IIC2523
Sistemas Distribuidos

Hernán F. Valdivieso López
(2025 - 2 / Clase 06)

Vectores Lógicos y Estados Globales

Temas de la clase

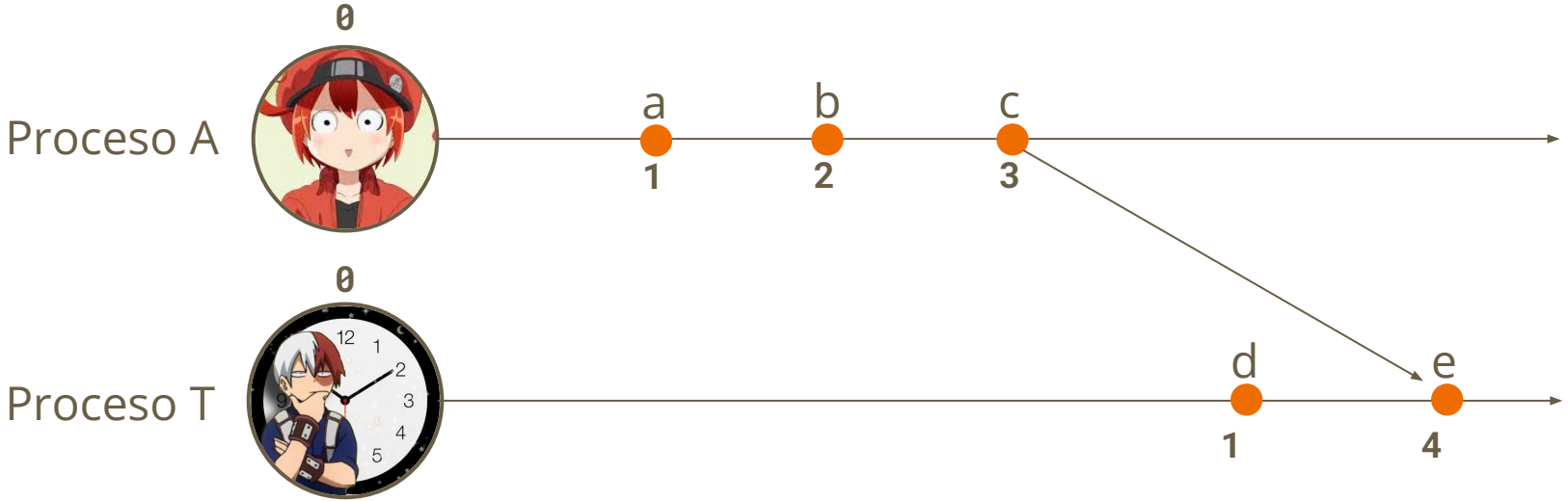
1. Relojes Vectoriales
2. Estados globales

Relojes Vectoriales

Sincronización de relojes - Algoritmos (Reloj Lógico)

Anteriormente.... vimos relojes Lógicos de Lamport

- Con la implementación anterior, si a → b entonces el contador al momento de ocurrir **a** será menor al momento de ocurrir **b**.
- Esta relación **es solo en una dirección**. **No podemos usar el contador para confiar plenamente en la causalidad.**



Sincronización de relojes - Algoritmos (Reloj Lógico)

Relojes Vectoriales

- Permiten establecer un orden parcial a los distintos sucesos o eventos.
  - Determinar el orden causal de eventos.
  - Identificar eventos concurrentes.
- Con eventos concurrentes nos referimos a que no sabemos si un evento ocurrió antes, después o al mismo tiempo que otro evento.
- **Podemos usar el vector para inferir el orden parcial.**

Sincronización de relojes - Algoritmos (Reloj Lógico)

Relojes Vectoriales

- Cada proceso P_i tiene su propio vector V_i en el que guarda tanto el estado de su reloj lógico como el de los demás procesos al momento de comunicarse.
  - $V_i[i]$ es el estado del reloj lógico de P_i
  - $V_i[z]$ es la última actualización del reloj lógico de P_z del que tiene constancia P_i
  - $V_i = V_k$ si $V_i[x] = V_k[x]$ para $x = 1,2,...,N$
  - $V_i \leq V_k$ si $V_i[x] \leq V_k[x]$ para $x = 1,2,...,N$
  - $V_i < V_k$ si $V_i \leq V_k$ y $V_i \neq V_k$

Sincronización de relojes - Algoritmos (Reloj Lógico)

Relojes Vectoriales - Reglas

- Cuando pasa un evento en el proceso P_i , se aumenta en una unidad el $V_i[i]$.
- Un evento es algo ocurrido únicamente en el proceso, el envío de un mensaje a otro proceso o la recepción de un mensaje.
- Cuando el proceso P_i recibe un vector V_z , primero se aumenta en una unidad el $V_i[i]$. Luego, se actualiza el vector V_i del siguiente modo:
  - $V_i[x] = \max(V_i[x], V_z[x])$ para $x = 1,2,...,N$

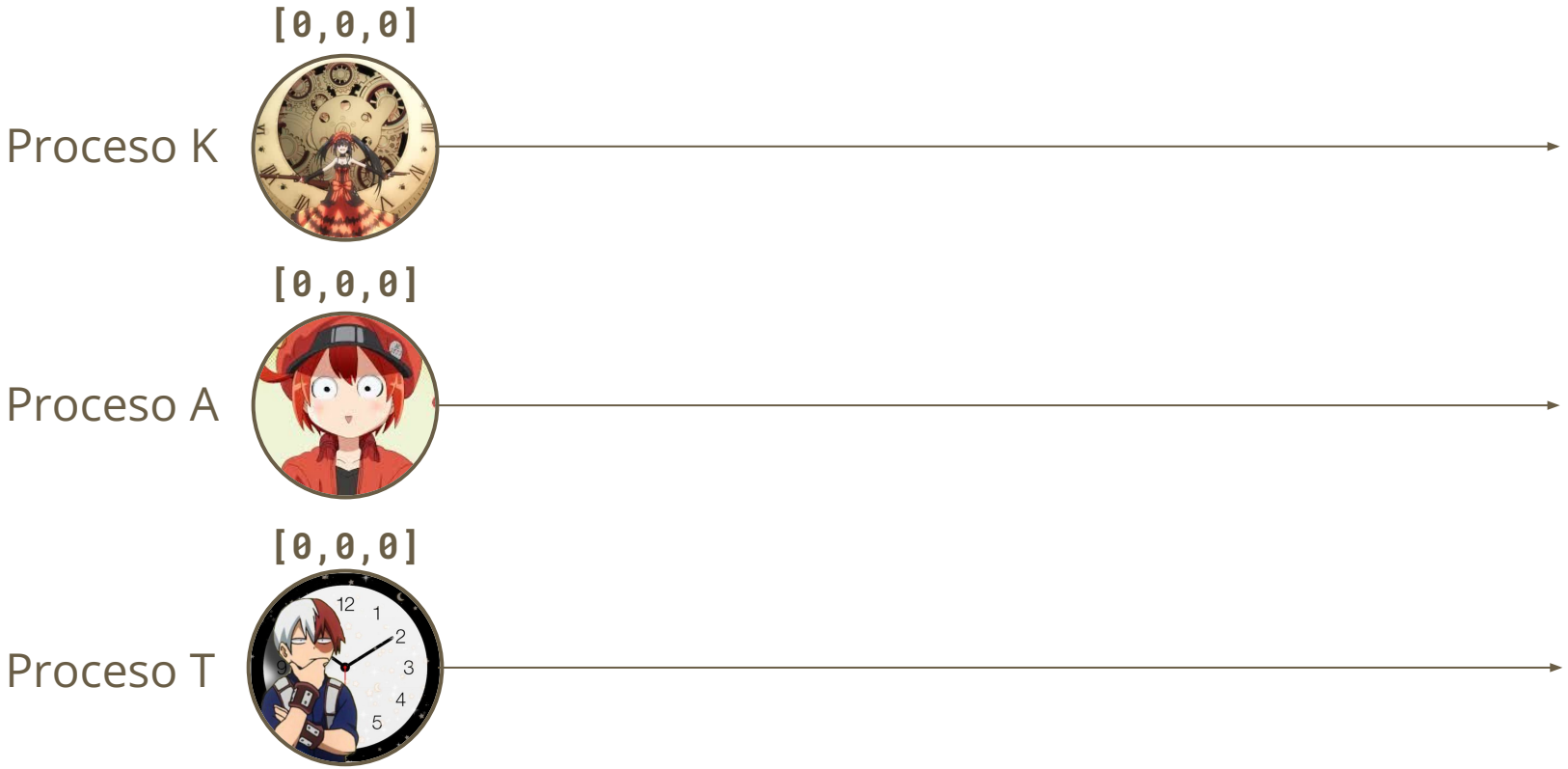
Sincronización de relojes - Algoritmos (Reloj Lógico)

Relojes Vectoriales - Reglas

- Cuando pasa un evento en el proceso P_i , se aumenta en una unidad el $V_i[i]$.
- Un evento es algo ocurrido únicamente en el proceso, el envío de un mensaje a otro proceso o la recepción de un mensaje.
- Cuando el proceso P_i recibe un vector V_z , primero se aumenta en una unidad el $V_i[i]$. Luego, se actualiza el vector V_i del siguiente modo:
  - $V_i[x] = max(V_i[x], V_z[x])$ para $x = 1,2,...,N$
- Con esto, se puede comparar 2 eventos (**a** y **b**) según sus vectores.
  - $V^a_i < V^b_k$ El evento b en P_k ocurrió después que el evento a en P_i
  - En otro caso, el evento a y b son "concurrentes".

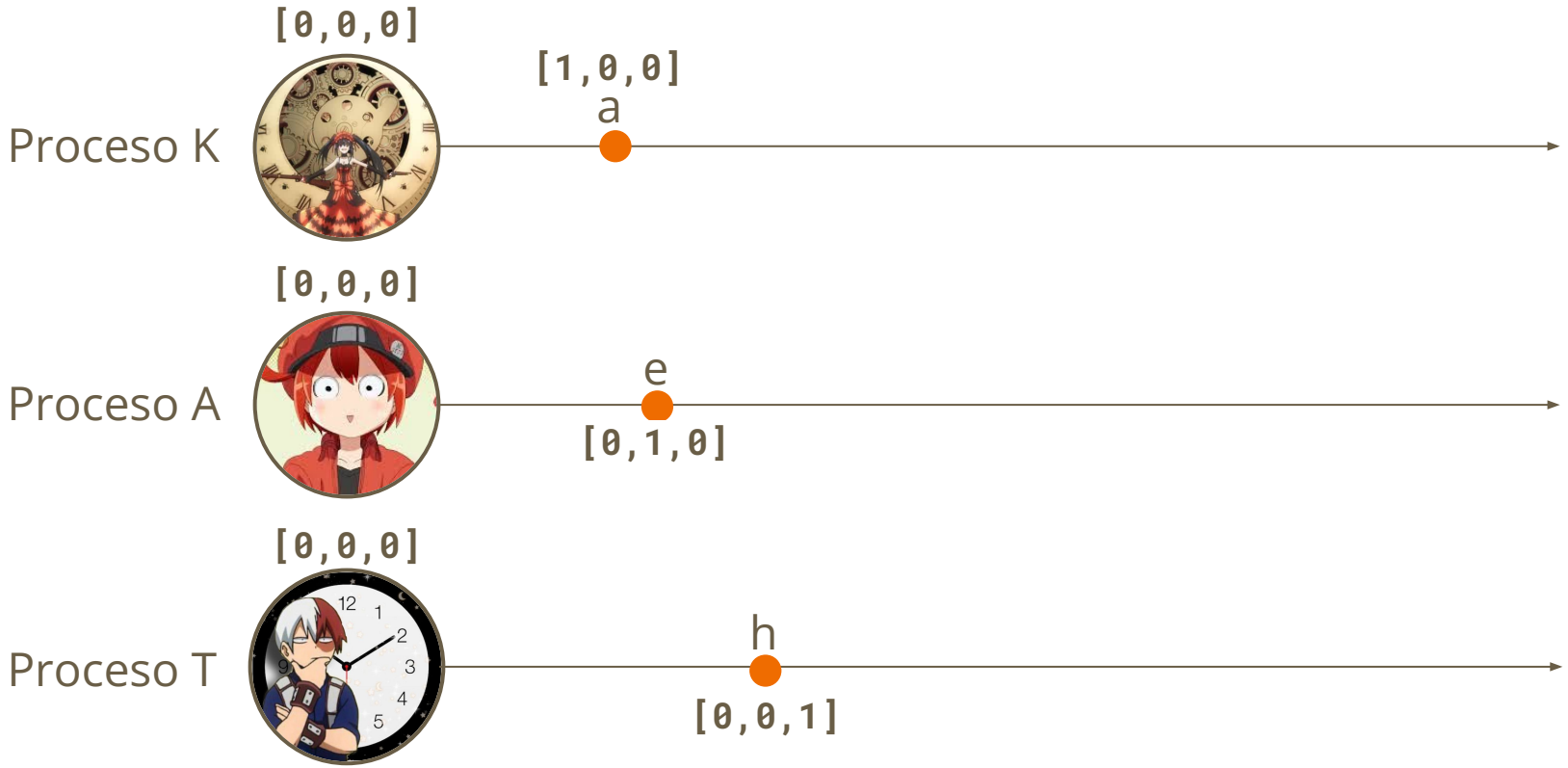
Sincronización de relojes - Algoritmos (Reloj Lógico)

Relojes Vectoriales



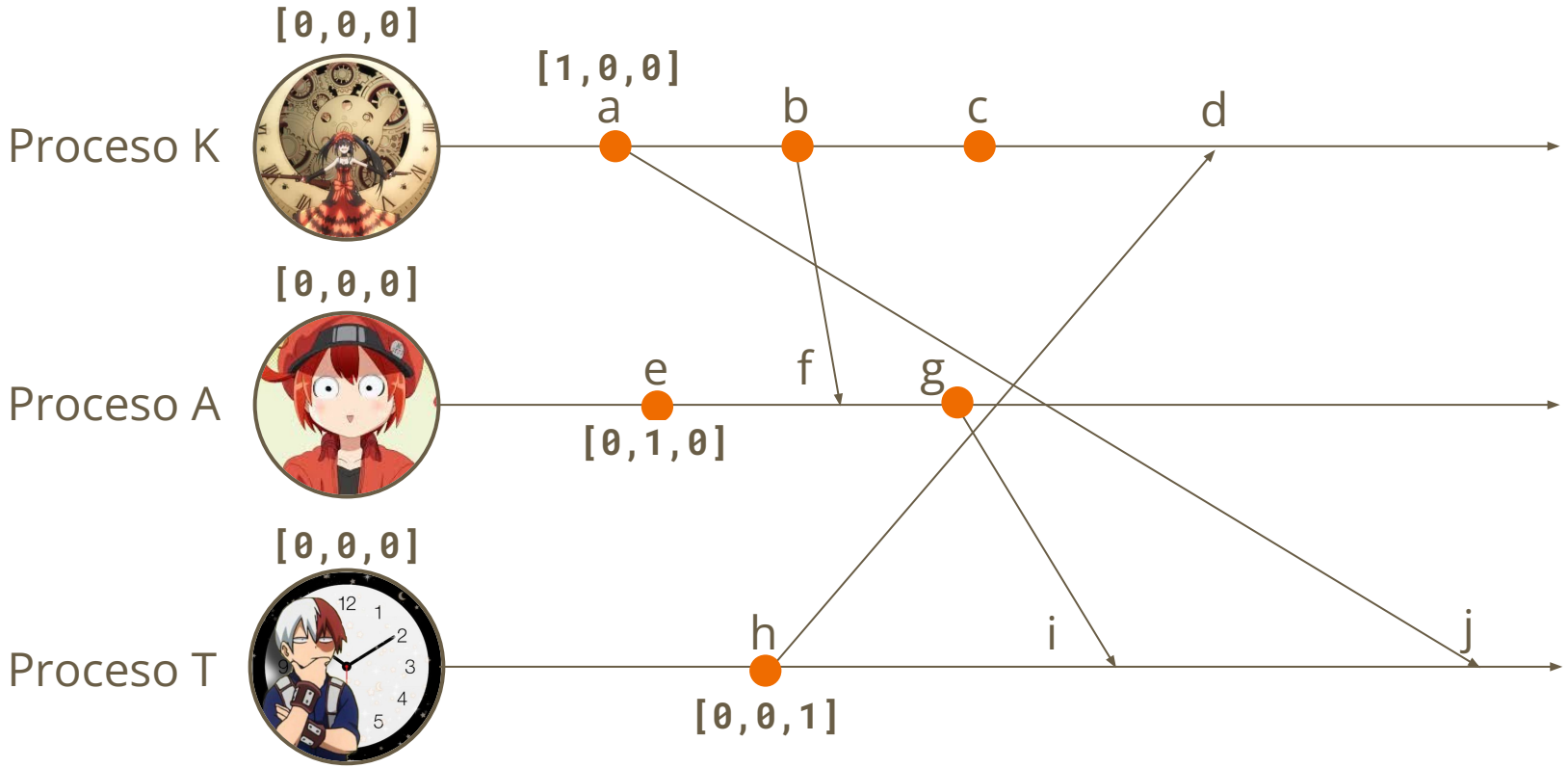
Sincronización de relojes - Algoritmos (Reloj Lógico)

Relojes Vectoriales



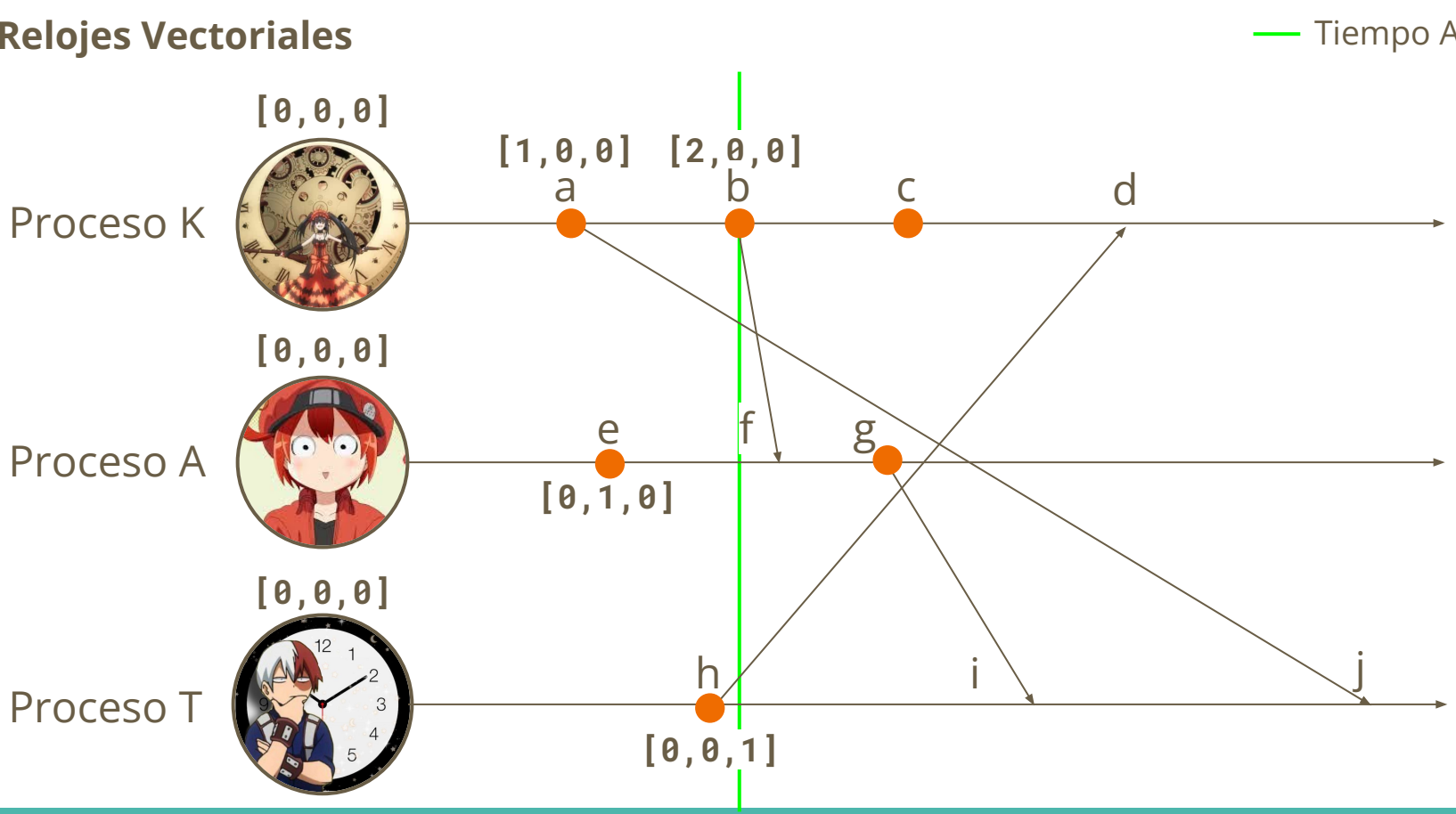
Sincronización de relojes - Algoritmos (Reloj Lógico)

Relojes Vectoriales



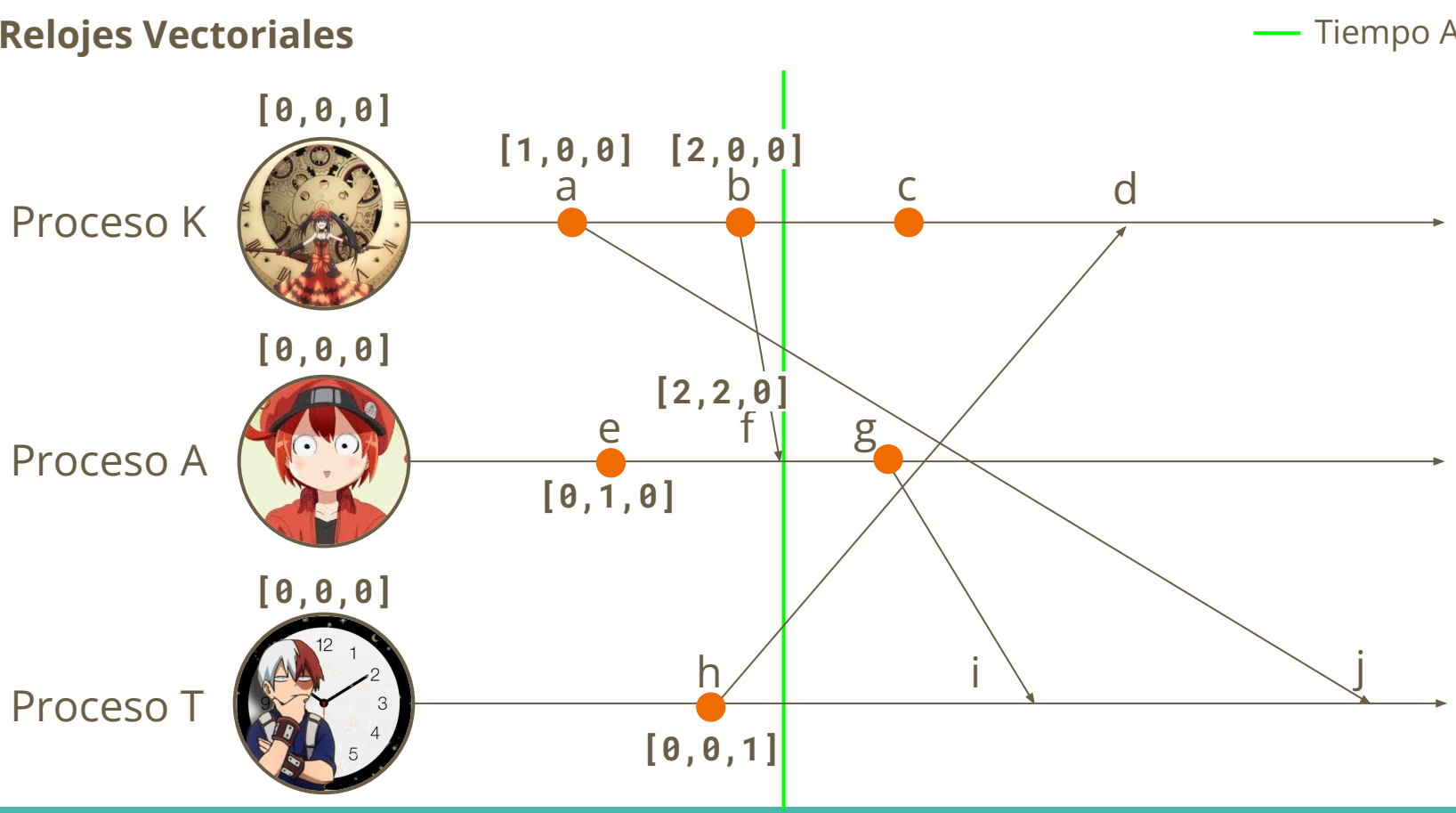
Sincronización de relojes - Algoritmos (Reloj Lógico)

Relojes Vectoriales



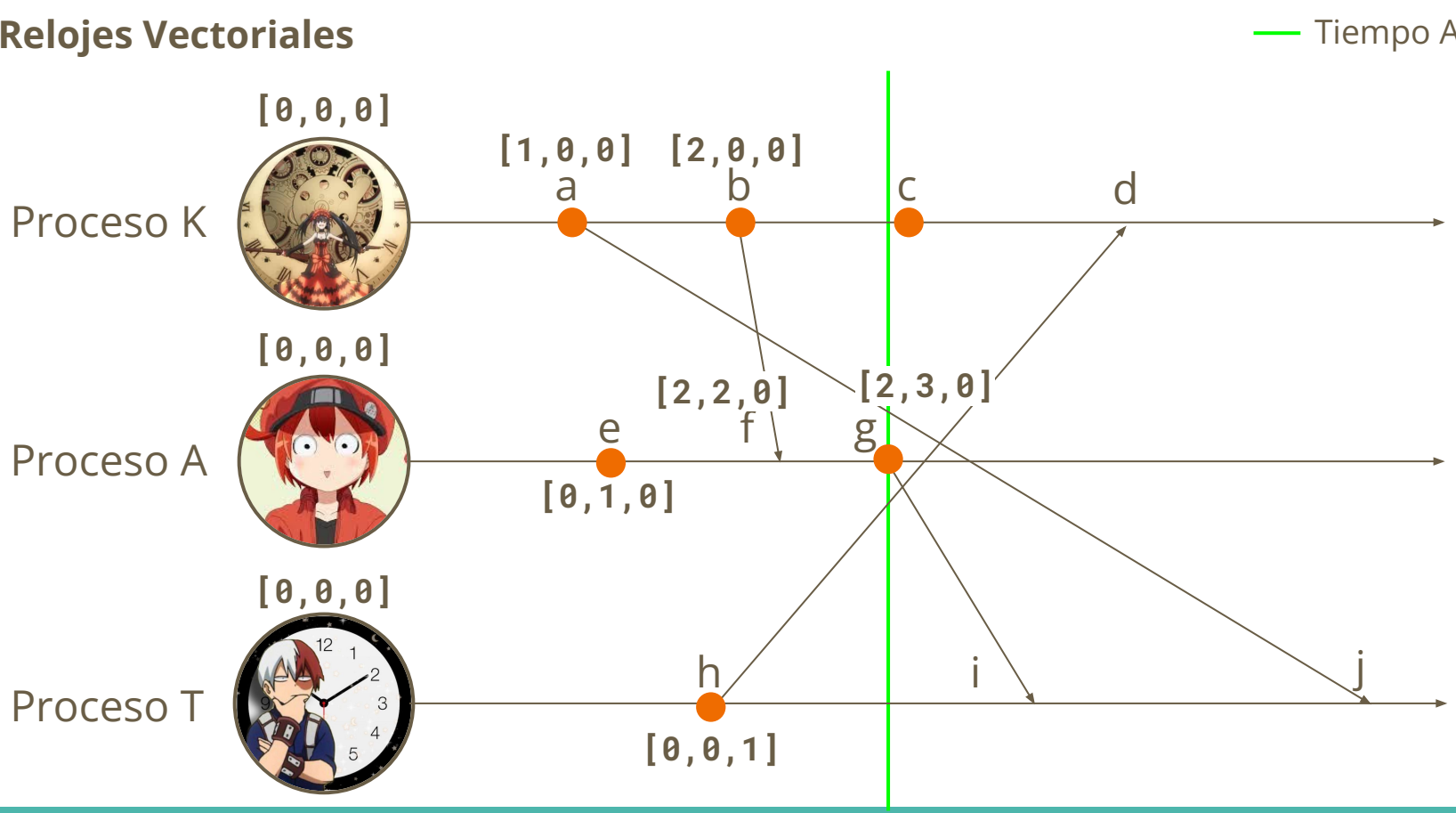
Sincronización de relojes - Algoritmos (Reloj Lógico)

Relojes Vectoriales



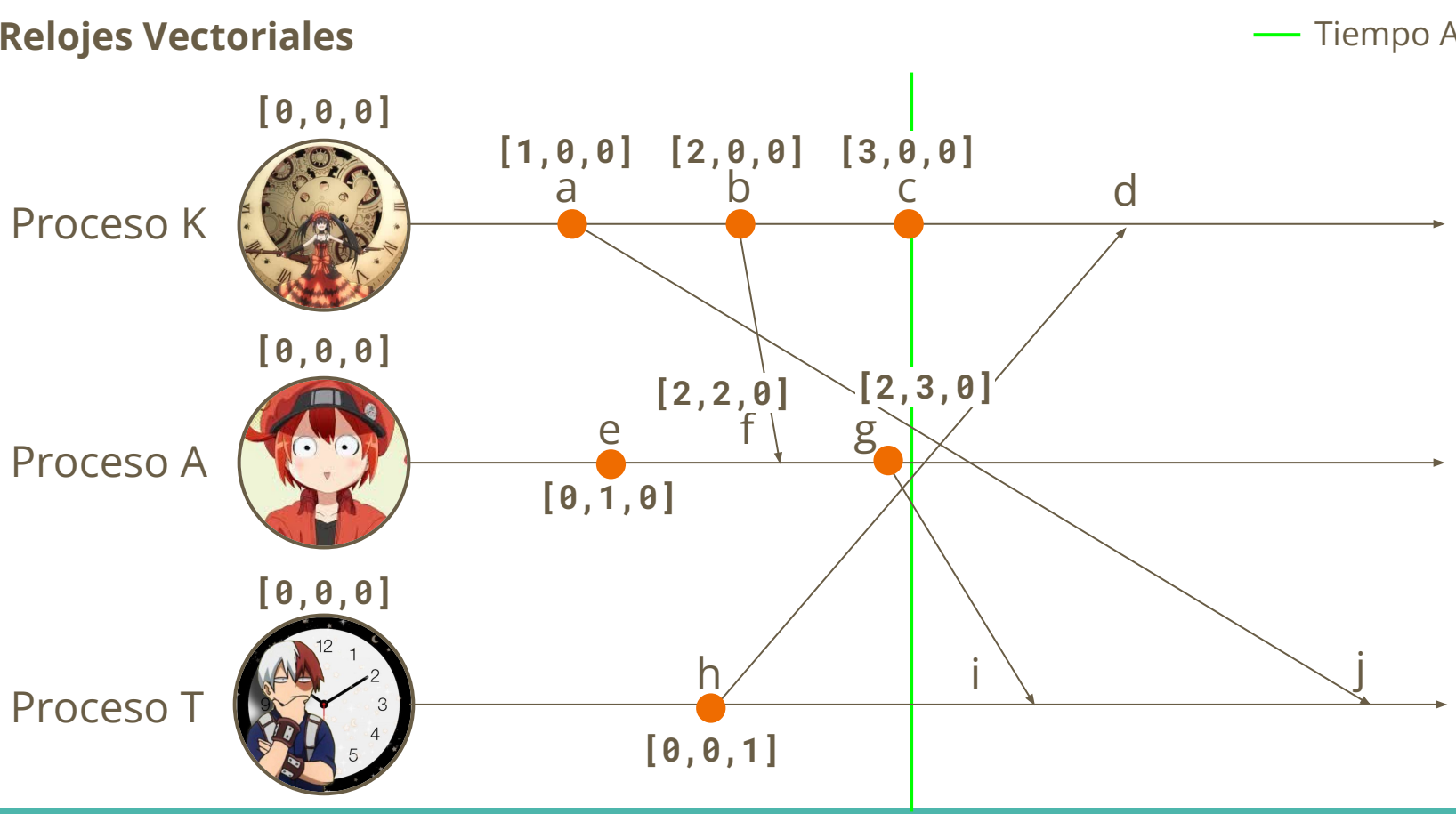
Sincronización de relojes - Algoritmos (Reloj Lógico)

Relojes Vectoriales



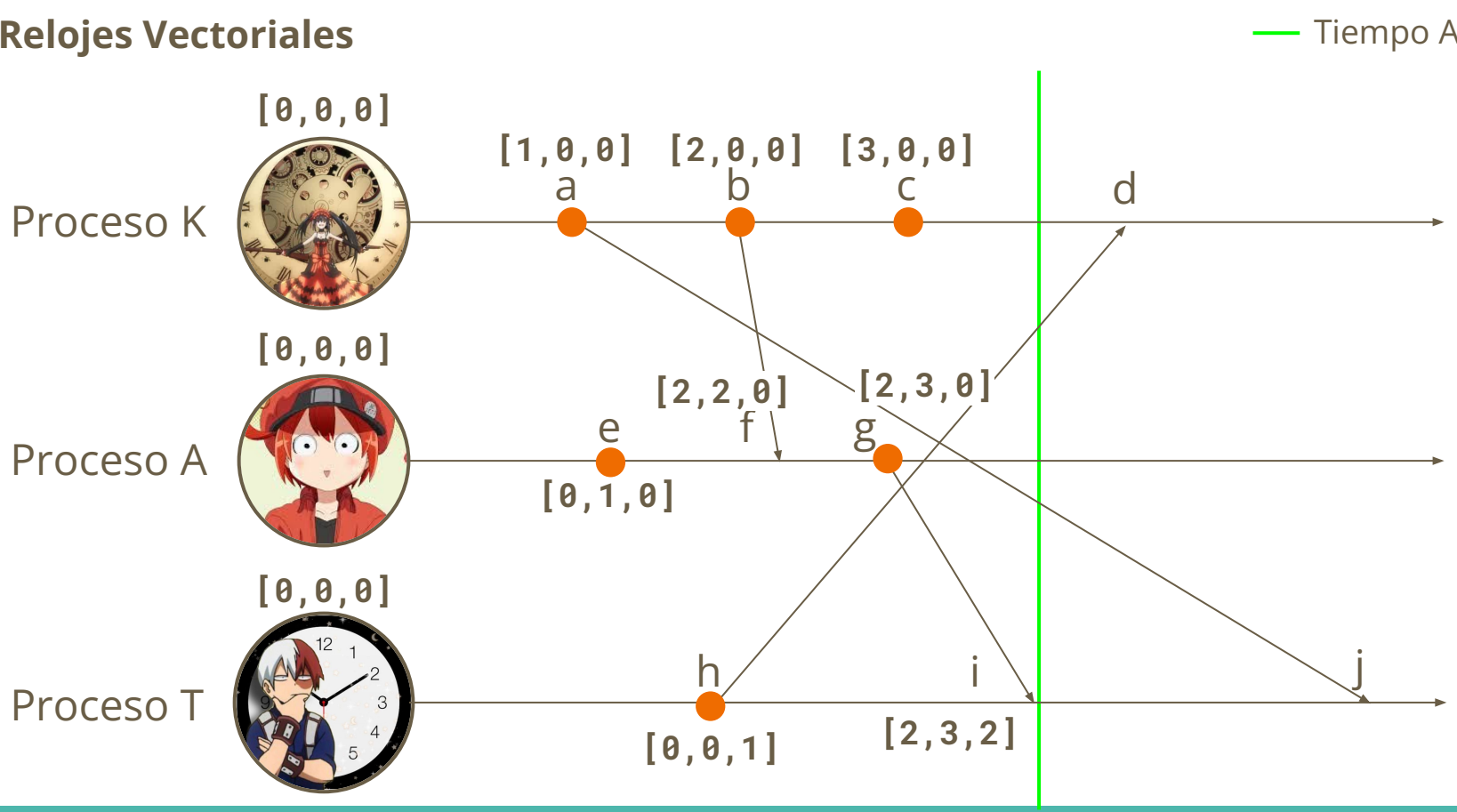
Sincronización de relojes - Algoritmos (Reloj Lógico)

Relojes Vectoriales



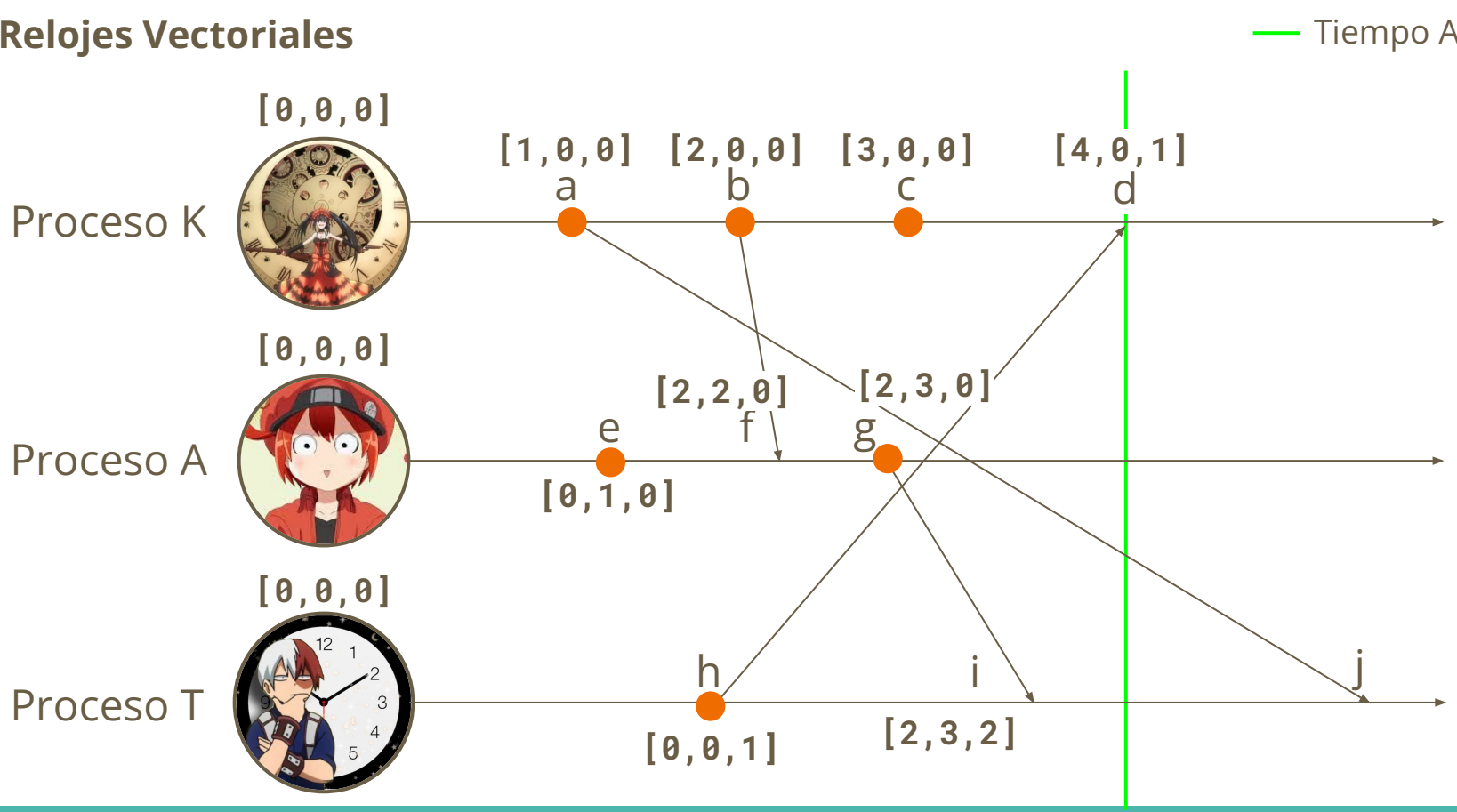
Sincronización de relojes - Algoritmos (Reloj Lógico)

Relojes Vectoriales



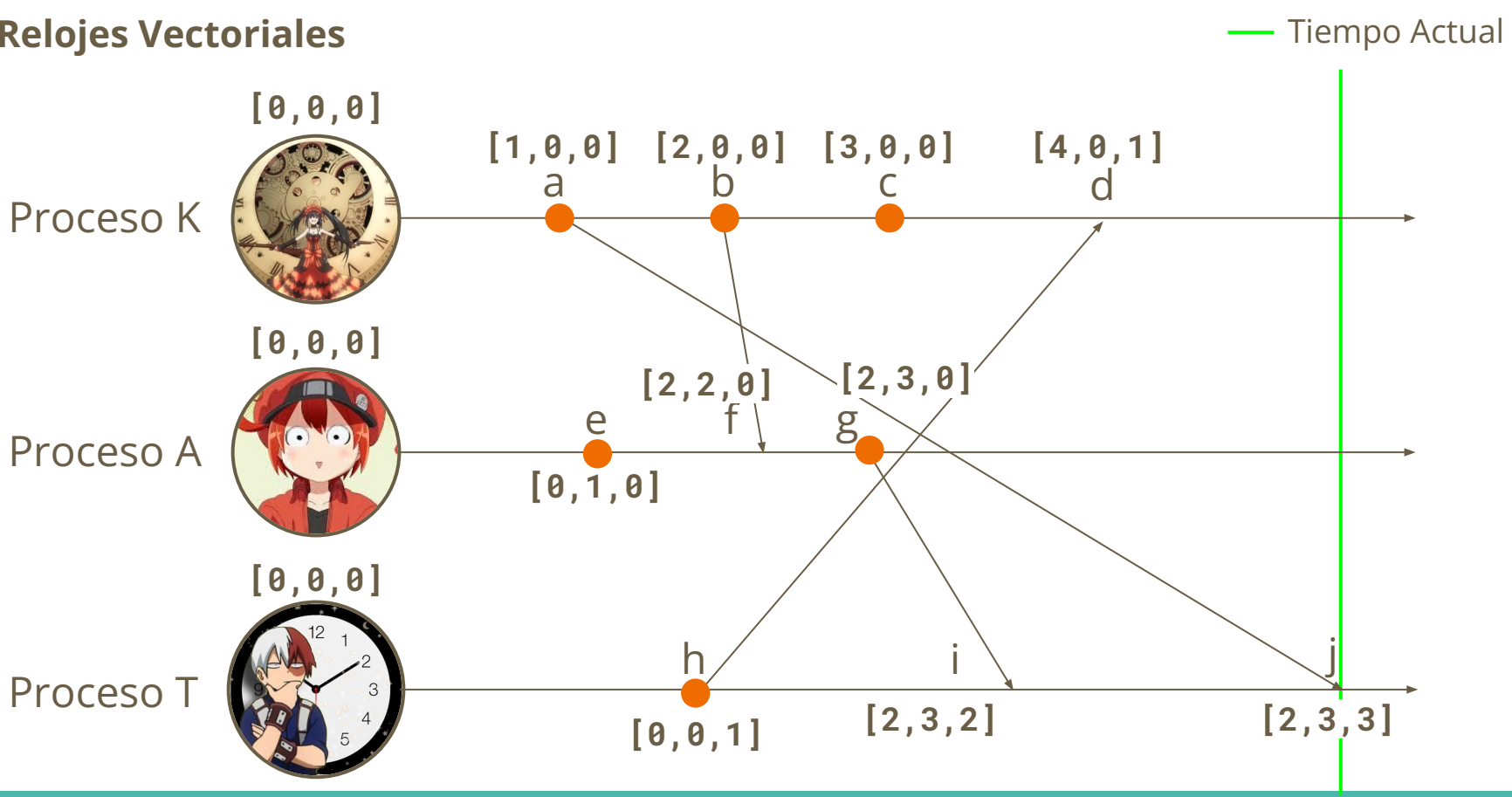
Sincronización de relojes - Algoritmos (Reloj Lógico)

Relojes Vectoriales



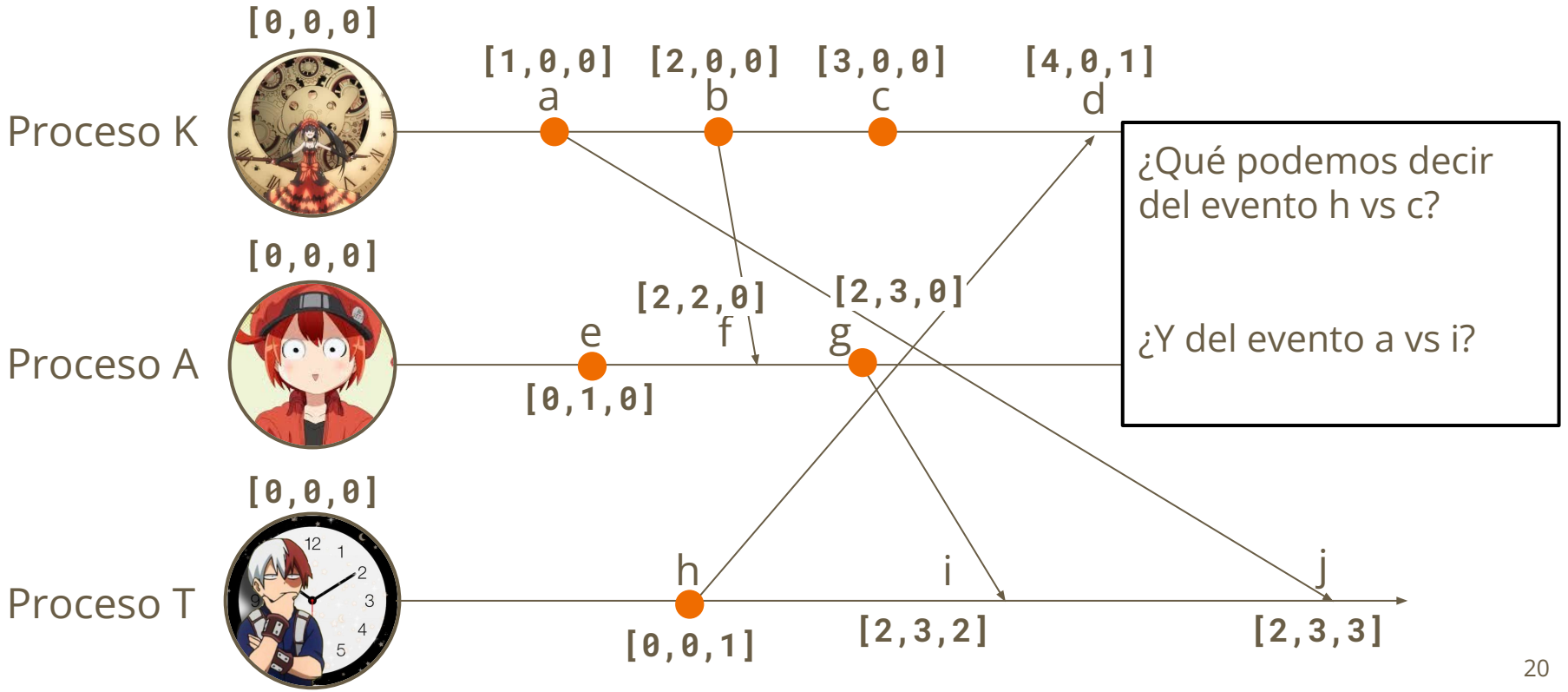
Sincronización de relojes - Algoritmos (Reloj Lógico)

Relojes Vectoriales



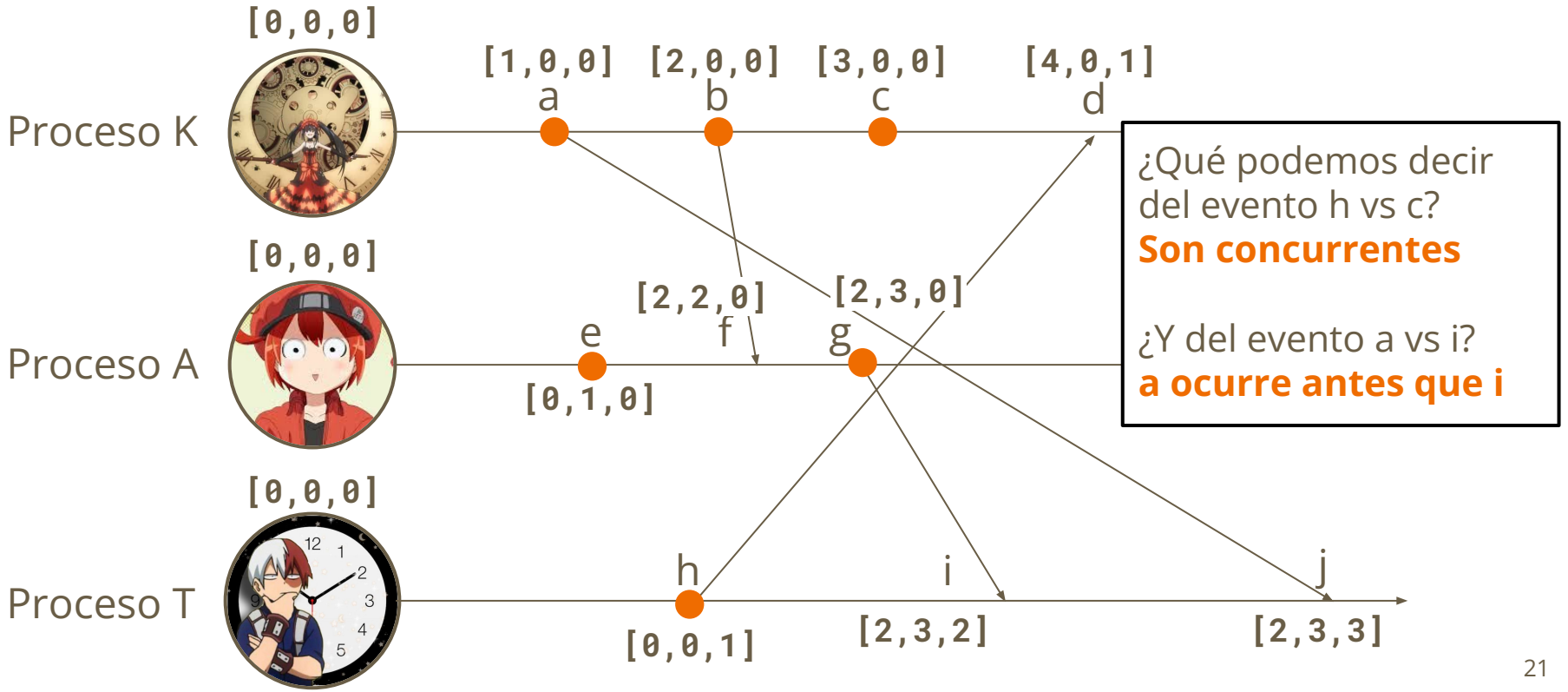
Sincronización de relojes - Algoritmos (Reloj Lógico)

Relojes Vectoriales



Sincronización de relojes - Algoritmos (Reloj Lógico)

Relojes Vectoriales



Estados Globales

¿Qué son?

Algoritmo Chandy-Lamport

Estados Globales

- También llamado "*Global Snapshot*".
- Consiste en los **estados locales** de cada nodo en el sistema distribuido, junto con los **mensajes en tránsito** en los canales de comunicación.
- En un computador, podemos definir un tiempo X para guardar todas las variables, y todos los procesos almacenan su estado en dicho tiempo X.

Estados Globales

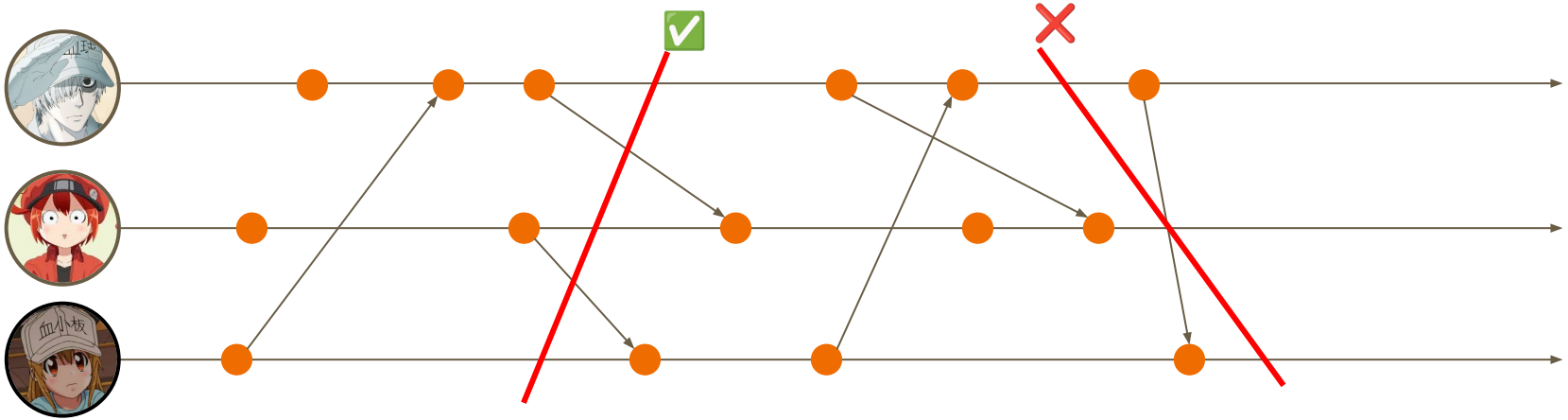
- También llamado "*Global Snapshot*".
- Consiste en los **estados locales** de cada nodo en el sistema distribuido, junto con los **mensajes en tránsito** en los canales de comunicación.
- En un computador, podemos definir un tiempo X para guardar todas las variables, y todos los procesos almacenan su estado en dicho tiempo X.
- Su principal problema es determinar esta "instantánea" con las características de un sistema distribuido:
  - Casi siempre habrán mensajes en tránsito.
  - Es inevitable el retraso entre los mensajes.
  - Como ya vimos... es difícil mantener todos los nodos sincronizados en todo momento.
  - Es casi imposible "pausar" todos los nodos en un mismo instante.

Estados Globales - ¿Por qué es crucial? 🤔

- **Recolección de basura distribuida**: Saber si un objeto no tiene referencias en ningún lugar del sistema.
  - Nodo A referencia un dato está disponible en nodo B, pero el nodo B no tiene ese dato en su estado actual porque lo eliminó.
- **Detección de interbloqueos (*deadlocks*)**: Identificar situaciones donde los nodos esperan recursos indefinidamente.
- **Depuración de sistemas distribuidos**: Entender comportamientos inesperados o violaciones de seguridad transitorias.
- ***Checkpoints***: Determinar puntos de seguridad para una base de datos distribuida.
- **Término de algoritmo**: Verificar si todos los nodos ya terminaron de ejecutar algún algoritmo distribuido.

Estados Globales - Corte Consistente

- Forma de definir un estado global, representándose como una "línea" a través de las historias de ejecución de los nodos.
- Un corte es **consistente** si:
  - Para cada evento, se contiene todos los eventos que lo "precedieron causalmente".
  - No muestra un "efecto sin una causa": Si un mensaje fue recibido antes del corte, debe aparecer también como enviado antes del corte.



Algoritmo de Chandy-Lamport

- Permite capturar un estado global consistente de un sistema distribuido sin necesidad de detener su ejecución normal.
- Su mecanismo central consiste en el envío de un mensaje marcador (*marker)*:
  - Estos marcadores viajan a través de los canales de comunicación y no afectan los mensajes de aplicación.
  - Todo par de nodos tiene un canal de comunicación para conversar.
- Compuesto por 3 fases: inicialización, propagación y finalización.

Algoritmo de Chandy-Lamport - Fase 1 (Inicialización)

- Nodo P_x inicia el algoritmo
  - Nodo P_x guarda su estado local y envía un *marker* a todos los demás canales de comunicación
  - Nodo P_x comienza a grabar todos los mensajes entrantes en sus canales de comunicación

Algoritmo de Chandy-Lamport - Fase 2 (Propagación)

Caso 1: Nodo P_y recibe por primera vez un marcador de algún nodo (P_z)

- Nodo P_y guarda su estado local y envía un marcador por todos sus canales de comunicación.
- Sea C_{zy} el canal de comunicación entre aquel que le mandó el primer marcador y este nodo, C_{zy} queda vacío.
- Nodo P_y comienza a grabar todos los mensajes entrantes en todos los demás canales de comunicación.

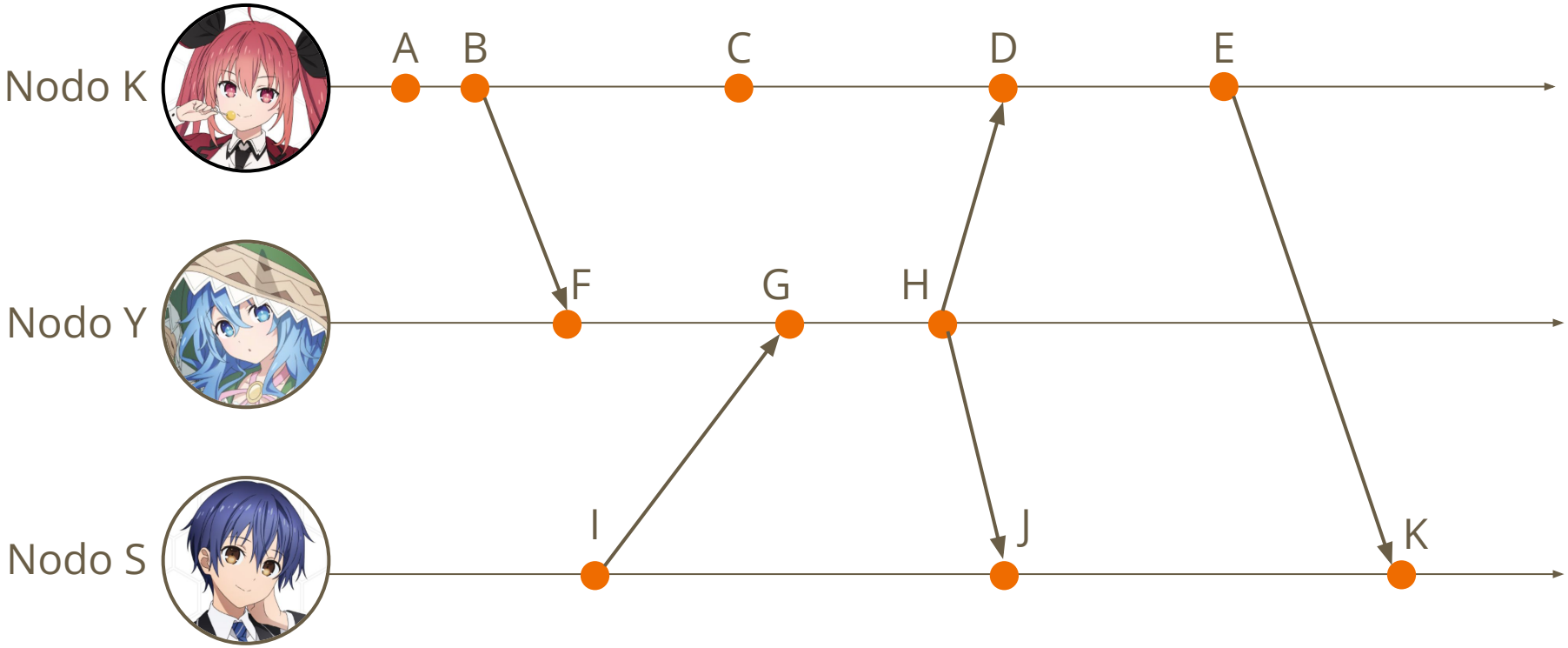
Caso 2: Nodo P_y recibe un marcador de algún nodo P_k luego de ya haber visto uno.

- Guarda el estado del canal C_{ky} como los mensajes recibidos desde que empezó la grabación de dicho canal.
- Deja de escuchar el canal C_{ky} para fines de este algoritmo.

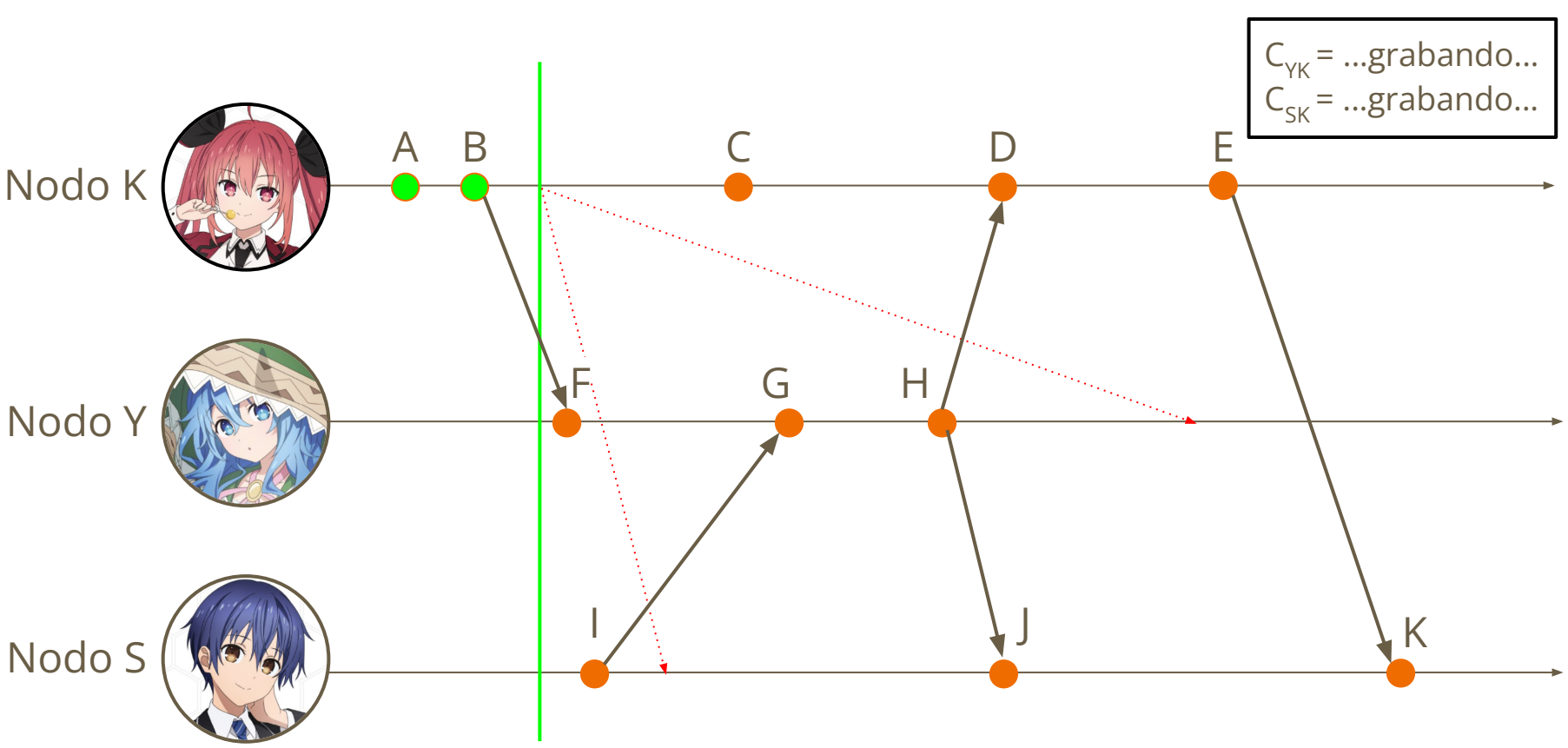
Algoritmo de Chandy-Lamport - Fase 3 (Finalización)

- El algoritmo termina cuando todos los nodos han:
  - Registrado su estado local.
  - Registrado el estado de todos sus canales de comunicación.
- Un nodo "líder" o central puede verificar que todos los nodos indiquen haber finalizado y solicitar los estados guardados para guardarlos en un solo lugar.

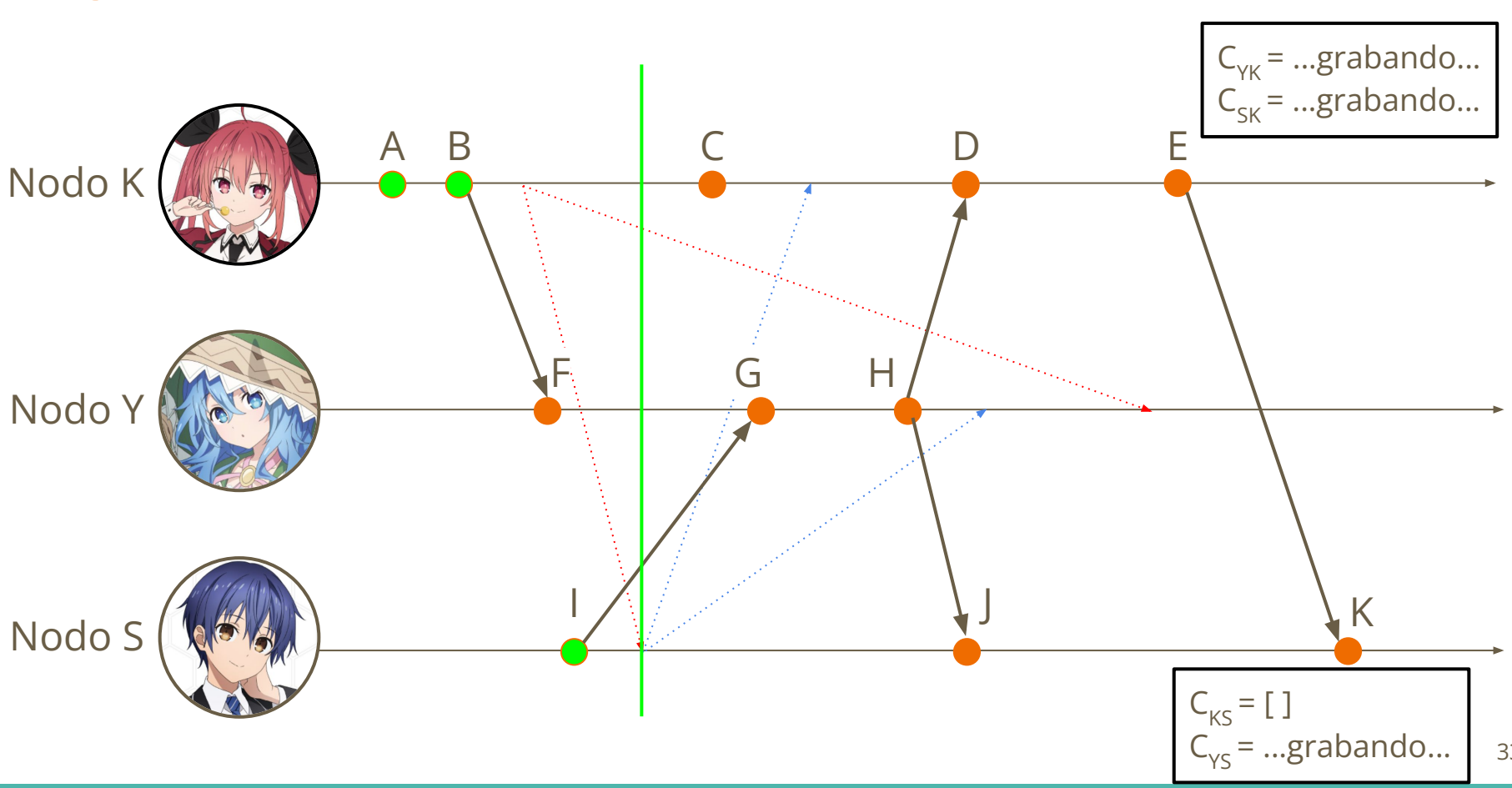
Algoritmo de Chandy-Lamport - Ejemplo



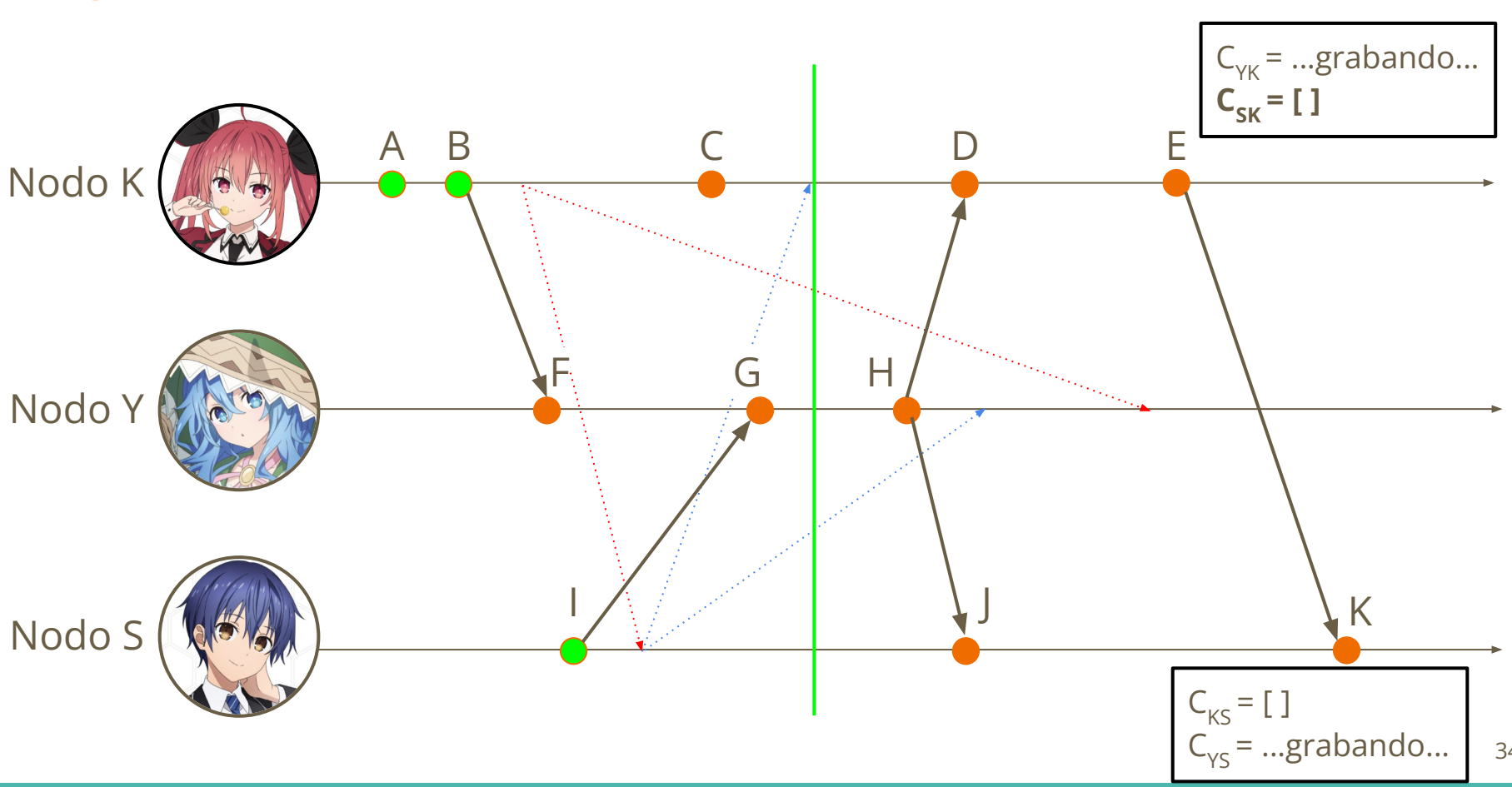
Algoritmo de Chandy-Lamport - Ejemplo



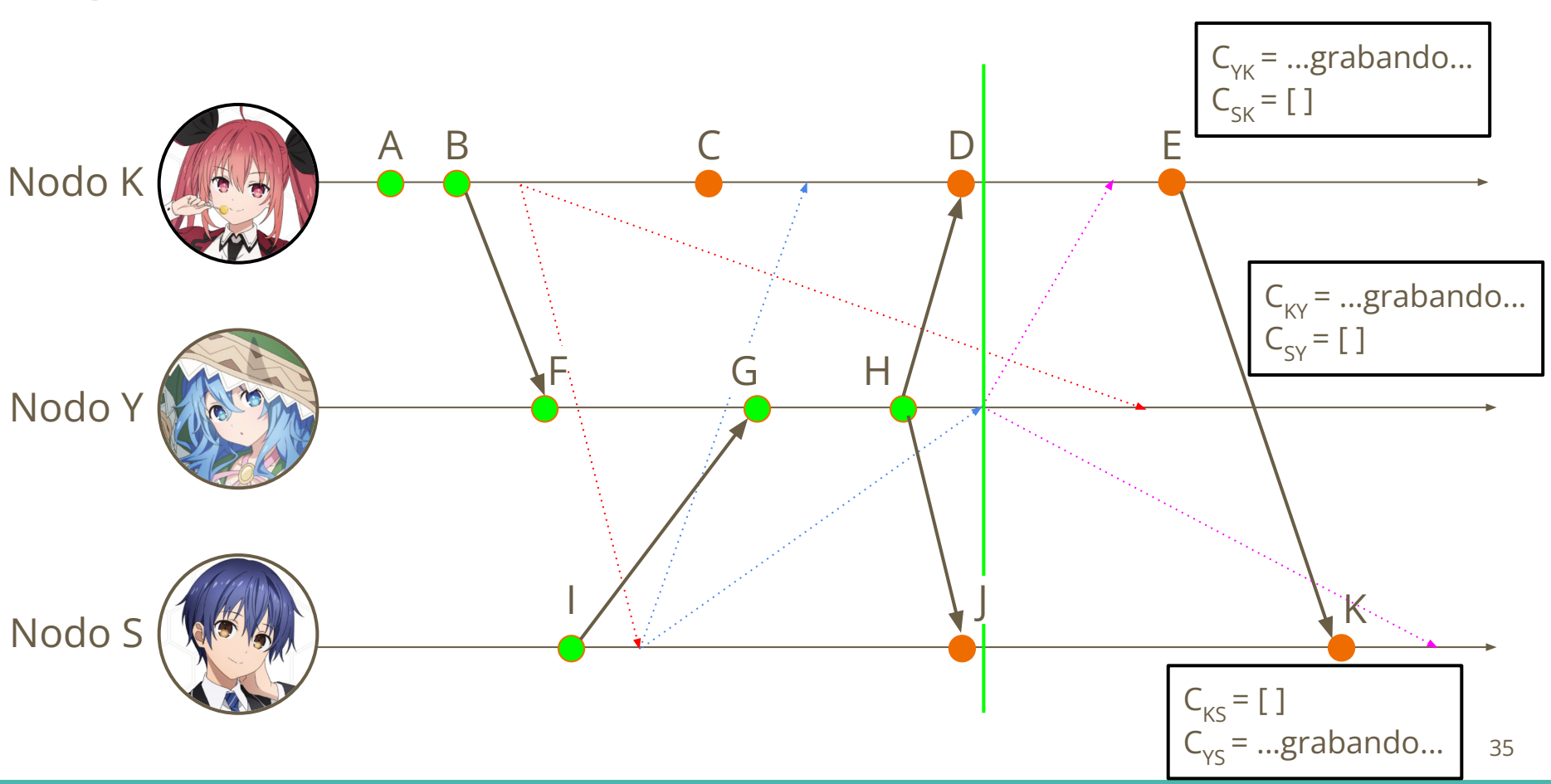
Algoritmo de Chandy-Lamport - Ejemplo



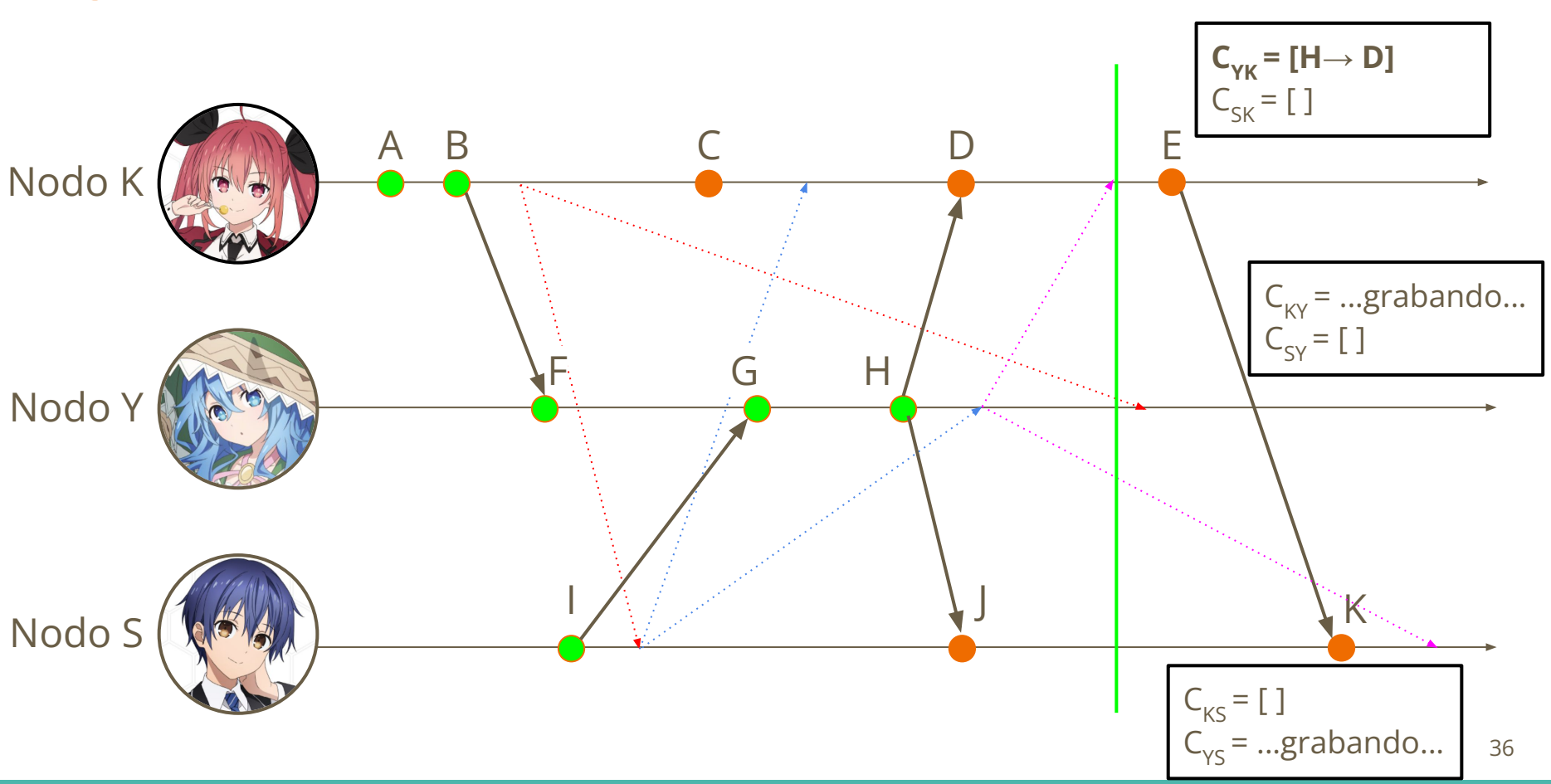
Algoritmo de Chandy-Lamport - Ejemplo



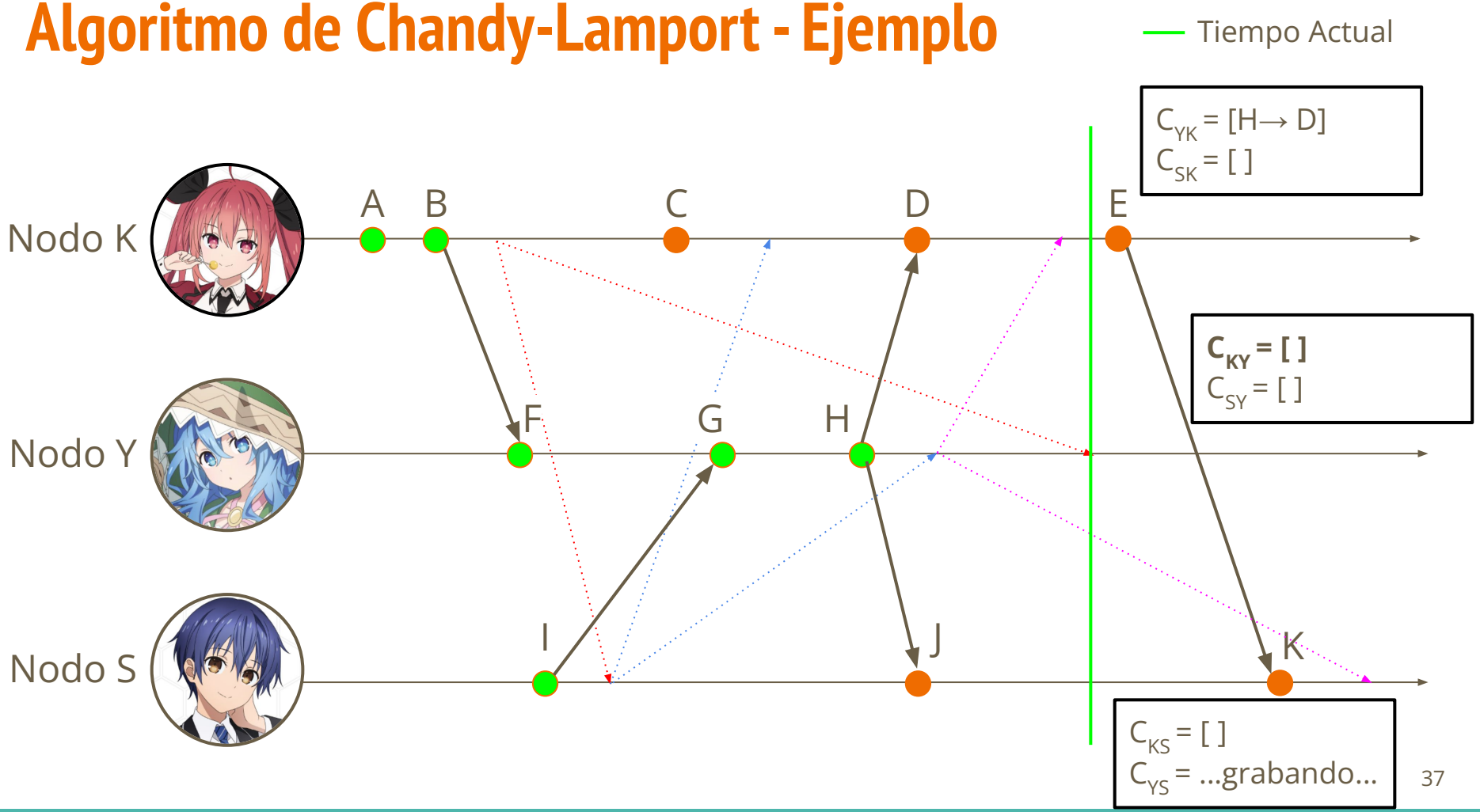
Algoritmo de Chandy-Lamport - Ejemplo



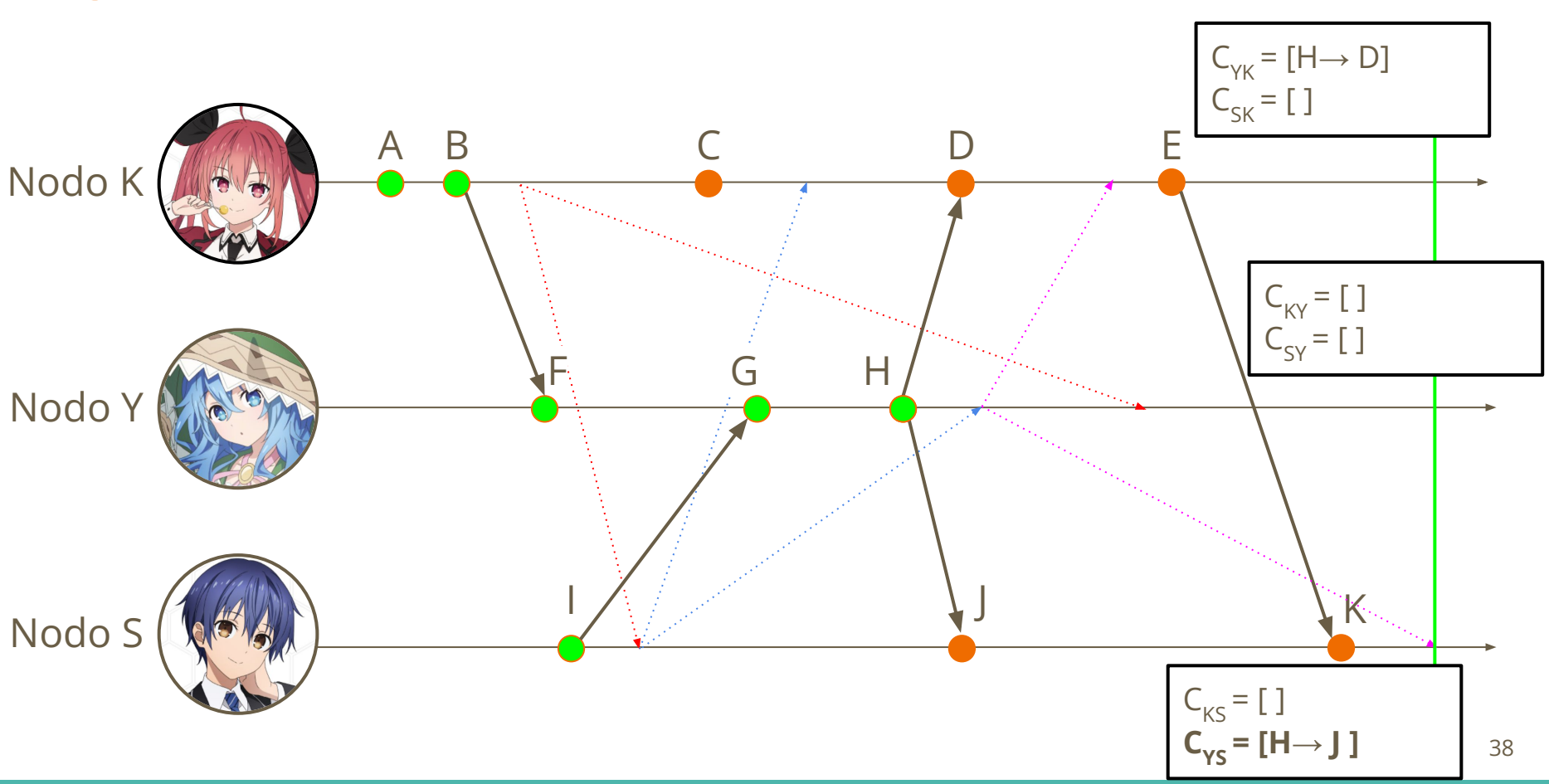
Algoritmo de Chandy-Lamport - Ejemplo



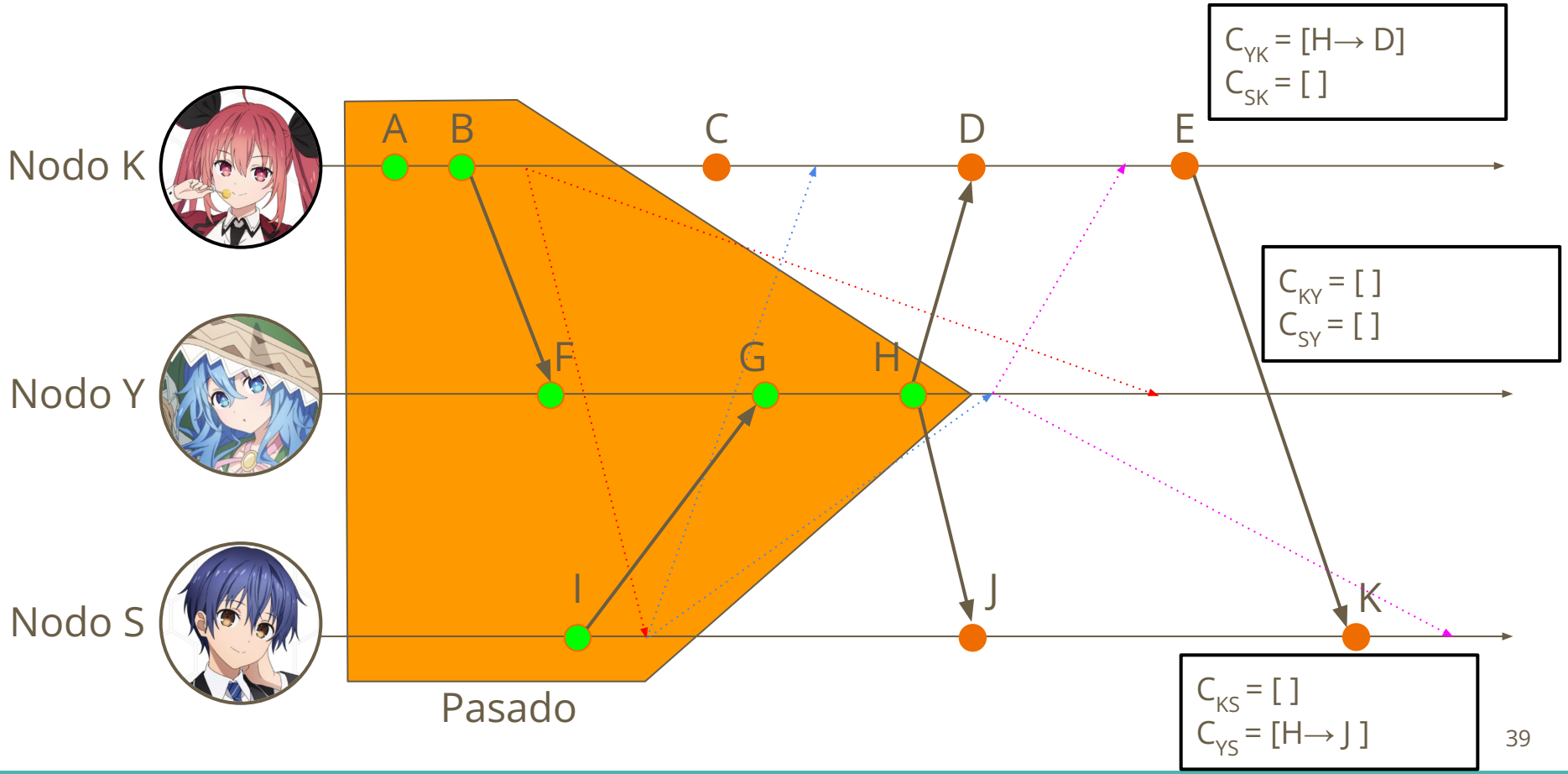
Algoritmo de Chandy-Lamport - Ejemplo



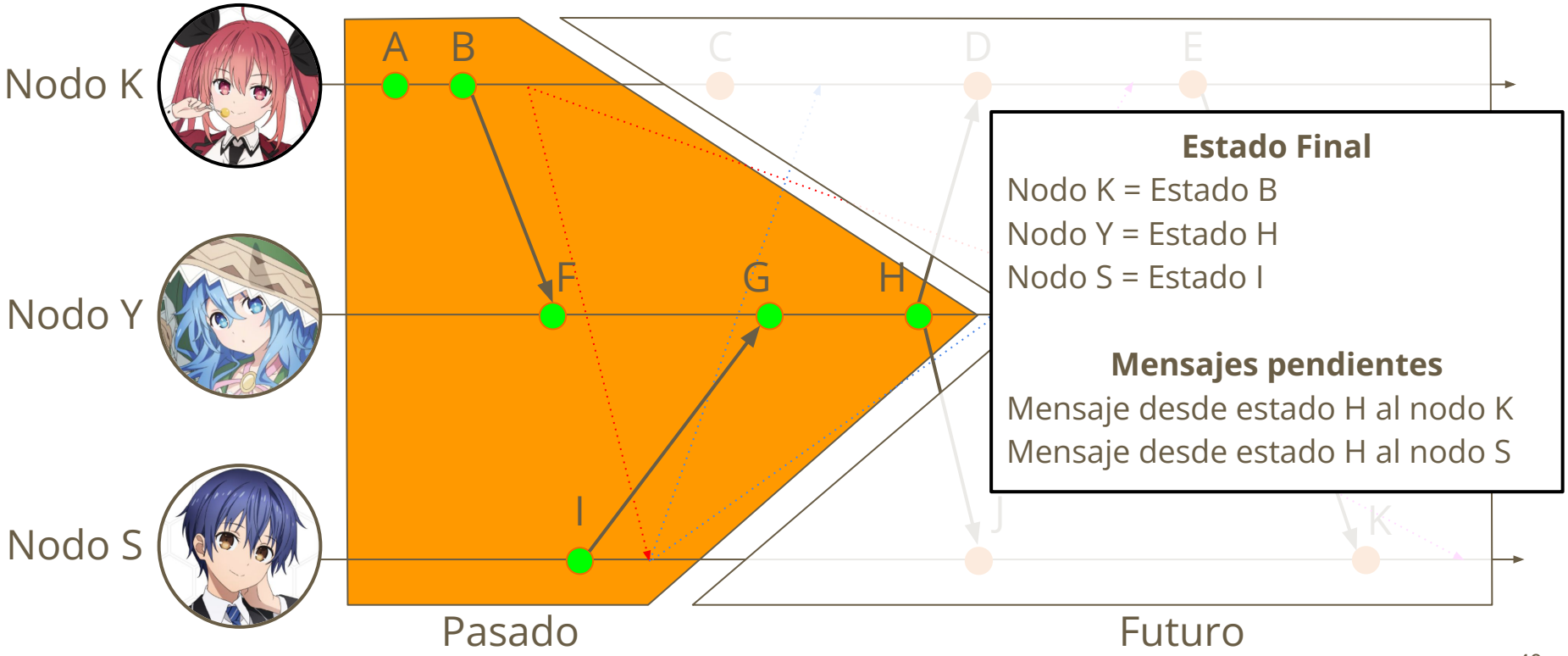
Algoritmo de Chandy-Lamport - Ejemplo



Algoritmo de Chandy-Lamport - Ejemplo



Algoritmo de Chandy-Lamport - Ejemplo



Algoritmo de Chandy-Lamport - Supuestos

Algunas condiciones del algoritmo son bien realista

- Cualquier nodo puede iniciar el algoritmo.
- El algoritmo de *snapshot* no interfiere con la ejecución normal de los nodos.
- Cada nodo registra su estado local y el estado de sus **canales de entrada**.
- El estado global se recolecta de manera distribuida.

Algoritmo de Chandy-Lamport - Supuestos

Algunas condiciones del algoritmo son bien realista

- Cualquier nodo puede iniciar el algoritmo.
- El algoritmo de *snapshot* no interfiere con la ejecución normal de los nodos.
- Cada nodo registra su estado local y el estado de sus **canales de entrada**.
- El estado global se recolecta de manera distribuida.

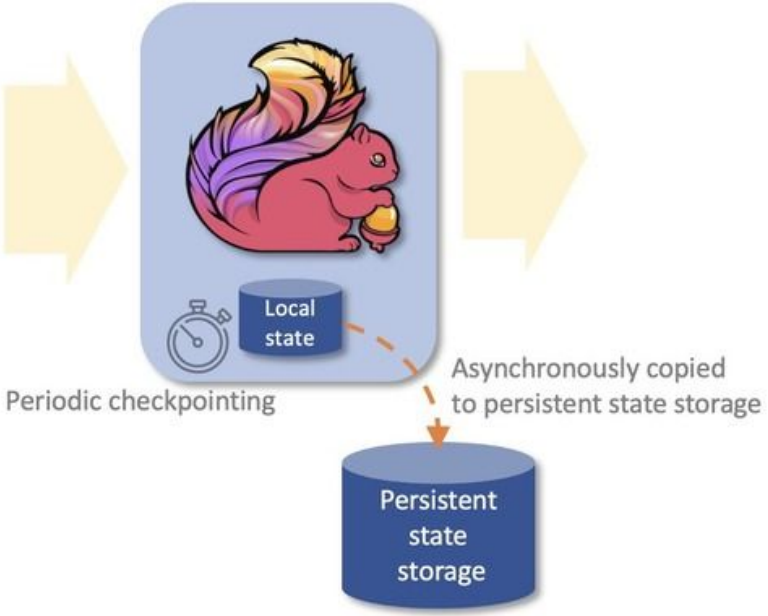
Otras, no tanto...

- Comunicación es confiable → No hay fallas durante la ejecución del algoritmo.
- Existe un canal de comunicación entre cualquier par de nodos del sistema.
- Canales de comunicación unidireccionales y entregan mensajes en orden FIFO.

Algoritmo de Chandy-Lamport - Hoy en día

Apache Flink

- Herramienta para procesar datos en tiempo real o por lotes, que puede recordar estados, escalar a muchos computadores, y está hecha para trabajar con flujos de datos grandes, veloces y vivos.
- Utiliza un mecanismo de *snapshots* está descrito en el paper: *Lightweight Asynchronous Snapshots for Distributed Dataflows*
- Está inspirado en el algoritmo que vimos hoy.



Fuente: ¿Qué es Apache Flink? y Stateful Stream Processing | Apache Flink

Próximos eventos

Próxima clase

- **Miércoles**: será para presentar la tarea 1.
- **Próximo Lunes**: Consenso en Sistemas Distribuidos
  - ¿Cómo nos ponemos de acuerdo en una operación distribuida?
  - Algoritmo Paxos.
  - ¿Qué pasa si tenemos un nodo traidor? 😵 🗡️

Evaluaciones

- El miércoles se libera la Tarea 1. No la pateen tanto al final.

IIC2523
Sistemas Distribuidos

Hernán F. Valdivieso López
(2025 - 2 / Clase 06)

Créditos (animes utilizados)

Date a Live




Boku no Hero Academia

Hataraku Saibou