



# **Sistemas Distribuidos**

## **Módulo 7**

**Nombres y Sistemas de Archivos  
en Sistemas Distribuidos**



# **Nombres**



# **Agenda**

## **1. Definiciones**

1. Nombre
2. Entidad
3. Dirección
4. Identificador

## **2. Espacio de Nombres**

## **3. Resolución de Nombres**

1. Mecanismo de Clausura
2. Montaje y Distribución
3. Implementación

## **4. Localización**



# Definiciones

**Nombre:** cadena de bits o caracteres que se usan para referirse a una entidad.

**Entidad:** casi cualquier cosa en un sistema distribuido.

Para operar con una entidad se necesita un **punto de acceso** a la misma.

**Dirección:** es el nombre que recibe el punto de acceso a una entidad

**Identificador** es un nombre que tiene las siguientes propiedades:

1. Un identificador se refiere a lo sumo a una entidad.
2. Cada entidad es referida a lo sumo por un identificador.
3. Un identificador siempre se refiere a una única entidad (nunca es reusado)

# Definiciones

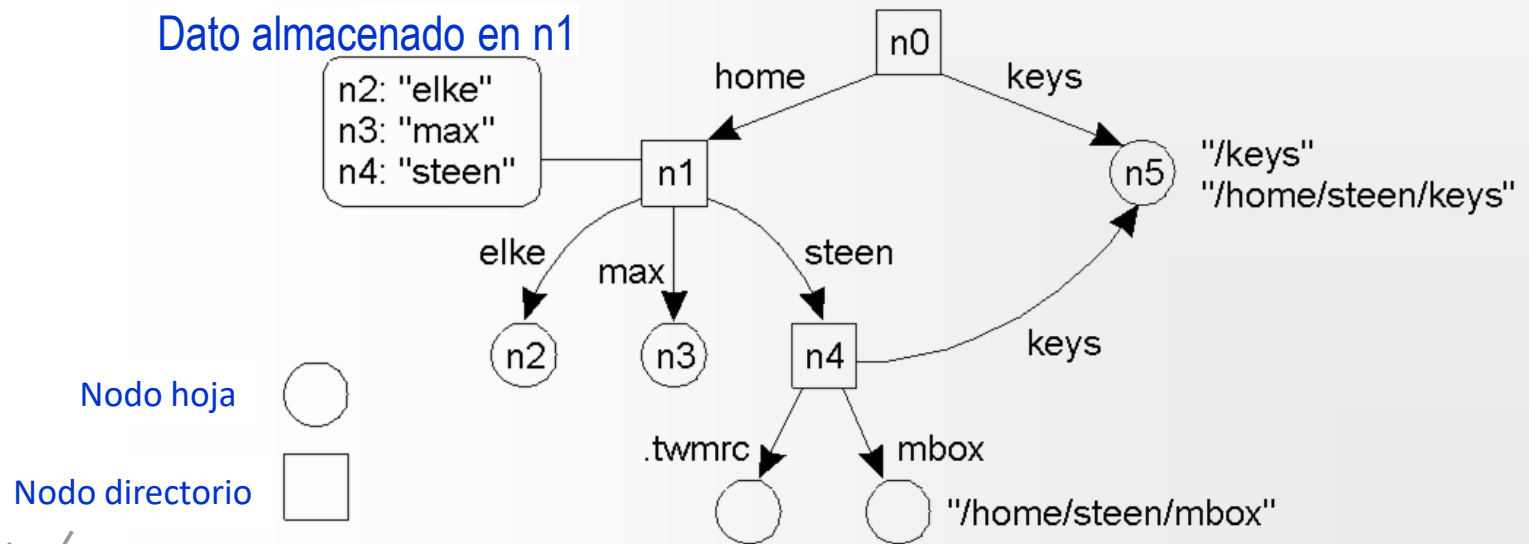
Los nombres en un sistema distribuido están organizados en lo que se llama ESPACIO DE NOMBRES.

Nombre global: es un nombre que denota una entidad no importa donde es usado el mismo en el sistema.

Nombre local: está asociado a un nombre relativo.

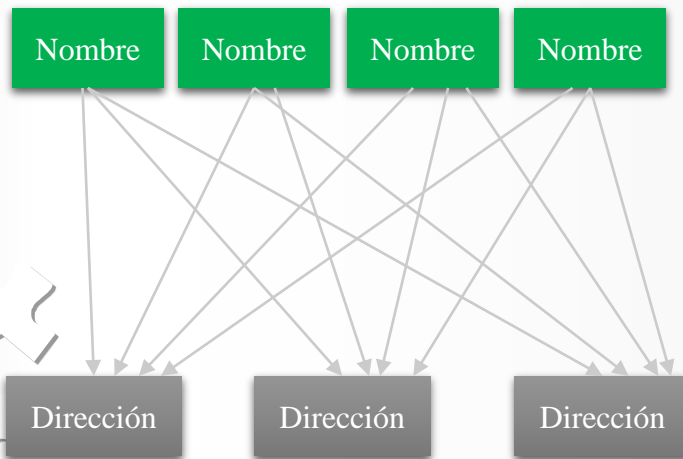
Alias: otro nombre para la misma entidad.

Dato almacenado en n1

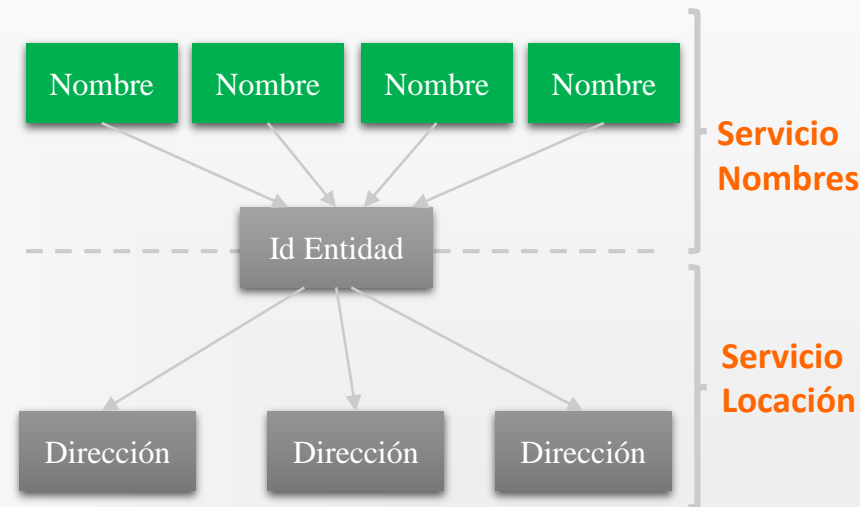


# Nombres versus Locación de Entidades

- Directo, nivel simple de mapeo entre nombres y direcciones.



- Nivel-T mapeo usando entidades.



Hay un mapa de nombres amigables a las direcciones.  
Cada vez que hay un cambio es necesario cambiar las tablas de mapeo.



# Nombres

- **Planos**: compuestos por una cadena de bits, convenientes para las computadoras.
- **Estructurados**: compuestos por un conjunto de nombres, fácilmente entendibles para las personas.
  - Organizados en espacio de nombres

# Ejemplo de Nombres

## URL

http://www.cdk5.net:8888/WebExamples/earth.html

## DNS lookup

Resource ID (IP number, port number, pathname)

55.55.55.55

8888

WebExamples/earth.html

## Network address

2:60:8c:2:b0:5a

## Web server

file

## Socket



# Ejemplo de Nombres y Recursos

**Uniform Resource Identifiers (URI)** ofrece una solución general para cualquier tipo de recurso. Hay dos clases especiales:

## URL: Uniform Resource Locator

- tipificado por el protocolo (http, ftp, nfs, etc.)
- parte del nombre es un servicio específico
- los recursos no se pueden mover entre dominios

## URN: Uniform Resource Name

- require un servicio de búsqueda de nombre

*formato: urn:<nameSpace>:<name-within-namespace>*

*ejemplos:*

*a) urn:ISBN:021-61918-0*

*b) urn:dcs.qmul.ac.uk:TR2000-56*

# Resolución de Nombres

Los *espacios de nombre* ofrecen un mecanismo apropiado para almacenar y recuperar información con respecto a las entidades por medio de nombres.

## Forma de resolución

$N:<\text{etiqueta}_1, \text{etiqueta}_2, \dots \text{etiqueta}_n>$



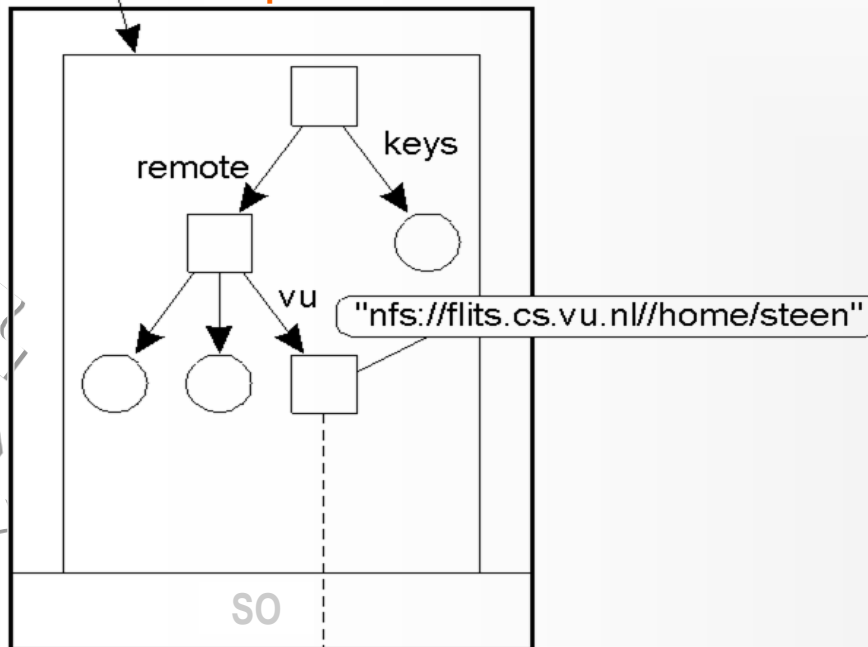
**Mecanismo de Clausura** – la resolución puede tener lugar solamente si se sabe cómo y en dónde buscar.

# Enlace y Montaje

- Montaje de un espacio de nombres remoto por medio de un protocolo específico.

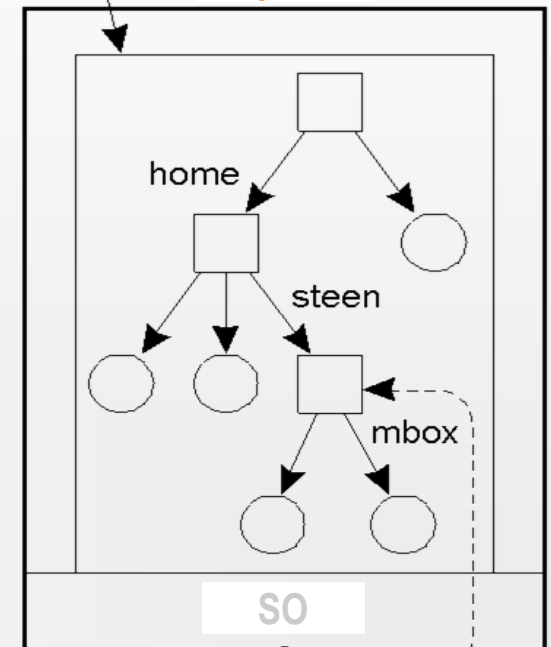
Servidor de nombres

Máquina A



Servidor de nombres para espacio de nombres externo

Máquina B

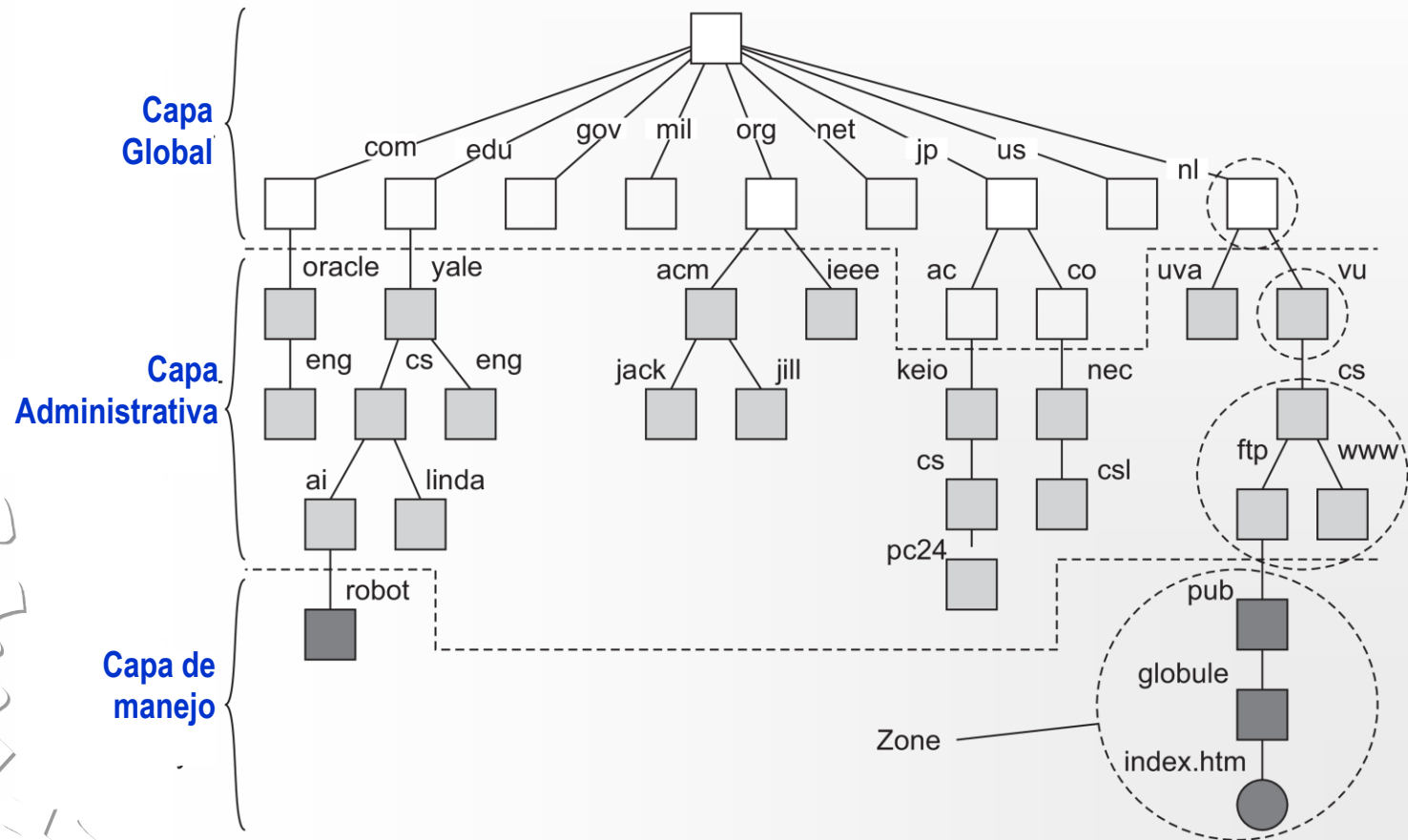


Referencias a un espacio de nombres externo

RED

# Distribución de Espacio de Nombres

- Un ejemplo de particionamiento del espacio de DNS, incluyendo los archivos accesibles por Internet, en tres capas.



# Distribución de Espacio de Nombres

- Una comparación entre servidores de nombres para implementar nodos de un espacio de nombres en una capa global, como una capa administrativa y una capa de manejo.

ITEM	GLOBAL	ADMINISTRATIVO	MANEJO
Escala geográfica de la Red	Mundial	Organización	Departamento
Número total de nodos	Pocos	Muchos	Número grande
Respuesta de búsquedas	Segundos	Milisegundos	Inmediato
Propagación de actualizaciones	Relajada	Inmediato	Inmediato
Número de réplicas	Muchas	Ninguna o pocas	Ninguno
Se aplica caché del lado del cliente?	Si	Si	Algunas veces



# Implementación Resolución de Nombres

Tipos de implementaciones:

- Resolución Iterativa
- Resolución Recursiva

# Implementación Resolución de Nombres

- El principio de Resolución Iterativa de nombres.

1. El cliente envía *resolver(dir; [etiqueta<sub>1</sub>; etiqueta<sub>2</sub>; ...; etiqueta<sub>n</sub>])* al *Server<sub>0</sub>* responsable de *dir*
2. El *Server<sub>0</sub>* resuelve *resolver(dir; [etiqueta<sub>1</sub>; etiqueta<sub>2</sub>; ...; etiqueta<sub>n</sub>])* → *dir<sub>1</sub>*, retorna la identificación (dirección) del *Server<sub>1</sub>*, el cual almacena *dir<sub>1</sub>*.
3. El cliente envía *resolver(dir<sub>1</sub>; [etiqueta<sub>2</sub>; ...; etiqueta<sub>n</sub>])* al *Server<sub>1</sub>* responsable de *dir<sub>1</sub>*

# Implementación Resolución de Nombres

- El principio de Resolución Recursiva de nombres.

1. El cliente envía  $\text{resolver}(\text{dir}; [\text{etiqueta}_1; \text{etiqueta}_2; \dots; \text{etiqueta}_n])$  al  $\text{Server}_0$  responsable de  $\text{Dir}$ .
2. El  $\text{Server}_0$  resuelve  $\text{resolver}(\text{dir}; [\text{etiqueta}_1; \text{etiqueta}_2; \dots; \text{etiqueta}_n]) \rightarrow \text{dir}_1$ , y envía  $\text{resolver}(\text{dir}_1; [\text{etiqueta}_2; \dots; \text{etiqueta}_n])$  al  $\text{Server}_1$  responsable de  $\text{dir}_1$ .
3. El  $\text{Server}_0$  espera el resultado del  $\text{Server}_1$ , y retorna el resultado al Cliente.





# Implementación Resolución de Nombres

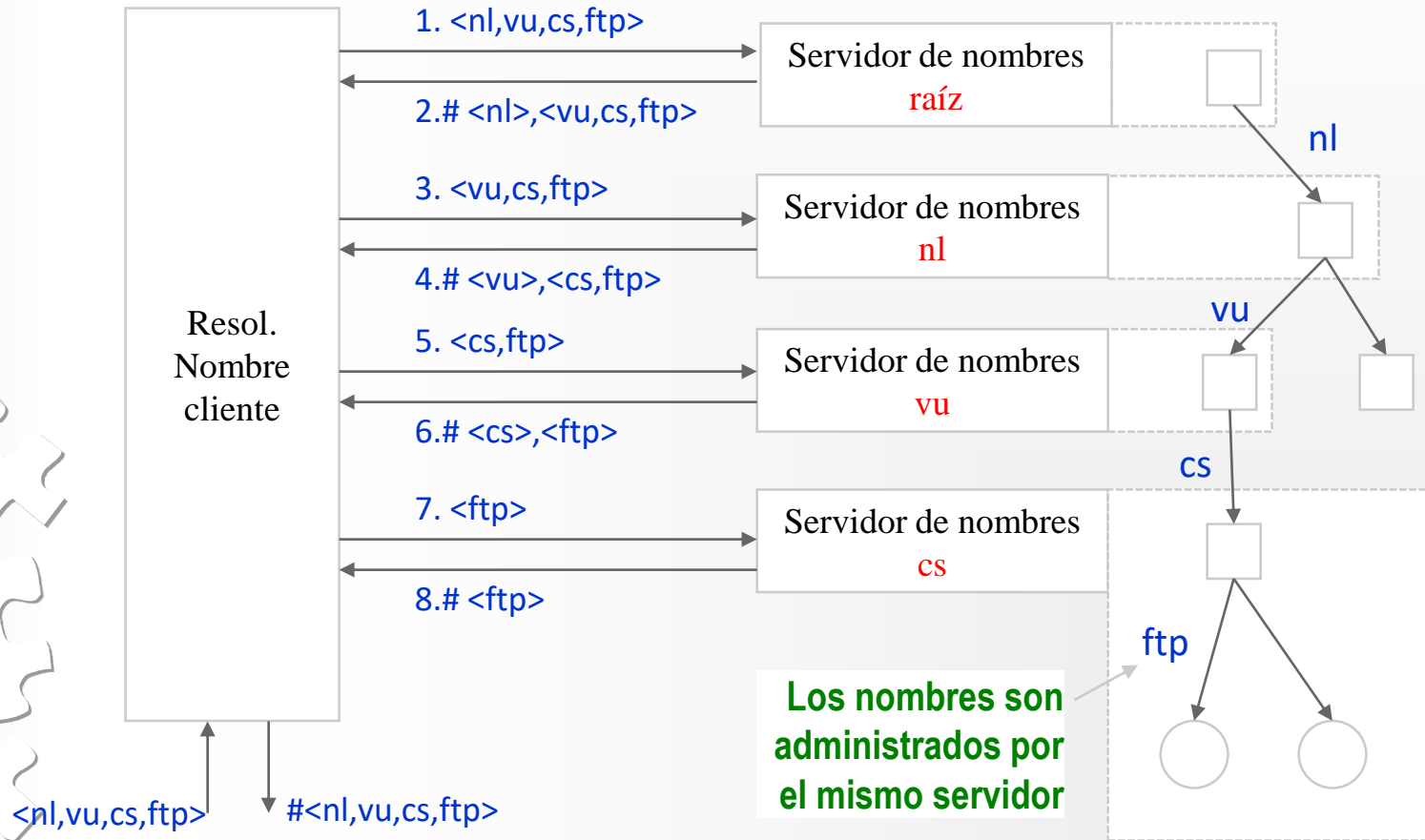
- Resolver el siguiente nombre.

Root:<*nl, vu, cs, ftp, pub, globe, index.html*>

Url: *ftp://ftp.cs.vu.nl/pub/globe/index.html*

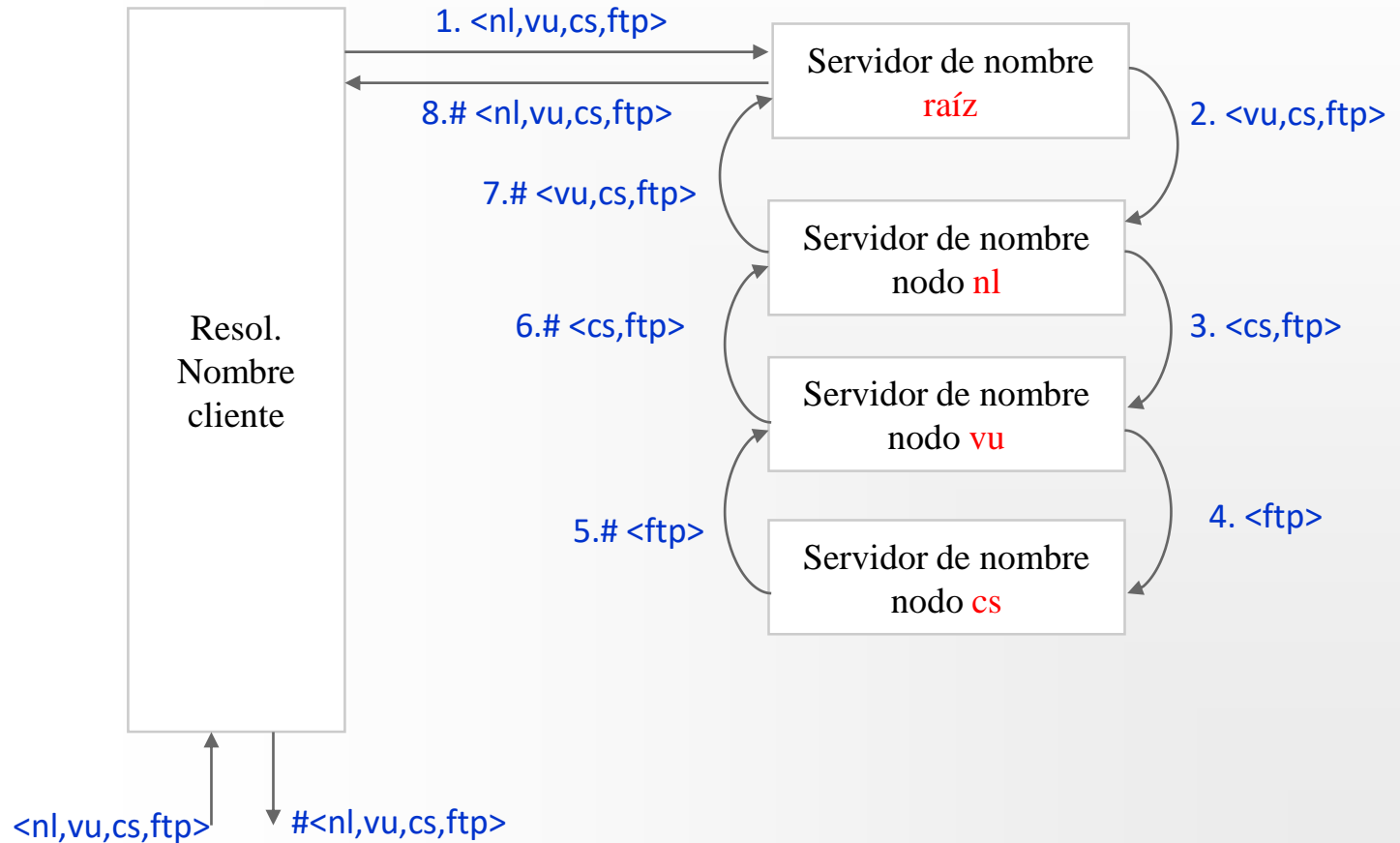
# Implementación Resolución de Nombres

## ○ Ejemplo: Resolución Iterativa de nombres.



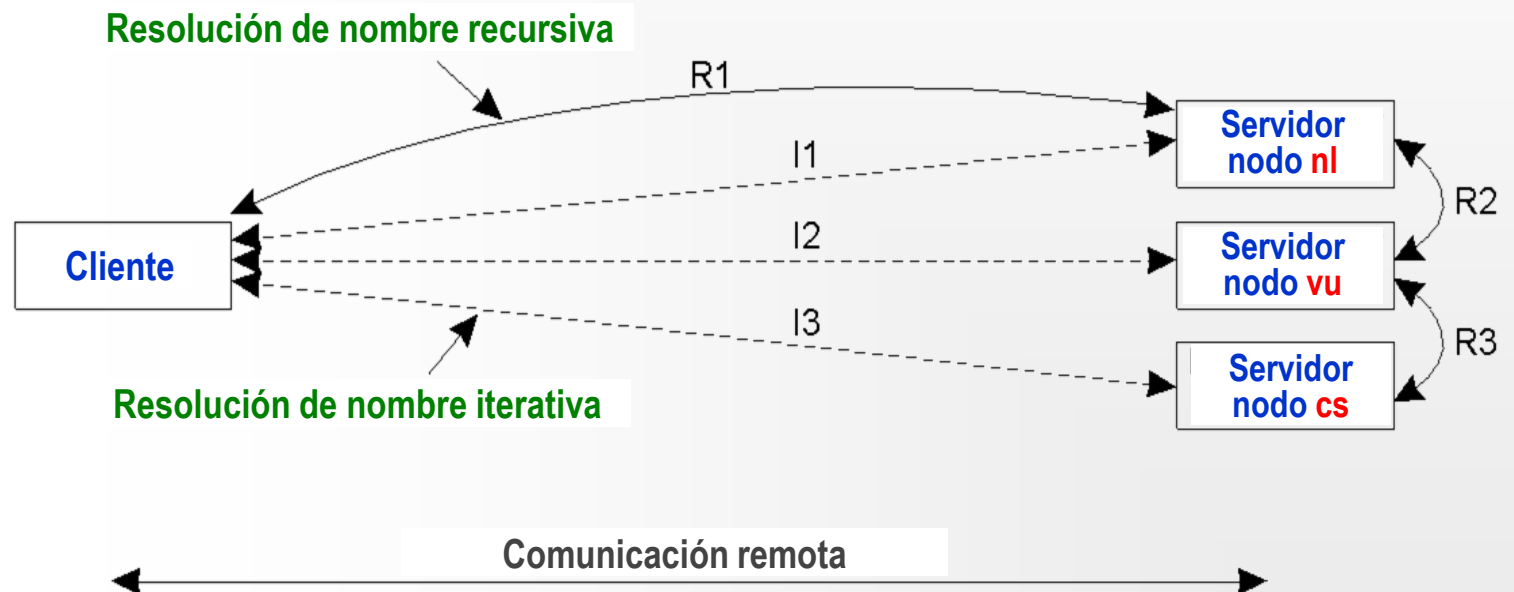
# Implementación Resolución de Nombres

- Ejemplo: Resolución recursiva de nombres.



# Implementación Resolución de Nombres

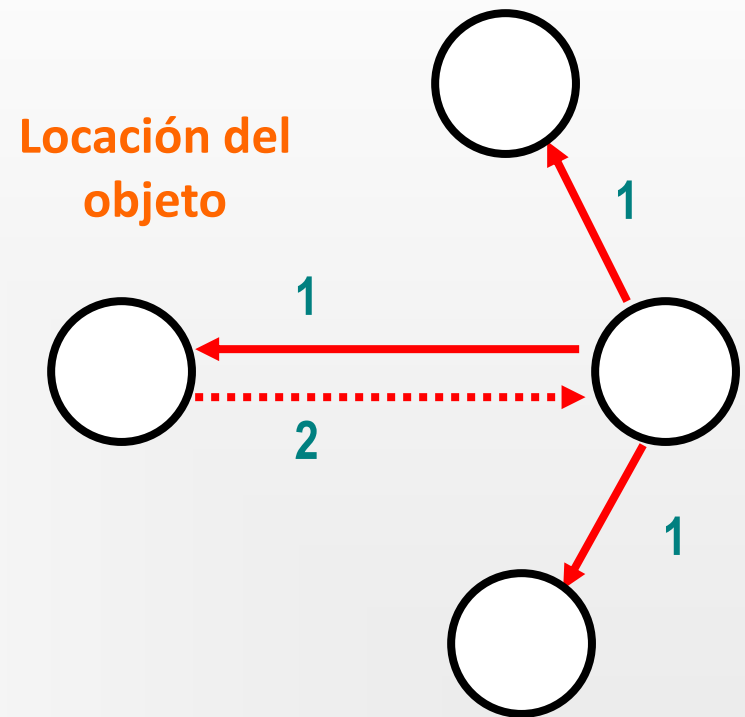
- Comparación entre resolución de nombres recursiva e iterativa con respecto al costo de comunicación.



# Localización de Nombres - Mecanismos

## Broadcast

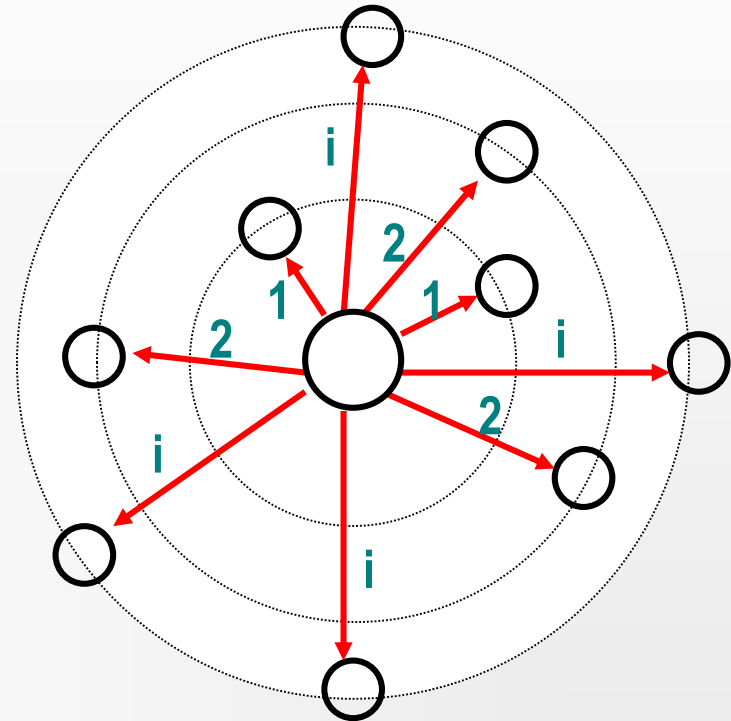
- 1 mensaje broadcast de requerimiento
- 2 respuesta del nodo en el que se encuentra el objeto



# Localización de Nombres - Mecanismos

## Broadcast en expansión de anillos

- 1 búsqueda en nodos a hop=1
- 2 búsqueda en nodos a hop=2 si la búsqueda a hop=1 falla
- i búsqueda en nodos a hop=i si la búsqueda a hop=i-1 falla

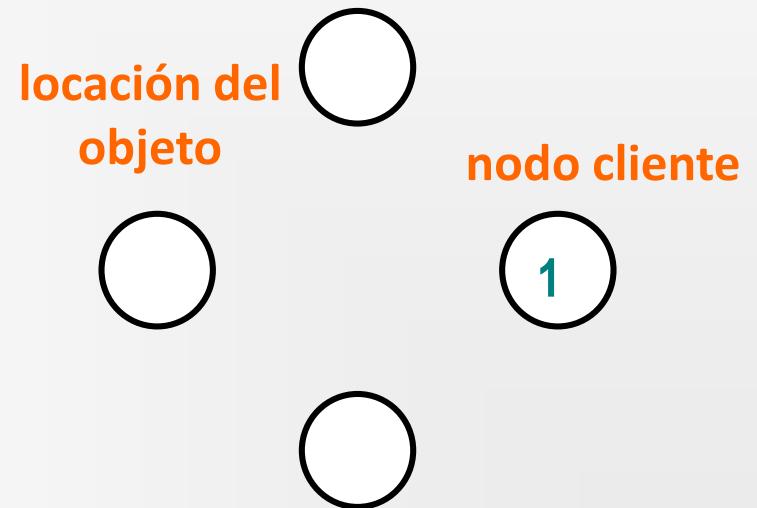


# Localización de Nombres - Mecanismos

## Codificar la locación en el UID

- Esconde información del nodo
- No se puede mover

1. extrae la locación del objeto de su UID. No se requiere intercambio de mensajes con ningún otro nodo

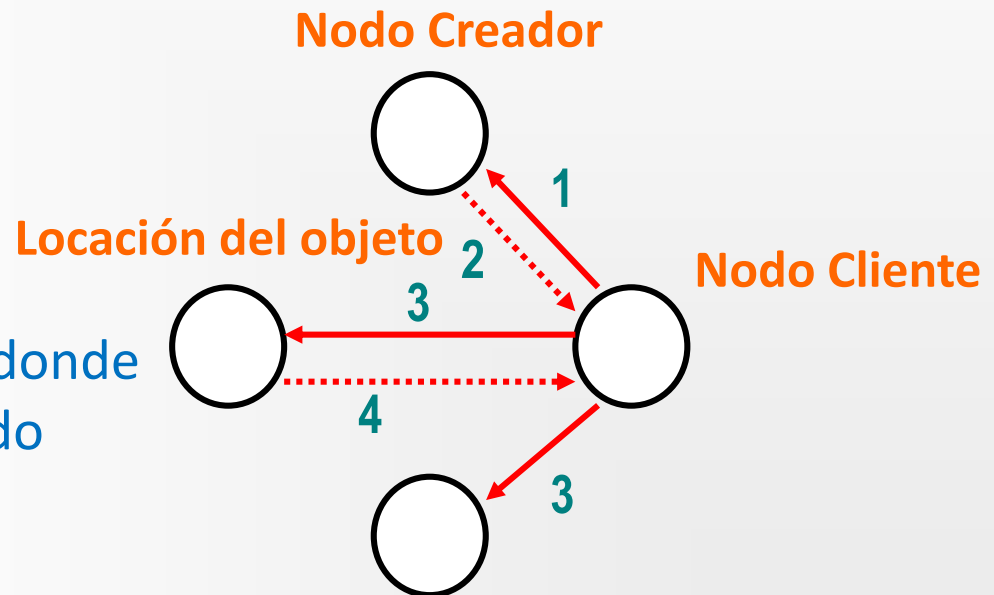


# Localización de Nombres - Mecanismos

## Buscar el nodo creador primero y luego broadcast

- Limita la escalabilidad pero es más flexible que el anterior

1. búsqueda del nodo creador
2. respuesta negativa
3. broadcast de requerimiento
4. respuesta del nodo donde el objeto está ubicado





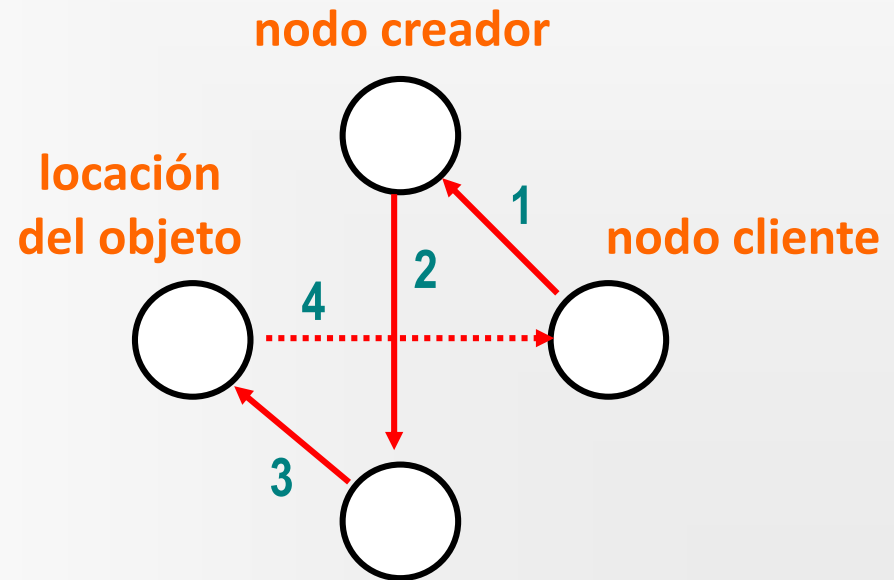
# Localización de Nombres - Mecanismos

## Uso de punteros de seguimiento

- El costo es directamente proporcional al largo de la cadena.
- Es difícil recuperar un id si se cayó un nodo intermedio. La confiabilidad es pobre.
- Hay sobrecarga extra.

1, 2 y 3 camino del mensaje

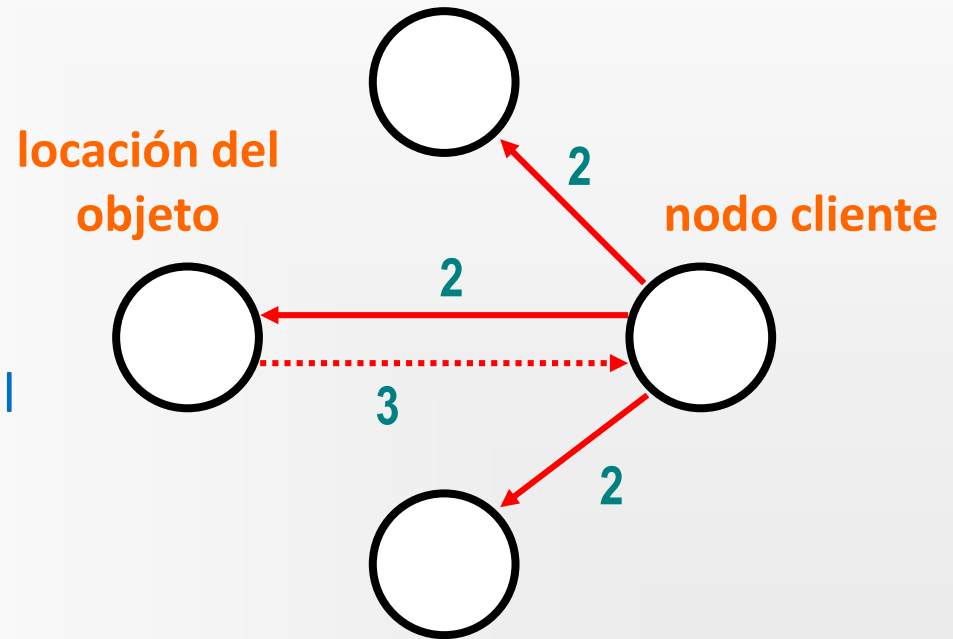
4 respuesta del nodo en donde está localizado el objeto



# Localización de Nombres

## Uso de caché y broadcast.

1. búsqueda en el caché local
2. broadcast de requerimiento
3. respuesta del nodo en donde está localizado el objeto





# **Sistemas de Archivos Distribuidos**



# Agenda

1. Objetivos
2. Servicios
3. Características
4. Modelos de archivos
5. Modos de acceso
6. Semánticas de consistencia
7. Esquema de caché y actualizaciones
8. Replicación
9. Tolerancia a fallas
  1. Servidores con estado y sin estado
10. Arquitecturas
  1. Cliente-servidor, en cluster, simétrico
11. Ejemplos: GFS y NFS



# Sistema de Archivos Distribuidos

Propósitos para el uso de archivos:

- Almacenaje de Información permanente
- Información compartida

También soporta:

- Compartir información remota
- Usuarios móviles
- Disponibilidad (réplicas)
- Estaciones de trabajo sin disco



# Sistema de Archivos Distribuidos

## SERVICIOS

- **Servicio de almacenaje.**
  - ✓ alocación y manejo del espacio
  - ✓ servicio de disco
  - ✓ servicio de “bloqueo”
- **Servicio de nombres.**
  - ✓ mapeo entre nombres externos e internos
- **Servicio de archivos.**
  - ✓ acceso
  - ✓ semántica de archivos compartidos
  - ✓ caching
  - ✓ replicación
  - ✓ control de concurrencia



# Sistema de Archivos Distribuidos

## CARACTERÍSTICAS DESEABLES DE LOS SAD

### TRANSPARENCIA

- de estructura: no se conoce el número de servidores ni sus lugares
- de acceso
- de nombres
- de ubicación
- de replicación
- de rendimiento (performance)



# Sistema de Archivos Distribuidos

## CARACTERÍSTICAS DESEABLES DE LOS SAD

- Movilidad de usuario
- Rendimiento
- Simplicidad y facilidad de uso
- Escalabilidad
- Alta disponibilidad
- Alta confiabilidad
- Integridad de datos
- Seguridad
- Heterogeneidad





# Sistema de Archivos Distribuidos

## Modelos de Archivos

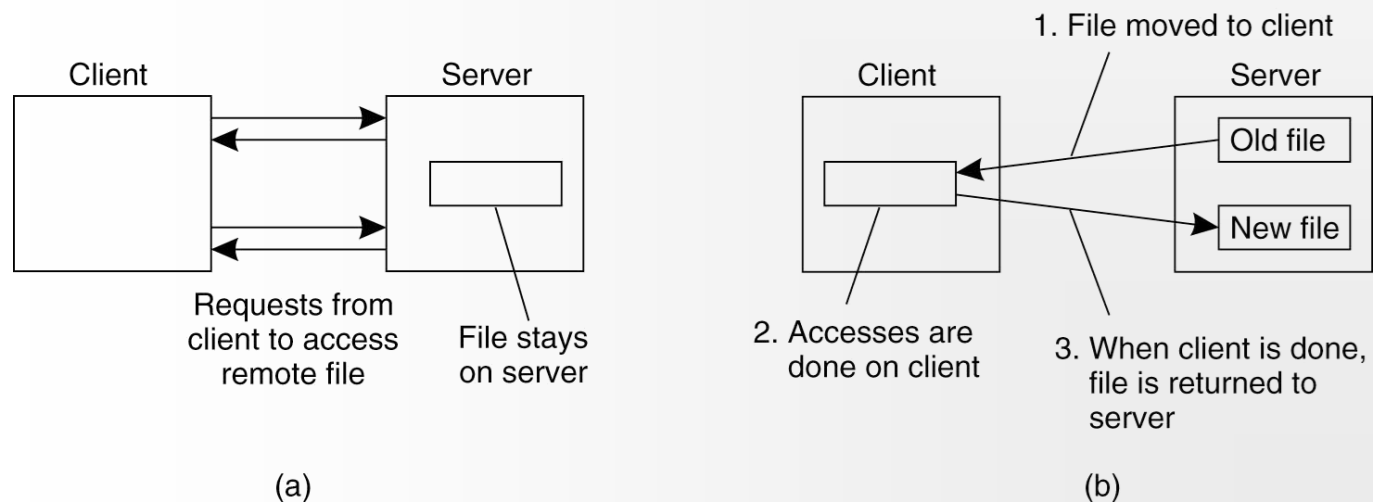
Los archivos pueden ser:

- **estructurados:** son raros hoy, son una secuencia de registros (indexados o no indexados)
- **no estructurados:** UNIX.
- **mutables**
- **inmutables**

# Sistema de Archivos Distribuidos

## Modelos de Acceso a Archivos

- Modelo de servicio remoto.
- Modelo de captura de datos (caching): trae consigo problemas de consistencia.





# Sistema de Archivos Distribuidos

## UNIDADES DE TRANSFERENCIA DE DATOS

- Modelo de transferencia a nivel de archivos.
- Modelo de transferencia bloques.
- Modelo de transferencia bytes.
- Modelo de transferencia por registros.

## SEMÁNTICA DE ARCHIVOS COMPARTIDOS

- Semántica UNIX
- Semántica de sesión
- Semántica de archivos inmutables
- Semántica de transacciones



# Sistema de Archivos Distribuidos

## Esquemas de caché

En centralizados:

- granularidad
- tamaño
- políticas de reemplazo

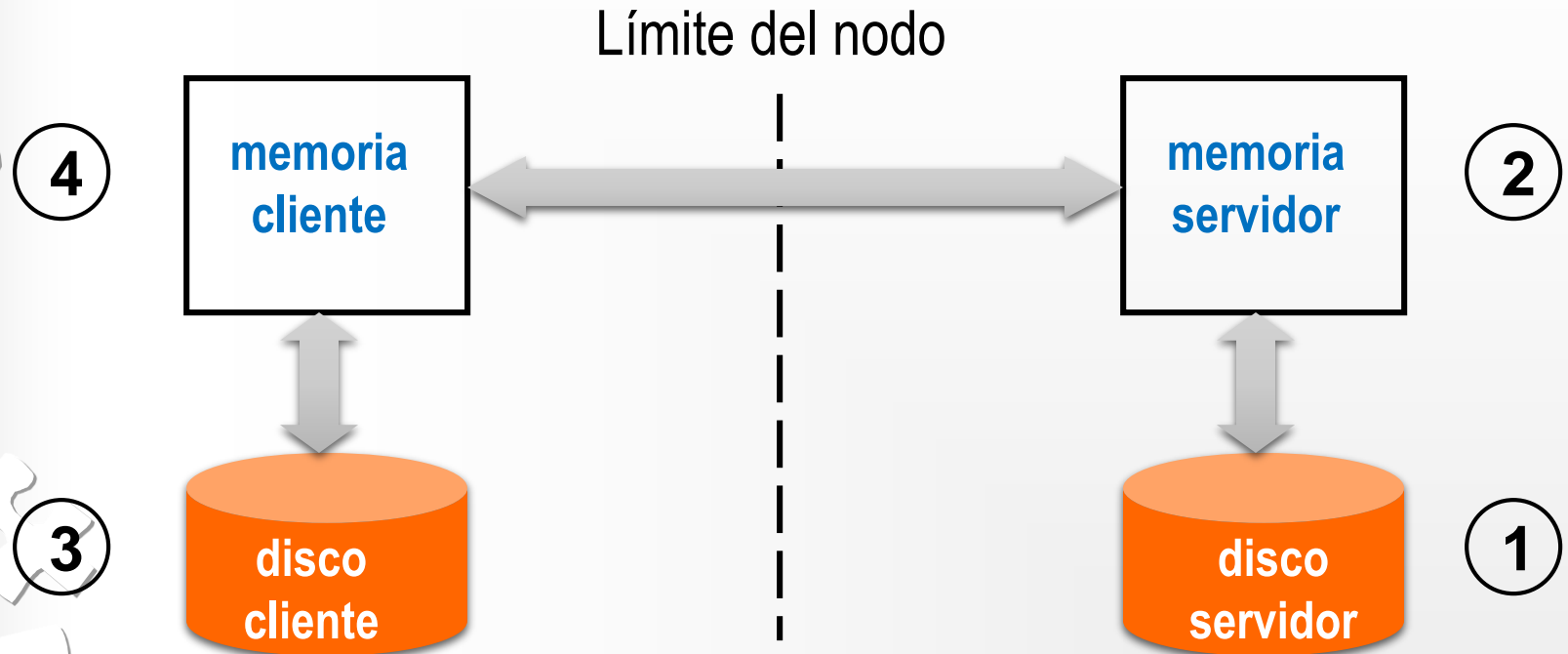
En distribuidos se agrega:

- locación
- propagación de la modificación
- validación

# Sistema de Archivos Distribuidos

Locación caché	Costo de acceso con éxito en la caché	Ventajas
Memoria del servidor	Un acceso por la red	<ul style="list-style-type: none"><li>▪Fácil de implementar</li><li>▪Totalmente transparente a los clientes</li><li>▪Fácil de mantener consistente el archivo original y los datos en el caché</li><li>▪Fácil para soportar semántica UNIX</li></ul>
Disco del cliente	Un acceso a disco	<ul style="list-style-type: none"><li>▪Confiabilidad en caso de “crash”</li><li>▪Gran capacidad de almacenaje</li><li>▪Adecuado para soportar operación sin conexión</li><li>▪Contribuye a la escalabilidad y confiabilidad</li></ul>
Memoria del cliente	-----	<ul style="list-style-type: none"><li>▪Máxima ganancia de rendimiento</li><li>▪Permite estaciones de trabajo sin disco</li><li>▪Contribuye a la escalabilidad y confiabilidad</li></ul>

# Sistema de Archivos Distribuidos



1. Sin caché
2. Caché localizado en la memoria del servidor
3. Caché localizado en el disco del cliente
4. Caché localizado en la memoria del cliente

ubicación  
original del  
archivo



# Sistema de Archivos Distribuidos

## PROPAGACIÓN DE LA MODIFICACIÓN

- Escritura inmediata (write through)
- Escritura diferida (delayed write)
  - escritura cuando se “echa” la información del caché
  - escritura periódica
  - escritura cuando se cierra

## VALIDACIÓN DE CACHÉS

- iniciado por el cliente
  - verificación antes de cada acceso
  - verificación periódica
  - verificación sólo cuando el archivo es abierto para el uso
- iniciado por el servidor



# Sistema de Archivos Distribuidos

## REPLICACIÓN – VENTAJAS

- se incrementa la disponibilidad
- se incrementa la confiabilidad
- mejora el tiempo de respuesta
- reduce el tráfico en la red
- mejora el procesamiento total
- buena escalabilidad
- operación autónoma





# Sistema de Archivos Distribuidos

## TRANSPARENCIA DE REPLICACIÓN

- Replicación explícita
- Replicación implícita/relajada

**Problema de actualización de múltiples copias**

Está relacionado con mantener consistentes las copias.



# Sistema de Archivos Distribuidos

## TOLERANCIA A LAS FALLAS

Hay propiedades que influyen directamente en un SAD para que sea tolerante a las fallas:

- Disponibilidad  $\Rightarrow$  replicación
- Robustez
- Almacenamiento estable:
  - volátil
  - discos
  - estable



# Sistema de Archivos Distribuidos

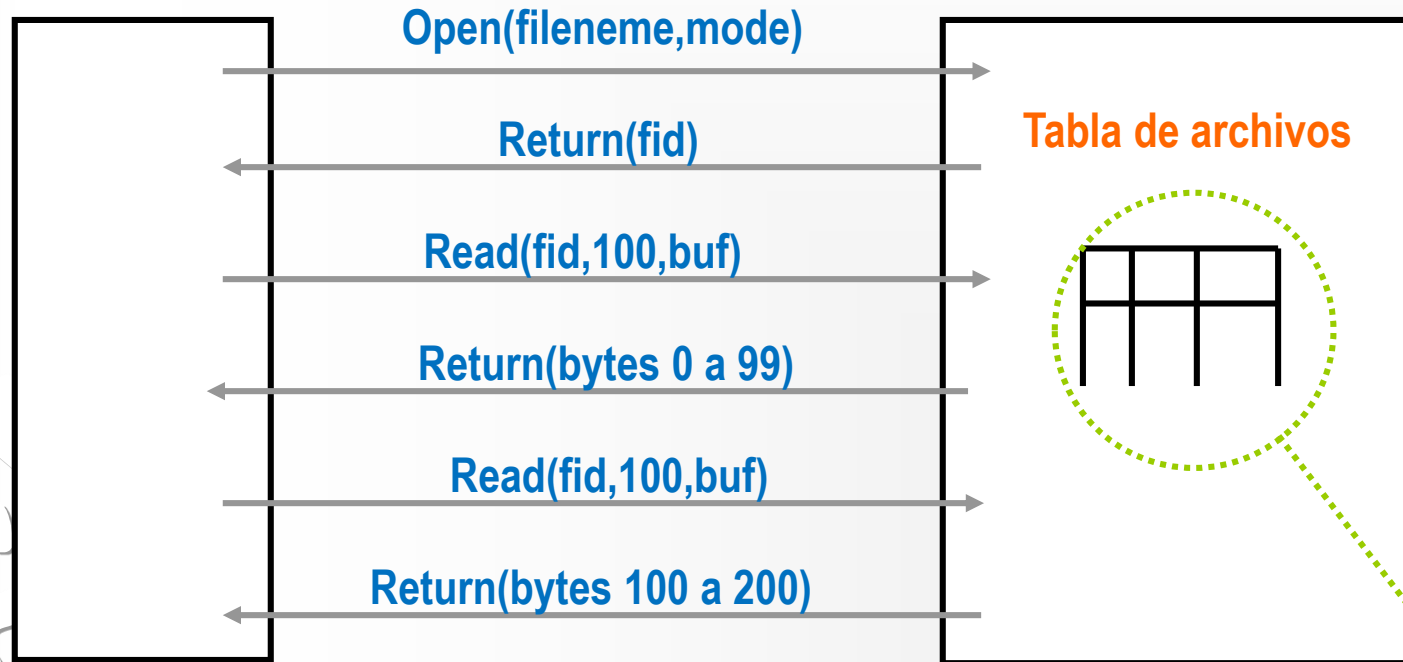
Efecto del paradigma de servicio en la tolerancia a las fallas

- Servidores **con estado**
- Servidores **sin estado**

# Sistema de Archivos Distribuidos – Servidor con estado

Proceso Cliente

Proceso Servidor



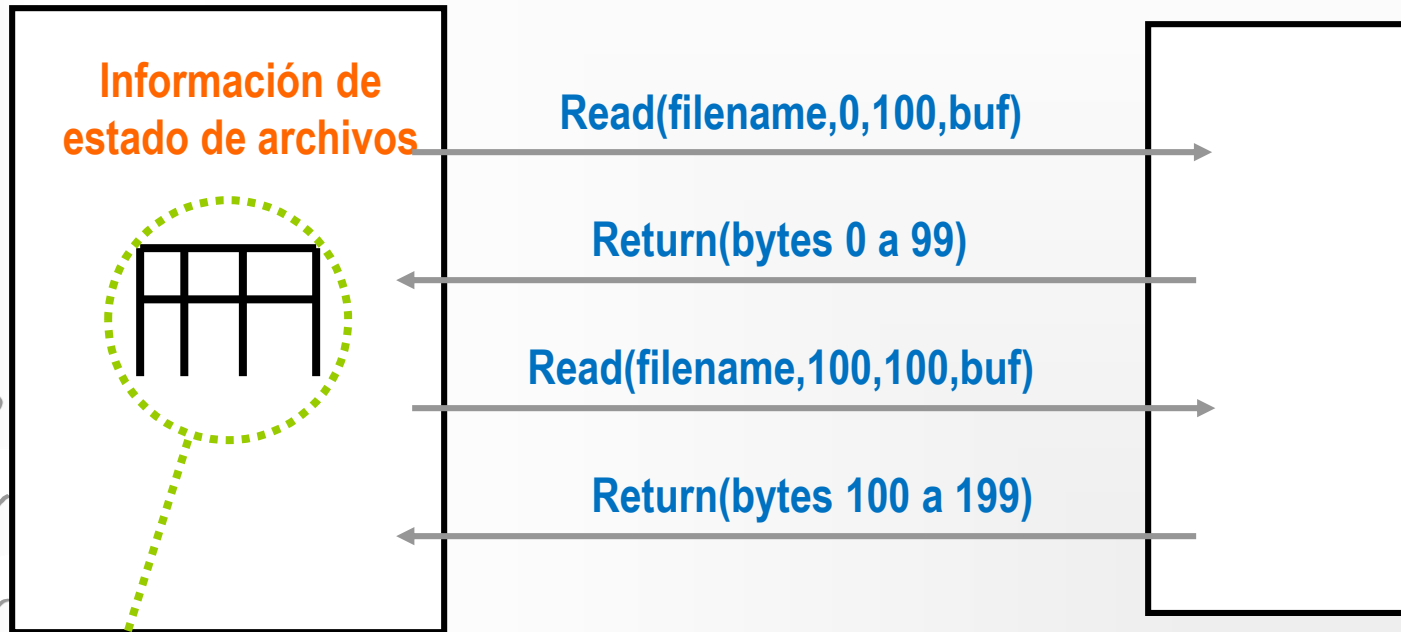
Dos subsecuentes lecturas de 100 bytes (200 bytes en total)

fid	Modo	Puntero R/W

# Sistema de Archivos Distribuidos – Servidor sin estado

Proceso Cliente

Proceso Servidor



Nombre de archivo	Modo	Puntero R/W

Dos subsecuentes lecturas de 100 bytes (200 bytes en total)



# Sistema de Archivos Distribuidos

Diferencias entre Servicios **con estado** y **sin estado**

## Recuperación de Fallas

- Un servidor con estados pierde, en un *crash*, todo su estado volátil.
  - Restaure el estado por un protocolo basado en un diálogo con clientes o aborta las operaciones que se estaban llevando a cabo cuando el crash ocurrió.
  - El servidor necesita estar al tanto de las fallas en los clientes para reclamar el espacio reservado para registrar el estado de los procesos de los clientes caídos.



# Sistema de Archivos Distribuidos

- Con un servidor sin estados, los efectos de fallas y recuperación en el servidor no son notables. Un nuevo servidor *encarnado* puede responder sin dificultad a un requerimiento autocontenido.

## Penalización por usar un servicio robusto sin estados:

- Mensajes de requerimiento más grandes
- Menor procesamiento de requerimientos
- Restricciones adicionales al diseño de un SAD

# Sistema de Archivos Distribuidos – Arquitectura Servicio de Archivo

## Componentes:

- Módulo cliente
- Servicio directorio
- Servicio de archivo plano

Máquina Cliente



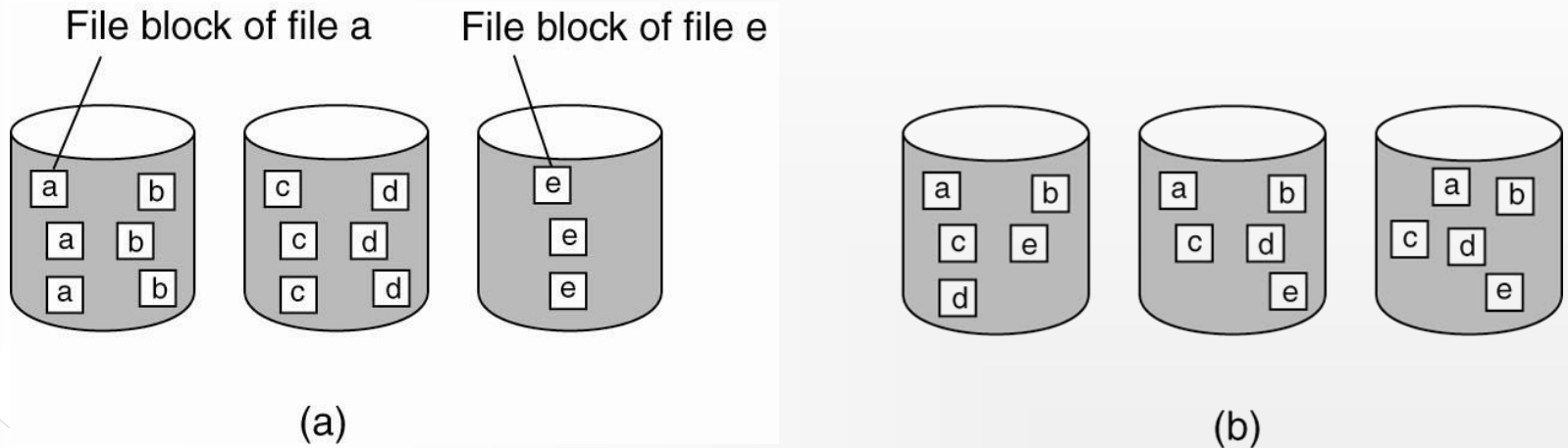
Máquina Servidor





# Sistema de Archivos Distribuidos

- **Arquitectura basadas en Cluster**

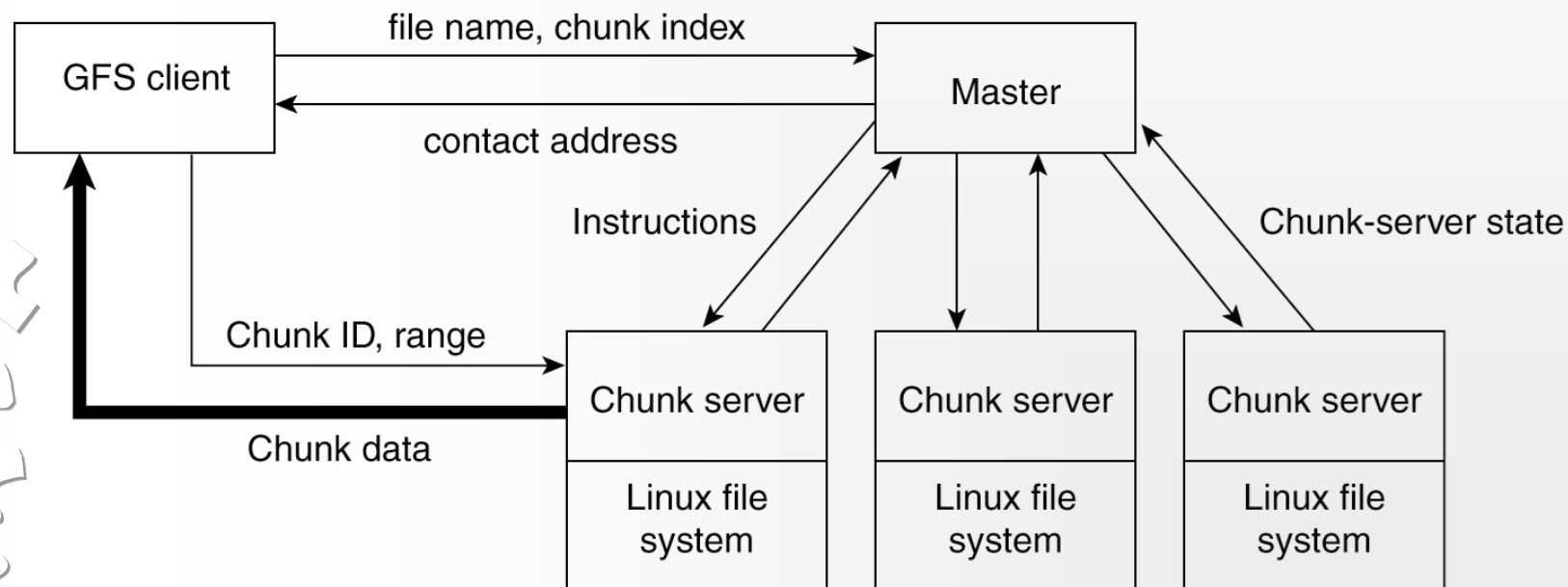


a) Distribución de archivos completos a través de varios servidores

b) Fragmentación de archivos para acceso paralelo

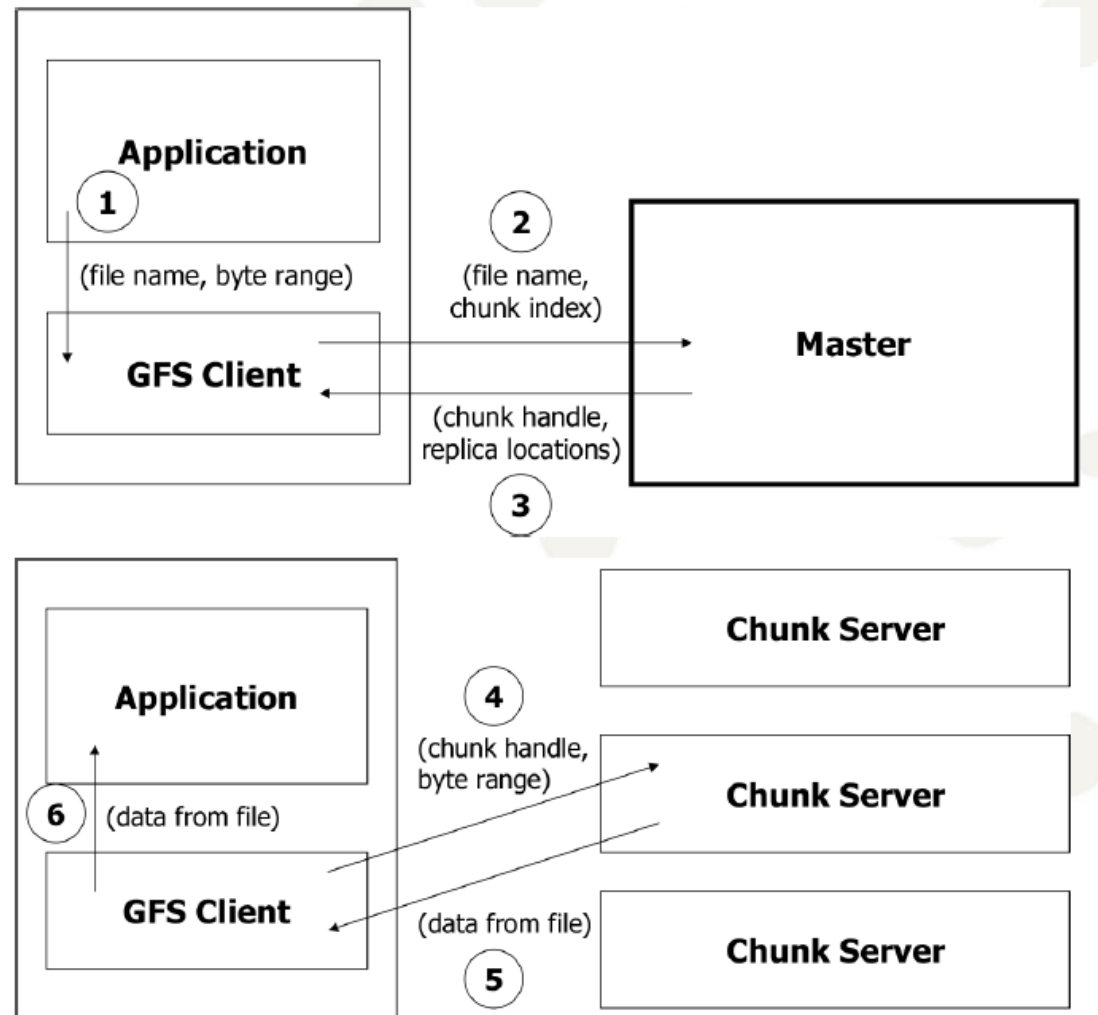
# Sistema de Archivos Distribuidos - GFS

- Arquitectura basadas en Cluster - Sistema de Archivos de Google (GFS)



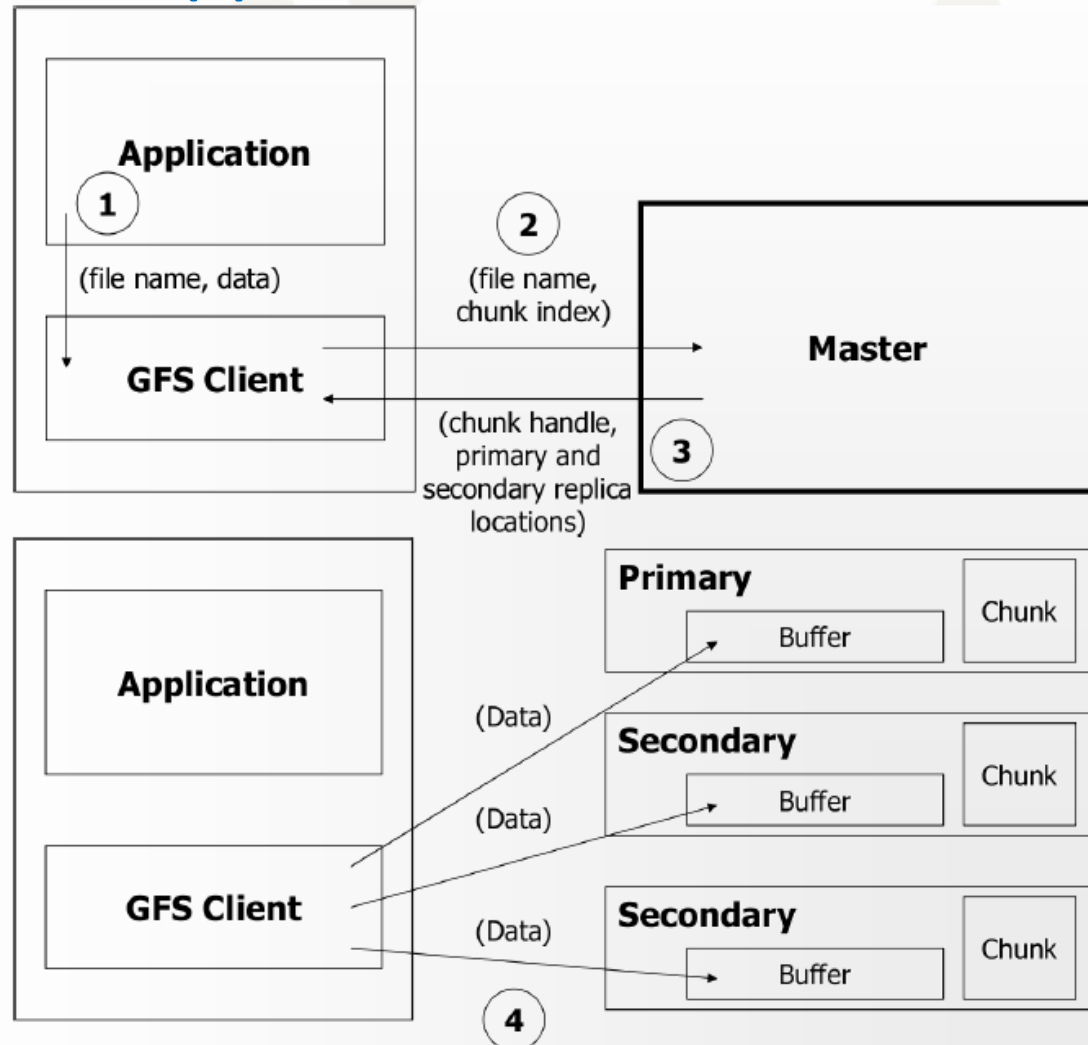
# Sistema de Archivos Distribuidos - GFS

## Operación Lectura



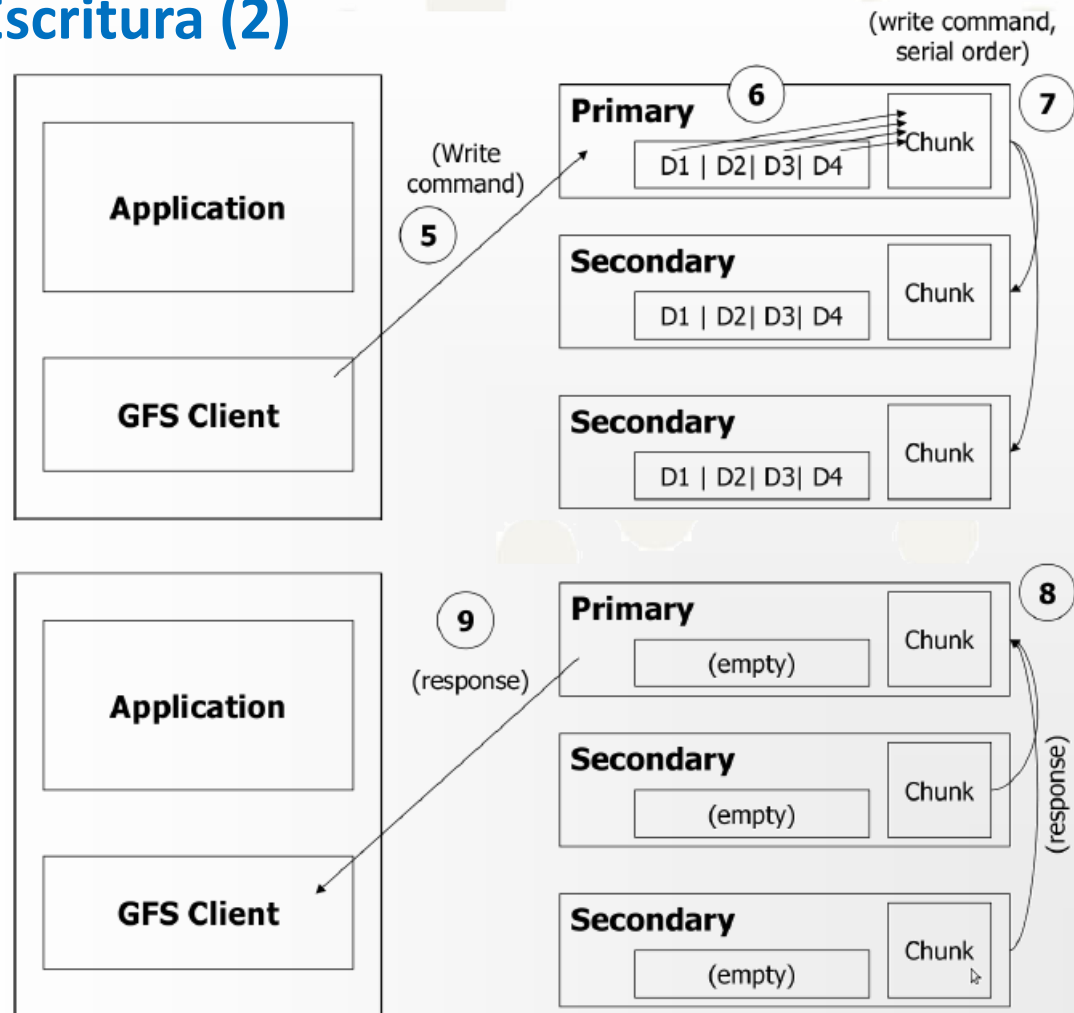
# Sistema de Archivos Distribuidos - GFS

## Operación Escritura (1)



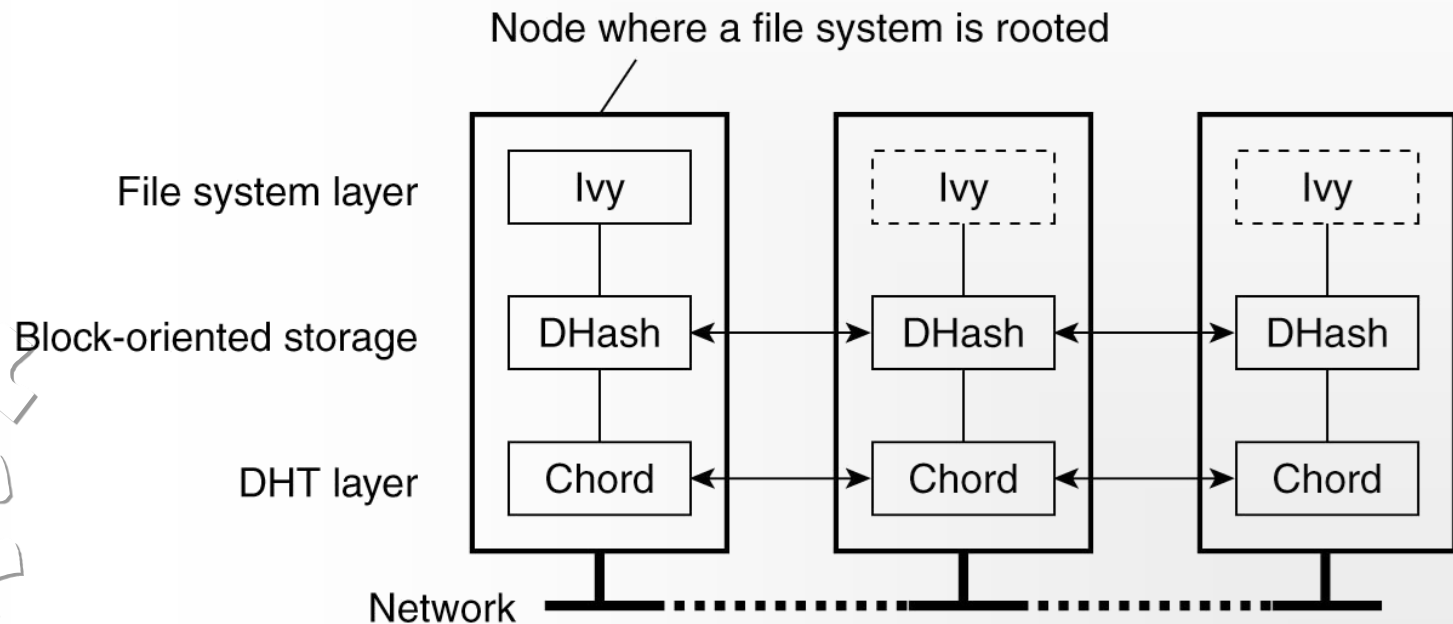
# Sistema de Archivos Distribuidos - GFS

## Operación Escritura (2)



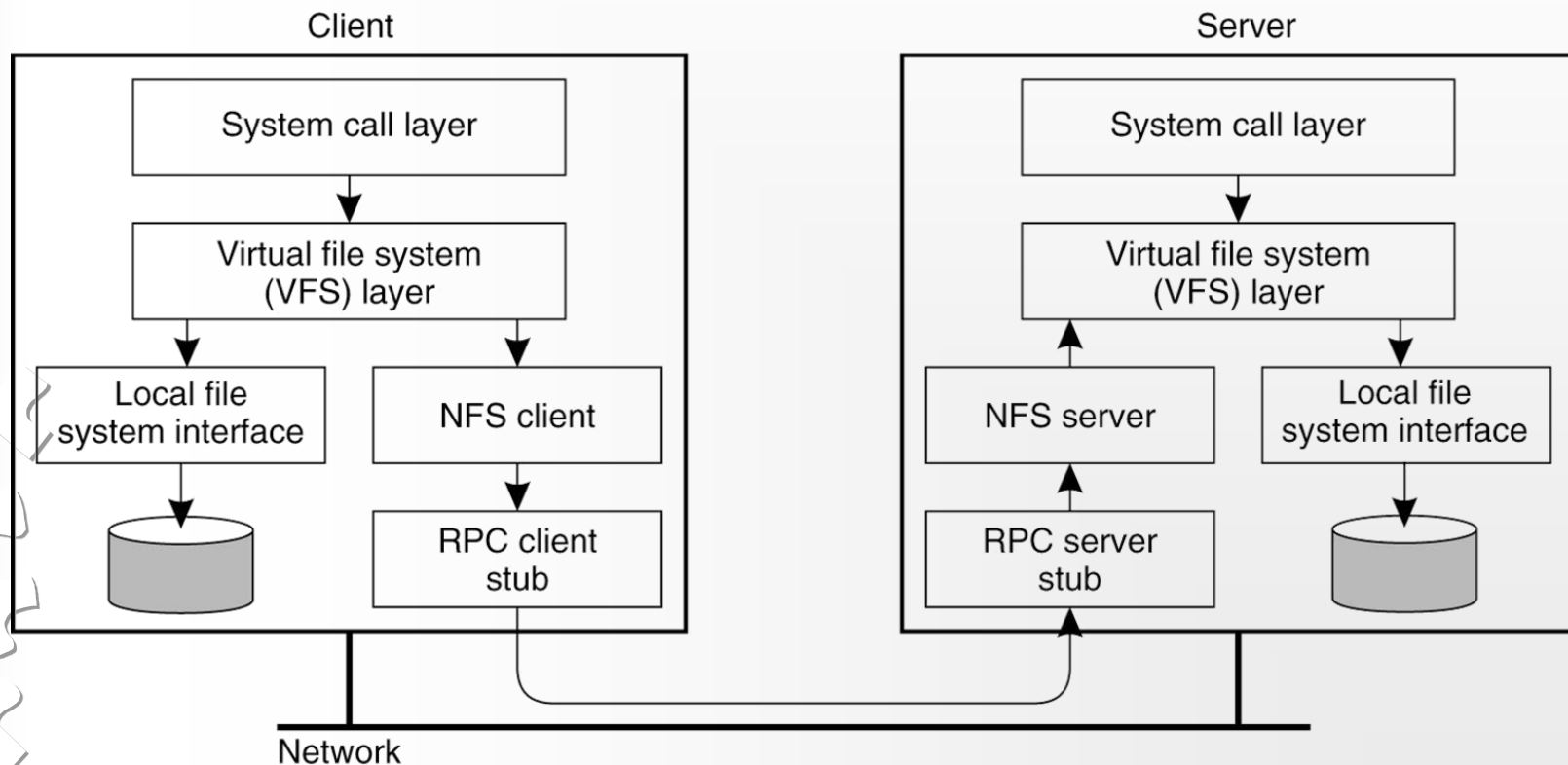
# Sistema de Archivos Distribuidos

## Arquitecturas simétricas - Sistema de archivo Distribuido Ivy



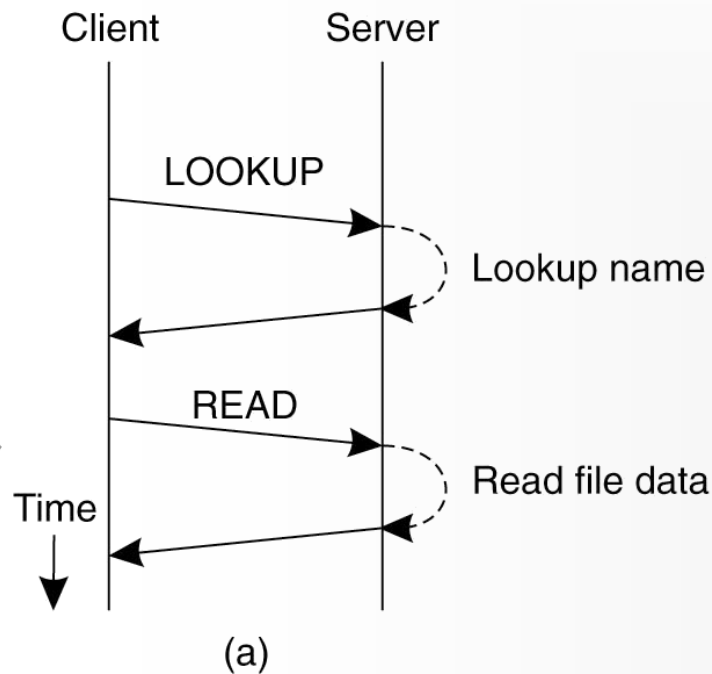
# Sistema de Archivos Distribuidos - NFS

## Arquitectura Cliente - Servidor - NFS

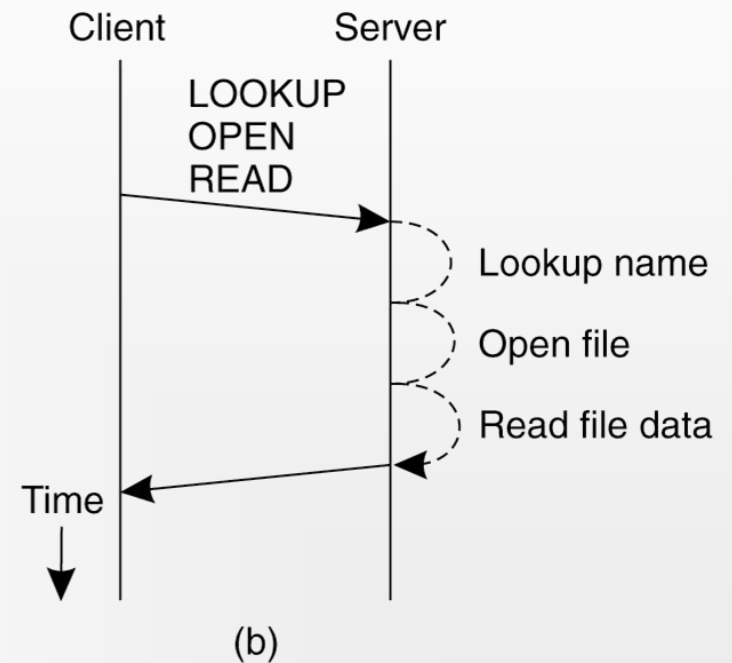


# Sistema de Archivos Distribuidos - NFS

## Comunicación en NFS



a) Lectura de datos en NFS vs3



b) Lectura de datos en NFS vs4





## Bibliografía:

- Sinha, P. K.; “Distributed Operating Systems: Concepts and Design”, IEEE Press, 1997.
- Tanenbaum, A.S.; van Steen, Maarten; “Distributed Systems: Principles and Paradigms”. 2<sup>nd</sup> Edition, Prentice Hall, 2007 and 1<sup>st</sup> Edition 2002.
- van Steen, Maarten; Tanenbaum, A.S.; “Distributed Systems”. 3<sup>rd</sup> Edition, Prentice Hall, 2017.
- Coulouris, G.F.; Dollimore, J. y T. Kindberg; “Distributed Systems: Concepts and Design”. 5th Edition Addison Wesley, 2011.
- Naushad UzZaman, Survey on Google File System, CSC 456 (Operating Systems), 2007