Caracterización de los sistemas distribuidos

Desarrollo de Sistemas en Red

“A distributed system is a collection of **autonomous** computing elements that **appears** to its users **as a single coherent system**.”

– van Steen & Tanenbaum

Definición de sistema distribuido

Elementos que destacan de la definición anterior:

- Autónomos: Independientes unos de otros
- Sin noción colectiva de tiempo
- Necesidad de gestionar la membresía y organización
- Transparencia: Un solo sistema coherente

"[. . .] uno en el que la falla de una computadora que ni siquiera sabía que existía puede hacer que su propia computadora sea inutilizable "

– Leslie Lamport

¿Qué caracteriza a un sistema distribuido?

Concurrencia de sus componentes

Falla independiente de sus componentes

Carencia de un reloj global

Concurrencia

- La ejecución concurrente de programas en una red de computadoras es la norma.
- Se puede incrementar la capacidad de manejar recursos compartidos añadiendo más recursos en la red.
- La coordinación de la ejecución de programas concurrentes que comparten recursos es un tópico importante y recurrente.

Carencia de un reloj global

- Cuando los programas necesitan cooperar éstos coordinan sus acciones mediante el intercambio de mensajes.

- La coordinación estrecha normalmente depende de una referencia común del tiempo en el que las acciones del programa ocurren.

- Existen límites en la exactitud en la que se pueden sincronizar los relojes.

  - Consecuencia del hecho de que la *única* comunicación es mediante el envío de mensajes a través de la red.

Falla independiente de sus componentes

- Todos los componentes del sistema pueden fallar por lo que es responsabilidad del diseñador planear las consecuencias de las posibles fallas.
- Las fallas en la red dan como resultado el aislamiento de una computadora pero eso no quiere decir que se ha detenido.
- Las fallas en algún lugar del sistema no se conocen inmediatamente por todos los componentes con los que se están comunicando.

¿Por qué se construyen sistemas distribuidos?

- La principal motivación para construir este tipo de sistemas es el deseo de **compartir recursos**
  - Recurso: Cosas que se pueden compartir de manera útil en una red de computadoras, ej.
    - **Hardware**: CPU, discos, impresoras
    - **Software**: archivos, bases de datos y objetos de datos de todo tipo

¿Cuáles son los retos de desarrollar este tipo de sistemas?

- **Heterogeneidad**: De todos sus componentes
- **Apertura**: ¿Cómo hacer sistemas extensibles?
- **Seguridad**: Protección de recursos y protección de información
- **Escalabilidad**: Incremento en la carga o número de usuarios
- **Manejo de fallas**: ¿Qué hacer si un componente falla?
- **Concurrencia**: Recursos a salvo en un ambiente concurrente
- **Transparencia**: Algunos aspectos invisibles
- **Calidad del servicio**: Desempeño, seguridad y disponibilidad

Tendencias en sistemas distribuidos

- El uso generalizado de tecnologías de red.
- La aparición del **cómputo ubicuo** y la movilidad de los usuarios.
- La creciente demanda de servicios multimedia.
- La visión de los sistemas distribuidos como un *commodity* (mercancía).

Móviles y sistemas distribuidos

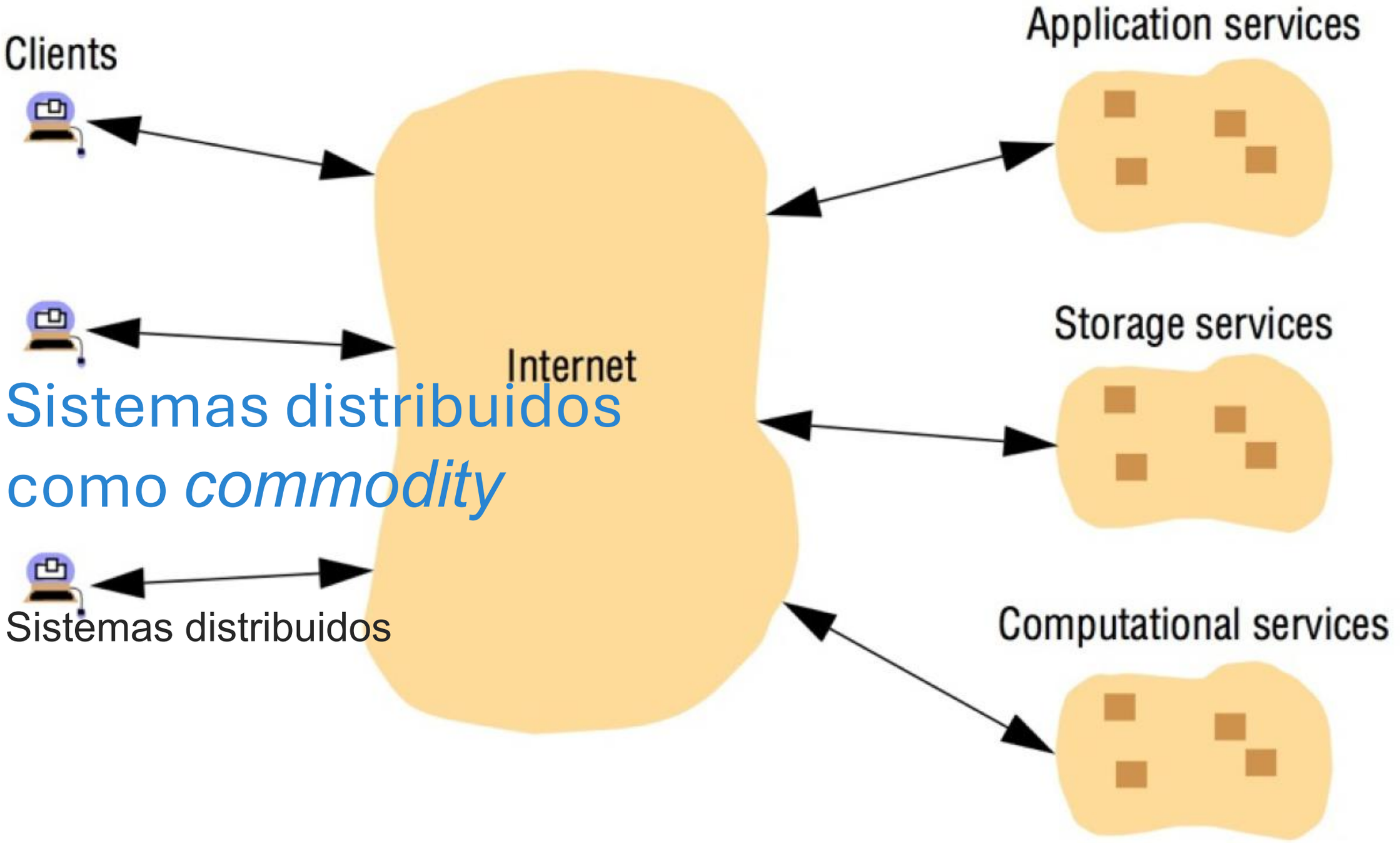
- Integración de dispositivos a los sistemas distribuidos:
  - Laptos, Handhelds, Wearables, Dispositivos empotrados, IoT
- Cómputo móvil:
  - Consciente de contexto y consciente de localización
- Interoperación espontánea
- Descubrimiento de servicios

Sistemas multimedia distribuidos

- Transmisiones en vivo o pre-ordenadas
- Video bajo demanda
- Bibliotecas musicales
- Audio y video conferencia, webcasting, telefonía IP
- Soportar un amplio rango de formatos de codificación y encriptación
- Mecanismos para asegurar la calidad del servicio

Sistemas distribuidos como *commodity* (mercancía)

- Los recursos que proveen los sistemas son rentados en lugar de ser propiedad de los usuarios finales.
- Incluye tanto recursos físicos como lógicos
  - Físico: Almacenamiento y procesamiento
  - Lógico: Servicios de email, calendarios distribuidos, etc.
- Cloud computing, Cluster computing, Grid computing

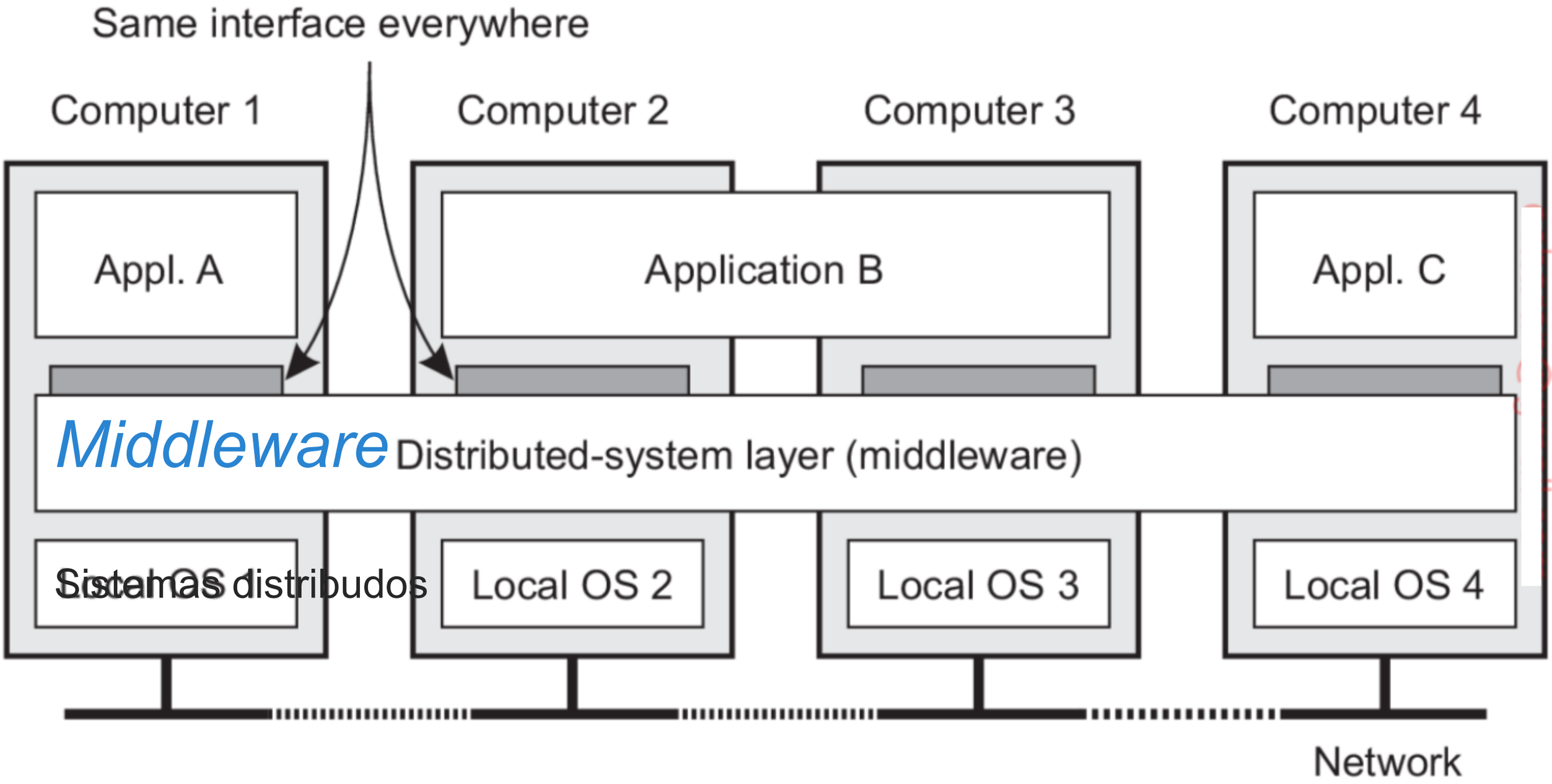


Sistemas distribuidos como *commodity*

Sistemas distribuidos

Middleware y los sistemas distribuidos

- Capa que se coloca lógicamente sobre los sistemas operativos de las computadoras que forman parte del sistema distribuido.
- Proporciona los medios para que los componentes de una sola aplicación distribuida se comuniquen entre sí.
- Oculta las diferencias en hardware y sistemas operativos de cada componente.


Same interface everywhere
Computer 1
Computer 2
Computer 3
Computer 4
Appl. A
Application B
Appl. C
Middleware
Distributed-system layer (middleware)
Local OS 1
Local OS 2
Local OS 3
Local OS 4
Network

- El ***middleware*** es para un SD lo que un sistema operativo es para una computadora:
- **Es un administrador** que ofrece recursos y otros servicios a aplicaciones de manera eficiente en una red:
  - Comunicación inter aplicación
  - Seguridad
  - Registro
  - Enmascaramiento y recuperación de fallos

- La principal diferencia con los sistemas operativos es que los servicios del middleware se ofrecen en un entorno de red.
- Se puede ver como un contenedor de componentes y funciones de uso común que ahora las aplicaciones no tienen que implementar por separado.

Servicios de middleware típico

- Comunicación
- Transacciones (atómicas)
- Composición de servicios
- Confiabilidad

Enfoque en compartir recursos

¿Qué recursos? ¿Por qué compartir?

- Se comparte para **reducir costos**.
- Se busca es compartir datos para las actividades diarias (archivos, bases de datos, etc.)
- Los diferentes recursos se administran de diferente manera pero existen algunos **requerimientos generales**:
  - Espacio de nombres para la identificación
  - Traducción de nombre a dirección de red
  - Sincronización de acceso múltiple

Algunos términos

- **Servicio**: Parte de un sistema que administra una colección de recursos relacionados y que presentan su funcionalidad a usuarios y aplicaciones.

  - Restringen el acceso a los recursos a través de un conjunto de operaciones bien definidas.

  - Servicios de archivos - Conjunto de operaciones para leer, escribir y eliminar archivos

- **Servidor**: Un programa en ejecución (*proceso*) en una computadora en red que acepta solicitudes de programas en otras computadoras para desempeñar un servicio y responder apropiadamente

- A los procesos que realizan la solicitud se les conoce como **clientes** y al enfoque de este tipo de sistemas se le llama **cómputo cliente-servidor**.
- Cuando un cliente envía una solicitud (en un mensaje) para realizar una operación decimos que se **invoca** una operación en el servidor, una **invocación remota**.
- En implementaciones OO los recursos pueden estar encapsulados como objetos y accedidos por objetos clientes, es decir, un *objeto cliente* invocando un método de un *objeto servicio.*

Retos de los sistemas distribuidos

Heterogeneidad

- Variedad y diferencia en:
  - Redes
  - Hardware
  - Sistemas operativos
  - Lenguajes de programación
  - Implementaciones de diferentes desarrolladores
- Es necesario acordar **estándares** y adoptarlos

Middleware

- Capa de software que provee una **abstracción de programación** y **enmascara la heterogeneidad** de redes, hardware, sistemas operativos y lenguajes de programación, ej.
  - The Common Object Request Broker Architecture (CORBA)
  - Java Remote Method Invocation (RMI)
- Proveen un modelo computacional uniforme para el uso de desarrolladores de servicios y aplicaciones distribuidas, ejemplos de modelos:
  - Invocación remota de objetos, notificación remota de eventos, acceso a SQL remoto y procesamiento de transacciones distribuidas

Heterogeneidad y código móvil

- **Código móvil**: Código de programación que puede ser transferido de una computadora a otra y ser ejecutada en la computadora destino.
  - Por ej. Applets de Java y programas de JavaScript.
- **Enfoque de máquina virtual**: Una manera de hacer el código ejecutable en una variedad de computadoras huéspedes
  - El computador de un lenguaje en particular genera código para una máquina virtual en lugar de un hardware en particular. Ej. Máquina Virtual de Java

Apertura

- En un **sistema cómputo** determina si un sistema puede ser extendido y reimplementado de diferentes formas.
- En los **sistemas distribuidos** está determinada principalmente por el grado en el que nuevos servicios para compartir recursos pueden ser añadidos y estar disponibles.
- Un **sistema abierto** necesita que sus interfaces principales sean publicadas.
- Los sistemas distribuidos abiertos tienen:
  - Mecanismos de comunicación uniformes
  - Interfaces publicadas para compartir recursos

- Los **sistemas abiertos** se caracterizan por el hecho de que sus interfaces principales se encuentran publicadas.

- Los **sistemas distribuidos abiertos** (SDA) se basan en proveer mecanismo de comunicación uniforme y publicar interfaces para el acceso a recursos compartidos.

- Se pueden construir de hardware y software heterogéneo posiblemente de diferentes fabricantes.

- La conformidad de cada componente con el estándar publicado debe probarse y verificarse.

Seguridad

1. **Confidencialidad**: Protección contra la revelación de información a personas no autorizadas.

2. **Integridad**: Protección contra la alteración y corrupción.

3. **Disponibilidad**: Protección contra la interferencia a los medios por los que se accede a los recursos.

- Los desafíos que aún no se han resuelto por completo
  - Ataques de denegación de servicios
  - Seguridad del código móvil

Escalabilidad

- Habilidad de trabajar bien cuando la carga del sistema o el número de usuarios se incrementa
- Retos al construir sistemas distribuidos escalables:
  - Controlar el costo de los recursos físicos
  - Controlar la pérdida desempeño
  - Prevenir que se agoten los recursos de software
  - Evitar cuellos de botella
- Técnicas: Replicación de datos, caching, ejecución concurrente/paralela

Manejo de fallas

- Las fallas ocurren tanto en hardware como en software.
- Las fallas en los sistemas distribuidos **son parciales**, es decir, algunos componentes dejan de funcionar mientras otros siguen funcionando.

Técnicas para manejar fallas

- Detectar las fallas - checksums, sospechar fallas
- Enmascar fallas
  - Los mensajes pueden ser retransmitidos
  - Los discos pueden ser replicados en una acción síncrona
- Tolerar fallas
- Recuperarse de fallas - roll-backs
- Redundancia - de componentes de hardware y software
- **Objetivo**: Alta disponiblidad

Concurrencia

- **Ejemplo**: Varios clientes tratando de acceder a recursos compartidos al mismo tiempo.

- Cualquier objeto con recursos compartidos en un sistema distribuido debe ser responsable de operar correctamente en un ambiente concurrente:

  - Sus operaciones deben estar sincronizadas de tal forma que sus datos se mantengan consistentes.

  - La sincronización puede lograrse a través de técnicas como los semáforos, se utilizan en la mayoría de SO.

Transparencia

- Se define como la **ocultación** de la **separación de componentes** en un sistema distribuido al usuario y al programador de la aplicación.
  - El sistema se percibe como un todo y no como una colección de componentes independientes.
- Tiene una gran influencia en el diseño de sistemas distribuidos.

Formas de transparencia

1. **De acceso**: Permite que los recursos locales y remotos sean accedidos utilizando las mismas operaciones, de forma idéntica

2. **De localización**: Se accede a los recursos sin saber su localización física o de red.

3. **De concurrencia**: Operaciones concurrentes de procesos utilizando recursos compartidos sin que se interfieran entre ellos.

4. **De replicación**: Múltiples instancias se ven como una sola.

5. **De fallas**: Ocultamiento de las fallas para completar tareas a pesar de ellas.
6. **De movilidad**: Movimiento de recursos/clientes dentro del sistema sin afectar las operaciones de usuarios o programas.
7. **De desempeño**: Permite reconfigurar el sistema para mejorar el desempeño conforme varía la carga.
8. **De escalabilidad**: Permite al sistema y aplicaciones expandirse en escala sin cambiar la estructura del sistema o algoritmos de la aplicación.

Aspectos importantes de la transparencia

- Los tipos más importantes son la de acceso y la de localización.

- Su presencia o ausencia **afectan** en gran medida la utilización de recursos distribuidos.

- Las dos juntas también se les conoce como **transparencia de red**.

- La transparencia oculta y convierte en anónimos los recurso que no son directamente relevantes en la tarea a realizar para usuarios y programadores de aplicaciones.

Calidad del servicio

- Los propiedades no funcionales más importantes de un sistema que afecta la **Calidad de un Servicio (QoS)** son:

    1. **Confiabilidad:** Permite que, en caso de que una computadora falle, otra la pueda sustituir en la realización de sus tareas asignadas.
    2. **Seguridad**: considerar todos los factores de riesgo a que se expone la información en un ambiente distribuido
    3. **Desempeño**: Está en referencia a los tiempos de respuesta de una aplicación
    4. **Adaptabilidad**: facilidad para incorporar cambios y extensiones al istema

Caso de estudio: La World Wide Web

- CERN 1989
- Estructura de hipertexto, hiperenlaces
- La Web es un sistema abierto
- Los estándares de contenido publican libremente y están ampliamente implementados
- Abierta en cuanto al tipo de recursos que pueden publicarse
- Componentes tecnológicos principales: HTML, URIs y HTTP

URI (*Universal Resource Identifier*)

- Identificar un recurso en la web
  - schema:identificador-específico-del-esquema
  - mailto:juan@uv.mx
  - ftp://ftp.descargas.com/software/prog.exe
- Para HTTP

http://nombreservidor[:puerto][/nombreRuta][?consulta][#fragmento]

http://www.w3.org/standards/faq.html#conformance

http://www.google.com/search?q=concurrencia

HTTP

- Interacciones solicitud-respuesta
- Tipos de contenido
- Un recurso por solicitud
- Control de acceso simple
- Páginas dinámicas
- Servicios web

Conclusión

- Los sistemas distribuidos están en todas partes.
- Internet permite el acceso a los recursos de los sistemas distribuidos en cualquier lugar.
- La principal motivación para construir sistemas distribuidos es compartir recursos.

- Las tres características de un sistema distribuido son:

    1. Los componentes operan de manera concurrente.

    2. Los componentes fallan de manera independiente.

    3. Los componentes no comparten un reloj global.

- La construcción de sistemas distribuidos presenta diferentes retos:

    - Heterogeneidad, Apertura, Seguridad, Escalabilidad, Manejo de fallas, Concurrencia, Transparencia y Calidad de servicio