



Instituto Politécnico Nacional
Escuela Superior de Cómputo

**Desarrollo de Sistemas Distribuidos
Unidad I:
Introducción**

M. en C. Hermes Francisco Montes Casiano
hermes.escom@gmail.com



Índice

1 Definición de un sistema distribuido

2 Objetivos y desafíos

3 Requisitos de diseño

4 Modelos de Sistema

5 Modelos fundamentales



ESCOM
Escuela Superior de Cómputo



Introducción

- De 1945, cuando comenzó la era moderna de las computadoras, a 1985, éstas eran grandes y caras.
- Hacia la mitad de la década de 1980, dos avances en la tecnología comenzaron a cambiar esa situación.
 - El desarrollo de poderosos microprocesadores.
 - La invención de las redes de computadoras de alta velocidad.
- El resultado de éstas tecnologías ha hecho factible y fácil poner a trabajar sistemas de cómputo compuestos por grandes cantidades de computadoras interconectadas mediante una red.



Definición de un Sistema Distribuido

Sistemas Distribuido

Es una colección de computadoras independientes que dan al usuario la impresión de construir un único sistema coherente.

Esta definición comprende diversos aspectos importantes:

- ① Un sistema distribuido consta de componentes autónomos.
- ② Los usuarios creen que realmente interactúan con un sistema único. *Ésto significa que de una manera o de otra los componentes autónomos necesitan colaborar entre sí.*



Definición de un Sistema Distribuido

En principio todos los sistemas distribuidos deberían tener los siguientes atributos:

- **Escalabilidad:** ésta característica es consecuencia de tener nodos independientes, pero al mismo tiempo, de ocultar cómo estas computadoras realmente forman parte del sistema como un todo.
 - **Disponibilidad:** un sistema distribuido estará disponible de forma continua, aunque tal vez algunas partes pudieran encontrarse fuera de operación.

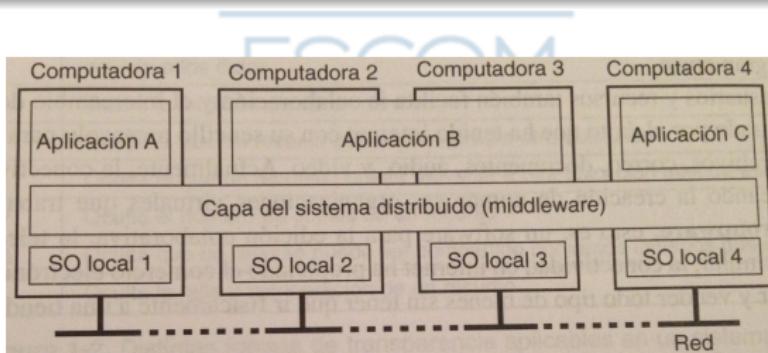


Definición de un Sistema Distribuido

computadoras y redes heterogéneas

Los sistemas distribuidos se organizan a menudo en términos de una capa de software:

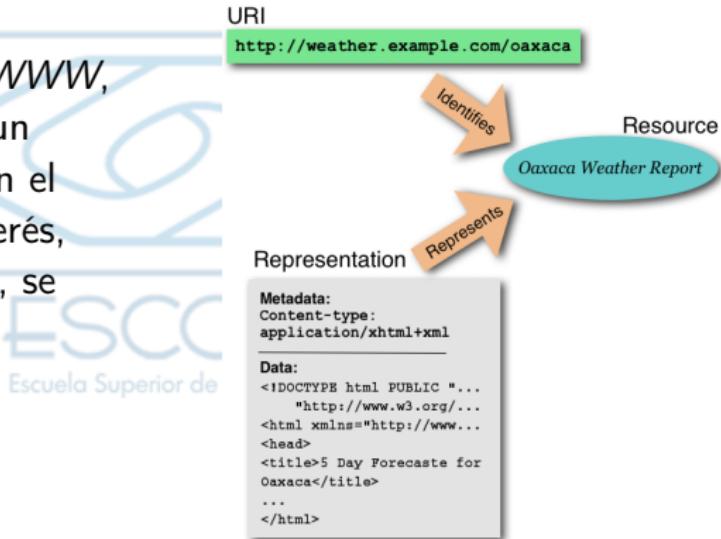
- Una capa de alto nivel (usuarios y aplicaciones)
- Una capa subyacente constituida por sistemas operativos y recursos básicos de comunicación.





Ejemplo: WWW

La World Wide Web (*WWW*, o simplemente *Web*) es un espacio de información en el que los elementos de interés, conocidos como recursos, se identifican mediante identificadores globales llamados *URI*.





Índice

1 Definición de un sistema distribuido



2 Objetivos y desafíos



3 Requisitos de diseño

4 Modelos de Sistema

5 Modelos fundamentales



Objetivo

¿En qué consiste el reto?

Un sistema distribuido debe hacer que los recursos sean fácilmente accesibles; debe ocultar de manera razonable el hecho de que los recursos están distribuidos por toda la red, debe ser abierto y debe ser escalable.

¡Atención!

Sólo porque es posible construir un sistema distribuido no significa que necesariamente sea buena idea.



Objetivo

Accesibilidad de recursos

- El principal objetivo de un sistema distribuido es facilitar a los usuarios y a las aplicaciones el acceso a los recursos remotos y compartirlos de manera controlada y eficiente.

¿Qué es un recurso?

Un recurso puede significar **casi** cualquier cosa: impresoras, computadoras, dispositivos de almacenamiento, datos, archivos, páginas web, redes, etc.

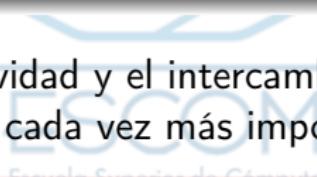


Objetivo

Accesibilidad de recursos

¿Por qué quiero compartir recursos?

- Una razón evidente es la económica.
- Conectar usuarios y recursos también facilita la colaboración y el intercambio de información (ej. la Internet.)
- Mientras la conectividad y el intercambio aumentan, la seguridad se vuelve cada vez más importante.



En la práctica

Los sistemas proporcionan poca protección en contra del espionaje o de la intrusión en las comunicaciones



Transparencia en la distribución

- 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.

Definición

Un sistema distribuido es transparente si es capaz de presentarse ante los usuarios y las aplicaciones como si se tratara de una sola computadora.





Transparencia en la distribución

Transparencia	Descripción
Acceso	Oculta diferencias en la representación de los datos y la forma en que un recurso accede a los datos
Ubicación	Oculta la localización de un recurso
Migración	Oculta que un recurso pudiera moverse a otra ubicación
Reubicación	Oculta que un recurso pudiera moverse a otra ubicación mientras está en uso
Replicación	Oculta el número de copias de un recurso
Concurrencia	Oculta que un recurso puede ser compartido por varios usuarios que compiten por él
Falla	Oculta la falla y recuperación de un recurso

Cuadro : Distintas formas de transparencia



Transparencia en la distribución

Leslie Lamport

Usted se da cuenta de que tiene un sistema distribuido cuando la falla de una computadora de la cual nunca había escuchado le impide realizar cualquier trabajo.





Transparencia en la distribución

Transparencia	Descripción
Acceso	Oculta diferencias en la representación de los datos y la forma en que un recurso accede a los datos
Ubicación	Oculta la localización de un recurso
Migración	Oculta que un recurso pudiera moverse a otra ubicación
Reubicación	Oculta que un recurso pudiera moverse a otra ubicación mientras está en uso
Replicación	Oculta el número de copias de un recurso
Concurrencia	Oculta que un recurso puede ser compartido por varios usuarios que compiten por él
Falla	Oculta la falla y recuperación de un recurso

Cuadro : Distintas formas de transparencia



Transparencia en la distribución

¿Es buena idea la transparencia?

Sería mucho mejor hacer una distribución explícita de modo que nunca se engañe al desarrollador de la aplicación ni al usuario para que crean que existe algo de transparencia.

Aspectos a considerar

La conclusión es que buscar la transparencia de distribución puede ser un buen objetivo cuando se diseñan e implementan sistemas distribuidos, pero se deben considerar aspectos tales como el rendimiento y la comprensibilidad.



Escalabilidad

- La escalabilidad de un sistema distribuido se puede medir de acuerdo con al menos 3 dimensiones:
 - 1 Un sistema puede ser escalable con respecto a su tamaño.
 - 2 Un sistema se puede escalar geográficamente
 - 3 Un sistema se puede escalar administrativamente





Escalabilidad

- La escalabilidad de un sistema distribuido se puede medir de acuerdo con al menos 3 dimensiones:
 - ① Un sistema puede ser escalable con respecto a su tamaño.
 - ② Un sistema se puede escalar geográficamente
 - ③ Un sistema se puede escalar administrativamente

Rendimiento

Desafortunadamente, con frecuencia un sistema escalable en una o más de éstas dimensiones exhibe alguna pérdida de rendimiento al escalarlo.

Escuela Superior de Cómputo





Grado de apertura

Definición

Un sistema distribuido abierto es un sistema que ofrece servicios de acuerdo con las reglas estándar que describen la sintaxis y la semántica de dichos servicios.

- Los servicios se especifican a través de interfaces, las cuales se definen como **lenguaje de definición de interfaes (IDL)**.
- Que la definición de las interfaces sean completas y neutrales es un factor muy importante para lograr la interoperabilidad y portabilidad.



Grado de apertura

Interoperabilidad: define la extensión mediante la cual dos implementaciones de sistemas o componentes de fabricantes distintos pueden coexistir y trabajar juntos si se apoyan en sus servicios mutuos.

Portabilidad: define la extensión mediante la cual una aplicación desarrollada para un sistema distribuido *A* se pueda ejecutar, sin ninguna modificación en un sistema distribuido *B* que comparte la misma interfaz que *A*.





Desafíos

Heterogeneidad: redes, hardware, sistemas operativos, lenguajes de programación, etc.

Seguridad: confidencialidad, integridad y disponibilidad. Ej. ataques de negación de servicio o código móvil.

Escalabilidad: control de costos de los recursos físicos, control de las pérdidas de prestaciones, evitar cuellos de botella, etc.

Extensibilidad: los sistemas ~~distribuidos~~ deben ser extensibles, el primer paso es la publicación de las interfaces de sus componentes, pero la integración de componentes heterogéneos es un reto auténtico.



Desafíos

Tratamiento de fallos: cada componente necesita estar al tanto de las formas posibles en que puede fallar cualquier componente de los que depende.

Concurrencia: cada componente debe estar diseñado para ser seguro en un entorno concurrente.

Transparencia: el objetivo es que ciertos aspectos de la distribución sean invisibles al programador de aplicaciones de modo que sólo necesite ocuparse del diseño de su aplicación particular.



Índice

1 Definición de un sistema distribuido



2 Objetivos y desafíos

3 Requisitos de diseño

4 Modelos de Sistema



5 Modelos fundamentales



Introducción

- La disponibilidad de redes de computadoras de prestaciones medias y la continua necesidad de compartir recursos condujeron al desarrollo de los **sistemas distribuidos** de los 70's y 80's.
- Los retos que surgen de la distribución de recursos aumentan la necesidad de la gestión de las **actualizaciones concurrentes**.

Requisitos de diseño

Los principales requisitos que se deben considerar para el diseño de un sistema distribuido son:

- 1 Capacidad de respuesta (Responsiveness)
- 2 Productividad (Throughput)
- 3 Calidad de Servicio (QoS)



Capacidad de respuesta (*Responsiveness*)

- Los usuarios de las aplicaciones interactivas necesitan rapidez y consistencia en las interfaces.

Tiempo de respuesta

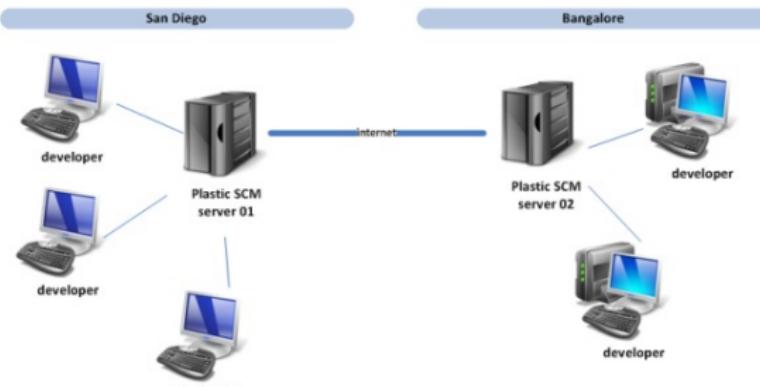
La velocidad a la que se genera una respuesta está determinada por:

- 1 La carga y prestaciones del servidor.
- 2 La red.
- 3 Los retardos de todos los componentes software implicados.
- 4 La comunicación entre los sistemas operativos.
- 5 Servicios *middleware*.



Capacidad de respuesta (*Responsiveness*)

Deployment View



Escuela Superior de Cómputo

¡¡Atención!!!

Para obtener buenos tiempos de respuesta, los sistemas deben estar compuestos de relativamente pocas capas de software y la cantidad de datos transferidos debe ser pequeña.



Productividad (*Throughput*)

- La productividad es la rapidez con la que se realiza el trabajo computacional.

Capacidad

La capacidad de un sistema distribuido depende de las velocidades de los procesos clientes y los servidores, además de las tasas de transferencia.

Escuela Superior de Computo

- La productividad de las capas de software que interbienen es bastante importante, así como de la red.



Productividad (*Throughput*)

- Uno de los propósitos es permitir que las aplicaciones y los procesos de servicio evolucionen concurrentemente.

Código móvil

La capacidad para ejecutar applets en un cliente elimina la carga del servidor web, permitiendo proporcionar mejor servicio.

- El balance de cargas puede implicar mover el trabajo parcialmente completado, como carga a un dispositivo alternativo.



Calidad de servicio QoS

- Las principales características no funcionales de los sistemas, que afectan a la calidad del servicio son:
 - ① Fiabilidad,
 - ② Seguridad y
 - ③ Prestaciones
- *QoS* se define en términos de la respuesta, la productividad computacional y proporcionar garantías oportunas.



Índice

1 Definición de un sistema distribuido



2 Objetivos y desafíos

3 Requisitos de diseño

4 Modelos de Sistema

ESCOM
Escuela Superior de Cómputo

5 Modelos fundamentales



Introducción

- Un modelo arquitectónico de un sistema distribuido trata sobre la colocación de sus partes y las relaciones entre ellas.
 - Cliente - Servidor
 - ① Particinamiento de datos o la replicación en servidores colaborativos.
 - ② El uso de caché para los datos en clientes y servidores proxy.
 - ③ Requisitos para añadir o eliminar dispositivos móviles de forma conveniente.
 - Modelo de procesos de igual a igual *Punto a Punto*.





Introducción

- No hay un tiempo global en un sistema distribuido.
 - La comunicación entre procesos se realiza por medio de mensajes.
 - La comunicación mediante mensajes sobre una red puede verse afectada por retrasos y es vulnerable.

Modelos

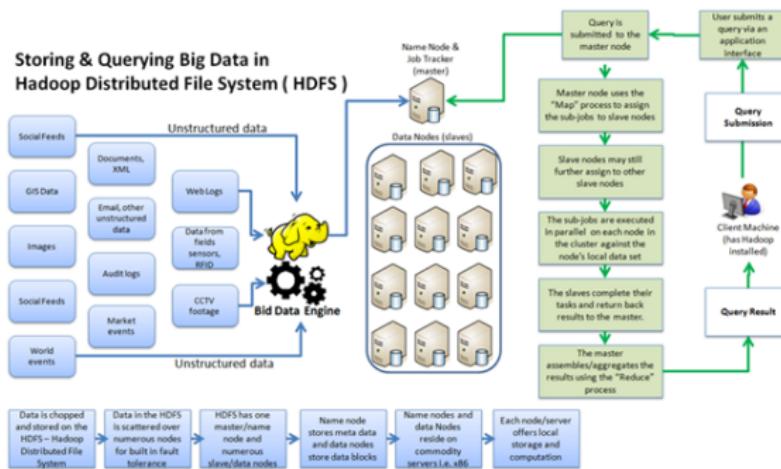
- ① **Modelo de interacción** trata de las prestaciones y la dificultad de poner límites temporales en un sistema distribuido.
 - ② **Modelo de fallos** intenta dar una especificación precisa de los fallos que se pueden producir en los procesos y en los canales de comunicación. (Comunicación fiable y procesos correctos).
 - ③ **El modelo de seguridad** discute sobre las posibles amenazas para los procesos y los canales de comunicación. Introduce el concepto de canal seguro.



Introducción

Diseño de sistemas

Los sistemas deben diseñarse para funcionar correctamente en el rango de circunstancias más amplio posible y considerando todas las dificultades y amenazas.





Introducción

Diseño de sistemas

Los sistemas deben diseñarse para funcionar correctamente en el rango de circunstancias más amplio posible y considerando todas las dificultades y amenazas.

Modelo arquitectónico

Un modelo arquitectónico define la forma en que los componentes de un sistema interactúan uno con otro y cómo están vinculados con la red de comunicaciones subyacente.



Dificultades y amenazas para los sistemas distribuidos

Modos de utilización muy variables: Los componentes de un sistema están sujetos a grandes variaciones en la carga de trabajo.

Amplio rango de entornos: Un sistema distribuido debe operar con hardware, sistemas operativos y redes heterogéneas.

Problemas internos: Relojes no sincronizados, actualizaciones conflictivas de datos, muchas formas de fallos en hardware y software implicando a los componentes individuales de un sistema.

Amenazas externas: Ataques a la seguridad y el secreto de los datos, denegación de servicio.

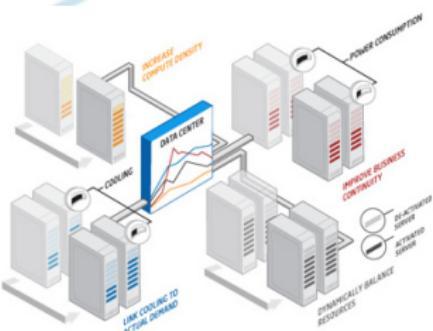


Objetivo de una arquitectura

- La arquitectura de un sistema es su estructura en términos de componentes especificados por separado.
- Se debe asegurar que la arquitectura satisfará las demandas presentes y previsibles sobre el sistema distribuidos.



ESCC
Escuela Superior de





Objetivo de una arquitectura

- Una simplificación inicial se obtiene clasificando los procesos entre: *servidores, clientes e iguales*.
- Clasificar a los procesos identifica las responsabilidades de cada uno y ayuda a valorar sus cargas de trabajo y a determinar el impacto de los fallos en cada uno de ellos.

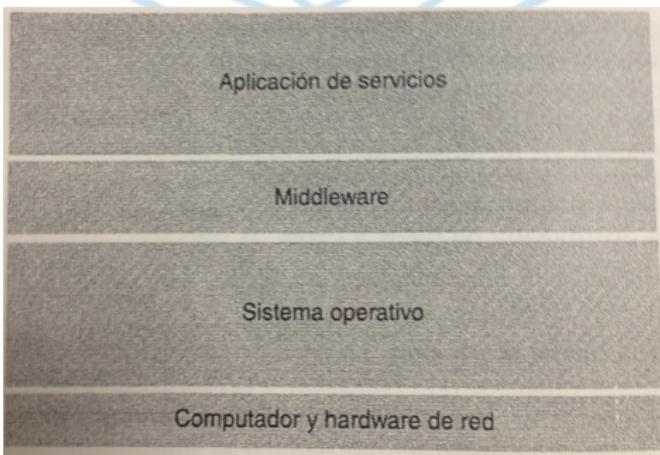
Resultado del análisis

Los resultados del análisis del modelo arquitectónico pueden utilizarse para especificar la distribución de los procesos de una forma que concuerde con los objetivos de **prestaciones y fiabilidad** del sistema resultante.



Capas de software

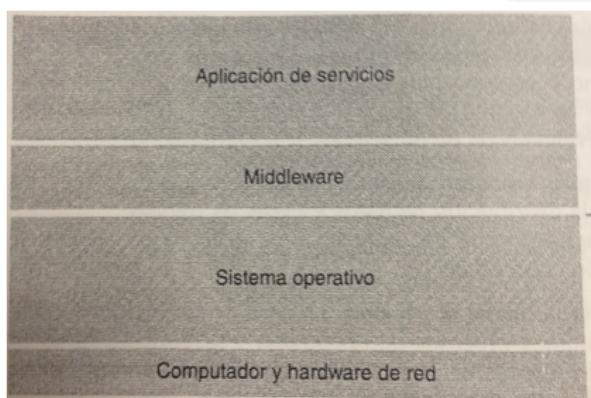
- El término arquitectura de software se refería a la estructuración del software como capas o módulos en un único dispositivo de cómputo.
- Recientemente se define en términos de los servicios ofrecidos en el mismo o en diferentes equipos.





Capas de software: Plataforma

- El nivel de hardware y las capas más bajas de software se denominan, a menudo, plataforma para sistemas distribuidos y aplicaciones.
- Las capas más bajas proporcionan servicios a las que están por encima de ellas y son implementadas independientemente en cada computadora.



- Windows para Intel x86
- Sun Os para Sun SPARC
- Solaris para Intel x86
- Mac OS para PowerPC
- Linux para x86



Capas de software: Middleware

- Su propósito es enmascarar la heterogeneidad y proporcionar un modelo de programación conveniente para los programadores de aplicaciones
- Se presenta mediante procesos u objetos en un conjunto de computadoras.
- Se ocupa de proporcionar bloques útiles para la construcción de componentes de software que puedan trabajar con otros.
- Mejora el nivel de las actividades de comunicación de los programas de aplicación:
 - ① procedimientos de invocación remota,
 - ② comunicación entre un grupo de procesos,
 - ③ notificación de eventos,
 - ④ replicación de datos, etc.



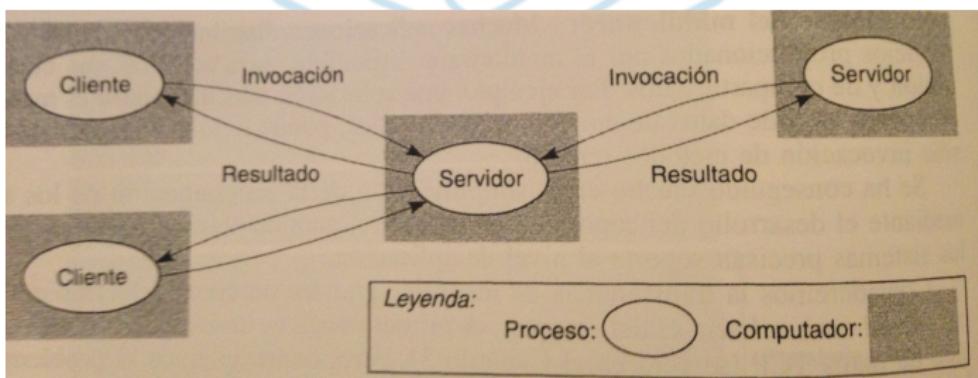
Arquitecturas de sistemas

- La división de responsabilidades entre los componentes del sistema y la ubicación de los componentes en las computadoras en la red, es quizá el aspecto más evidente del diseño de un sistema distribuido.
- Sus implicaciones fundamentales están en las **prestaciones, fiabilidad y seguridad** del sistema resultante.
- Los procesos con responsabilidades bien definidas interactúan con otros para realizar una actividad útil.



Modelo cliente-servidor

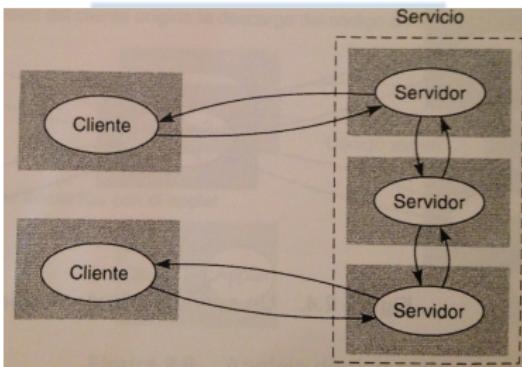
- Históricamente es la más importante y continua siendo la más utilizada.
- Los servidores pueden, a su vez, ser clientes de otros servidores





Servicios proporcionados por múltiples servidores

- Los servicios pueden implementarse como distintos procesos de servidor en computadoras separadas interaccionando para proporcionar un servicio a los procesos clientes.
- Los servidores pueden dividir el conjunto de objetos en los que está basado el servicio y distribuirselos entre ellos mismos
- Pueden mantener copias replicadas de ellos en varias máquinas.





Servicios proporcionados por múltiples servidores

- Los servicios pueden implementarse como distintos procesos de servidor en computadoras separadas interaccionando para proporcionar un servicio a los procesos clientes.
- Los servidores pueden dividir el conjunto de objetos en los que está basado el servicio y distribuirselos entre ellos mismos
- Pueden mantener copias replicadas de ellos en varias máquinas.

Replicación

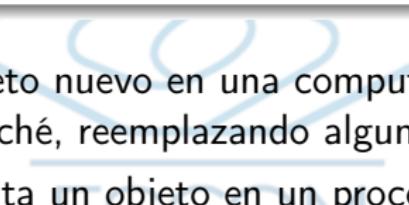
La **replicación** se utiliza para aumentar las prestaciones y disponibilidad y para mejorar la tolerancia a fallos.



Servidores proxy y cachés

Caché

Una cache es un almacén de objetos de datos utilizados recientemente y que se encuentra más próximo que los objetos en sí.

- 
- Escuela Superior de Cómputo
- ① Al recibir un objeto nuevo en una computadora se añade al almacén de la caché, reemplazando algunos objetos existentes.
 - ② Cuando se necesita un objeto en un proceso cliente, el servicio caché comprueba inicialmente la caché y le proporciona el objeto de una copia actualizada.
 - ③ Si no, se buscará una copia actualizada.

Las cachés pueden estar ubicadas en cada cliente o en un servidor proxy que puede compartirse desde varios clientes.



Servidores proxy y cachés

- ① Al recibir un objeto nuevo en una computadora se añade al almacen de la caché, reemplazando algunos objetos existentes.
- ② Cuando se necesita un objeto en un proceso cliente, el servicio caché comprueba inicialmente la caché y le proporciona el objeto de una copia actualizada.
- ③ Si no, se buscará una copia actualizada.

Las cachés pueden estar ubicadas en cada cliente o en un servidor proxy que puede compartirse desde varios clientes.

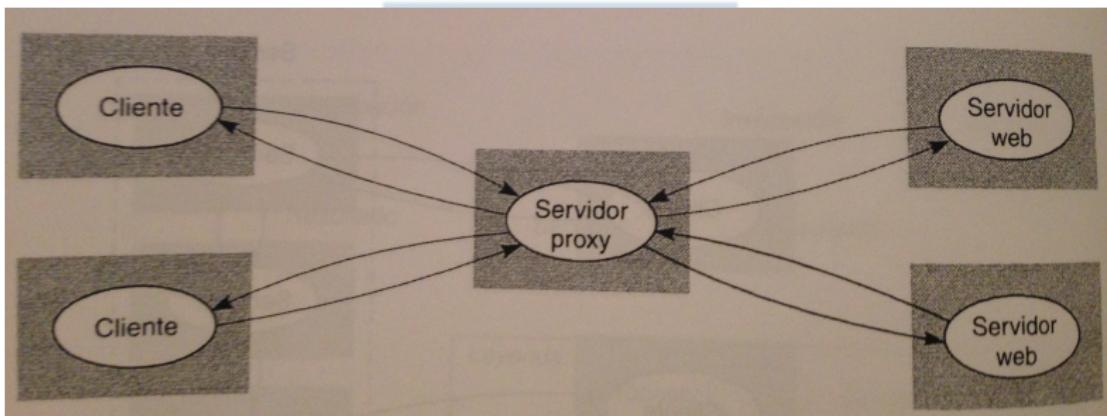
Escuela Superior de Cómputo

Objetivo

El propósito de los servidores proxy es incrementar la disponibilidad y prestaciones del servicio, reduciendo las cargas de en redes y en servidores web.



Servidores proxy y cachés

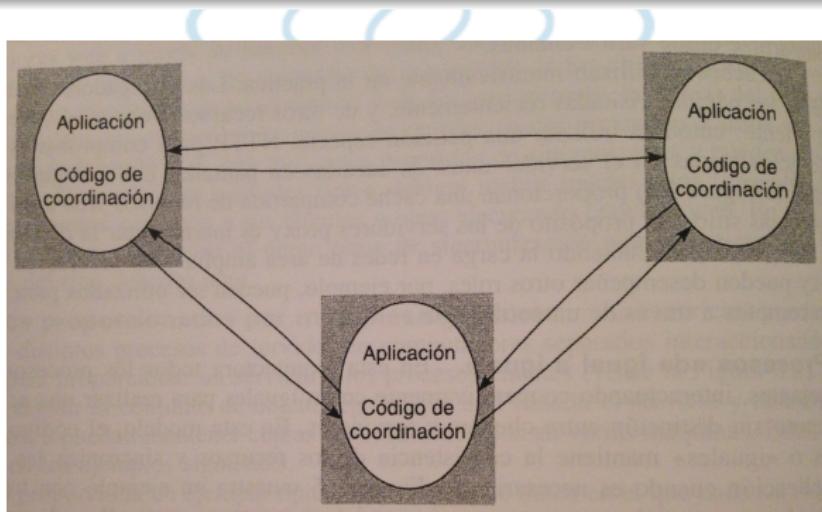




Procesos de igual a igual

Definición

Los procesos desempeñan tareas semejantes, interactuando cooperativamente como iguales para realizar una actividad distribuida o cómputo **sin distinción entre clientes y servidores**.





Procesos de igual a igual

- El código en los procesos mantiene la consistencia de los recursos y sincroniza las acciones a nivel de la aplicación cuando es necesario.
- n procesos podrán interactuar entre ellos, dependiendo del patrón de comunicación y de los requisitos de la aplicación.
- Eliminar el proceso servidor reduce los retardos de comunicación entre procesos al acceder a objetos locales.



Variaciones en el modelo Cliente - Servidor

Se pueden distinguir algunas variaciones al modelo con base en:

- El uso de código móvil y agentes móviles
- La necesidad de los usuarios de equipo de bajo costo y con recurso hardware limitados.
- El requisito de añadir o eliminar de una forma conveniente dispositivos móviles.

Ejemplos:

- ① Código móvil
- ② Agentes móviles
- ③ Computadoras en red
- ④ Clientes ligeros





Índice

1 Definición de un sistema distribuido



2 Objetivos y desafíos

3 Requisitos de diseño

4 Modelos de Sistema



5 Modelos fundamentales



Introducción

Modelo

Un modelo contiene los ingredientes que necesitamos para comprender y razonar sobre comportamiento de un sistema.

- ¿Cuáles son las principales entidades del sistema?
- ¿Cómo interactuan?
- ¿Cuáles son las características qye afetan sy comportamiento individual y colectivo?

Objetivo

- ① Hacer explícitas todas las premisas relevantes sobre los sistemas que se modelan.
- ② Hacer generalizaciones respecto a lo que es posible o no.



Introducción

Los aspectos de los sistemas distribuidos que se desean capturar en un modelo son:

Interacción: los procesos interactúan por medio de paso de mensajes (comunicación y sincronización). El **modelo de interacción** debe reflejar hechos como que la comunicación tiene lugar con retrasos.

Fallo: la correcta operación de un sistema distribuido se ve amenazada allá donde aparezca un fallo. Un modelo debe definir y clasificar los fallos.

Seguridad: la naturaleza modular de los sistemas distribuidos y su extensibilidad los expone a ataques tanto de agentes externos como internos. El modelo de seguridad define y clasifica los modos en que pueden tener lugar tales ataques.

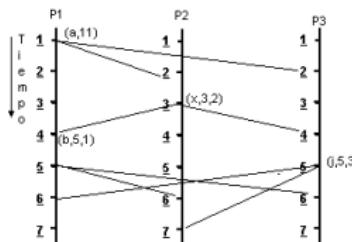


Modelo de interacción

- La mayoría de los desarrolladores están familiarizados con el concepto de algoritmo.
- Los programas simples están controlados por algoritmos en los que los pasos son estrictamente secuenciales.
- Los sistemas distribuidos son más complejos. Su comportamiento y el estado puede describirse mediante un algoritmo distribuido: *una definición de los pasos que hay que llevar a cabo por cada uno de los procesos de los que consta el sistema.*

ESCOM

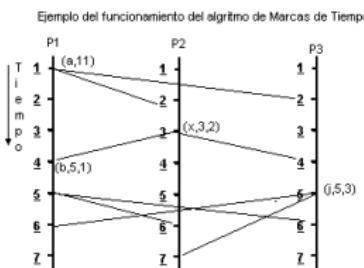
Ejemplo del funcionamiento del algoritmo de Marcas de Tiempo





Modelo de interacción

- La tasa con que procede cada proceso y la temporización de cada mensaje no puede predecirse.
- También es complicado describir todos los estados de un algoritmo distribuido, debido a la naturaleza y fuentes de los errores.
- En un sistema distribuido toda la acción la soportan los procesos que interactúan. Cada proceso tiene su propio estado, que consiste en el conjunto de datos a los que puede acceder y actualizar, incluyendo las variables internas al programa.





Prestaciones de los canales de comunicación

- Los canales de comunicación se implementan mediante *streams* o por un simple paso de mensajes sobre una red de computadoras.

Latencia: El retardo entre el envío de un mensaje por un proceso y su recepción por otro se denomina latencia.

- El tiempo que toma en llegar a su destino el primero de una trama de bits.
 - El retardo en acceder a la red, que es grande cuando la red está muy cargada.
 - El tiempo empleado por los servicios de comunicación del sistema operativo.

Ancho de banda: es la cantidad total de información que puede transmitirse en un intervalo de tiempo.

Jitter: es la variación en el tiempo invertido en completar el reparto de una serie de mensajes.



Reloj de computadoras y eventos de temporización

- Cada computadora de un sistema distribuido tiene su propio reloj interno.
 - Los relojes presentan diferencias con respecto al tiempo perfecto y sus tasas de *precisión* difieren.

Tasa de deriva

El término tasa de deriva del reloj alude a la proporción en que el reloj de una computadora difiere del reloj de referencia perfecto.

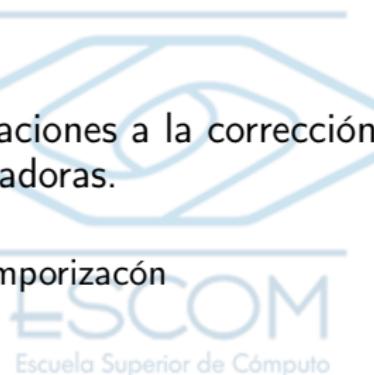
Intento de sincronización

Si los relojes de todas las computadoras de un sistema distribuido se ponen a una misma hora a la vez, a partir de un tiempo sus relojes variarán en una magnitud significativa a **menos que se apliquen correcciones**.



Reloj de computadoras y eventos de temporización

- Hay varias aproximaciones a la corrección de los tiempos de reloj de las computadoras.
 - ① GPS
 - ② Mensajes de temporización





Variantes del modelo de interacción

- En un sistema distribuido es difícil establecer cotas sobre el tiempo que debe tomar la ejecución de un proceso, el reparto de un mensaje o la deriva del reloj.
 - **Sistemas distribuidos síncronos:** cuenta con las siguientes restricciones:
 - ① El tiempo de ejecución de cada etapa de un proceso tiene ciertos límites inferior y superior.
 - ② Cada mensaje transmitido sobre un canalse recibe en un tiempo limitado conocido.
 - ③ Cada proceso tiene un reloj local cuya tasa de deriva sobre el tiempo real tiene un límite conocido.

Garantizar lo límites

Es difícil obtener valores realistas y además dar garantías sobre los valores elegidos. A menos que se puedan garantizar los límites, cualquier diseño basado en dichos valores no será fiable.



Variantes del modelo de interacción

- En un sistema distribuido es difícil establecer cotas sobre el tiempo que debe tomar la ejecución de un proceso, el reparto de un mensaje o la deriva del reloj.
 - **Sistemas distribuidos asíncronos:** un sistema distribuido asíncrono es aquel en el que no existen limitaciones sobre:
 - ① La velocidad de procesamiento, *a cada paso de un proceso le tomará un intervalo de tiempo arbitrariamente largo.*
 - ② Los retardos de transmisión de mensajes, *un mensaje puede recibirse tras un tiempo arbitrariamente largo.*
 - ③ Las tasas de deriva de un reloj son arbitrariamente largas.



Variantes del modelo de interacción

- En un sistema distribuido es difícil establecer cotas sobre el tiempo que debe tomar la ejecución de un proceso, el reparto de un mensaje o la deriva del reloj.
 - **Sistemas distribuidos asíncronos:**
 - El modelo asíncrono no presume nada sobre los intervalos de tiempo involucrados en cualquier ejecución (modelo de internet).
 - Los sistemas distribuidos reales son frecuentemente asíncronos a causa de la necesidad de los procesos de compartir procesadores y de los canales de comunicación de compartir la red.



Modelo de Fallo

Definición

El modelo de fallo define las formas en que puede ocurrir un fallo para darnos una comprensión de los efectos de los fallos.

Fallo por omisión: se refieren a casos en los que los procesos o los canales de comunicación no consiguen realizar acciones que se supone pueden hacer.

Fallo arbitrario: también denominado fallo *bizantino* se emplea para describir la peor semántica de fallo posible, en la que puede ocurrir cualquier tipo de error.

Fallo de temporización: se aplican en los sistemas distribuidos síncronos donde se establecen límites en el tiempo de ejecución de un proceso, en el tiempo de reparto de un mensaje y en la tasa de derivación de un reloj.



Ejemplo: Fallos por omisión y fallos arbitrarios

Clase de fallo	Afecta a	Descripción
Fallo-parada	Proceso	El proceso para y permanece parado. Otros procesos pueden detectar este estado.
Ruptura	Proceso	El proceso para y permanece parado. Otros procesos pueden no ser capaces de detectar este estado.
Omisión	Canal	Un mensaje insertado en el búfer de mensajes salientes nunca llega al búfer de mensajes entrantes del otro extremo.
Omisión de envío	Proceso	Un proceso completa y envía, pero el mensaje no es colocado en el búfer de mensajes salientes.
Omisión de recepción	Proceso	El mensaje es colocado en la cola de mensajes del proceso, pero el proceso no lo recibe.
Arbitrario (Bizantino)	Proceso o canal	El proceso/canal presenta un comportamiento arbitrario.

- En un SD pueden fallar tanto los procesos como los canales de comunicación:
- Los fallos pueden categorizarse según su severidad.



Ejemplo: Fallo de temporización

Clase de fallo	Afecta a	Descripción
Reloj	Proceso	El reloj local del proceso excede el límite de su tasa de deriva sobre el tiempo real.
Prestaciones	Proceso	El proceso excede el límite sobre el intervalo entre dos pasos.
Prestaciones	Canal	La transmisión de un mensaje toma más tiempo que el límite permitido.





Modelo de seguridad

- El modelo arquitectónico da la base para un modelo de seguridad.

Definición

La seguridad de un sistema distribuido puede lograrse asegurando los procesos y los canales empleados para sus interacciones y protegiendo los objetos que encapsulan contra el acceso no autorizado.



Modelo de seguridad

Criptografía La criptografía es la ciencia de mantener los mensajes seguros y la encriptación es el proceso de trastocar un mensaje de modo que quede oculto el contenido.

Autenticación Es el proceso de detectar y comprobar la identidad de una entidad de seguridad examinando las credenciales del usuario y validando esas credenciales contra alguna autoridad.

Canales seguros Es un canal de comunicación que conecta un par de procesos, cada uno de los cuales actúa en representación de un principal (proceso o usuario con un rol definido).



Propiedades de un canal seguro

- ① Cada proceso conoce bien la identidad del principal en cuya representación se ejecuta un proceso.
 - ② Un canal seguro asegura la privacidad y la integridad (protección contra la manipulación) de datos transmitidos por él.
 - ③ Cada mensaje incluye un sello de carácter temporal, de tipo físico o lógico, para prevenir el reenvío o la reordenación de los mensajes.