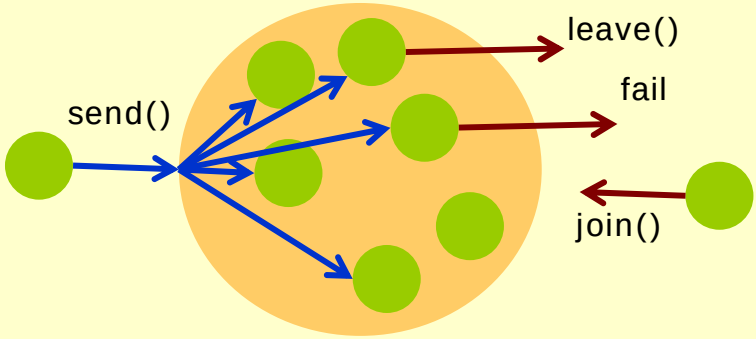
IPC en Grupo

Índice

- Grupos, tipología, tipos de comunicación y utilidad
- Soporte de los modos de comunicación
  - Multidifusión IP
- Multicast con/sin búfer (en el receptor)
- Semánticas de envío
- Tipos de multicast
  - Atomicidad de multidifusión
  - Fiabilidad de la multidifusión
  - Según el ordenamiento
    - Relojes lógicos de Lamport

Presentación

- La comunicación en grupo permite comunicarse con un conjunto de procesos en abstracto.
  - La composición del grupo es dinámica



Tipos de comunicación

- Unicast (unidifusión)
  - 1 ↔ 1 punto a punto
- Anycast
  - 1 → el primero que lo reciba (aparece en IPv6)
- Netcast
  - 1→muchos, pero uno cada vez
- Multicast (multidifusión)
  - 1→ muchos, soportado por el nivel de red
- Broadcast (difusión)
  - 1→ todos, abarcando toda la red

Tipología de grupo

- ¿Quién puede enviar un mensaje?
  - Grupos cerrados: solo los miembros aplicaciones paralelas, ...
  - Grupos abiertos: cualquiera servidores redundantes, ...
- ¿Cómo se toman las decisiones del grupo?
  - Grupos pares: entre todos (o la mayoría) precisa de algoritmos de elección
  - Grupos jerárquicos: por un coordinador es asimétrico y centralizado

Tipología de grupos

- Existe una función de pertenencia al grupo (*membership function*)
- ¿Quién decide quién pertenece al grupo?
  - Pertenencia centralizada: el responsable del grupo decide, y el resto no tiene porqué intervenir.
  - Pertenencia descentralizada: el grupo recibe y vota las solicitudes.
- Los abandonos del grupo por caída, máxime en el segundo caso, plantean problemas.

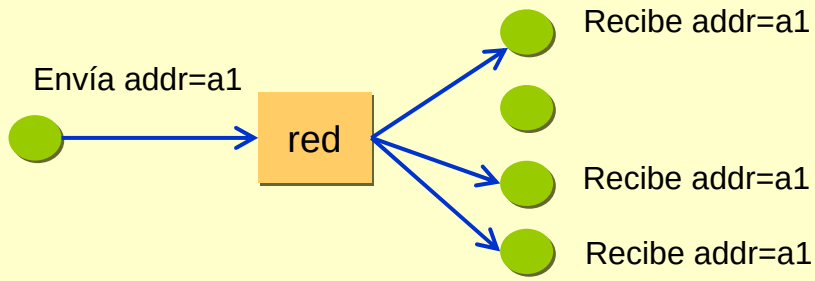
Comunicación en grupo – utilidad

- Tolerancia a fallos: para servidores replicados
- Ubicación de objetos en servicios distribuidos
- Aumento de prestaciones: mediante replicas de objetos
- Actualizaciones múltiples.

Soporte de los modos de comunicación

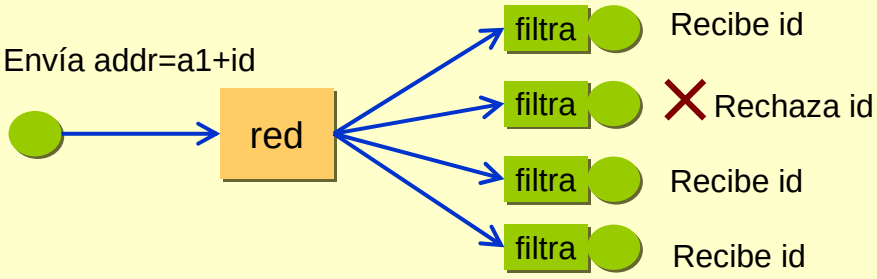
- Hardware:
  - Unicast
  - Multicast
  - Broadcast
  - Anycast, en IPv6
- Software
  - Multicast: típicamente “netcast”

Multicast hardware



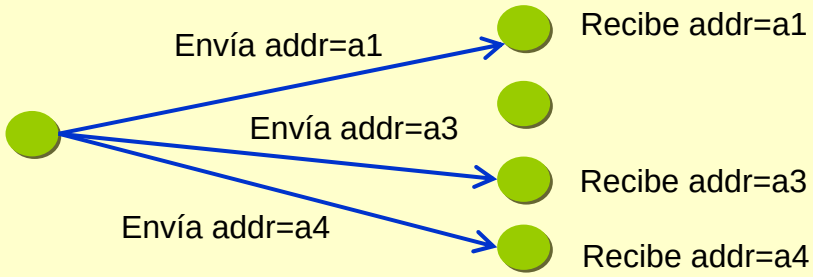
- Los miembros del grupo escuchan una dirección de red.
  - Plantea problemas a los routers, para distribuir mensajes de multicast. Está resuelto en los "buenos"
  - Evita tener que conocer la "*función de pertenencia*"

Multicast por broadcast hardware



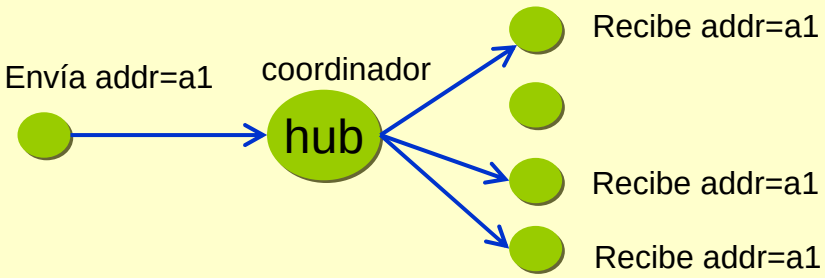
- La noción de grupo descansa fuertemente en la noción de red para el broadcast.
  - Gasta tiempo de cada host de la red en el filtrado.
  - Requiere cierta programación de drivers de dispositivos

Multicast por netcast (software)



- El emisor debe ser consciente de la “función de pertenencia”, y tener una tabla con las relaciones de direcciones de red.
  - La función de pertenencia es difícil de gestionar.

Multicast software vía hub



- Un hub (concentrador) realiza las funciones de envío (conoce la función de pertenencia).
- Sobre dicho hub, suele, además recaer ciertas funciones de coordinación “imprescindibles”

Multidifusión IP

1110	ID multicast 28 bits

Dirección de clase D

- Multicast dinámico (los routers añaden bajo demanda otros routers a su tabla de rutado) IGMP (Internet Group Management Protocol), RFC-1112.
- Las tarjetas LAN lo soportan:
  - Por filtrado hash
  - Por tablas multicast (limitado)
- Para más, véase (Kurose & Ross, “Redes de Computadoras”).

Multicast con/sin búfer (en receptor)

- El multicast es suele ser asíncrono:
  - El emisor puede no esperar a todos los receptores.
  - El emisor puede no conocer a todos los receptores.
- Con el multicast sin búfer, se pierde si el receptor no está a la escucha.
- Con el multicast con búfer, cualquier proceso receptor puede recibir el mensaje asíncronamente.

Semánticas de envío

- Send to all: una copia a cada proceso, con búfer.
- Bulletin board: El modelo de canal retiene una copia (post); y el receptor la toma cuando puede.
  - Se pueden recuperar mensajes según su relevancia y disponibilidad.
  - Se puede fijar un tiempo de caducidad en cada mensaje según la necesidad del emisor.
  - Ejemplo: floating pool processor.

Tipos de multicast

- Según la atomicidad
- Según la fiabilidad que necesitemos:
  - Fiable
  - No fiable
- Según el ordenamiento temporal
  - Global
  - Total
  - Causal
  - Sincronizado
  - No ordenado

Atomicidad de multidifusión

- El mensaje debe ser recibido por todos, para que ocurra. (Es muy costoso)
- Alternativas en caso de fallo:
  - Se invalida la multidifusión
  - Se excluye del grupo al elemento fallido.
- Los fallos se pueden producir en cualquier punto, y momento (incluso en el emisor)
  - El emisor necesita el ACK de los receptores.
  - Los receptores necesitan el ACK del COMMIT.

Atomicidad de multidifusión

- Suele darse mediante dos fases:
  1. Reparto garantizado del mensaje a **todos**
     1. Se reparte el mensaje y se esperan los ACK
     2. Se repite para los procesos sin ACK, cierto número de veces y con tiempos límite de espera.
        - Si no todos responden, se decide si excluirlos, o
        - Invalidar el multicast (según el efecto, deseado)
  2. Completado de la operación (COMMIT).
- Se guardan copias de las acciones en logs históricos, por posibles interrupciones.

N-Fiabilidad

0-Fiable: no se espera respuesta (Multicast asíncrono) (p.ej.: reloj)

1-Fiable: Se espera 1 respuesta (Multicast asíncrono, o anycast) (p.ej.: un cálculo)

(m entre n)-Fiable: m respuestas entre n miembros (para las votaciones) (p.ej.: fiabilidad y consistencia de réplicas)

Totalmente-Fiable: se esperan todas las respuestas (p.ej.: actualización de una base de datos replicada)

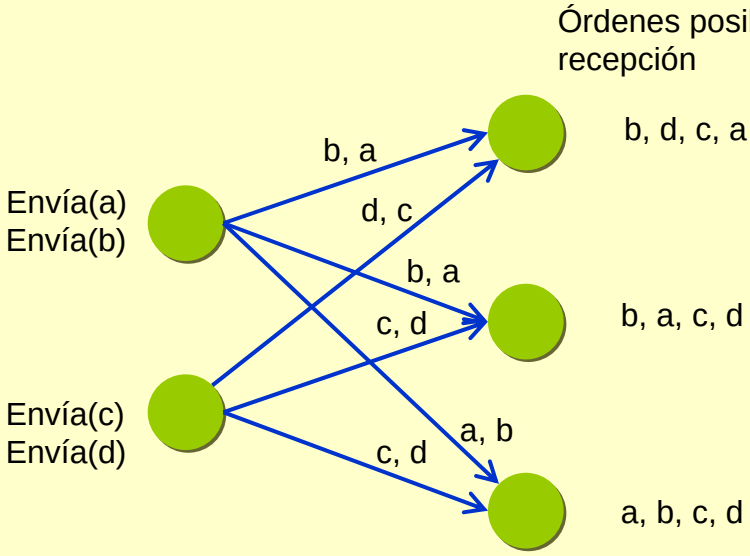
Multicast No fiable

- Se envía en las mejores condiciones que proporciona el multicast nativo, y
- Se espera que llegue a los receptores.

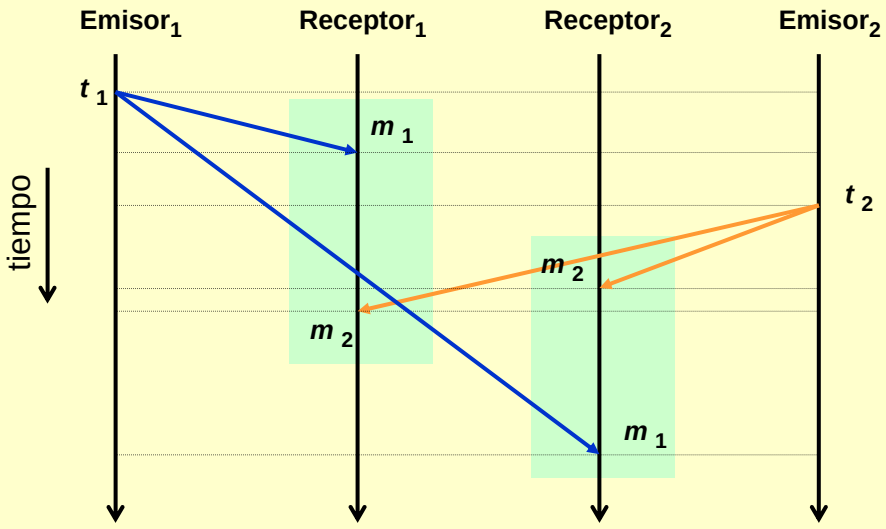
Multicast fiable

- Se envía el mensaje y se espera un cierto tiempo (t-corto) para recibir ACK de cada miembro.
- Se reintenta si algún miembro no envía ACK expirado el t-corto
  - Durante cierto tiempo (t-largo)
  - Cierto número de veces razonable.
- Otra posibilidad es usar números de secuencia y ACK negativos.

El desordenamiento es posible



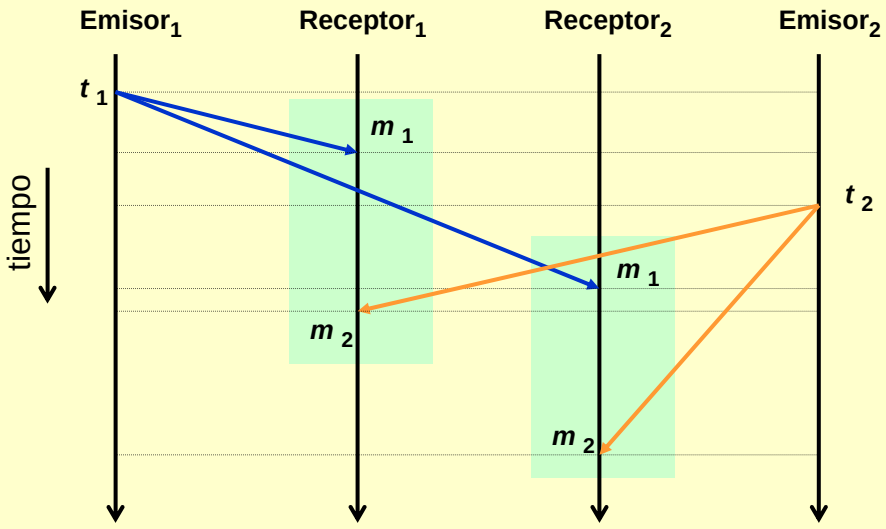
El desordenamiento es posible



Alternativas al desordenamiento

- Ordenamiento global, o absoluto
  por orden de envío
- Ordenamiento total, o consistente
  el mismo desorden en cada receptor
- Ordenamiento causal
  por ordenes parciales de subsecuencias
  de mensajes

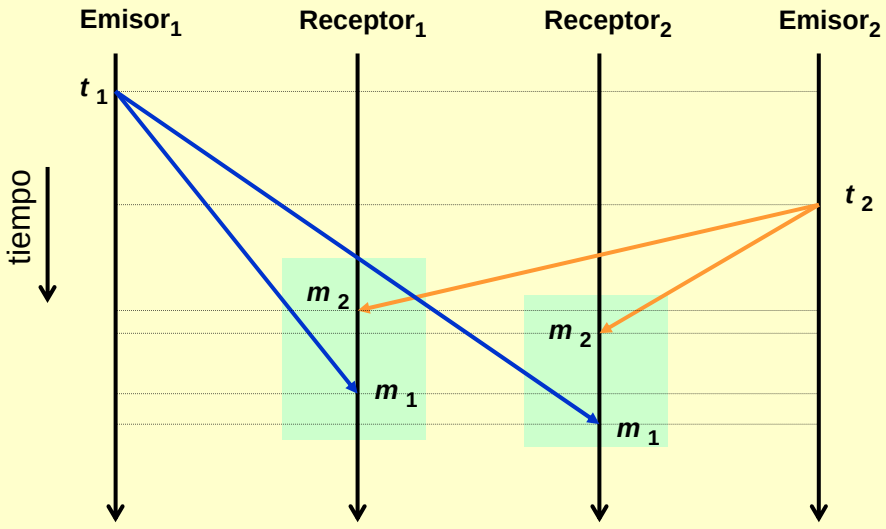
Ordenamiento global (absoluto)



Ordenamiento global (absoluto)

- Todos los mensajes llegan en el mismo orden en que fueron enviados. (Difícil-fícil)
- Forma:
  - Los relojes están sincronizados y se añaden timestamps a los mensajes.
  - Los mensajes se reciben en búfer de ventana deslizante.
  - Periódicamente se reparten a los receptores los mensajes recibidos con anterioridad a cierto retraso T_d.
  - T_d es ligeramente superior al mayor retraso tolerable.

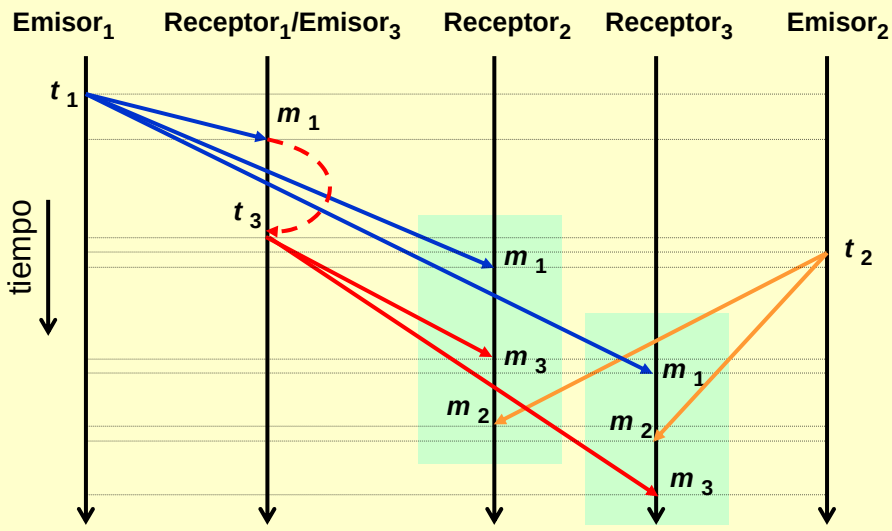
Ordenamiento total (consistente)



Ordenamiento total (consistente)

- Los mensajes se reciben en todos los receptores en el mismo orden.
- Métodos:
  - Mediante un hub secuenciador único y un método de ventana deslizante en cada receptor. (sufre de centralización)
  - Mediante ABCAST (de ISIS):
    - Se asigna un número de secuencia mediante un acuerdo distribuido del emisor con el grupo.

Ordenamiento causal



Ordenamiento causal

- Es un ordenamiento parcial (por grupos de mensajes).
  - Los mensajes relacionados causalmente llevan una marca temporal que permite al receptor decidir el orden relativo.
- Métodos:
  - Relojes lógicos de Lamport para timestamps (marcas temporales)
  - CBCAST (ISIS)

Relojes lógicos de Lamport

- Cada proceso tiene una variable natural (reloj) local, puesta a cero (al principio)
- Cada emisor pone su timestamp (ts) al emitir cada mensaje, e incrementa el reloj en 1 (u otra cantidad fija, aleatoria, ...).
- Cada vez que un proceso recibe un mensaje:
  - Si el ts asociado es mayor que el local, se copia éste valor en el reloj, y se incrementa en 1 (u otra cantidad fija, aleatoria, ...).

Relojes lógicos de Lamport

- Asegura que si se emite un mensaje relacionado, tendra un ts posterior: **ordenamiento parcial**
  → "huída hacia delante" de relojes.
- Sufre el problema de los ts idénticos
  - No ordena los eventos dispares
  - Solución: añadir el PID de cada proceso:
    - Se convierte en un ordenamiento arbitrario
    - Solución: relojes vectoriales.