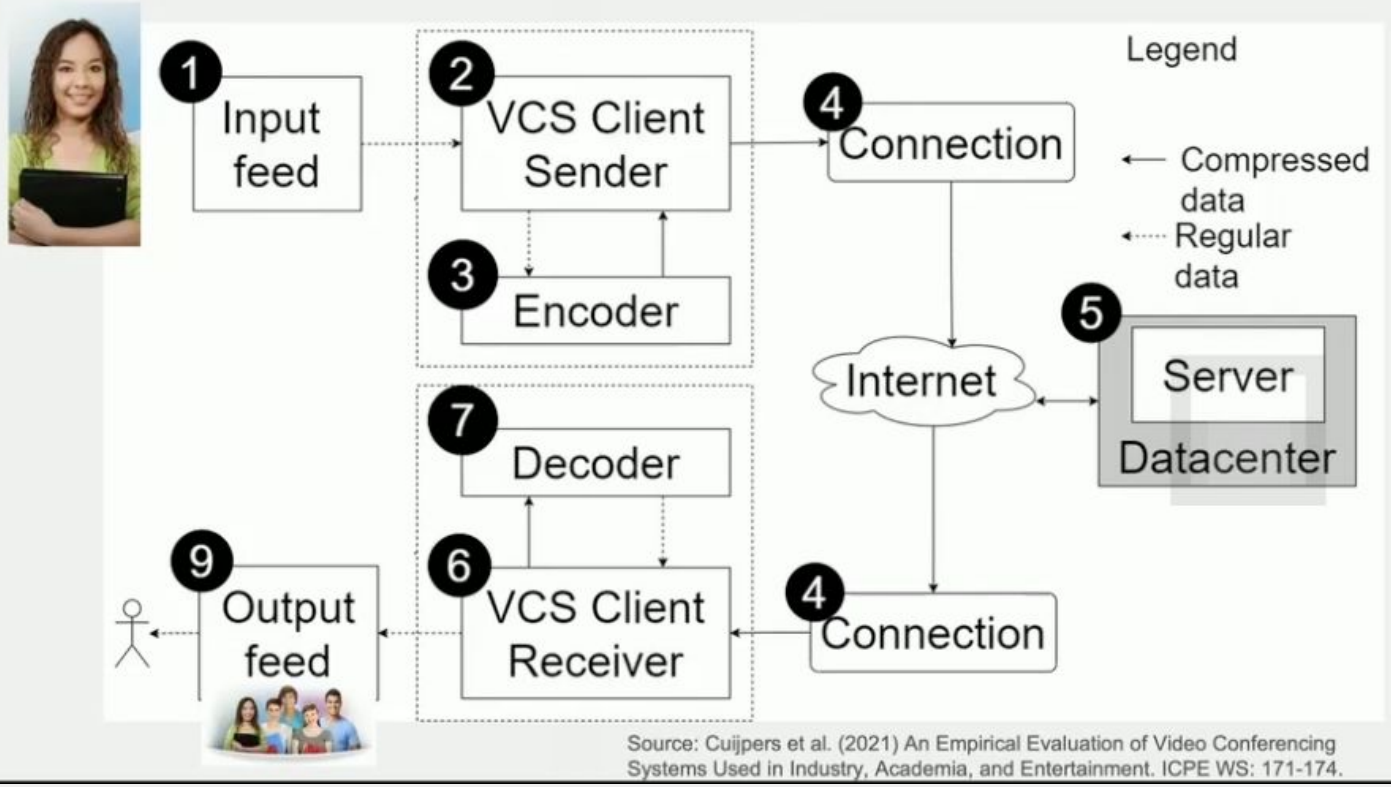
Acceso a recursos compartidos

- Sincronizar hilos locales y procesos remotos
- Estado inconsistente entre servidores

Desarrollo y reproducción de errores

- El tiempo es un factor variable
- Múltiples componentes de hardware y software involucrados

Example: A Video-Conferencing System



Source: Cuijpers et al. (2021) An Empirical Evaluation of Video Conferencing Systems Used in Industry, Academia, and Entertainment. ICPE WS: 171-174.

Definición de Sistema Distribuido (SD)

Un sistema distribuido es aquel en el que los procesos ubicados en una red de computadoras, comunican y coordinan sus acciones solo pasando mensajes.

Esta definición conduce a las siguientes características especialmente significativas de los sistemas distribuidos:

- Concurrencia de componentes.
- Falta de un reloj global.
- Fallas independientes de componentes.

Coulouris, George(2012).DISTRIBUTED SYSTEMS,Concepts and Design ,USA, Press Pearson.

Definición de Sistema Distribuido (SD)

Hardware: Conjunto de procesadores sin memoria común conectados por una red.

- Sistema débilmente acoplado
- No existe un reloj común
- Dispositivos de E/S asociados a cada procesador
- Fallos independientes de componentes del SD
- Carácter heterogéneo

Ventajas de los Sistemas Distribuidos

- Economía: Buena relación rendimiento/coste
  - Avances en la tecnología de microprocesadores y redes de área local.
- Alto rendimiento: Procesamiento paralelo.
- Soporte de aplicaciones inherentemente distribuidas.
  - Por ejemplo: empresa distribuida geográficamente
- Capacidad de crecimiento: Escalabilidad.
- Fiabilidad y disponibilidad: Tolerancia a fallos.
- Carácter abierto y heterogéneo:
  - Estándares de interoperabilidad.
- Compartir recursos y datos.

Desventajas de los Sistemas Distribuidos

- Necesidad de un nuevo tipo de software:
  - Más complejo.
  - No hay todavía un acuerdo sobre cómo debe ser.
- Red de interconexión introduce nuevos problemas:
  - Pérdida de mensajes y saturación.
  - Latencia puede provocar que al recibir un dato ya esté obsoleto.
  - La red es un elemento crítico.
- Seguridad y confidencialidad

Aplicaciones de los Sistemas Distribuidos

- Entornos empresariales: redes corporativas e *intranets*:
  - Sustituye a los clásicos *mainframes*.
- Entornos de computación de altas prestaciones:
  - Procesamiento paralelo, alternativa a costosos *supercomputadores*.
- Servicios con alta disponibilidad y rendimiento.
- Sistemas distribuidos de gestión de bases de datos
- Aplicaciones multimedia.
- Sistemas industriales distribuidos y aplicaciones de control.
- Internet: un enorme sistema distribuido.

Paradigmas de SD

- ***Cluster Computing*:**
  - Dedicados a tareas específicas:
    - Altas prestaciones.
    - Alta disponibilidad.
  - Sistema homogéneo (a menudo dedicado):
    - Nodos PCs.
    - LAN (de propósito general o específicas).
  - Problemática: Grado de acoplamiento, servicios distribuidos.
- ***Grid Computing*:**
  - Aprovechamiento de recursos creando un uniprocesador virtual.
  - Restringido a una serie de tareas.
  - Diferentes ámbitos:
    - Desde intradepartamentales.
    - Hasta intercorporativos.
  - Problemática: Coordinación, seguridad, carácter dinámico.

Propiedades de los SD

En general el desarrollo de sistemas distribuidos intenta poner solución a los siguientes objetivos:

- Transparencia.
- Rendimiento.
- Escalabilidad.
- Flexibilidad.
- Fiabilidad y tolerancia a fallos.

Transparencia

Un objetivo importante de un sistema distribuido es ocultar el hecho de que sus procesos y recursos están físicamente distribuidos a través de múltiples computadoras. Decimos que un sistema distribuido es transparente si es capaz de presentarse ante los usuarios y las aplicaciones como si se tratara de una sola computadora. (Tanenbaum, 2008)

Transparencia

Existen varios perfiles de transparencia:

- **Acceso**: Manera de acceder a recurso local igual que a remoto.
- **Posición**: Se accede a los recursos sin conocer su localización.
- **Migración**: Recursos pueden migrar sin afectar a los usuarios.
- **Concurrencia**: Acceso concurrente no afecta a los usuarios.
- **Replicación**: La existencia de réplicas no afecta a los usuarios.
- **Fallos**: La ocurrencia de fallos no afecta a los usuarios.
- **Crecimiento**: El crecimiento del sistema no afecta a los usuarios.
- **Heterogeneidad**:Carácter heterogéneo no afecta a los usuarios.

Rendimiento

Rendimiento para un **servicio multiusuario**:

- Objetivo: Rendimiento no peor que un sistema centralizado

Rendimiento para la **ejecución paralela** de aplicaciones:

- Objetivo: Rendimiento proporcional a procesadores empleados

Factores:

- Uso de esquemas de *caching*
  - Intentar que muchos accesos se hagan localmente
- Uso de esquemas de replicación
  - Reparto de carga entre componentes replicados
- En ambos casos: Coste de mantener la coherencia

Escalabilidad

Capacidad de crecimiento.

Tamaño o Carga: Un sistema distribuido nos hace fácil el ampliar y reducir sus recursos para acomodar (a conveniencia), cargas más pesadas o más ligeras según se requiera.

Geográfica: Un sistema geográficamente escalable, es aquel que mantiene su utilidad y usabilidad, sin importar que tan lejos estén sus usuarios o recursos.

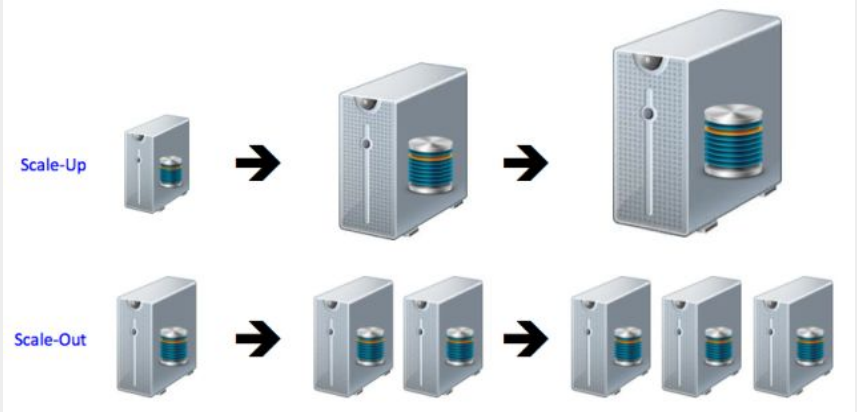
Administrativa: Un sistema distribuido nos hace fácil el ampliar y reducir sus recursos para acomodar (a conveniencia), cargas más pesadas o más ligeras según se requiera.

Escalabilidad

Tipos de Escalabilidad.

Escalabilidad vertical

La escalabilidad vertical se basa en añadir más recursos,actualizar o modernizar los sistemas existentes para otorgar más capacidad a un equipo.



Escalabilidad horizontal

Aunque es más complejo de implementar y administrar, en líneas generales la escalabilidad horizontal se basa en la integraciòn de nuevo equipo a la solución.

Tanenbaum, Andrew, 2008, SISTEMAS DISTRIBUIDOS ,Principios y Paradigmas,Editorial Pearson.

Flexibilidad

SOD debe ser adaptable:

- facilidad para incorporar cambios y extensiones al sistema

Uso preferible de arquitectura microkernel

Importancia de **sistemas abiertos**:

- Sus interfaces y protocolos deberían ser públicos.
- Contrario a *"tecnología propietaria"*.
- Uso de estándares siempre que sea posible.
- Disponibilidad de su código fuente (libremente o no).
- Regulación por parte de un colectivo (usuarios u organizaciones) y no por particulares (fabricantes).

Fiabilidad y tolerancia a fallos

Una característica sobresaliente de los sistemas distribuidos que los distingue de los sistemas de una sola máquina es la noción de falla parcial. En un sistema distribuido, una falla parcial puede acontecer cuando falla un componente.

Esta falla puede afectar la operación de algunos componentes, al tiempo que otros más no se ven afectados en absoluto. Por contraste, en un sistema no distribuido, una falla a menudo es total en el sentido de que afecta a todos los componentes y fácilmentepuede echar abajo al sistema.(Tanenbaum,2008)

Fiabilidad y tolerancia a fallos

Un objetivo importante en el diseño de sistemas distribuidos es construirlos de manera que puedan recuperarse automáticamente de fallas parciales sin que se afecte seriamente el desempeño total. (Tanenbaum,2008)

Fiabilidad y tolerancia a fallos

Para entender el rol de la tolerancia a fallas en los sistemas distribuidos, primero se tiene que examinar a fondo lo que en realidad significa el que un sistema distribuido tolere fallas. Ser tolerante a las fallas está fuertemente relacionado con lo que se llama sistemas fiables. Fiabilidad es un término que comprende varios requerimientos útiles para los sistemas distribuidos incluidos los siguientes (Kopetz y Verissimo, 1993):

1. Disponibilidad
2. Confiabilidad
3. Seguridad
4. Mantenimiento

Fiabilidad y tolerancia a fallos

Fiabilidad: Disponibilidad se define como la propiedad de que un sistema está listo para ser utilizado de inmediato. En general, se refiere a la probabilidad de que el sistema esté operando correctamente en cualquier momento dado y se encuentre disponible para realizar sus funciones a nombre de sus usuarios. En otros términos, un sistema altamente disponible es uno que muy probablemente funcionará en un instante dado.

Fiabilidad y tolerancia a fallos

Fiabilidad: Confiabilidad se refiere a la propiedad de que un sistema sea capaz de funcionar de manera continua sin fallar. Por contraste con la disponibilidad, la confiabilidad se define en función de un intervalo de tiempo en lugar de un instante en el tiempo. (Tanenbaum, 2008)

Fiabilidad y tolerancia a fallos

Fiabilidad: Seguridad se refiere a la situación en que no acontece nada catastrófico cuando un sistema deja de funcionar correctamente durante un tiempo. Por ejemplo, se requiere que muchos sistemas de control de proceso, como los utilizados para controlar plantas de energía nuclear o enviar personas al espacio exterior, proporcionen un alto grado de seguridad. Si tales sistemas de control fallan temporalmente durante sólo un breve momento, los efectos podrían ser desastrosos. Muchos ejemplos del pasado (y probablemente más que aún no acontecen) demuestran lo difícil que es construir sistemas seguros.

Fiabilidad y tolerancia a fallos

Fiabilidad: Mantenimiento se refiere a cuán fácil puede ser reparado un sistema que falló.

Un sistema altamente mantenible también puede ser altamente disponible, en especial si las fallas pueden ser detectadas y reparadas en forma automática. Sin embargo,, la recuperación automática de fallas es fácil de expresar pero difícil de realizar.

FALLA en SD

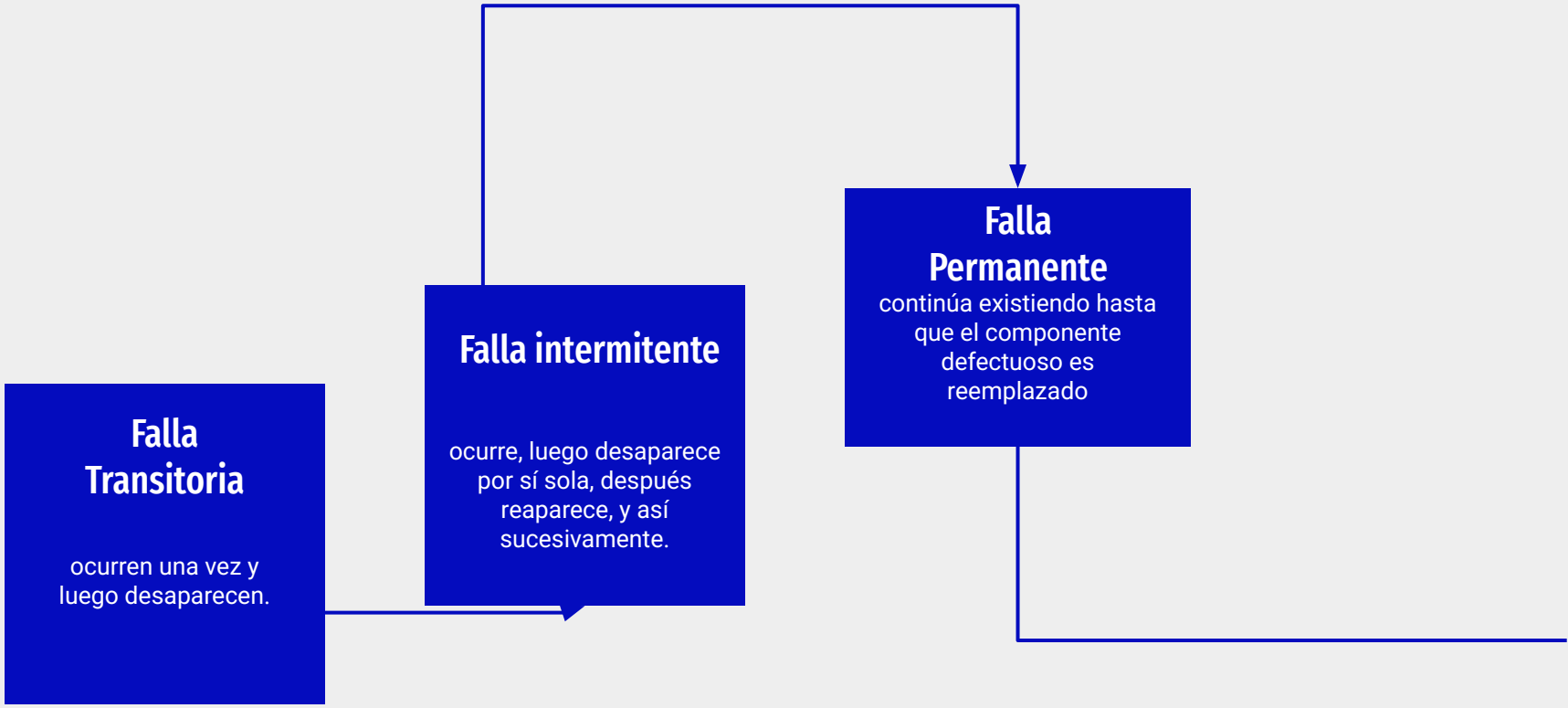
Definición de FALLA.

Se dice que un sistema falla cuando no puede cumplir sus propósito.

Un **error** es una parte del estado de un sistema que puede conducir a una falla

tolerancia a fallas significa que un sistema puede proveer sus servicios incluso en presencia de fallas

Conceptos básicos Fiabilidad : Fallas.



Tipos de fallas

Tipo de falla	Descripción
Falla de congelación	Un servidor se detiene, pero estaba trabajando correctamente hasta que se detuvo
Falla de omisión	Un servidor no responde a las peticiones entrantes
Omisión de recepción	Un servidor no recibe los mensajes entrantes
Omisión de envío	Un servidor no envía mensajes
Falla de tiempo	La respuesta de un servidor queda fuera del intervalo de tiempo especificado
Falla de respuesta	La respuesta de un servidor es incorrecta
Falla de valor	El valor de la respuesta está equivocado
Falla de transición de estado	El servidor se desvía del flujo de control correcto
Falla arbitraria	Un servidor puede producir respuestas arbitrarias en tiempos arbitrarios

Figura 8-1. Diferentes tipos de fallas.

Tolerancia a Fallas

Para que un sistema sea tolerante a fallas, lo mejor que se puede hacer es tratar de ocultar ante otros procesos la ocurrencia de fallas. La técnica clave para disfrazar las fallas es utilizar redundancia.

Tres clases son posibles: **redundancia de información, redundancia de tiempo, y redundancia física** [vea también Johnson (1995)]. Con redundancia de información, se agregan bits adicionales para recuperar los bits mutilados. Por ejemplo, se puede agregar un código Hamming a los datos transmitidos para recuperarlos del ruido presente en la línea de transmisión.

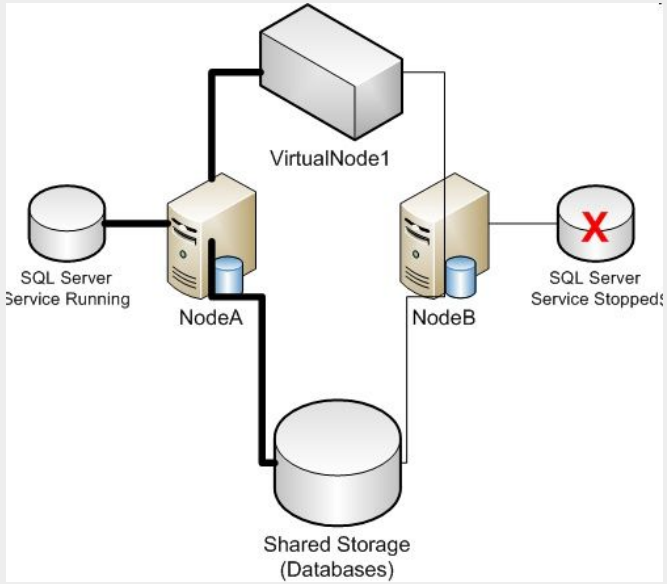
Con redundancia de tiempo, se realiza una acción, y luego, si es necesario, se vuelve a realizar. Las transacciones (vea el capítulo 1) utilizan este método. Si se aborta una transacción, puede rehacerse sin ningún perjuicio. La redundancia de tiempo resulta especialmente útil cuando las fallas son transitorias e intermitentes

Tolerancia a Fallas

Con redundancia física, se agrega equipo o procesos adicionales para que el sistema en su conjunto tolere la pérdida o el mal funcionamiento de algunos componentes. La redundancia física puede realizarse entonces en el hardware o el software. Por ejemplo, se pueden agregar procesos adicionales al sistema de modo que si algunos se congelan, el sistema pueda seguir funcionando correctamente. En otros términos, al replicar procesos se logra un alto grado de tolerancia a fallas.

Tolerancia a Fallas

Básicamente un cluster es uno o más servidores (conocidos como nodos),que actuan y son administrados como uno.



Consistencia

1. Problemas relacionados con la replicación

- La red de interconexión es una nueva fuente de fallos
- La seguridad del sistema es más vulnerable
- La gestión del estado global es más compleja/costosa

2. Problemas para mantener la consistencia

- distribución física: varias copias, cada una con su estado
- errores y/o retardos en las comunicaciones
- ausencia de reloj global: ¿cómo ordenar eventos?

3. Técnicas: transacciones, comunicación a grupos

4. Para un rendimiento aceptable: relajar consistencia

Componentes de Sistemas distribuidos.

Hardware.

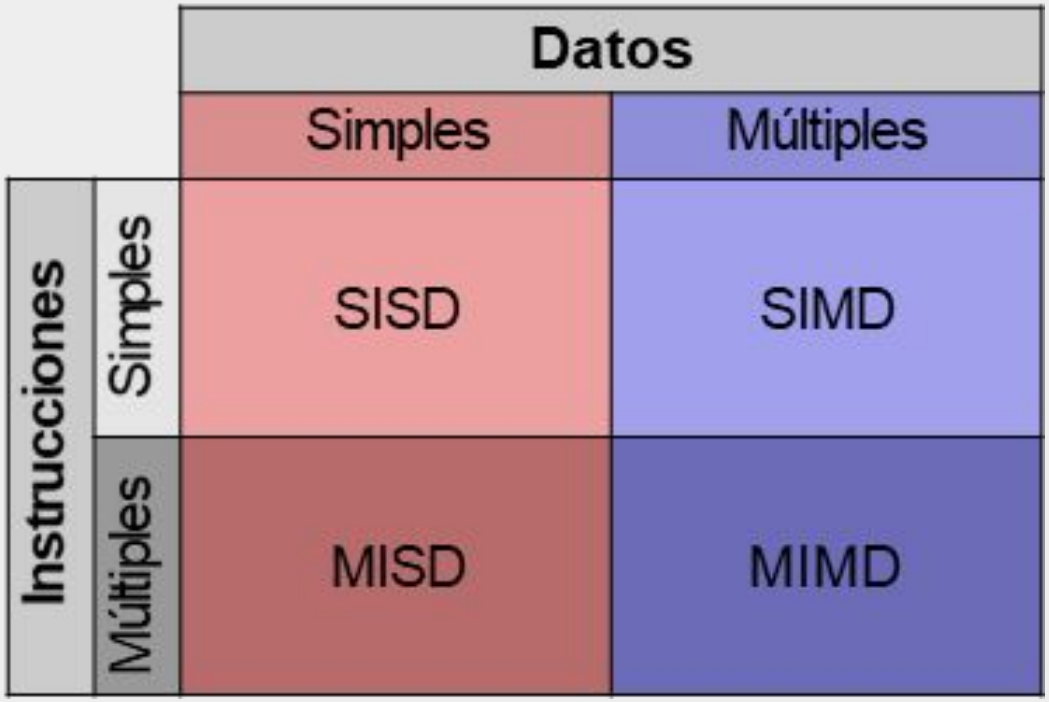
Software

Taxonomía de Flynn

Es la clasificación más extendida del paralelismo

Distingue entre instrucciones y datos

Estos pueden ser simples o múltiples

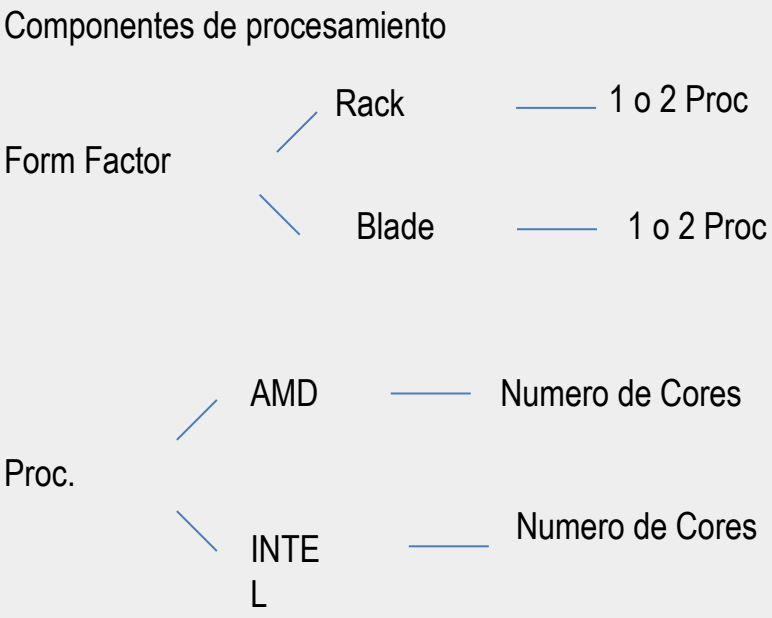


Componentes de Hardware.

Componentes de Interconexión.

- Tecnologìa
  - Ethernet — 1Gb,10Gb,40Gb,100Gb
  - Infiniband — 10Gb,54,100Gb,200Gb
- Medio
  - Fibra
    - LC
    - SC
  - Cobre

Componentes de Hardware Procesamiento.



Componentes de Hardware : Almacenamiento.

3 Dimensiones :

1. Uso

Actualmente se usan y creando más datos que nunca.

Para poner esto en perspectiva, ¡los últimos dos años se crearon más datos que todos los años anteriores combinados! Las redes sociales, los dispositivos móviles y los dispositivos inteligentes combinados en Internet de las cosas (IoT) aceleran enormemente la creación de datos.

. Algunos ejemplos de datos que usamos, almacenamos y administramos son: documentos, fotos, películas, aplicaciones y con el auge de la virtualización y especialmente el Centro de Datos Definido por Software (SDDC) también completan las infraestructuras en código como backend para la nube. En los últimos dos años, creamos más datos en todo el mundo que todos los años digitales anteriores.

https://nexenta.com/three-dimensions-storage-sizing-design-%E2%80%93-part-1-overview

Componentes de Hardware : Almacenamiento.

3 Dimensiones :

2.Carga de trabajo

Para que sea más fácil y rápido trabajar con datos, se han creado aplicaciones. En la actualidad, usamos, protegemos y administramos muchos tipos diferentes de cargas de trabajo que se ejecutan en nuestros sistemas de almacenamiento. La cantidad de trabajo que se espera realizar en un período de tiempo específico, tiene varias características que definen el tipo de carga de trabajo.

https://nexenta.com/three-dimensions-storage-sizing-design-%E2%80%93-part-1-overview

Componentes de Hardware : Almacenamiento.

- **Velocidad**: medida en IOPS (entrada / salida por segundo), define las IO por segundo. Las E / S de lectura y escritura pueden tener diferentes patrones (por ejemplo, secuenciales y aleatorios). Cuanto mayor sea la IOPS, mejor será el rendimiento.

- **Rendimiento:** medido en MB / s, define la velocidad de transferencia de datos, también llamada ancho de banda. Cuanto mayor sea el rendimiento, más datos se pueden procesar por segundo.

- **Respuesta**: medida en tiempo como latencia ns / us / ms, define la cantidad de tiempo que el IO necesita para completarse. Cuanto menor es la latencia, más rápido responde un sistema / aplicación / datos, más fluido se ve para el usuario. Hay muchas medidas de latencia que se pueden tomar a lo largo de toda la ruta de datos.

https://nexenta.com/three-dimensions-storage-sizing-design-%E2%80%93-part-1-overview

Componentes de Hardware : Almacenamiento.

3 Dimensiones :

3.Protecciòn.

Redundancia: medida en número de copias de los datos o metadatos que pueden reconstruir los datos al original. p.ej. medidas como, por ejemplo, paridad, duplicación, múltiples objetos.

Disponibilidad: medida en RTO y RPO. La cantidad máxima de datos que podemos perder se denomina Objetivo de punto de recuperación (RPO) y la cantidad máxima de tiempo que lleva restaurar los niveles de servicio Objetivo de tiempo de recuperación (RTO).

https://nexenta.com/three-dimensions-storage-sizing-design-%E2%80%93-part-1-overview

Componentes de Software.

Problema para la integración: heterogeneidad

Solución: sistemas abiertos.

- especificación pública de su interfaz
- estándares: oficiales vs de facto (OSI vs TCP/IP)

Lafuente, Alberto . Larrea, Mikel,2018, Sistemas Distribuidos. Introducción -Madrid, Dpto. ATC, UPV/EHU

Componentes de Software.

Propiedades de los sistemas abiertos:

Interoperabilidad:

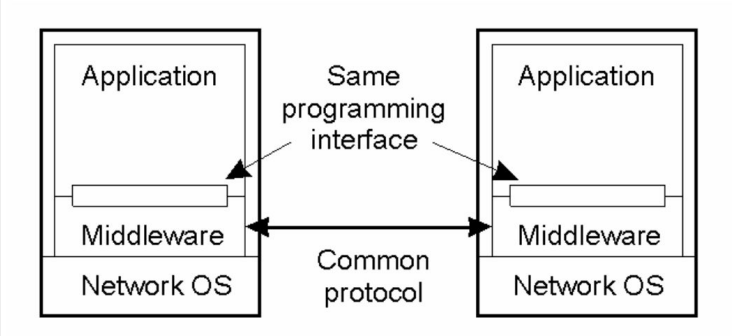
- protocolos estándar: TCP/IP, RPC/XDR
- lenguajes de definición de interfaces: CORBA IDL
  1. Tendencia actual: XML/SOAP (Servicios Web)

- Transportabilidad de aplicaciones
  1. POSIX (código fuente, entre máquinas Unix)
  2. Java (código 'ejecutable', entre JVMs)

- Transportabilidad de usuarios: GUI, NIS

Componentes de Software: Middleware.

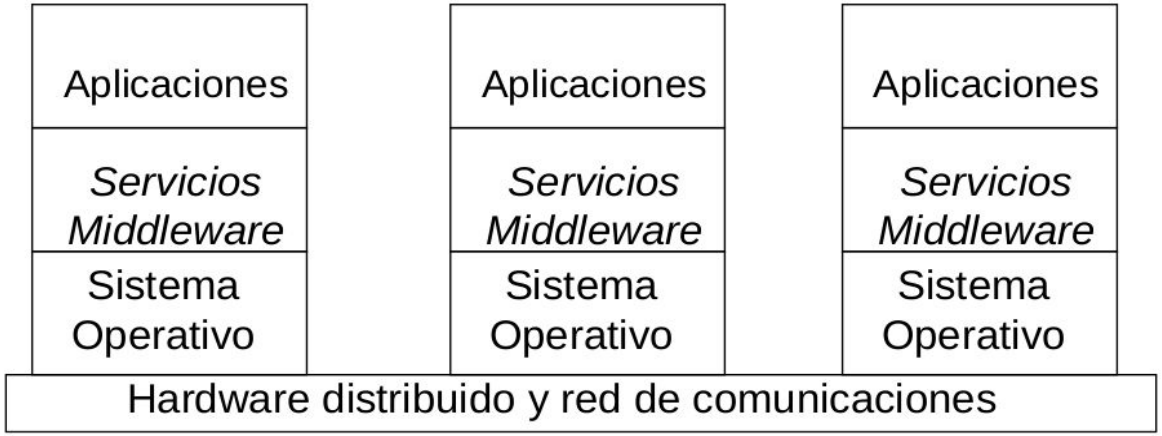
Middleware es software que se sitúa entre un sistema operativo y las aplicaciones que se ejecutan en él. Básicamente, funciona como una capa de traducción oculta para permitir la comunicación y la administración de datos en aplicaciones distribuidas. A veces, se le denomina “plumbing” (tuberías), porque conecta dos aplicaciones para que se puedan pasar fácilmente datos y bases de datos por una “canalización”. El uso de middleware permite a los usuarios hacer solicitudes como el envío de formularios en un explorador web o permitir que un servidor web devuelva páginas web dinámicas en función del perfil de un usuario.



https://azure.microsoft.com/es-es/overview/what-is-middleware/

Componentes de Software : Estructura.

a) visión del usuario



b) estructura del sistema (visión del diseñador)

- *Servicios Middleware*: soporte RPC/RMI, soporte a comunicación uno-a-muchos, sincronización de tiempos y ordenación de eventos, consistencia (replicación), servicios de nombres, de seguridad…

Lafuente, Alberto . Larrea, Mikel,2018, Sistemas Distribuidos. Introducción -Madrid, Dpto. ATC, UPV/EHU

Componentes de Software : Soporte de la Comunicación.

Hay que diferenciar entre distribución física de la memoria y modelo de comunicación:

- el grado de acoplamiento determinará el soporte necesario para implementar el modelo de comunicación.
- el modelo de comunicación puede estar basado tanto en memoria compartida como en paso de mensajes.

Sistemas con memoria física compartida

- variables compartidas, buzones FIFO.

Sistemas con memoria física distribuida

- paso de mensajes (protocolos de red).

Componentes de Software : Soporte de la Comunicación.

Grado de acoplamiento y modelo de comunicación

Grado de acoplamiento / Modelo de comunicación	**Memoria física compartida**	**Memoria física distribuida**
Memoria compartida	Variables compartidas	Memoria compartida distribuida (DSM), objetos distribuidos (RMI)
Paso de mensajes	pipes, colas FIFO, sockets UNIX	sockets INET, MPI, RPC

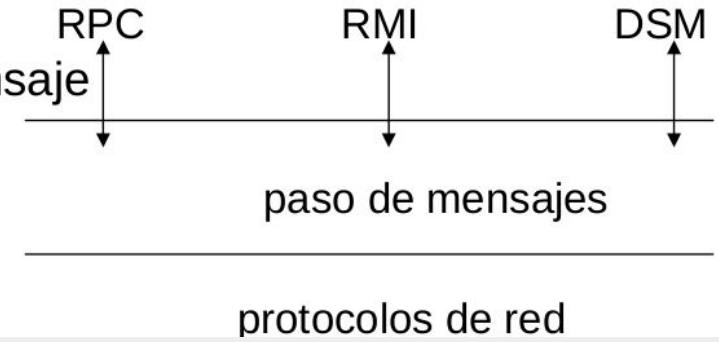
Mecanismos de comunicación

Lafuente, Alberto . Larrea, Mikel,2018, Sistemas Distribuidos. Introducción -Madrid, Dpto. ATC, UPV/EHU

Componentes de Software : Soporte de la Comunicación.

Implementación del modelo de comunicación

- Paso de mensajes estándar básico: sockets INET
  - modelo cliente/servidor
  - bloqueante / no bloqueante
  - fiable (TCP) / no fiable (UDP)
  - punto-a-punto / broadcast / multicast (IP Multicast)
- RPC, RMI, DSM:
  - basados en paso de mensaje



Componentes de Software : Sistemas Operativos.

Soporte del sistema operativo

- Propiedades deseables: abierto y flexible
  - desarrollo, ubicación y gestión eficiente de servicios
- Los SO clásicos (UNIX) son monolíticos: todos los servicios en el kernel, única interfaz de llamadas al sistema, políticas de gestión predeterminadas
- Alternativas y tendencias
  - Emulación hardware: SO huésped sobre SO anfitrión
  - Microkernels (Mach): servicios fuera del kernel
  - PDAs, teléfonos móviles: versiones adaptadas de SO comerciales (Mobile, Palm, Symbian) + navegador web

Lafuente, Alberto . Larrea, Mikel,2018, Sistemas Distribuidos. Introducción -Madrid, Dpto. ATC, UPV/EHU

Componentes de Software : Sistemas Operativos.

Sistema Operativo de red operativo: también llamado N.O.S (en inglés, Network Operating System), es un software que permite la interconexión de ordenadores para poder acceder a los servicios y recursos, hardware y software, creando redes de computadoras. Al igual que un equipo no puede trabajar sin un sistema operativo, una red de equipos no puede funcionar sin un sistema operativo de red. Consiste en un software que posibilita la comunicación de un sistema informático con otros equipos en el ámbito de una red.

Sistema operativo distribuido **:** es la unión lógica de un grupo de sistemas operativos sobre una colección de nodos computacionales independientes, conectados en red, comunicándose y físicamente separados. Cada nodo contiene de forma individual un subconjunto específico de los programas que componen el sistema operativo distribuido. Cada subconjunto es una combinación de dos proveedores de servicios distintos.Ejemplos: Inferno, Plan9.

Lafuente, Alberto . Larrea, Mikel,2018, Sistemas Distribuidos. Introducción -Madrid, Dpto. ATC, UPV/EHU